MOTIX™ TLE989x/TLE988x

Microcontroller with CAN-FD and NFET Driver for BLDC Applications
AK step

User manual

About this document

This user manual is addressed to embedded hardware and software developers. It provides the reader with detailed descriptions about the behavior of the TLE989x/TLE988x functional units and their interaction.

The manual describes the functionality of the superset device of the TLE989x/TLE988x family. For the available functionality (features) of a specific TLE989x/TLE988x derivative (derivative device), please refer to the respective datasheet.

Table of contents

1 Overview

The TLE989x/TLE988x has following features:

- Arm® Cortex®[1)]-M3 core system
  - Up to 60 MHz CPU and system frequency
  - Arm® NVIC interrupt controller with 32 interrupt requests and 32 levels
  - Arm® Coresight debug with 2 hardware breakpoints and 2-wire interface (SWD)
  - Arm® µDMA direct memory access controller with 8 channels
  - Arm® SysTick system timer (24-bit)
- Single system power supply connected to battery supply (VS pin)
  - Operating range from 5.5 V to 28 V, extended operating range from 3 V to 40 V
  - Low-dropout voltage regulators (LDO) for pad and CAN supply (VDDP, VCAN) and core supply (VDDC)
  - 5 V low-dropout voltage regulator for on-board loads (VDDEXT)
- On-chip clock generation
  - Low power oscillators as clock source in startup and power saving modes, also used as independent safe watchdog timer clock
  - High precision oscillator as base and fallback clock source for system with clock watchdog
  - Oscillator circuit for external crystal/resonator for accurate clock source with clock watchdog
  - Two low jitter phase lock loop circuits (PLL0/1) with programmable prescaler for system clock with loss-of-lock detection and fallback clock
- Control state machine for switching the system states
  - Active mode: system fully operational with power saving options for frequency and peripherals; bridge driver in active mode, brake mode or off; current consumption typ. 20 mA at V_S (MCU and CAN active, bridge driver off)
  - Stop mode: MCU subsystem stopped with monitoring and communication peripherals listening
  - Sleep mode: MCU subsystem unpowered with wake monitoring active; wake-up time typ. 2 ms and typ. 30 µA at V_S
  - Wake capabilities for stop and sleep modes via cyclic timer event or CAN/MON event
- On-chip memory
  - Up to 256 KByte FLASH1 for non-volatile code and data storage with ECC
  - Up to 32 KByte FLASH0 for non-volatile code and data storage with ECC, EEPROM emulation support
  - 1024 Byte 100 Time Programmable Memory with ECC (100 TP)
  - Up to 32 KByte RAM with ECC
  - BootROM for startup firmware, bootstrap loader (BSL) and flash routines
  - Key storage for supporting security routines
- Security features
  - Secured boot mechanism as anchor for in-field software updates
  - CMAC and AES functions
  - Key storage with key management support
  - Layered access right management

1) Arm and Cortex are trademarks of ARM Limited, UK

- Communication features
  - MultiCAN+ protocol handler with CAN-FD support (up to 2 MBaud) and 32 message objects
  - CAN-FD transceiver compliant to ISO11898-2 and ISO11898-5 up to 5 MBaud
  - 2x full duplex asynchronous serial interface (UART0/1) with LIN support
  - 2x synchronous serial channel (SSC0/1) up to 30 MHz (master mode) and 15 MHz (slave mode)
- Wake-up capable high voltage monitoring input(s) (MONx) with input range of -28 V to 40 V (with series resistor)
- General-purpose I/O Ports (GPIO) with push-pull, open-drain and pull-up/down arrangement
  - TQFP-48: 8 GPIOs (incl. RESET, SWD)
  - LQFP-64: 16 GPIOs (incl. RESET, SWD)
- General-purpose input Ports (GPI) with pull-up/down arrangement
  - TQFP-48: 7 GPIs (incl. XTALI/O)
  - LQFP-64: 10 GPIs (incl. XTALI/O)
- Optimized functionality for BLDC motor control
  - 3-phase bridge driver for N-Channel MOSFETs with programmable current driven output stage, various diagnosis and protection features in on and off state. The bridge driver allows an EMC and thermally optimized switching behavior for MOSFETs of up to 6 x 150 nC at 20 kHz
  - 2-stage charge pump operating down to V_{SD} = 5.4 V allowing motor operation for wide supply range from V_S = 4.4 V to 28 V (cranking and load dump situation)
  - High speed current sense amplifier (CSA) for single shunt current measurement in ground path with programmable gain
  - Current sense comparator (CSC) with programmable threshold for fast overcurrent detection and safe switch off request
  - 12-bit ADC (ADC1) for measurement of eight high and ten middle voltage inputs with deterministic sample trigger, four time-triggered sequences and digital postprocessing
  - 3x BEMF comparators for sensorless block commutation
  - Capture/compare unit (CCU7) with five 16-bit timers for sophisticated 3-phase PWM pattern generation
- Sensor interface
  - 14-bit Sigma Delta ADC (SDADC) for rotary sensors with two differential channels supporting AMR/GMR/TMR type sensors
- General purpose timer
  - GPT12 (five 16-bit), Timer 20 (16 bit), Timer 21 (16 bit)
- Monitoring ADC
  - 10-bit ADC (ADC2) for background monitoring of five external and eight internal voltages with programmable threshold, warning flag indication, shut down and interrupt request
- Fail-safe mechanism and error handling
  - Power-on and undervoltage/brown-out reset generator
  - Supervision of all system supply voltages
  - Clock monitoring for master clock, external clock, system clock and PLL with error handling
  - Overtemperature detection sensing the junction temperature at two die locations with warning flag indication and automatic error handling
  - Drain source monitoring of bridge driver for detection of short circuit (in on/off state) and open load diagnosis (in off state)

  - All memories (flash and RAM) with single bit error correction and double bit correction (SECDET)
  - 2x window watchdog (FS_WDT and SYSWDT) with independent clock source
  - Safe switch off for bridge driver at severe system malfunction (FS_WDT overflow, shunt overcurrent, supply under-/overvoltage, failure input active) and failure indication according to ISO 26262 Safety Element out of Context for safety requirements up to ASIL-B
- Temperature range T_J: -40°C up to 175°C
- Packages TQFP-48 and LQFP-64
- Green package (RoHS compliant)
- AEC qualified (Grade 0)

2 Block diagram

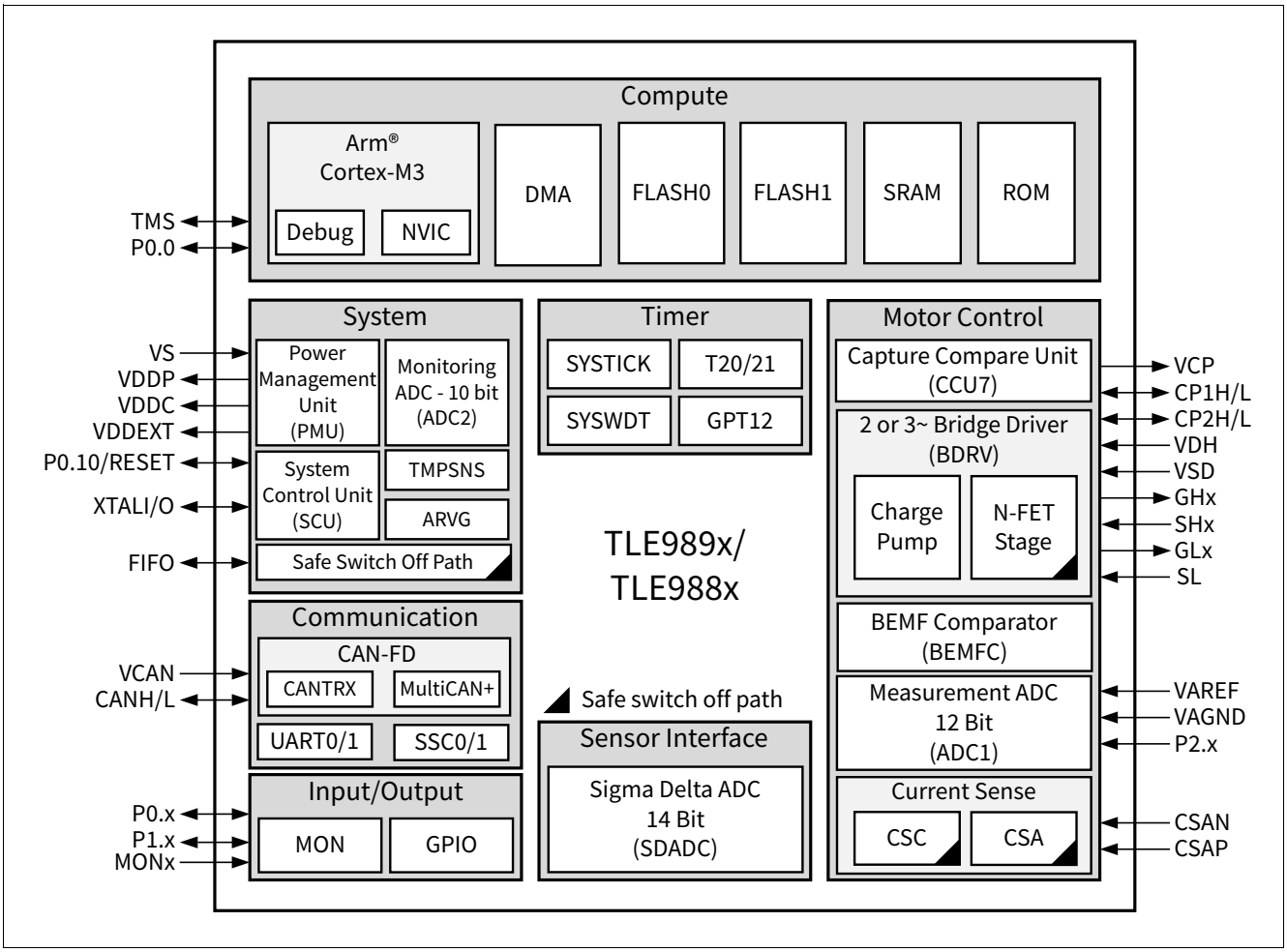


Figure 1 Block diagram TLE989x/TLE988x

3 Product definitions

3.1 Device pinout

3.1.1 Device pinout 48 pins

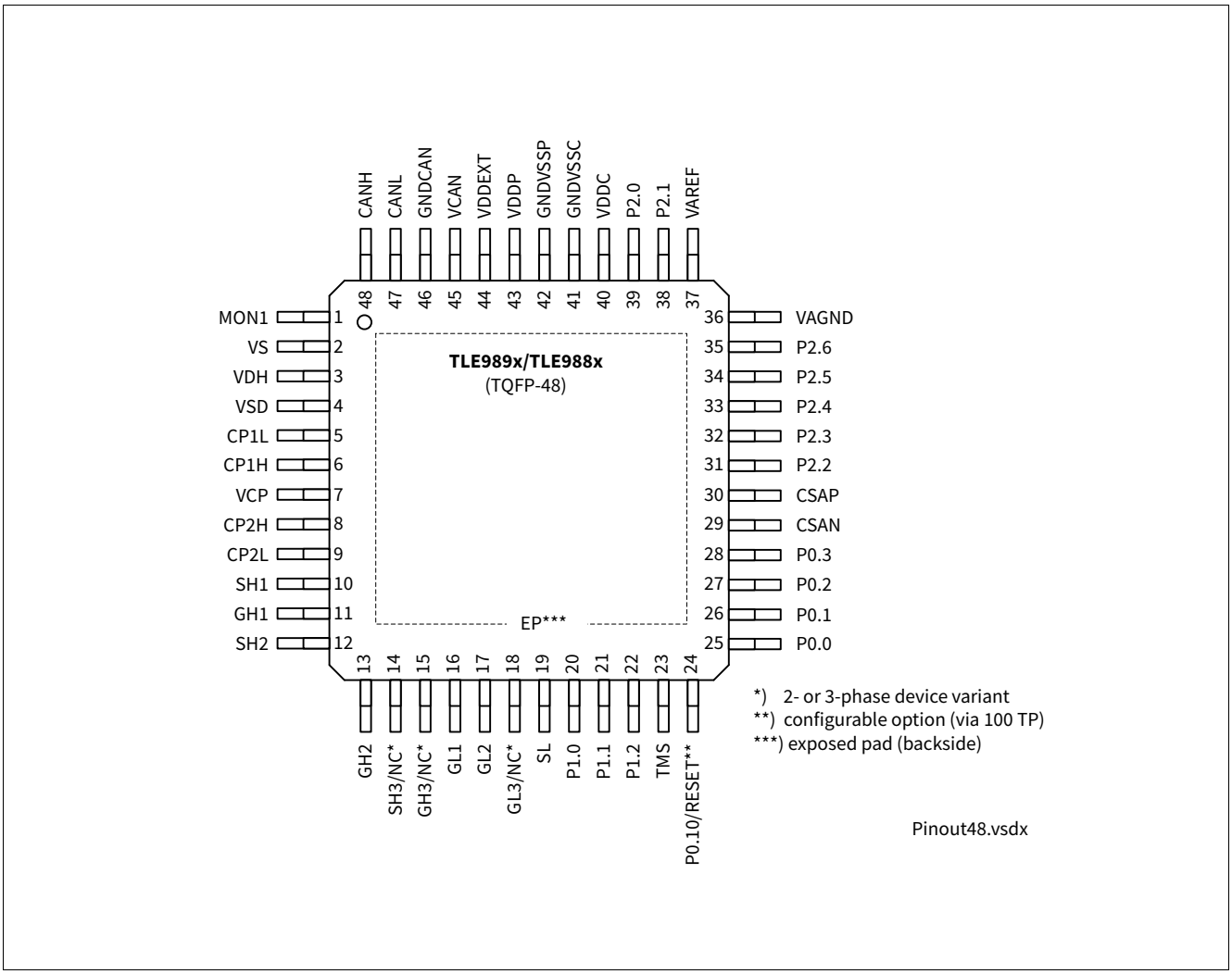


Figure 2 Pinout for 48 pin package

3.1.2 Device pinout 64 pins

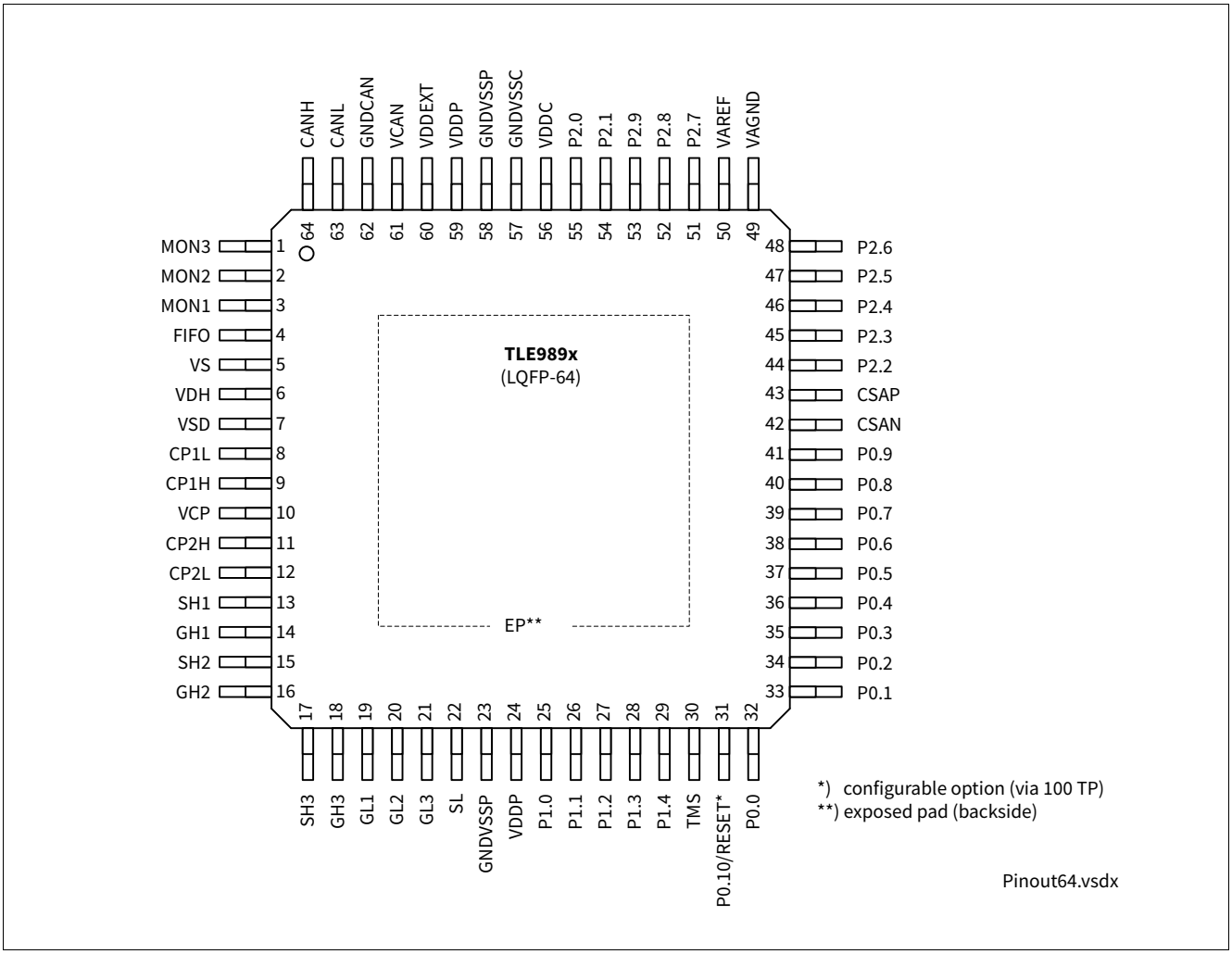


Figure 3 Pinout for 64 pin package

3.2 Device packages and ordering information

The device is offered in following package(s), see **Table 1**.

Table 1 Device packages

Name	Number of pins	Body size [mm²]	Pin pitch [mm]	Epad	Designed for automatic lead tip inspection (LTI)
TQFP-48	48	7 x 7	0.5	yes	yes
LQFP-64	64	10 x 10	0.5	yes	yes

Ordering information

This user manual covers the products with different package markings. Each marking has a separate ordering number. The different markings are described in **Table 2**. For more details please refer to the datasheet.

Table 2 Ordering info

Marking	Package
TLE989x Grade-0 (3 ph)	
TLE9893-2QKW62S	LQFP-64
TLE9893QKW62S	LQFP-64
TLE9893-2QTW62S	TQFP-48
TLE9891-2QTW61	TQFP-48
TLE9891-2QTW60	TQFP-48
TLE989x Grade-1 (3 ph)	
TLE9893-2QTA62S	TQFP-48
TLE9893-2QTA62	TQFP-48
TLE9891QTA61	TQFP-48
TLE988x Grade-0 (2 ph)	
TLE9883-2QTW62S	TQFP-48
TLE9881-2QTW60	TQFP-48
TLE988x Grade-1 (2 ph)	
TLE9883QTA62	TQFP-48

JTAG ID

The JTAG ID of the TLE989x/TLE988x is 0x1021 F083.

Customer ID

The Customer ID contains the device specific variant information. It can be read using a firmware API routine, refer to the firmware user manual. The decoding of the Customer ID is described in the following figure.

Byte 0 Grade		Byte 1 Design Step		Byte 2 Package, Variant		Byte 3 Family	
Grade 0	20_H	AA-Step	AA_H	48-pin	$X7_H$	TLE988x	06_H
Grade 1	00_H	AB-Step	AB_H	64-pin	XB_H	TLE989x	07_H
		AK-Step	BA_H	TLE98x1	$1X_H$		
				TLE98x3	$3X_H$		

Note: An 'X' within a hexadecimal value represents a "dont'care" position.

Figure 4 Customer ID decoding

3.3 Pin definitions

The functions and default states of the external pins are provided in **Table 3**.

The following pin types exist:

- I/O: Input or output
- I: Input only
- O: Output only
- P: Power supply

After reset, all pins have a defined setting. The following options are possible:

- Input with pull-up device enabled (I/PU)
- Input with pull-down device enabled (I/PD)
- Input with both pull-up and pull-down devices disabled (I/HiZ)
- Input floating to a voltage level (float)
- Input/Output with driver off (HiZ)
- Output with driver off and pull-down device enabled (PD)
- Output with driver off floating to a voltage level (float)
- Power supply (powered or GND)

Table 3 Pin definitions and functions

Symbol	Pin no.		Type	Reset state	Function	Description
	64	48				
General purpose I/Os						
P0.0	32	25	I/O	I/HiZ	GPIO	Connect to SWDCLK for debugging; Leave open if not used
P0.1	33	26	I/O	I/HiZ	GPIO	Leave open if not used
P0.2	34	27	I/O	I/HiZ	GPIO	Leave open if not used
P0.3	35	28	I/O	I/HiZ	GPIO	Leave open if not used
P0.4	36	–	I/O	I/HiZ	GPIO	Leave open if not used
P0.5	37	–	I/O	I/HiZ	GPIO	Leave open if not used
P0.6	38	–	I/O	I/HiZ	GPIO	Leave open if not used
P0.7	39	–	I/O	I/HiZ	GPIO	Leave open if not used
P0.8	40	–	I/O	I/HiZ	GPIO	Leave open if not used
P0.9	41	–	I/O	I/HiZ	GPIO	Leave open if not used
P0.10 or RESET	31	24	I/O	I/HiZ I/PU	GPIO or pin RESET	Configurable option (via 100TP); Leave open if not used
P1.0	25	20	I/O	I/HiZ	GPIO	Leave open if not used
P1.1	26	21	I/O	I/HiZ	GPIO	Leave open if not used
P1.2	27	22	I/O	I/HiZ	GPIO	Leave open if not used
P1.3	28	–	I/O	I/HiZ	GPIO	Leave open if not used
P1.4	29	–	I/O	I/HiZ	GPIO	Leave open if not used

Table 3 Pin definitions and functions (cont'd)

Symbol	Pin no.		Type	Reset state	Function	Description
	64	48				
Analog inputs						
P2.0	55	39	I	I/HiZ	GPI	Leave open if not used
P2.1	54	38	I	I/HiZ	GPI	Leave open if not used
P2.2	44	31	I	I/HiZ	GPI	Leave open if not used
P2.3	45	32	I	I/HiZ	GPI	Leave open if not used
P2.4	46	33	I	I/HiZ	GPI	Leave open if not used
P2.5	47	34	I	I/HiZ	GPI	Leave open if not used
P2.6	48	35	I	I/HiZ	GPI	Leave open if not used
P2.7	51	–	I	I/HiZ	GPI	Leave open if not used
P2.8	52	–	I	I/HiZ	GPI	Leave open if not used
P2.9	53	–	I	I/HiZ	GPI	Leave open if not used
CSAN	42	29	I	I/HiZ	CSA negative input	Connect via shunt resistor to CSAP; Connect to GNDVSSP if not used
CSAP	43	30	I	I/HiZ	CSA positive input	Connect via shunt resistor to CSAN; Connect to GNDVSSP if not used
Fail input/output						
FIFO	4	-	I/O	I/HiZ	Fail in /fail out	Connect via resistor to an external signal; Connect via 10 k pull-up to VDDP if not used
High-Voltage Monitoring inputs						
MON1	3	1	I	I/HiZ	HV monitor input 1	Connect via resistor to an external signal; Connect to GNDVSSP if not used
MON2	2	–	I	I/HiZ	HV monitor input 2	Connect via resistor to an external signal; Connect to GNDVSSP if not used
MON3	1	–	I	I/HiZ	HV monitor input 3	Connect via resistor to an external signal; Connect to GNDVSSP if not used
CAN interface						
CANH	64	48	I/O	HiZ	CAN high bus	Connect resistor to CANL; Leave open if not used
CANL	63	47	I/O	HiZ	CAN low bus	Connect resistor to CANH; Leave open if not used
VCAN	61	45	P	–	Supply input for CAN transceiver	Connect to VDDP; Connect capacitor to GNDCAN
GNDCAN	62	46	P	–	Ground for CAN transceiver	Connect to GNDVSSP; Connect capacitor to VCAN

Table 3 Pin definitions and functions (cont'd)

Symbol	Pin no.		Type	Reset state	Function	Description
	64	48				
Bridge Driver						
SL	22	19	I	GND	BDRV ground	Source low side FETs; Connect to GNDVSSP if not used
GL1	19	16	O	PD	BDRV gate low side 1	Connect to gate of low side MOSFET 1; Leave open if not used
GL2	20	17	O	PD	BDRV gate low side 2	Connect to gate of low side MOSFET 2; Leave open if not used
NC	21	18	–	–	–	2-phase device variant; Leave open
GL3	21	18	O	PD	BDRV gate low side 3	3-phase device variant; Connect to gate of low side MOSFET 3; Leave open if not used
SH1	13	10	I	Float to GND+1*Vdiode	BDRV source high 1	Connect to source of high side MOSFET 1; Leave open if not used
GH1	14	11	O	PD	BDRV gate high 1	Connect to gate of high side MOSFET 1; Leave open if not used
SH2	15	12	I	Float to GND+1*Vdiode	BDRV source high 2	Connect to source of high side MOSFET 2; Leave open if not used
GH2	16	13	O	PD	BDRV gate high 2	connect to gate of high side MOSFET 2; Leave open if not used
NC	17	14	–	–	–	2-phase device variant; Leave open
SH3	17	14	I	Float to GND+1*Vdiode	BDRV source high 3	3-phase device variant; Connect to source of high side MOSFET 3; Leave open if not used;
NC	18	15	–	–	–	2-phase device variant; Leave open
GH3	18	15	O	PD	BDRV gate high 3	3-phase device variant; Connect to gate of high side MOSFET 3; Leave open if not used;
Charge pump						
CP1L	8	5	O	HiZ	CP stage 1 out low	Connect external capacitor to CP1H; Leave open if not used
CP1H	9	6	O	Float to VSD-1*Vdiode	CP stage 1 out high	Connect external capacitor to CP1L; Leave open if not used
VCP	10	7	P	Float to VSD-1*Vdiode	charge pump output voltage	Connect via capacitor to star point of DC link high; Connect to VSD if not used
CP2L	12	9	O	HiZ	CP stage 2 out low	Connect external capacitor to CP2H; Leave open if not used

Table 3 Pin definitions and functions (cont'd)

Symbol	Pin no.		Type	Reset state	Function	Description
	64	48				
CP2H	11	8	O	Float to VSD-1*Vdiode	CP stage 2 out high	Connect external capacitor to CP2L; Leave open if not used
VSD	7	4	P	–	BDRV supply input	Connect with RC filter from star point of DC link high; Connect to VS if not used
VDH	6	3	I	I/HiZ	BRDV sense input	Connect with RC filter from star point of DC link high; Connect to SL if not used
Other pins						
TMS	30	23	I/O	I/PD	Test mode select input	Connect to SWDIO for debugging; Connect to GNDVSSP if not used
Power supply						
VS	5	2	P	–	Supply input	Connect via reverse polarity diode to VBAT; Connect capacitor to GNDVSSP
VDDP	24, 59	43	P	–	Output of VDDP regulator	Connect capacitor to GNDVSSP; Connect to VCAN
VDDC	56	40	P	–	Output of VDDC regulator	Connect capacitor to GNDVSSC
VDDEXT	60	44	P	–	Output of VDDEXT regulator	Connect capacitor to GNDVSSP; Connect to sensor supply input
GNDVSSP	23, 58	42	P	–	Ground of VDDP regulator	Connect capacitor to VDDP; Connect to module GND; Do not connect to GNDVSSC
GNDVSSC	57	41	P	–	Ground of VDDC regulator	Connect capacitor to VDDC; Do not connect to GNDVSSP
VAGND	49	36	P	–	Reference ground for mixed signal peripherals	If VREF5V is used: do not connect to GNDVSSP; If external reference is used: connect to GNDVSSP; Always connect via capacitor to VAREF
VAREF	50	37	P	–	Optional output of VREF5V regulator; Reference input for ADC1, SDADC, CSA, CSC	Connect to capacitor to VAGND; Optionally connect to VDDEXT or other reference; Leave open if not used
EP	–	–	P	–	Exposed pad	Connect to GNDVSSP

3.4 Special pin functions

3.4.1 RESET and FIFO pins

The following reset and fail-safe pins are available:

- RESET pin: P0.10 can be configured via a config sector setting (via 100TP) as bidirectional RESET function (default for P0.10 is GPIO)
- FIFO pin (only in 64 pin variant): this is a dedicated pin with bidirectional safe switch off (SSO) function

3.4.2 Programming

The device flash modules can be programmed using the following interfaces:

- Via standard Cortex SWD interface (pins TMS and P0.0, latched at start up, bootlatch) and SWD protocol
- Via bootstrap loader (BSL) interface (pins CANH and CANL) and UART protocol over CAN transceiver

Note: TMS is a dedicated pin. P0.0 is configured as SWDCLK in case TMS is latched high.

3.4.3 Debugging

The device can be debugged via standard Cortex SWD interface (pins TMS and P0.0) and SWD protocol.

Note: TMS is a dedicated pin. P0.0 is configured as SWDCLK in case TMS is latched high.

3.4.4 Clock input

- An external crystal or resonator can be connected to P2.0/XTALI and P2.1/XTALO
- An external digital clock can be connected to P2.0/XTALI

3.4.5 Analog reference

- The pins VAREF and VAGND serve as buffer for the analog reference voltage and analog reference ground for ADC1, CSA, CSC and SDADC
- A buffer capacitor (C_{VAREF}, P_ARVG_03_03) has to be placed externally

3.5 Device startup

After a device reset, the BootROM firmware is executed to initialize the device. Depending on the reset type occurred, the BootROM firmware has a different startup procedure/behavior. In general, the different reset sources can be categorized into three main reset types. Namely the COLD reset, WARM reset, and HOT reset. The associated reset sources are listed in **Table 4**.

Table 4 Reset categories and their associated reset sources

	Reset category		
	COLD	**WARM**	**HOT**
Reset source	VMSUO_UV_RST	SLEEPX_RST	LOCKUP_RST
	MCLK_WD_RST	PIN_RST	SOFT_RST
	VDDC_UV_RST	TMS_RST	WDT_MCU_RST
	VDDP_UV_RST		STOPEX_RST[1)]
	FS_SLEEP_RST		
	FSWDT_RST		
	SEC_STACK_OV_RST		

1) The STOPEX_RST does not reset the VTOR register, hence the reset vector points to the last programmed value which is usually pointing to User Code or User BSL.

The following diagram shows the complete startup flow from VS power-on to user code execution.

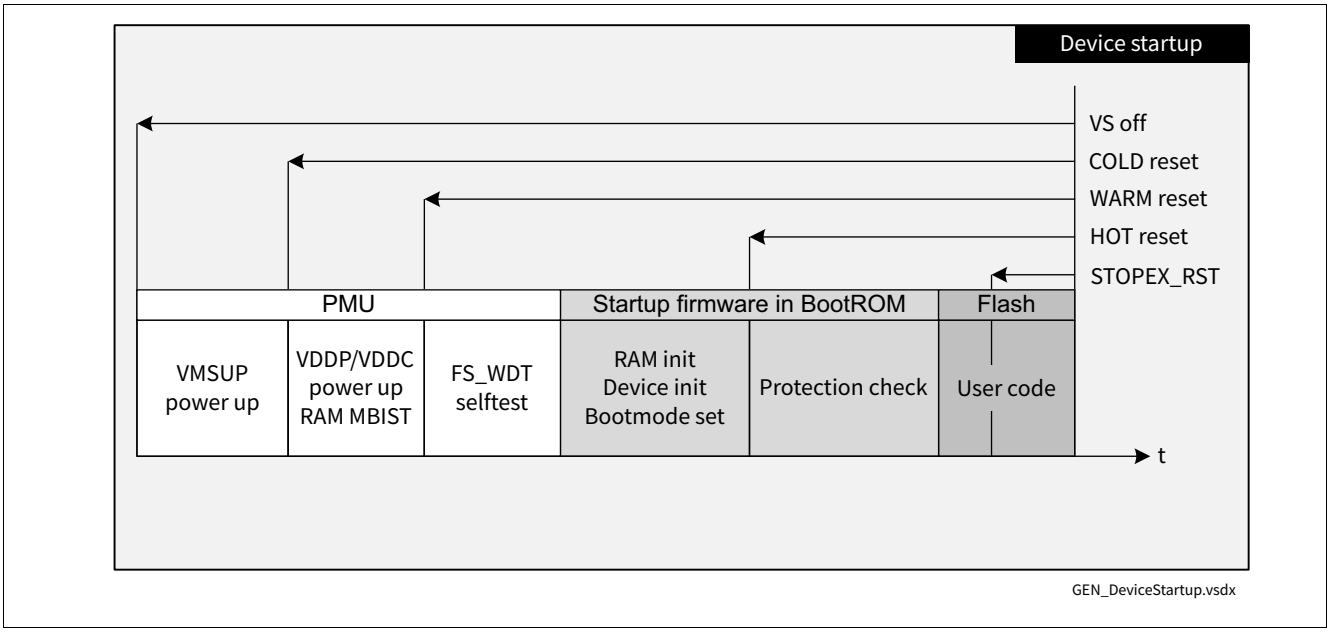


Figure 5 Device startup flow

The startup procedure for all reset sources of type COLD and WARM are depicted in **Figure 6**. For all other reset sources, the HOT reset applies as illustrated in **Figure 7**.

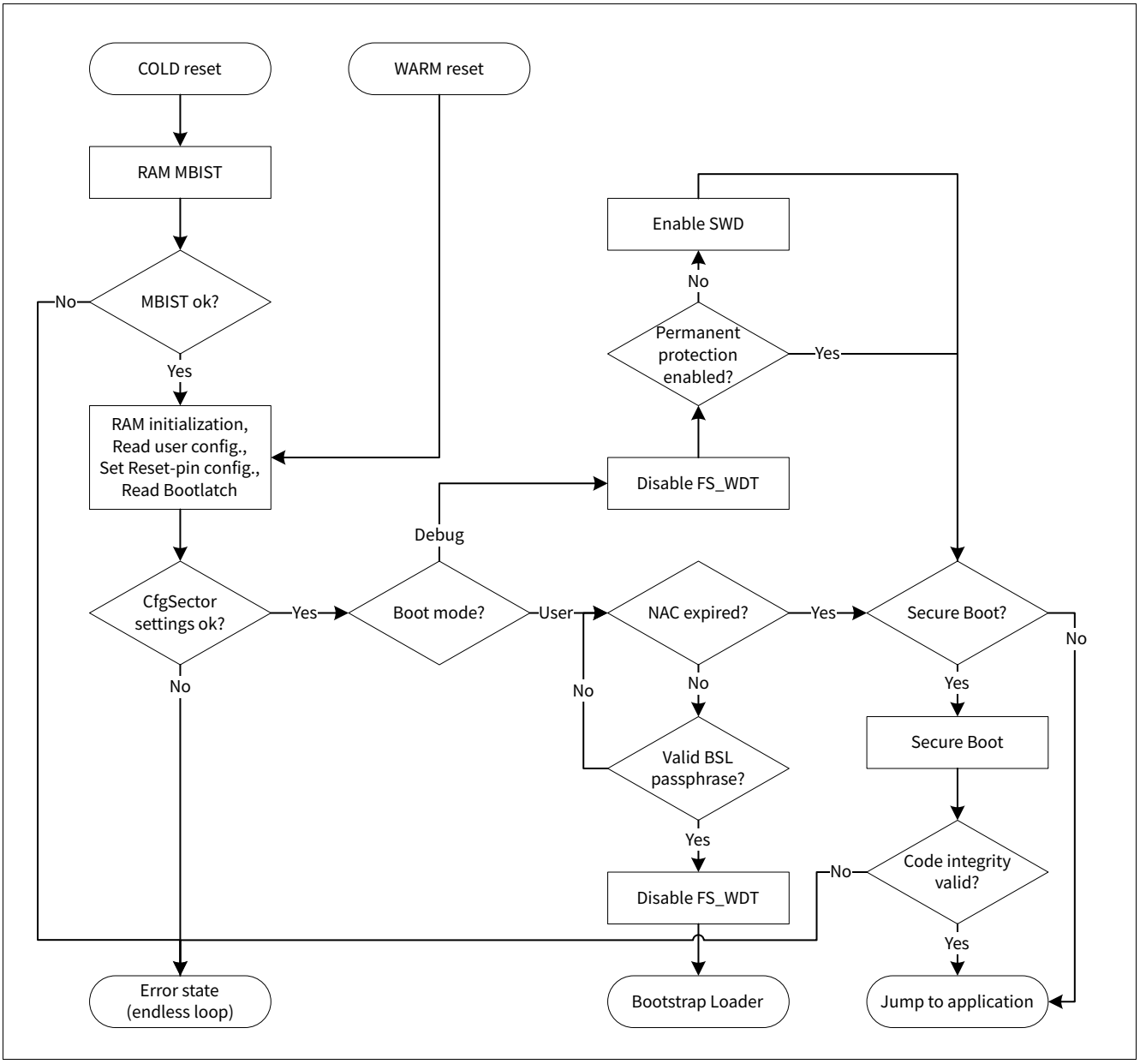


Figure 6 Device bootup flow after COLD and WARM reset

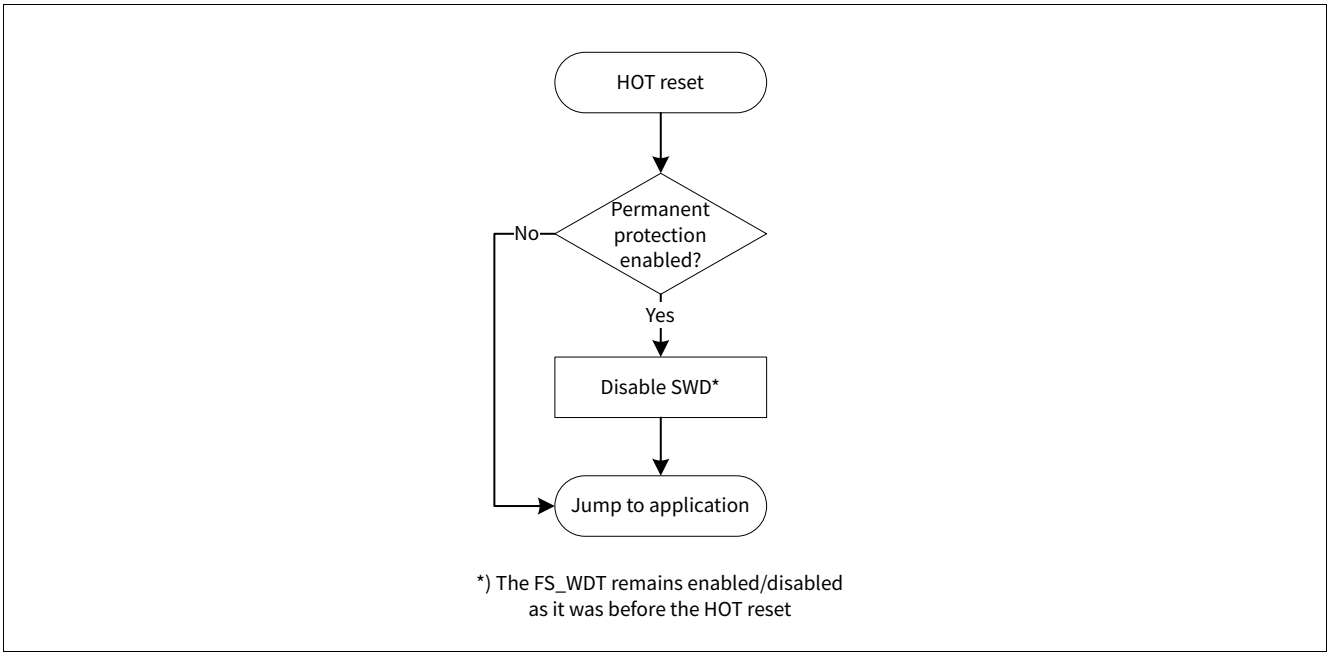


Figure 7 Device bootup flow after HOT reset

The execution of the BootROM firmware has a certain execution time until execution is handed over to the user application. This execution time depends on the reset category and other bootup configurations. A timing diagram for resets of the category COLD and WARM is depicted in **Figure 8**. The execution time after a reset of category HOT is negligible.

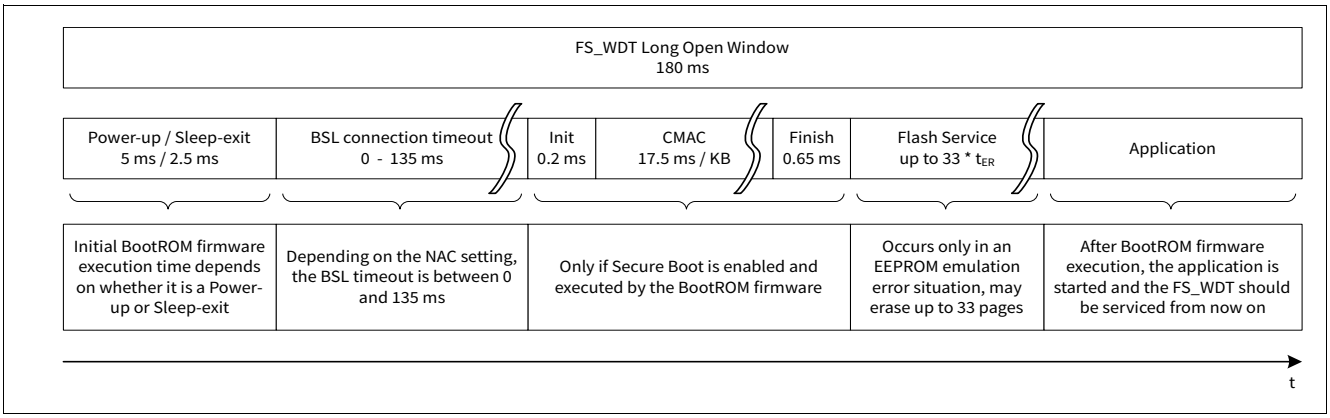


Figure 8 Device bootup timing

It is important to note that the FS_WDT is starting with a Long Open Window before the BootROM firmware is started. Hence, the execution time needed by the BootROM firmware is already consuming a portion of time of the Long Open Window. Depending on the execution time, it can be necessary that the FS_WDT needs to be serviced immediately after BootROM firmware execution. This is also limiting the code size that can be secured by Secure Boot.

Note: The Power-up, Sleep-exit, and FS_WDT Long Open Window timings refer to typical values of MCLK. The timings for the BSL connection timeout and Secure Boot refer to typical values of HP_CLK.

3.6 Brown-out

The integrated VDDP regulator will enter dropout operation as the VS pin voltage is dropping below the minimum supply voltage (P_GEN_09_01). As a consequence the regulator will enter dropout and can no longer maintain its output voltage within the regulation limits.

The MCU subsystem remains fully functional down to the minimum extended supply voltage range (P_GEN_09_03 and P_GEN_09_14).

Care should be taken while operating following peripherals under low-supply conditions:

- Derated electrical performance for VDDEXT, VREF5V (VAREF), CSA, CSC, SDADC, MON, BDRV
- Derated ADC1 electrical performance (relating to a drift on the VREF5V (VAREF) reference)
- CAN transceiver interface

Figure 9 illustrates the operation under low-supply (brown-out) conditions:

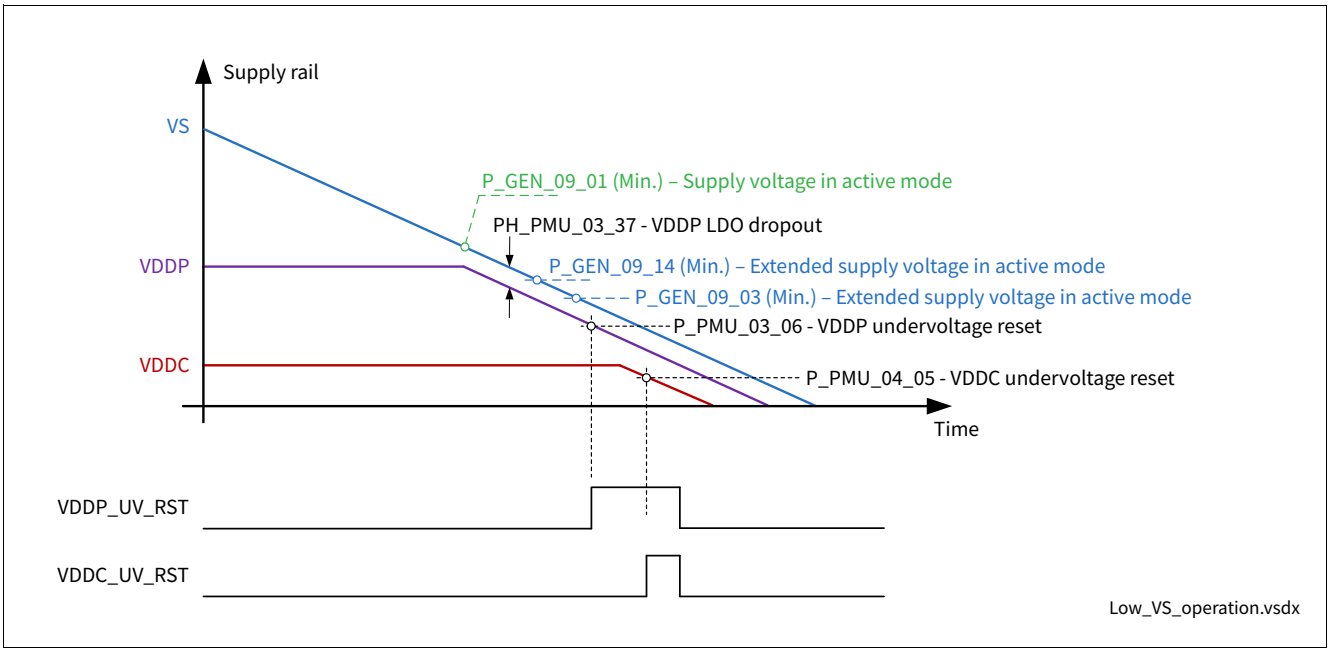


Figure 9 Operation under low-supply (brown-out) conditions

3.7 Interrupt request mapping

Modules which can generate an interrupt request are either directly connected to the NVIC or via SCU. The interrupt mapping can be found in **Table 5**. The detection, selection, enabling and handling of the requests is module specific and is described within the corresponding module chapter. The node pointer scheme is described within the SCU. The interrupt processing is described within the NVIC chapter.

Table 5 Assignment of interrupt requests to NVIC IRQ inputs

Module	Module IRQ event status	Number of module IRQ events	Number of module IRQs	Node pointer location	IRQ name	NVIC IRQ input
T21	IRQ via ored events EXF21, TF21	2	1	T21	T21_IRQ	31
DMA	DMACHx, DMATRERR (x = 0-7)	9	9	SCU	INP7_IRQ1	30
					INP7_IRQ0	29
MultiCAN	CAN_IR[2:0] via CAN_NIPR0 and CAN_MOIPRn TXOK, RXOK, EWRN, BOFF, LEC[6], LLE, LOE, CVCOV, TXRQ[32], RXRQ[32] MITR0/1/2	80	3	MultiCAN	INT_O2	28
					INT_O1	27
					INT_O0	26
SSC1	EIR (TEIR, REIR, PEIR, BEIR), TIR, RIR	6	3	SCU	INP6_IRQ1	25
SSC0	EIR (TEIR, REIR, PEIR, BEIR), TIR, RIR	6	3		INP6_IRQ0	24
UART1	RI, TI, LIN1EOFSYN, LIN1ERRSYN	4	4	SCU	INP5_IRQ1	23
UART0	RI, TI, LIN0EOFSYN, LIN0ERRSYN	4	4		INP5_IRQ0	22
SCU	EXTINTxR, EXTINTxF (x = 0, 1, 2, 3)	8	8	SCU	INP4_IRQ1	21
					INP4_IRQ0	20
BEMFC	IRQ via ored events PHx_ZCFALL_IS, PHx_ZCRISE_IS (x = 1, 2, 3)	6	3	SCU	INP3_IRQ1	19
SDADC	IRQ[1:0] via SDADC.INP RESx_IS, CMPx_UP_IS, CMPx_LO_IS (x = 0, 1)	6	2		INP3_IRQ0	18
ADC1	IRQ[3:0] via ADC1.INP[3:0]		4	ADC1	INP_IRQ3	17
	CH[19:0],	20			INP_IRQ2	16
	CMP_UP_[3:0], CMP_LO_[3:0],	8			INP_IRQ1	15
	SQ[3:0], WFR[3:0], COLL[3:0]	12			INP_IRQ0	14

Table 5 Assignment of interrupt requests to NVIC IRQ inputs

Module	Module IRQ event status	Number of module IRQ events	Number of module IRQs	Node pointer location	IRQ name	NVIC IRQ input
MON	IRQ via ored events MONxR, MONxF (x = 1, 2, 3)	6	3	SCU	INP2_IRQ1	13
					INP2_IRQ0	12
ADC2	IRQ[1:0] via ADC2.INP[1:0]		2	ADC2	INP_IRQ1	11
	CMP_UP_[7:0], CMP_LO_[7:0],	16				
	SQ[3:0]	4			INP_IRQ0	10
T20	IRQ via ored events EXF2, TF2	2	1	T20	T20_IRQ	9
GPT12	GPT1T2, GPT1T3, GPT1T4, GPT2T5, GPT2T6, GPT2CR	6	6	SCU	INP1_IRQ1	8
					INP1_IRQ0	7
MCU/ MEMCTRL	IRQ via ored events NVM0OPC, NVM1OPC	2	1	MEMCTRL	MEM_IRQ	6
CCU7	SR[3:0] via CCU7.INP[3:0] STR, WHE, CHE, TRPF, T12PM, T12OM, ICC72F, ICC72R, ICC71F, ICC71R, ICC70F, ICC70R, ICC72BF, ICC72BR, ICC71BF, ICC71BR, ICC70BF, ICC70BR, T13PM, T13CM, T14PM, T14CM, T15PM, T15CM, T16PM, T16CM, LBE	27	4	CCU7	INP_IRQ3	5
					INP_IRQ2	4
					INP_IRQ1	3
					INP_IRQ0	2

Table 5 Assignment of interrupt requests to NVIC IRQ inputs

Module	Module IRQ event status	Number of module IRQ events	Number of module IRQs	Node pointer location	IRQ name	NVIC IRQ input
CANTRX	IRQ via ored events BUS_TO_IS, TXD_TO_IS, OT_IS, BUS_ACT_IS	4	6	SCU	INP0_IRQ1	1
BDRV	IRQ0 via ored events HBx_ACTDRV_IS IRQ1 via ored events LS/HSx_OC_IS, LS/HSx_DS_IS, HBx_ASEQ_IS, SEQ_ERR_IS, VCP_LOTH2_IS (x = 1, 2, 3)	20				
CSC	IRQ via ored events CSC_OC_IS, SEL_ERR_IS	2			INP0_IRQ0	0
PMU	IRQ via ored events VDDP: UVWARN_IS, OV_IS VDDC: UVWARN_IS, OV_IS VDDEXT: UV_IS, OT_IS	6				
ARVG	OC_IS	1				

3.8 Exception request mapping

Modules which can generate an exception request are either directly connected to the NVIC or via SCU. The exception mapping can be found in **Table 6**. The detection, selection, enabling and handling of the requests is module specific and is described within the corresponding module chapter. The node pointer scheme is described within the SCU. The exception processing is described within the NVIC chapter.

Table 6 Assignment of exception requests

Module	Event name (in module)	Number of events		Direct or via SCU	Request signal connected to NVIC	IRQ number
SYSTICK	TICKINT	1		Direct	EXCEPT	-1
MCU	PendSV	1		Direct	EXCEPT	-2
–	–	–		–	–	-3
MCU	Debug	1		Direct	EXCEPT	-4
MCU	SVCall	1		Direct	EXCEPT	-5
–	–	–		–	–	-6
–	–	–		–	–	-7
–	–	–		–	–	-8
–	–	–		–	–	-9
MCU	Usage fault	1		Direct	–	-10
MCU	Bus fault	1		Direct	–	-11
MCU	MemManage fault	1		Direct	–	-12
MCU	Hard fault	1		Direct	–	-13
SCU	NMIRQ via ored events (NMIXTAL, NMIPLL0, NMIPLL1)	3	1	SCU	NMI	-14
MCU/ MEMCTRL	NMIRQ via ored events (NMIDS, NMIPS, NMICD, NMINVM0, NMINVM1, NMIMAP0, NMIMAP1, NMIWDT, NMISTOF)	9	1			
ADC2	IRQ3 via ADC2.INP3 CMPUP0, CMPLO0, CMPLO1, CMPUP2, CMPLO2, CMPLO3, CMPUP4	7	1			

3.9 DMA request mapping

Modules which can generate a DMA request are connected via SCU to the DMA. The assignment of DMA request events from modules to the DMA controller can be found in found in **Table 7**.

Table 7 DMA mapping

DMA mapping	Event name	DMA_ SRQ [0]	DMA_ SRQ [1]	DMA_ SRQ [2]	DMA_ SRQ [3]	DMA_ SRQ [4]	DMA_ SRQ [5]	DMA_ SRQ [6]	DMA_ SRQ [7]
CCU7	T12 CM_70	x	–	x	–	x	–	–	–
	T12 CM_71	–	x	–	x	–	x	–	–
	T12 CM_72	x	–	x	–	x	–	–	–
	T12_PM	–	x	–	x	–	x	–	–
	T12_ZM	x	–	x	–	x	–	–	–
	T13 CM_73	–	x	–	x	–	x	–	–
	T13_PM	x	–	x	–	x	–	–	–
	T13_ZM	–	x	–	x	–	x	–	–
	T14 CM_74	x	–	x	–	x	–	–	–
	T14_PM	–	x	–	x	–	x	–	–
	T15 CM_75	x	–	x	–	x	–	–	–
	T15_PM	–	x	–	x	–	x	–	–
	T16 CM_76	x	–	x	–	x	–	–	–
	T16_PM	–	x	–	x	–	x	–	–
	CHE	x	–	x	–	x	–	–	–
ADC1	CH0	–	–	x	–	x	–	x	–
	CH1	–	–	–	x	–	x	–	x
	CH2	–	–	x	–	x	–	x	–
	CH3	–	–	–	x	–	x	–	x
	CH4	–	–	x	–	x	–	x	–
	CH5	–	–	–	x	–	x	–	x
	CH6	–	–	x	–	x	–	x	–
	CH7	–	–	–	x	–	x	–	x
	SQ0	–	–	x	–	x	–	x	–
	SQ1	–	–	–	x	–	x	–	x
	CMPLO0	–	–	x	–	x	–	x	–
	CMPUP0	–	–	–	x	–	x	–	x
SDADC	RES0	–	–	x	–	x	–	x	–
	RES1	–	–	–	x	–	x	–	x
Timer20	T20F	–	–	–	–	–	–	x	–
Timer21	T21F	–	–	–	–	–	–	–	x

Table 7 DMA mapping (cont'd)

DMA mapping	Event name	DMA_ SRQ [0]	DMA_ SRQ [1]	DMA_ SRQ [2]	DMA_ SRQ [3]	DMA_ SRQ [4]	DMA_ SRQ [5]	DMA_ SRQ [6]	DMA_ SRQ [7]
GPT12	GPT12T2	x	–	–	–	–	–	–	–
	GPT12T3	–	x	–	–	–	–	–	–
	GPT12T4	–	–	x	–	–	–	–	–
	GPT12T5	–	–	–	x	–	–	–	–
	GPT12T6	–	–	–	–	x	–	–	–
	GPT12CR	–	–	–	–	–	x	–	–
SSC0	TIR0	x	–	x	–	–	–	–	–
	RIR0	–	x	–	x	–	–	–	–
SSC1	TIR1	–	–	–	–	x	–	x	–
	RIR1	–	–	–	–	–	x	–	x
UART0	RI	x	–	x	–	–	–	–	–
	TI	–	x	–	x	–	–	–	–
UART1	RI	–	–	–	–	x	–	x	–
	TI	–	–	–	–	–	x	–	x
MultiCAN	CAN_IR0	x	–	–	–	–	–	–	–
	CAN_IR1	–	x	–	–	–	–	–	–

3.10 Interconnects between modules

Some modules exchange events and signals directly with each other. The module interconnects can be taken from following chapters **Chapter 3.10.2** to **Chapter 3.10.21**.

3.10.1 Alternate functions at GPIOs

GPIOs have following functions:

- Output function written to the GPIOs output port register (Out_0)
- Input function read from the GPIOs input port register (Inp_0)
- Alternate output function from a peripheral via the GPIOs ALTSEL register (Out_1 to Out_7)
- Alternate input function to a peripheral via the peripheral's INSEL register (Inp_0 to Inp_7)

The alternate output and input functions assignment can be taken from **Figure 10**.

GPIO	48er	64er	Out_0	Out_1	Out_2	Out_3	Out_4	Out_5	Out_6	Out_7
P0.0	25	32	P0.0	CCU7.CC72	T21.EXF2	UART0.RXDO				
P0.1	26	33	P0.1	UART0.TXD	GPT12.T6OUT	MultiCAN.TXDC	CANTRX.RXD	SSC0.MTSR	BEMFC.PH1_ZC_STS	CCU7.CC72
P0.2	27	34	P0.2	UART1.TXD	GPT12.T3OUT	SSC0.MRST	SSC1.CS0	BEMFC.PH2_ZC_STS	CCU7.COUT73	
P0.3	28	35	P0.3	SSC0.SCLK	BEMFC.PH3_ZC_STS	CCU7.COUT70	SCU.CLKOUT	ADC1.CMPLO3	UART0.TXD	SDADC.CMP1
P0.4	-	36	P0.4	CCU7.CC70	SDADC.DOUT0	SSC0.MTSR	UART1.TXD	GPT12.T3OUT		
P0.5	-	37	P0.5	CCU7.COUT70	SDADC.DOUT0	SSC0.SCLK	SDADC.CMP1			
P0.6	-	38	P0.6	CCU7.CC71	SDADC.DOUT1	SSC0.MRST	SSC1.CS3	SDADC.CMP0	ADC1.CMPUP3	SSC0.MTSR
P0.7	-	39	P0.7	CCU7.COUT71	SDADC.DOUT1	SSC1.SCLK	MultiCAN.TXDC	UART0.TXD	GPT12.T6OUT	ADC2.CMPUP7
P0.8	-	40	P0.8	CCU7.COUT72	SDADC.MCLK	SSC1.MTSR	ADC1.CMPLO1			
P0.9	-	41	P0.9	CCU7.CC72	SDADC.MCLK	SSC1.MRST	SSC0.CS3	ADC1.CMPLO0	SSC1.MTSR	
P0.10	24	31	P0.10	PMU.RESET	SSC1.CS1	SSC0.CS1	ADC1.CMPLO2	T20.EXF2	CCU7.COUT72	ADC2.CMPLO7
P1.0	20	25	P1.0	SSC1.SCLK	CCU7.CC71	GPT12.T6OUT	UART1.RXDO	ADC1.CMPUP2	SDADC.CMP0	ADC2.CMPLO6
P1.1	21	26	P1.1	SSC1.MTSR	CCU7.COUT71	UART1.TXD	GPT12.T3OUT	CANTRX.RXD	ADC1.CMPUP1	ADC2.CMPUP6
P1.2	22	27	P1.2	SSC1.MRST	CCU7.CC70	SSC0.CS0	SCU.CLKOUT	BEMFC.PH3_ZC_STS	ADC1.CMPUP0	SSC1.MTSR
P1.3	-	28	P1.3	SSC0.CS2	CCU7.COUT73	MultiCAN.TXDC	UART0.TXD	GPT12.T6OUT	BEMFC.PH1_ZC_STS	
P1.4	-	29	P1.4	SSC1.CS2	SCU.CLKOUT	BEMFC.PH2_ZC_STS	T21.EXF2			
P2.0	39	55								
P2.1	38	54		SCU.XTALO						
P2.2	31	44								
P2.3	32	45								
P2.4	33	46								
P2.5	34	47								
P2.6	35	48								
P2.7	-	51								
P2.8	-	52								
P2.9	-	53								

Legend
CCU7
GPT12
T20
T21
UART0
UART1
SSC0
SSC1
SCU/PMU
LIN
SDADC
MultiCAN
BEMFC
CANTRX
ADC2
ADC1
GPIO
MCU
BDRV

GPIO	48er	64er	Inp_0	Inp_1	Inp_2	Inp_3	Inp_4	Inp_5	Inp_6	Inp_7	Inp_8
P0.0	25	32	P0.0	MCU.SWDCLK	UART1.RXDA	T21.T2EXA	CCU7.T12HRA	SCU.EXINT2D	T20.T2A	GPT12.T6EUDA	
P0.1	26	33	P0.1	SSC0.MTSRA	CCU7.CC72INA	GPT12.T2INA	SSC1.STARTA	UART1.STARTA	SCU.EXINT3A	SSC0.MRSTD	
P0.2	27	34	P0.2	SSC0.MRSTA	CCU7.T13HRA	SCU.EXINT1A	GPT12.T2EUDA	SSC1.SLSA	BDRV.INC6	BEMFC.INC6	
P0.3	28	35	P0.3	UART0.RXDA	MultiCAN.RXDCD	CANTRX.TXDD	T20.T2EXA	SSC0.SCLKA	SCU.EXINT0C	T21.T2A	
P0.4	-	36	P0.4	CCU7.T14HRA	SSC0.MTSRB	SDADC.DIN0A	GPT12.T6INA	CCU7.CC70INC	UART0.STARTA	SCU.EXINT3B	
P0.5	-	37	P0.5	CCU7.T15HRA	SSC0.SCLKB	SDADC.DIN1A		SCU.EXINT2B	GPT12.T2INB		
P0.6	-	38	P0.6	CCU7.T16HRA	SSC0.MRSTB	SCU.EXINT1B	GPT12.T5INA	SSC1.SLSD	CCU7.CC71INC		
P0.7	-	39	P0.7	CCU7.T14HRB	SSC1.SCLKB	GPT12.T5EUDA	GPT12.T6EUDB	SCU.EXINT3C			
P0.8	-	40	P0.8	CCU7.T15HRB	SSC1.MTSRB	MultiCAN.RXDCB	UART0.RXDB	T20.T2EXB	SCU.EXINT0B	T21.T2B	
P0.9	-	41	P0.9	CCU7.T16HRB	SSC1.MRSTB	SCU.EXINT0D	GPT12.T6INB	SSC0.SLSD	CCU7.CC72INC		
P0.10	24	31	P0.10	PMU.RESET							
P1.0	20	25	P1.0	SSC1.SCLKA	CCU7.CCPOS2A	GPT12.T3INA	CCU7.CC71INA	SSC0.SLSA	SCU.EXINT3D		
P1.1	21	26	P1.1	SSC1.MTSRA	CCU7.CCPOS1A	GPT12.T3EUDA	CCU7.CTRAPA	SCU.EXINT1C	GPT12.CAPINA	SSC0.STARTA	
P1.2	22	27	P1.2	SSC1.MRSTA	CCU7.CCPOS0A	GPT12.T4INA	UART1.RXDB	T21.T2EXB	CCU7.CC70INA	SCU.EXINT2C	
P1.3	-	28	P1.3	SDADC.DIN0B		GPT12.T2EUDB	SSC0.SLSC				
P1.4	-	29	P1.4	SDADC.DIN1B	MultiCAN.RXDCC	UART0.RXDC	SCU.EXINT0A	T21.T2C	SSC1.SLSC	T20.T2EXC	
P2.0	39	55	P2.0	ADC1.AN19	SCU.XTALI	CCU7.T12HRB	GPT12.T4EUDA				
P2.1	38	54	P2.1	ADC1.AN20	CCU7.T13HRB	GPT12.T4EUDB					
P2.2	31	44	P2.2	ADC1.AN26	SDADC.IN0NA	CCU7.CCPOS2B	GPT12.T3INB	CCU7.CC70INB	BDRV.INC1	BEMFC.INC1	
P2.3	32	45	P2.3	ADC1.AN22	SDADC.IN1NA	CCU7.CCPOS0B	BDRV.INC2	CCU7.CC71INB	SCU.EXINT1D	BEMFC.INC2	
P2.4	33	46	P2.4	ADC1.AN23	SDADC.IN0PA	CCU7.CCPOS1B	GPT12.T3EUDB	CCU7.CC72INB	SSC1.SLSB	BDRV.INC3	BEMFC.INC3
P2.5	34	47	P2.5	ADC1.AN24	SDADC.IN1PA	SSC0.MRSTC	BDRV.INC4	BEMFC.INC4			
P2.6	35	48	P2.6	ADC1.AN25	SDADC.IN0NB	SSC1.MRSTC	CCU7.CTRAPB	SSC0.SLSB	BDRV.INC5	BEMFC.INC5	
P2.7	-	51	P2.7	ADC1.AN21	SDADC.IN1NB	CCU7.CCPOS0C					
P2.8	-	52	P2.8	SDADC.IN0PB	CCU7.CCPOS1C	SSC1.MRSTD	UART1.RXDC	T21.T2EXC	T20.T2B		
P2.9	-	53	P2.9	SDADC.IN1PB	CCU7.CCPOS2C		SCU.EXINT2A	T21.T2D			

Figure 10 GPIO alternate functions overview

3.10.2 PMU interconnections

Table 8 PMU interconnections

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
Reset					
PMU	RESET	<-i/o->	GPIO	P0_10	Pin RESET
Clock					
PMU	MCLK	out -->	SCU	MCLK	Independent master clock from PMU
PMU	MCLK	out -->	CSA	CLKIN	Independent master clock from PMU
PMU	MCLK	out -->	CSC	CLKIN	Independent master clock from PMU
PMU	MCLK	out -->	CANTRX	MCLK	Silence time reference clock from PMU
PMU	REF_CLK	out -->	SCU	REF_CLK	Independent reference clock from PMU
Fail-safe					
PMU	FIFO	<-i/o->	pin	FIFO	Fail I / Fail Out pin
PMU	SSONOUT	out -->	BDRV	SAFE_ENABLE	Safe enable of BDRV
PMU	FASTDIS	out -->	BDRV	SSO_HCDIS	Safe shutdown pre-discharge
PMU	SYS_OT	<-- in	ADC2	CMPUP5	Fail sleep request
PMU	HP_CLK_EN	out -->	SCU	HP_CLK_EN	Enable for HPOSC
PMU	HP_CLK_FAIL	<-- in	SCU	HP_CLK_FAIL	Fail sleep request
PMU	CSC_BIST_FAIL	<-- in	CSC	CSC_BIST_FAIL	Result of built-in self-test
PMU	CSC_EN	<-- in	CSC	CSC_EN	CSC comparator enable signal after self-test
PMU	CSC_OC	<-- in	CSC	CSC_OC	CSC monitor
PMU	VGEN_ADC	out -->	ADC2	AN12	Central PMU bandgap reference voltage measured at ADC2
Wake					
PMU	WAKEVSDOV	<-- in	BDRV	VSD_OV	VSD wake event
PMU	WAKECAN	<-- in	CANTRX	WAKECAN	CAN wake event
PMU	WAKEMON3	<-- in	MON	MON3O	MON3 wake event
PMU	WAKEMON2	<-- in	MON	MON2O	MON2 wake event
PMU	WAKEMON1	<-- in	MON	MON1O	MON1 wake event
PMU	P0_x	<-- in	GPIO	P0_x	GPIO P0.x wake event
PMU	P1_x	<-- in	GPIO	P1_x	GPIO P1.x wake event
PMU	P2_x	<-- in	GPIO	P2_x	GPIO P2.x wake event

3.10.3 SCU interconnections

Table 9 SCU clock connections

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SCU	HP_CLK_EN	<-- in	PMU	HP_CLK_EN	Enable for HPOSC
SCU	MCLK	<-- in	PMU	MCLK	Independent clock from PMU to SCU
SCU	REF_CLK	<-- in	PMU	REF_CLK	Independent clock from PMU to SCU
SCU	CPU_CLK	out -->	MCU	CPU_CLK	MCU subsystem and AHB bus clock
SCU	PER_CLK	out -->	GPIO	PER_CLK	Peripheral clock PER_CLK
SCU	SYS0_CLK	out -->	ADC1	CLKIN	Input clock for ADC1
SCU	SYS0_CLK	out -->	ADC2	CLKIN	Input clock for ADC2
SCU	SYS0_CLK	out -->	CCU7	CLKIN	Input clock for CCU7
SCU	SYS0_CLK	out -->	SDADC	CLKIN	Input clock for SDADC
SCU	SYS0_CLK	out -->	SSC0	SSC_CLK	Peripheral clock SYS0_CLK
SCU	SYS0_CLK	out -->	SSC1	SSC_CLK	Peripheral clock SYS0_CLK
SCU	SYS0_CLK	out -->	T20	T2_CLK	Peripheral clock SYS0_CLK
SCU	SYS0_CLK	out -->	T21	T2_CLK	Peripheral clock SYS0_CLK
SCU	SYS0_CLK	out -->	GPT12	GPT_CLK	Peripheral clock SYS0_CLK
SCU	MI_CLK	out -->	BDRV	MI_CLK	Peripheral clock MI_CLK
SCU	TFILT_BDRV_CLK	out -->	BDRV	CLKIN	Filter input clock for MultiCAN
SCU	MI_CLK	out -->	BEMFC	MI_CLK	Peripheral clock MI_CLK
SCU	TFILT_BDRV_CLK	out -->	BEMFC	CLKIN	Peripheral clock TFILT_BDRV_CLK
SCU	UART_CLK	out -->	UART0	UART_CLK	Peripheral clock UART_CLK
SCU	UART_CLK	out -->	UART1	UART_CLK	Peripheral clock UART_CLK
SCU	PER_CLK	out -->	MultiCAN	SYN_CLK	Synchronous input clock for MultiCAN
SCU	CAN_CLK	out -->	MultiCAN	ASYN_CLK	Asynchronous input clock for MultiCAN
SCU	PER_CLK	out -->	CANTRX	CLKIN	Peripheral clock PER_CLK
SCU	TFILT_CLK	out -->	CANTRX	TFILT_CLK	Peripheral clock TFILT_CLK

Table 10 SCU interrupt to peripheral

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SCU	INP1_IRQ1	out -->	UART0	STARTB	Node pointer 1 interrupt request 1
SCU	INP1_IRQ1	out -->	UART1	STARTB	Node pointer 1 interrupt request 1
SCU	INP1_IRQ0	out -->	ADC1	TRGSQ1G	Node pointer 1 interrupt request 1
SCU	INP1_IRQ1	out -->	ADC1	TRGSQ2G	Node pointer 1 interrupt request 1
SCU	INP1E	<-- in	GPT12	T6IRQ	Node pointer 1 input E
SCU	INP2_IRQ1	out -->	SSC1	STARTD	Node pointer 2 interrupt request 1 to trigger SSC1

Table 10 SCU interrupt to peripheral (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SCU	INP2_IRQ1	out -->	UART1	STARTD	Node pointer 2 interrupt request 1 to trigger UART1
SCU	INP1_IRQ1	out -->	SSC0	STARTB	Node pointer 1 interrupt request 1
SCU	INP1_IRQ1	out -->	SSC1	STARTB	Node pointer 1 interrupt request 1

Table 11 SCU fail-safe

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SCU	HP_CLK_FAIL	out -->	PMU	HP_CLK_FAIL	Fail sleep request
SCU	SYS_OT_WARN	<-- in	ADC2	CMPLO5	Fail sleep request
SCU	BDRV_SD	out -->	BDRV	BDRV_SD	Shutdown request from SCU to BDRV

Table 12 SCU alternate functions

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SCU	XTALI	<-- in	GPIO	P2_0	External oscillator input
SCU	XTALO	out -->	GPIO	P2_1	External oscillator output
SCU	CLKOUT	out -->	GPIO	P1_2	Clockout function
SCU	CLKOUT	out -->	GPIO	P0_3	Clockout function
SCU	CLKOUT	out -->	GPIO	P1_4	Clockout function

Table 13 SCU external interrupts

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SCU	EXINT3A	<-- in	GPIO	P0_1	GPIO interrupt request 3
SCU	EXINT3B	<-- in	GPIO	P0_4	GPIO interrupt request 3
SCU	EXINT3C	<-- in	GPIO	P0_7	GPIO interrupt request 3
SCU	EXINT3D	<-- in	GPIO	P1_0	GPIO interrupt request 3
SCU	EXINT2A	<-- in	GPIO	P2_9	GPIO interrupt request 2
SCU	EXINT2B	<-- in	GPIO	P0_5	GPIO interrupt request 2
SCU	EXINT2C	<-- in	GPIO	P1_2	GPIO interrupt request 2
SCU	EXINT2D	<-- in	GPIO	P0_0	GPIO interrupt request 2
SCU	EXINT1A	<-- in	GPIO	P0_2	GPIO interrupt request 1
SCU	EXINT1B	<-- in	GPIO	P0_6	GPIO interrupt request 1
SCU	EXINT1C	<-- in	GPIO	P1_1	GPIO interrupt request 1
SCU	EXINT1D	<-- in	GPIO	P2_3	GPIO interrupt request 1
SCU	EXINT0A	<-- in	GPIO	P1_4	GPIO interrupt request 0
SCU	EXINT0B	<-- in	GPIO	P0_8	GPIO interrupt request 0

Table 13 SCU external interrupts (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SCU	EXINT0C	<-- in	GPIO	P0_3	GPIO interrupt request 0
SCU	EXINT0D	<-- in	GPIO	P0_9	GPIO interrupt request 0

3.10.4 UART0 interconnections

Table 14 UART0 interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
UART0	RXDA	<-- in	GPIO	P0_3	UART0 receive data input option A via GPIO
UART0	RXDB	<-- in	GPIO	P0_8	UART0 receive data input option B via GPIO
UART0	RXDC	<-- in	GPIO	P1_4	UART0 receive data input option C via GPIO
UART0	RXDD	<-- in	CANTRX	RXD	UART0 receive data input option D via CANTRX
UART0	TXD	out -->	GPIO	P0_1	UART0 transmit data to GPIO
UART0	TXD	out -->	GPIO	P0_7	UART0 transmit data to GPIO
UART0	TXD	out -->	GPIO	P1_3	UART0 transmit data to GPIO
UART0	TXD	out -->	GPIO	P0_3	UART0 transmit data to GPIO (half duplex option)
UART0	TXD	out -->	CANTRX	TXDB	UART0 (mode 1) transmit data to GPIO
UART0	RXDO	out -->	GPIO	P0_0	UART0 (mode 0) transmit data to GPIO
UART0	STARTA	<-- in	GPIO	P0_4	UART0 transmit trigger input option A via GPIO
UART0	STARTB	<-- in	SCU	INP1_IRQ1	UART0 transmit trigger input option B via GPT12
UART0	STARTC	<-- in	CCU7	SR3	UART0 transmit trigger input option C via CCU7
UART0	STARTD	<-- in	–	–	UART0 transmit trigger input option D

3.10.5 UART1 interconnections

Table 15 UART1 interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
UART1	RXDA	<-- in	GPIO	P0_0	UART1 receive data input option A via GPIO
UART1	RXDB	<-- in	GPIO	P1_2	UART1 receive data input option B via GPIO

Table 15 UART1 interconnects (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
UART1	RXDC	<-- in	GPIO	P2_8	UART1 receive data input option C via GPIO
UART1	RXDD	<-- in	MON	MON1O	UART1 receive data input option D via MON1
UART1	TXD	out -->	GPIO	P0_2	UART1 (mode 1) transmit data to GPIO
UART1	TXD	out -->	GPIO	P1_1	UART1 (mode 1) transmit data to GPIO
UART1	TXD	out -->	GPIO	P0_4	UART1 (mode 1) transmit data to GPIO
UART1	RXDO	out -->	GPIO	P1_0	UART1 (mode 0) transmit data to GPIO
UART1	STARTA	<-- in	GPIO	P0_1	UART1 transmit trigger input option A via GPIO
UART1	STARTB	<-- in	SCU	INP1_IRQ1	UART1 transmit trigger input option B via GPT12
UART1	STARTC	<-- in	CCU7	SR3	UART1 transmit trigger input option C via CCU7
UART1	STARTD	<-- in	SCU	INP2_IRQ1	UART1 transmit trigger input option D via MONx

3.10.6 SSC0 interconnections

Table 16 SSC0 interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SSC0	CS0	out -->	GPIO	P1_2	SSC0 chip select 0 (as master)
SSC0	CS1	out -->	GPIO	P0_10	SSC0 chip select 1 (as master)
SSC0	CS2	out -->	GPIO	P1_3	SSC0 chip select 2 (as master)
SSC0	CS3	out -->	GPIO	P0_9	SSC0 chip select 3 (as master)
SSC0	MTSR	out -->	GPIO	P0_1	SSC0 master transmit slave receive (as master)
SSC0	MTSR	out -->	GPIO	P0_4	SSC0 master transmit slave receive (as master)
SSC0	MTSR	out -->	GPIO	P0_6	SSC0 master transmit slave receive (as master)
SSC0	MTSRA	<-- in	GPIO	P0_1	SSC0 master transmit slave receive input option A (as slave)
SSC0	MTSRB	<-- in	GPIO	P0_4	SSC0 master transmit slave receive input option B (as slave)
SSC0	MTSRC	<-- in	GPIO	P2_5	SSC0 master transmit slave receive input option C (as slave)
SSC0	MTSRD	<-- in	–	–	SSC0 master transmit slave receive input option D (as slave)

Table 16 SSC0 interconnects (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SSC0	MRST	out -->	GPIO	P0_2	SSC0 master receive slave transmit (as slave)
SSC0	MRST	out -->	GPIO	P0_6	SSC0 master receive slave transmit (as slave)
SSC0	MRSTA	<-- in	GPIO	P0_2	SSC0 master receive slave transmit input option A (as master)
SSC0	MRSTB	<-- in	GPIO	P0_6	SSC0 master receive slave transmit input option A (as master)
SSC0	MRSTC	<-- in	GPIO	P2_5	SSC0 master receive slave transmit input option C (as master)
SSC0	MRSTD	<-- in	GPIO	P0_1	SSC0 master receive slave transmit input option D (as master)
SSC0	SCLK	out -->	GPIO	P0_3	SSC0 shift clock (as master)
SSC0	SCLK	out -->	GPIO	P0_5	SSC0 shift clock (as master)
SSC0	SCLKA	<-- in	GPIO	P0_3	SSC0 shift clock input option A (as slave)
SSC0	SCLKB	<-- in	GPIO	P0_5	SSC0 shift clock input option B (as slave)
SSC0	SLSA	<-- in	GPIO	P1_0	SSC0 slave select input option A (as slave)
SSC0	SLSB	<-- in	GPIO	P2_6	SSC0 slave select input option B (as slave)
SSC0	SLSC	<-- in	GPIO	P1_3	SSC0 slave select input option C (as slave)
SSC0	SLSD	<-- in	GPIO	P0_9	SSC0 slave select input option D (as slave)
SSC0	STARTA	<-- in	GPIO	P1_1	SSC0 transmit trigger input option A via GPIO
SSC0	STARTB	<-- in	SCU	INP1_IRQ1	SSC0 transmit trigger input option B via GPT12
SSC0	STARTC	<-- in	CCU7	SR3	SSC0 transmit trigger input option C via CCU7
SSC0	STARTD	<-- in	–	–	SSC0 transmit trigger input option D

3.10.7 SSC1 interconnections

Table 17 SSC1 interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SSC1	CS0	out -->	GPIO	P0_2	SSC1 chip select 0 (as master)
SSC1	CS1	out -->	GPIO	P0_10	SSC1 chip select 1 (as master)

Table 17 SSC1 interconnects (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SSC1	CS2	out -->	GPIO	P1_4	SSC1 chip select 2 (as master)
SSC1	CS3	out -->	GPIO	P0_6	SSC1 chip select 3 (as master)
SSC1	MTSR	out -->	GPIO	P1_1	SSC1 master transmit slave receive (as master)
SSC1	MTSR	out -->	GPIO	P0_8	SSC1 master transmit slave receive (as master)
SSC1	MTSR	out -->	GPIO	P0_9	SSC1 master transmit slave receive (as master)
SSC1	MTSR	out -->	GPIO	P1_2	SSC1 master transmit slave receive (as master)
SSC1	MTSRA	<-- in	GPIO	P1_1	SSC0 master transmit slave receive input option A (as slave)
SSC1	MTSRB	<-- in	GPIO	P0_8	SSC0 master transmit slave receive input option B (as slave)
SSC1	MTSRC	<-- in	–	–	SSC0 master transmit slave receive input option C (as slave)
SSC1	MTSRD	<-- in	–	–	SSC0 master transmit slave receive input option D (as slave)
SSC1	MRST	out -->	GPIO	P1_2	SSC1 master receive slave transmit (as slave)
SSC1	MRST	out -->	GPIO	P0_9	SSC1 master receive slave transmit (as slave)
SSC1	MRSTA	<-- in	GPIO	P1_2	SSC1 master receive slave transmit option A (as master)
SSC1	MRSTB	<-- in	GPIO	P0_9	SSC1 master receive slave transmit option B (as master)
SSC1	MRSTC	<-- in	GPIO	P2_6	SSC1 master receive slave transmit option C (as master)
SSC1	MRSTD	<-- in	GPIO	P2_8	SSC1 master receive slave transmit option D (as master)
SSC1	SCLK	out -->	GPIO	P1_0	SSC1 shift clock output (as master)
SSC1	SCLK	out -->	GPIO	P0_7	SSC1 shift clock output (as master)
SSC1	SCLKA	<-- in	GPIO	P1_0	SSC1 shift clock input option A (as master)
SSC1	SCLKB	<-- in	GPIO	P0_7	SSC1 shift clock input option B (as master)
SSC1	SLSA	<-- in	GPIO	P0_2	SSC1 slave select input option A (as slave)
SSC1	SLSB	<-- in	GPIO	P2_4	SSC1 slave select input option B (as slave)

Table 17 SSC1 interconnects (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SSC1	SLSC	<-- in	GPIO	P1_4	SSC1 slave select input option C (as slave)
SSC1	SLSD	<-- in	GPIO	P0_6	SSC1 slave select input option D (as slave)
SSC1	STARTA	<-- in	GPIO	P0_1	SSC1 transmit trigger input option A via GPIO
SSC1	STARTB	<-- in	SCU	INP1_IRQ1	SSC1 transmit trigger input option B via GPT12
SSC1	STARTC	<-- in	CCU7	SR3	SSC1 transmit trigger input option C via CC7
SSC1	STARTD	<-- in	SCU	INP2_IRQ1	SSC1 transmit trigger input option D via MONx

3.10.8 MultiCAN interconnections

Table 18 MultiCAN alternate functions

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
MultiCAN	TXDC	out -->	CANTRX	TXDA	Transmit data output to CANTRX
MultiCAN	TXDC	out -->	GPIO	P0_1	Transmit data output
MultiCAN	TXDC	out -->	GPIO	P0_7	Transmit data output
MultiCAN	TXDC	out -->	GPIO	P1_3	Transmit data output
MultiCAN	RXDCA	<-- in	CANTRX	RXD	Receive data input option A via CANTRX
MultiCAN	RXDCB	<-- in	GPIO	P0_8	Receive data input option A
MultiCAN	RXDCC	<-- in	GPIO	P1_4	Receive data input option A
MultiCAN	RXDCD	<-- in	GPIO	P0_3	Receive data input option A

3.10.9 CANTRX interconnections

Table 19 CANTRX alternate functions

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
CANTRX	RXD	out -->	MultiCAN	RXDCA	Receiver data output to MultiCAN
CANTRX	RXD	out -->	GPIO	P0_1	Receiver data output
CANTRX	RXD	out -->	GPIO	P1_1	Receiver data output
CANTRX	RXD	out -->	T20	T2EXD	Receiver data output to T2 for baud-rate capture
CANTRX	RXD	out -->	GPT12	T4INB	Receiver data output to GPT12 (PWM over CANTRX)

Table 19 CANTRX alternate functions (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
CANTRX	RXD	out -->	UART0	RXDD	Receiver data output to UART0 (UART over CANTRX)
CANTRX	TXDA	<-- in	MultiCAN	TXDC	Transmitter data input option A via MultiCAN
CANTRX	TXDB	<-- in	UART0	TXD	Transmitter data input option B via UART0 (UART over CANTRX)
CANTRX	TXDC	<-- in	GPT12	T3OUT	Transmitter data input option C via GPT12 (PWM over CANTRX)
CANTRX	TXDD	<-- in	GPIO	P0_3	Transmitter data input option D via GPIO

Table 20 CANTRX fail-safe and wake

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
CANTRX	WAKECAN	out -->	PMU	WAKECAN	Wake event
CANTRX	VCAN_UV	<-- in	ADC2	CMPLO3	Voltage monitoring status signal

3.10.10 MON interconnections

Table 21 MON interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
MON	MON1O	out -->	UART1	RXDD	To UART1 as HV UART
MON	MON1O	out -->	T21	T2EXD	To T21 as capture input
MON	MON1O	out -->	PMU	WAKEMON1	To PMU as wake event
MON	MON2O	out -->	PMU	WAKEMON2	To PMU as wake event
MON	MON3O	out -->	PMU	WAKEMON3	To PMU as wake event

3.10.11 ADC1 interconnections

Table 22 ADC1 analog inputs

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
ADC1	AN0	<-- in	PMU	VS	Analog input, ATT_TYP0, uncalibrated
ADC1	AN1	<-- in	PMU	VS	Analog input, ATT_TYP1
ADC1	AN2	<-- in	BDRV	VDH	Analog input, ATT_TYP0, uncalibrated
ADC1	AN3	<-- in	BDRV	VDH	Analog input, ATT_TYP1
ADC1	AN4	<-- in	BDRV	SH1	Analog input, ATT_TYP1
ADC1	AN5	<-- in	BDRV	SH1	Analog input, ATT_TYP1
ADC1	AN6	<-- in	BDRV	SH2	Analog input, ATT_TYP1

Table 22 ADC1 analog inputs (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
ADC1	AN7	<-- in	BDRV	SH2	Analog input, ATT_TYP1
ADC1	AN8	<-- in	BDRV	SH3	Analog input, ATT_TYP1
ADC1	AN9	<-- in	BDRV	SH3	Analog input, ATT_TYP1
ADC1	AN10	<-- in	MON	MON1	Analog input, ATT_TYP0, uncalibrated
ADC1	AN11	<-- in	MON	MON1	Analog input, ATT_TYP1
ADC1	AN12	<-- in	MON	MON2	Analog input, ATT_TYP0, uncalibrated
ADC1	AN13	<-- in	MON	MON2	Analog input, ATT_TYP1
ADC1	AN14	<-- in	MON	MON3	Analog input, ATT_TYP0, uncalibrated
ADC1	AN15	<-- in	MON	MON3	Analog input, ATT_TYP1
ADC1	AN16	<-- in	ARVG	VREV1V2	Analog input, ATT_TYP2
ADC1	AN17	<-- in	–	–	–
ADC1	AN18	<-- in	CSA	OUT	Analog input, ATT_TYP2
ADC1	AN19	<-- in	GPIO	P2_0	Analog input, ATT_TYP2
ADC1	AN20	<-- in	GPIO	P2_1	Analog input, ATT_TYP2
ADC1	AN21	<-- in	GPIO	P2_7	Analog input, ATT_TYP2
ADC1	AN22	<-- in	GPIO	P2_3	Analog input, ATT_TYP2
ADC1	AN23	<-- in	GPIO	P2_4	Analog input, ATT_TYP2
ADC1	AN24	<-- in	GPIO	P2_5	Analog input, ATT_TYP2
ADC1	AN25	<-- in	GPIO	P2_6	Analog input, ATT_TYP2
ADC1	AN26	<-- in	GPIO	P2_2	Analog input, ATT_TYP2

Table 23 ADC1 trigger and gating

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
ADC1	TRGSQ0A	<-- in	ADC1	TRGSW0	Sample trigger input for SQ0
ADC1	TRGSQ0B	<-- in	CCU7	T12ZM	Sample trigger input for SQ0
ADC1	TRGSQ0C	<-- in	CCU7	T12PM	Sample trigger input for SQ0
ADC1	TRGSQ0D	<-- in	CCU7	SR3	Sample trigger input for SQ0
ADC1	TRGSQ0E	<-- in	CCU7	T13PM	Sample trigger input for SQ0
ADC1	TRGSQ0F	<-- in	CCU7	T16CM	Sample trigger input for SQ0
ADC1	TRGSQ0G	<-- in	CCU7	CHE	Sample trigger input for SQ0
ADC1	TRGSQ0H	<-- in	ADC1	EVSQ0	Sample trigger input for SQ0
ADC1	TRGSQ0I	<-- in	ADC1	EVSQ3	Sample trigger input for SQ0
ADC1	TRGSQ1A	<-- in	ADC1	TRGSW1	Sample trigger input for SQ1
ADC1	TRGSQ1B	<-- in	CCU7	T12ZM	Sample trigger input for SQ1
ADC1	TRGSQ1C	<-- in	CCU7	T12CM0	Sample trigger input for SQ1
ADC1	TRGSQ1D	<-- in	CCU7	T12CMB2	Sample trigger input for SQ1

Table 23 ADC1 trigger and gating (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
ADC1	TRGSQ1E	<-- in	CCU7	T14PM	Sample trigger input for SQ1
ADC1	TRGSQ1F	<-- in	CCU7	T13CM	Sample trigger input for SQ1
ADC1	TRGSQ1G	<-- in	SCU	INP1_IRQ0	Sample trigger input for SQ1
ADC1	TRGSQ1H	<-- in	ADC1	EVSQ1	Sample trigger input for SQ1
ADC1	TRGSQ1I	<-- in	ADC1	EVSQ0	Sample trigger input for SQ1
ADC1	TRGSQ2A	<-- in	ADC1	TRGSW2	Sample trigger input for SQ2
ADC1	TRGSQ2B	<-- in	CCU7	T12ZM	Sample trigger input for SQ2
ADC1	TRGSQ2C	<-- in	CCU7	T12CM1	Sample trigger input for SQ2
ADC1	TRGSQ2D	<-- in	CCU7	T12CMB0	Sample trigger input for SQ2
ADC1	TRGSQ2E	<-- in	CCU7	T15PM	Sample trigger input for SQ2
ADC1	TRGSQ2F	<-- in	CCU7	T14CM	Sample trigger input for SQ2
ADC1	TRGSQ2G	<-- in	SCU	INP1_IRQ1	Sample trigger input for SQ2
ADC1	TRGSQ2H	<-- in	ADC1	EVSQ2	Sample trigger input for SQ2
ADC1	TRGSQ2I	<-- in	ADC1	EVSQ1	Sample trigger input for SQ2
ADC1	TRGSQ3A	<-- in	ADC1	TRGSW3	Sample trigger input for SQ3
ADC1	TRGSQ3B	<-- in	CCU7	T12ZM	Sample trigger input for SQ3
ADC1	TRGSQ3C	<-- in	CCU7	T12CM2	Sample trigger input for SQ3
ADC1	TRGSQ3D	<-- in	CCU7	T12CMB1	Sample trigger input for SQ3
ADC1	TRGSQ3E	<-- in	CCU7	T16PM	Sample trigger input for SQ3
ADC1	TRGSQ3F	<-- in	CCU7	T15CM	Sample trigger input for SQ3
ADC1	TRGSQ3G	<-- in	CCU7	SR3	Sample trigger input for SQ3
ADC1	TRGSQ3H	<-- in	ADC1	EVSQ3	Sample trigger input for SQ3
ADC1	TRGSQ3I	<-- in	ADC1	EVSQ2	Sample trigger input for SQ3
ADC1	GTSQ0A	<-- in	ADC1	GTSW0	Gating input for SQ0
ADC1	GTSQ0B	<-- in	CCU7	CC70ST	Gating input for SQ0
ADC1	GTSQ0C	<-- in	CCU7	CC71STB	Gating input for SQ0
ADC1	GTSQ0D	<-- in	CCU7	C73ST	Gating input for SQ0
ADC1	GTSQ1A	<-- in	ADC1	GTSW1	Gating input for SQ1
ADC1	GTSQ1B	<-- in	CCU7	CC71ST	Gating input for SQ1
ADC1	GTSQ1C	<-- in	CCU7	CC70STB	Gating input for SQ1
ADC1	GTSQ1D	<-- in	CCU7	C74ST	Gating input for SQ1
ADC1	GTSQ2A	<-- in	ADC1	GTSW2	Gating input for SQ2
ADC1	GTSQ2B	<-- in	CCU7	CC72ST	Gating input for SQ2
ADC1	GTSQ2C	<-- in	CCU7	CC72STB	Gating input for SQ2
ADC1	GTSQ2D	<-- in	CCU7	CC75ST	Gating input for SQ2
ADC1	GTSQ3A	<-- in	ADC1	GTSW3	Gating input for SQ3
ADC1	GTSQ3B	<-- in	CCU7	CC73ST	Gating input for SQ3

Table 23 ADC1 trigger and gating (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
ADC1	GTSQ3C	<-- in	CCU7	CC73ST	Gating input for SQ3
ADC1	GTSQ3D	<-- in	CCU7	CC76ST	Gating input for SQ3
ADC1	EXTSTRA	<-- in	CCU7	T12ZM	Synchronization request for sequencer
ADC1	EXTSTRB	<-- in	CCU7	T12PM	Synchronization request for sequencer
ADC1	EXTSTRC	<-- in	CCU7	T13PM	Synchronization request for sequencer
ADC1	EXTSTRD	<-- in	CCU7	T14PM	Synchronization request for sequencer
ADC1	EXTSTRE	<-- in	CCU7	T15PM	Synchronization request for sequencer
ADC1	EXTSTRF	<-- in	CCU7	T16PM	Synchronization request for sequencer
ADC1	EXTSTE	<-- in	CCU7	T12STE	Synchronization enable for sequencer

Table 24 ADC1 alternate functions

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
ADC1	CMPLO3	out -->	GPIO	P0_3	CMP3 lower threshold level output
ADC1	CMPLO3	out -->	CCU7	CCPOS2C	CMP3 lower threshold level output
ADC1	CMPUP3	out -->	GPIO	P0_6	CMP3 upper threshold level output
ADC1	CMPUP3	out -->	–	–	CMP3 upper threshold level output
ADC1	CMPLO2	out -->	GPIO	P0_10	CMP2 lower threshold level output
ADC1	CMPLO2	out -->	CCU7	CCPOS1C	CMP2 lower threshold level output
ADC1	CMPUP2	out -->	GPIO	P1_0	CMP2 upper threshold level output
ADC1	CMPUP2	out -->	–	–	CMP2 upper threshold level output
ADC1	CMPLO1	out -->	GPIO	P0_8	CMP1 lower threshold level output
ADC1	CMPLO1	out -->	CCU7	CCPOS0C	CMP1 lower threshold level output
ADC1	CMPUP1	out -->	GPIO	P1_1	CMP1 upper threshold level output
ADC1	CMPUP1	out -->	–	–	CMP1 upper threshold level output
ADC1	CMPLO0	out -->	GPIO	P0_9	CMP0 lower threshold level output
ADC1	CMPLO0	out -->	CCU7	CTRAPC	CMP0 lower threshold level output
ADC1	CMPUP0	out -->	GPIO	P1_2	CMP0 upper threshold level output
ADC1	CMPUP0	out -->	–	–	CMP0 upper threshold level output

3.10.12 ADC2 interconnections

Table 25 ADC2 analog inputs

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
ADC2	AN0	<-- in	PMU	VS	Analog input, ATT_TYP0
ADC2	AN1	<-- in	BRDV	VSD	Analog input, ATT_TYP3
ADC2	AN2	<-- in	BRDV	VCP	Analog input, ATT_TYP2

Table 25 ADC2 analog inputs (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
ADC2	AN3	<-- in	BRDV	VSD	Analog input, ATT_TYP1
ADC2	AN4	<-- in	MON	MON1	Analog input, ATT_TYP1
ADC2	AN5	<-- in	MON	MON2	Analog input, ATT_TYP1
ADC2	AN6	<-- in	MON	MON3	Analog input, ATT_TYP1
ADC2	AN7	<-- in	PMU	VDDEXT	Analog input, ATT_TYP4
ADC2	AN8	<-- in	PMU	VCAN	Analog input, ATT_TYP4
ADC2	AN9	<-- in	PMU	VDDP	Analog input, ATT_TYP4
ADC2	AN10	<-- in	PMU	VDDC	Analog input, ATT_TYP4
ADC2	AN11	<-- in	ARVG	VAREF	Analog input, ATT_TYP4
ADC2	AN12	<-- in	PMU	VGEN_ADC	Analog input, ATT_TYP4
ADC2	AN13	<-- in	TEMPS	TEMP0	Analog input, ATT_TYP4
ADC2	AN14	<-- in	TEMPS	TEMP1	Analog input, ATT_TYP4

Table 26 ADC2 fail-safe

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
ADC2	CMPLO5	out -->	SCU	SYS_OT_WARN	CMP5 lower threshold level output
ADC2	CMPUP5	out -->	PMU	SYS_OT	CMP5 upper threshold level output
ADC2	CMPLO4	out -->	–	–	CMP4 lower threshold level output
ADC2	CMPUP4	out -->	BDRV	CP_OTSD	CMP4 upper threshold level output to shutdown CP at OT
ADC2	CMPLO3	out -->	CANTRX	VCAN_UV	CMP3 lower threshold to monitor VCAN for UV
ADC2	CMPUP3	out -->	–	-	CMP3 upper threshold level output
ADC2	CMPLO2	out -->	BDRV	VCP_LOTH1	CMP2 lower threshold to monitor VCP for UV
ADC2	CMPUP2	out -->	BDRV	VCP_UPTH	CMP2 upper threshold to monitor VCP for OV
ADC2	CMPLO1	out -->	BDRV	VSD_LOTH	CMP1 lower threshold to monitor VSD for UV
ADC2	CMPUP1	out -->	–	–	CMP1 upper threshold level output
ADC2	CMPLO0	out -->	BDRV	VSD_CP1ST	CMP0 lower threshold to select CP stage
ADC2	CMPUP0	out -->	BDRV	VSD_UPTH	CMP0 upper threshold to monitor VSD for OV

Table 27 ADC2 alternate functions

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
ADC2	CMPLO7	out -->	GPIO	P0_10	CMP7 lower threshold level output to GPIO
ADC2	CMPUP7	out -->	GPIO	P0_7	CMP7 upper threshold level output to GPIO
ADC2	CMPLO6	out -->	GPIO	P1_0	CMP6 lower threshold level output to GPIO
ADC2	CMPUP6	out -->	GPIO	P1_1	CMP6 upper threshold level output to GPIO

3.10.13 CSA interconnections

Table 28 CSA interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
CSA	OUT	out -->	ADC1	AN18	Output signal
CSA	OUT	out -->	CSC	INP	Output signal
CSA	OUTOFF	out -->	CSC	INOFF	Output offset signal

3.10.14 CSC interconnections

Table 29 CSC interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
CSC	INOFF	<-- in	CSA	OUTOFF	Input offset
CSC	INP	<-- in	CSA	OUT	Positive input
CSC	CSC_BIST_FAIL	out -->	PMU	CSC_BIST_FAIL	Result of built-in self-test
CSC	CSC_EN	out -->	PMU	CSC_EN	CSC comparator enable signal after self-test
CSC	CSC_OC	out -->	CCU7	CTRAPD	Output (level)
CSC	CSC_OC	out -->	PMU	CSC_OC	Output (level)

3.10.15 BEMFC interconnections

Table 30 BEMFC alternate functions

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
BEMFC	PH3_ZC_STS	out -->	GPIO	P0_3	Phase 3 zero cross status
BEMFC	PH3_ZC_STS	out -->	GPIO	P1_2	Phase 3 zero cross status
BEMFC	PH3_ZC_STS	out -->	CCU7	CCPOS2D	Phase 3 zero cross status
BEMFC	PH2_ZC_STS	out -->	GPIO	P0_2	Phase 2 zero cross status

Table 30 BEMFC alternate functions (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
BEMFC	PH2_ZC_STS	out -->	GPIO	P1_4	Phase 2 zero cross status
BEMFC	PH2_ZC_STS	out -->	CCU7	CCPOS1D	Phase 2 zero cross status
BEMFC	PH2_ZC_STS	out -->	GPT12	T3EUDD	Phase 2 zero cross status
BEMFC	PH1_ZC_STS	out -->	GPIO	P0_1	Phase 1 zero cross status
BEMFC	PH1_ZC_STS	out -->	GPIO	P1_3	Phase 1 zero cross status
BEMFC	PH1_ZC_STS	out -->	CCU7	CCPOS0D	Phase 1 zero cross status
BEMFC	PH1_ZC_STS	out -->	GPT12	T3IND	Phase 1 zero cross status
BEMFC	PHXZC_TRIG	out -->	GPT12	CAPINB	Selected phase zero cross trigger output
BEMFC	INA1	<-- in	CCU7	CC70	PMW trigger for comparator
BEMFC	INA2	<-- in	CCU7	CC71	PMW trigger for comparator
BEMFC	INA3	<-- in	CCU7	CC72	PMW trigger for comparator
BEMFC	INB1	<-- in	CCU7	COUT70	PMW trigger for comparator
BEMFC	INB2	<-- in	CCU7	COUT71	PMW trigger for comparator
BEMFC	INB3	<-- in	CCU7	COUT72	PMW trigger for comparator
BEMFC	TRIGA	<-- in	CCU7	T12PM	Sample trigger for comparator in 0/100%
BEMFC	TRIGB	<-- in	CCU7	SR3	Sample trigger for comparator in 0/100%

3.10.16 SDADC interconnections

Table 31 SDADC alternate GPIO functions

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SDADC	IN0PA	<-- in	GPIO	P2_4	Positive input channel 0 option A
SDADC	IN0PB	<-- in	GPIO	P2_8	Positive input channel 0 option B
SDADC	IN0NA	<-- in	GPIO	P2_2	Negative input channel 0 option A
SDADC	IN0NB	<-- in	GPIO	P2_6	Negative input channel 0 option B
SDADC	IN1PA	<-- in	GPIO	P2_5	Positive input channel 1 option A
SDADC	IN1PB	<-- in	GPIO	P2_9	Positive input channel 1 option B
SDADC	IN1NA	<-- in	GPIO	P2_3	Negative input channel 1 option A
SDADC	IN1NB	<-- in	GPIO	P2_7	Negative input channel 1 option B
SDADC	DIN0A	<-- in	GPIO	P0_4	Data input for CIC filter channel 0 option A
SDADC	DIN0B	<-- in	GPIO	P1_3	Data input for CIC filter channel 0 option B
SDADC	DIN1A	<-- in	GPIO	P0_5	Data input for CIC filter channel 1 option A

Table 31 SDADC alternate GPIO functions (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SDADC	DIN1B	<-- in	GPIO	P1_4	Data input for CIC filter channel 1 option B
SDADC	DOUT0	out -->	GPIO	P0_4	Modulator data output channel 0
SDADC	DOUT0	out -->	GPIO	P0_5	Modulator data output channel 0
SDADC	DOUT1	out -->	GPIO	P0_6	Modulator data output channel 1
SDADC	DOUT1	out -->	GPIO	P0_7	Modulator data output channel 1
SDADC	MCLK	out -->	GPIO	P0_8	Modulator clock output
SDADC	MCLK	out -->	GPIO	P0_9	Modulator clock output
SDADC	CMP0	out -->	GPIO	P1_0	Comparator output channel 0
SDADC	CMP0	out -->	GPIO	P0_6	Comparator output channel 0
SDADC	CMP1	out -->	GPIO	P0_3	Comparator output channel 1
SDADC	CMP1	out -->	GPIO	P0_5	Comparator output channel 1

Table 32 SDADC interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
SDADC	CMP0	out -->	GPT12	T3INC	Use as incremental interface for SDADC
SDADC	CMP0	out -->	CCU7	CC70IND	Comparator output channel 0
SDADC	CMP1	out -->	GPT12	T3EUDC	Use as incremental interface for SDADC
SDADC	CMP1	out -->	CCU7	CC71IND	Comparator output channel 1
SDADC	TRGSD0A	<-- in	CCU7	SR3	External trigger input A channel 0
SDADC	TRGSD0B	<-- in	CCU7	SR2	External trigger input B channel 0
SDADC	TRGSD1A	<-- in	CCU7	SR3	External trigger input A channel 1
SDADC	TRGSD1B	<-- in	CCU7	SR2	External trigger input B channel 1

3.10.17 T20 interconnections

Table 33 T20 interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
T20	EXF2	out -->	GPIO	P0_10	T20 capture event to GPIO
T20	T2A	<-- in	GPIO	P0_0	T20 count input option A via GPIO
T20	T2B	<-- in	GPIO	P2_8	T20 count input option B via GPIO
T20	T2C	<-- in	–	–	–
T20	T2D	<-- in	–	–	–
T20	T2EXA	<-- in	GPIO	P0_3	T20 capture input option A via GPIO
T20	T2EXB	<-- in	GPIO	P0_8	T20 capture input option B via GPIO

Table 33 T20 interconnects (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
T20	T2EXC	<-- in	GPIO	P1_4	T20 capture input option C via GPIO
T20	T2EXD	<-- in	CANTRX	RXD	T20 capture input option D (CAN baud-rate)

3.10.18 T21 interconnections

Table 34 T21 interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
T21	EXF2	out -->	GPIO	P0_0	T21 capture event to GPIO
T21	EXF2	out -->	GPIO	P1_4	T21 capture event to GPIO
T21	T2A	<-- in	GPIO	P0_3	T21 count input option A via GPIO
T21	T2B	<-- in	GPIO	P0_8	T21 count input option B via GPIO
T21	T2C	<-- in	GPIO	P1_4	T21 count input option C via GPIO
T21	T2D	<-- in	GPIO	P2_9	T21 count input option D via GPIO
T21	T2EXA	<-- in	GPIO	P0_0	T21 capture input option A via GPIO
T21	T2EXB	<-- in	GPIO	P1_2	T21 capture input option B via GPIO
T21	T2EXC	<-- in	GPIO	P2_8	T21 capture input option C via GPIO
T21	T2EXD	<-- in	MON	MON1O	T21 capture input option D (PWM via MON1)

3.10.19 GPT12 interconnections

Table 35 GPT12 alternate GPIO function

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
GPT12	T2INA	<-- in	GPIO	P0_1	T2 count input option A
GPT12	T2INB	<-- in	GPIO	P0_5	T2 count input option B
GPT12	T2EUDA	<-- in	GPIO	P0_2	T2 count direction option A
GPT12	T2EUDB	<-- in	GPIO	P1_3	T2 count direction option B
GPT12	T3INA	<-- in	GPIO	P1_0	T3 input option A
GPT12	T3INB	<-- in	GPIO	P2_2	T3 input option B
GPT12	T3EUDA	<-- in	GPIO	P1_1	T3 count direction option A
GPT12	T3EUDB	<-- in	GPIO	P2_4	T3 count direction option B
GPT12	T4INA	<-- in	GPIO	P1_2	T4 input option A
GPT12	T4EUDA	<-- in	GPIO	P2_0	T4 count direction option A
GPT12	T4EUDB	<-- in	GPIO	P2_1	T4 count direction option B
GPT12	T5INA	<-- in	GPIO	P0_6	T5 input option A
GPT12	T5EUDA	<-- in	GPIO	P0_7	T5 count direction option A

Table 35 GPT12 alternate GPIO function (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
GPT12	T6INA	<-- in	GPIO	P0_4	T6 count direction option A
GPT12	T6INB	<-- in	GPIO	P0_9	T6 count direction option B
GPT12	T6EUDA	<-- in	GPIO	P0_0	T6 input option A
GPT12	T6EUDB	<-- in	GPIO	P0_7	T6 input option B
GPT12	CAPINA	<-- in	GPIO	P1_1	Capture input option A
GPT12	T3OUT	out -->	GPIO	P0_2	T3 output option
GPT12	T3OUT	out -->	GPIO	P1_1	T3 output option
GPT12	T3OUT	out -->	GPIO	P0_4	T3 output option
GPT12	T6OUT	out -->	GPIO	P0_1	T6 output option
GPT12	T6OUT	out -->	GPIO	P0_7	T6 output option
GPT12	T6OUT	out -->	GPIO	P1_0	T6 output option
GPT12	T6OUT	out -->	GPIO	P1_3	T6 output option

Table 36 GPT12 interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
GPT12	T3INC	<-- in	SDADC	CMP0	Use as incremental interface for SDADC
GPT12	T3IND	<-- in	BEMFC	PH1_ZC_STS	Count BEMC.PH1_ZC_STS events
GPT12	T3EUDC	<-- in	SDADC	CMP1	Use as incremental interface for SDADC
GPT12	T3EUDD	<-- in	BEMFC	PH2_ZC_STS	Capture BEMFC.PH2_ZC_STS events
GPT12	T4INB	<-- in	CANTRX	RXD	T4 input option B (PWM over CANTRX)
GPT12	T4INC	<-- in	CCU7	C74ST	Gate T4 via CCU7.T14
GPT12	T4IND	<-- in	CCU7	SR3	Count CCU7 events
GPT12	T4EUDC	<-- in	–	–	T4 count direction option C
GPT12	T4EUDD	<-- in	–	–	T4 count direction option D
GPT12	T5INB	<-- in	–	–	T5 input option B
GPT12	T5EUDB	<-- in	–	–	T5 count direction option B
GPT12	CAPINB	<-- in	BEMFC	PHXZC_TRIG	Capture input option B
GPT12	CAPINC	<-- in	internal	–	Read trigger from T3
GPT12	CAPIND	<-- in	internal	–	Read trigger from T2, T3, T4
GPT12	T3OUT	out -->	CCU7	T12HRD	Start CCU7.T12 via T3
GPT12	T3OUT	out -->	CANTRX	TXDC	PWM output to CANTRX
GPT12	T6OUT	out -->	CCU7	T12HRE	Start CCU7.T12 via T6

3.10.20 CCU7 interconnections

Table 37 CCU7 peripheral interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
CCU7	SR2	out -->	SDADC	TRGSD0B	Service request 2
CCU7	SR2	out -->	SDADC	TRGSD1B	Service request 2
CCU7	SR3	out -->	SDADC	TRGSD0A	Service request 3
CCU7	SR3	out -->	SDADC	TRGSD1A	Service request 3
CCU7	SR3	out -->	SSC0	STARTC	Service request 3
CCU7	SR3	out -->	SSC1	STARTC	Service request 3
CCU7	SR3	out -->	UART0	STARTC	Service request 3
CCU7	SR3	out -->	UART1	STARTC	Service request 3
CCU7	SR3	out -->	GPT12	T4IND	Service request 3
CCU7	SR3	out -->	BEMFC	TRIGB	Service request 3
CCU7	CC70	out -->	BEMFC	INA1	PWM channel CC70
CCU7	CC70	out -->	BDRV	INA1	PWM channel CC70
CCU7	CC71	out -->	BDRV	INA2	PWM channel CC71
CCU7	CC71	out -->	BEMFC	INA2	PWM channel CC71
CCU7	CC72	out -->	BEMFC	INA3	PWM channel CC72
CCU7	CC72	out -->	BDRV	INA3	PWM channel CC72
CCU7	COUT70	out -->	BEMFC	INB1	PWM channel COUT70
CCU7	COUT70	out -->	BDRV	INB1	PWM channel COUT70
CCU7	COUT71	out -->	BEMFC	INB2	PWM channel COUT71
CCU7	COUT71	out -->	BDRV	INB2	PWM channel COUT71
CCU7	COUT72	out -->	BDRV	INB3	PWM channel COUT72
CCU7	COUT72	out -->	BEMFC	INB3	PWM channel COUT72
CCU7	CC70IND	<-- in	SDADC	CMP0	Capture SDADC.CMP0 event
CCU7	CC71IND	<-- in	SDADC	CMP1	Capture SDADC.CMP1 event
CCU7	CC72IND	<-- in	–	–	–
CCU7	CCPOS0C	<-- in	ADC1	CMPLO1	CCPOS0 input option C via GPIO
CCU7	CCPOS0D	<-- in	BEMFC	PH1_ZC_STS	CCPOS0 input option D via BEMFC
CCU7	CCPOS1C	<-- in	ADC1	CMPLO2	CCPOS1 input option C via GPIO
CCU7	CCPOS1D	<-- in	BEMFC	PH2_ZC_STS	CCPOS1 input option D via BEMFC
CCU7	CCPOS2C	<-- in	ADC1	CMPLO3	CCPOS2 input option C via GPIO
CCU7	CCPOS2D	<-- in	BEMFC	PH3_ZC_STS	CCPOS2 input option D via BEMFC
CCU7	CTRAPC	<-- in	ADC1	CMPLO0	Trap input option C via inverted ADC1.CMPLO0
CCU7	CTRAPD	<-- in	CSC	CSC_OUT	Trap input option C via inverted CSC.OUT
CCU7	T12HRC	<-- in	GPT12	T2IRQ	Start T12 option C via GPT12.T2

Table 37 CCU7 peripheral interconnects (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
CCU7	T12HRD	<-- in	GPT12	T3OUT	Start T12 option D via GPT12.T3
CCU7	T12HRE	<-- in	GPT12	T6OUT	Start T12 option E via GPT12.T6
CCU7	T12HRF	<-- in	–	–	–
CCU7	T12HRG	<-- in	–	–	–
CCU7	T12HRH	<-- in	–	–	–
CCU7	T13HRC	<-- in	GPT12	T2IRQ	Start T13 option C via GPT12.T2
CCU7	T13HRD	<-- in	–	–	–
CCU7	T13HRE	<-- in	–	–	–
CCU7	T13HRF	<-- in	–	–	–
CCU7	T13HRG	<-- in	–	–	–
CCU7	T13HRH	<-- in	–	–	–
CCU7	T14HRC	<-- in	GPT12	T2IRQ	Start T14 option C via GPT12.T2
CCU7	T14HRD	<-- in	–	–	–
CCU7	T14HRE	<-- in	–	–	–
CCU7	T14HRF	<-- in	–	–	–
CCU7	T14HRG	<-- in	–	–	–
CCU7	T14HRH	<-- in	–	–	–
CCU7	T15HRC	<-- in	GPT12	T2IRQ	Start T15 option C via GPT12.T2
CCU7	T15HRD	<-- in	–	–	–
CCU7	T15HRE	<-- in	–	–	–
CCU7	T15HRF	<-- in	–	–	–
CCU7	T15HRG	<-- in	–	–	–
CCU7	T15HRH	<-- in	–	–	–
CCU7	T16HRC	<-- in	GPT12	T2IRQ	Start T16 option C via GPT12.T2
CCU7	T16HRD	<-- in	–	–	–
CCU7	T16HRE	<-- in	–	–	–
CCU7	T16HRF	<-- in	–	–	–
CCU7	T16HRG	<-- in	–	–	–
CCU7	T16HRH	<-- in	–	–	–
CCU7	T12CM0	out -->	ADC1	TRGSQ1C	T12 compare match 70
CCU7	T12CM1	out -->	ADC1	TRGSQ2C	T12 compare match 71
CCU7	T12CM2	out -->	ADC1	TRGSQ3C	T12 compare match 72
CCU7	T12CMB0	out -->	ADC1	TRGSQ2D	T12 compare match 70B
CCU7	T12CMB1	out -->	ADC1	TRGSQ3D	T12 compare match 71B
CCU7	T12CMB2	out -->	ADC1	TRGSQ1D	T12 compare match 72B
CCU7	T12OM	out -->	–	–	T12 one match
CCU7	T12PM	out -->	ADC1	TRGSQ0C	T12 period match

Table 37 CCU7 peripheral interconnects (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
CCU7	T12PM	out -->	ADC1	EXTSTRB	T12 period match
CCU7	T12PM	out -->	BEMFC	TRG6	T12 period match
CCU7	T12ZM	out -->	ADC1	TRGSQ0B	T12 zero match
CCU7	T12ZM	out -->	ADC1	TRGSQ1B	T12 zero match
CCU7	T12ZM	out -->	ADC1	TRGSQ2B	T12 zero match
CCU7	T12ZM	out -->	ADC1	TRGSQ3B	T12 zero match
CCU7	T12ZM	out -->	ADC1	EXTSTRA	T12 zero match
CCU7	T13CM	out -->	ADC1	TRGSQ1F	T13 compare match
CCU7	T13PM	out -->	ADC1	TRGSQ0E	T13 period match
CCU7	T13PM	out -->	ADC1	EXTSTRC	T13 period match
CCU7	T14CM	out -->	ADC1	TRGSQ2F	T14 compare match
CCU7	T14PM	out -->	ADC1	TRGSQ1E	T14 period match
CCU7	T14PM	out -->	ADC1	EXTSTRD	T14 period match
CCU7	T15CM	out -->	ADC1	TRGSQ3F	T15 compare match
CCU7	T15PM	out -->	ADC1	TRGSQ2E	T15 period match
CCU7	T15PM	out -->	ADC1	EXTSTRE	T15 period match
CCU7	T16CM	out -->	ADC1	TRGSQ0F	T16 compare match
CCU7	T16PM	out -->	ADC1	TRGSQ3E	T16 period match
CCU7	T16PM	out -->	ADC1	EXTSTRF	T16 period match
CCU7	CHE	out -->	ADC1	TRGSQ0G	Correct hall event
CCU7	T12STE	out -->	ADC1	EXTSTE	T12 shadow transfer enable output
CCU7	CC70ST	out -->	ADC1	GTSQ0B	State bit channel 70
CCU7	CC71ST	out -->	ADC1	GTSQ1B	State bit channel 71
CCU7	CC72ST	out -->	ADC1	GTSQ2B	State bit channel 72
CCU7	CC70STB	out -->	ADC1	GTSQ1C	State bit channel 70
CCU7	CC71STB	out -->	ADC1	GTSQ0C	State bit channel 71
CCU7	CC72STB	out -->	ADC1	GTSQ2C	State bit channel 72
CCU7	C73ST	out -->	ADC1	GTSQ0D	State bit channel 73
CCU7	C73ST	out -->	ADC1	GTSQ3B	State bit channel 73
CCU7	C74ST	out -->	GPT12	T4INC	State bit channel 74
CCU7	C74ST	out -->	ADC1	GTSQ1D	State bit channel 74
CCU7	C75ST	out -->	ADC1	GTSQ2D	State bit channel 75
CCU7	C76ST	out -->	ADC1	GTSQ3D	State bit channel 76

Table 38 CCU7 alternate GPIO functions

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
CCU7	CC70	out -->	GPIO	P1_2	PWM channel CC70
CCU7	CC70	out -->	GPIO	P0_4	PWM channel CC70
CCU7	CC71	out -->	GPIO	P1_0	PWM channel CC71
CCU7	CC71	out -->	GPIO	P0_6	PWM channel CC71
CCU7	CC72	out -->	GPIO	P0_0	PWM channel CC72
CCU7	CC72	out -->	GPIO	P0_1	PWM channel CC72
CCU7	CC72	out -->	GPIO	P0_9	PWM channel CC72
CCU7	COUT70	out -->	GPIO	P0_3	PWM channel COUT70
CCU7	COUT70	out -->	GPIO	P0_5	PWM channel COUT70
CCU7	COUT71	out -->	GPIO	P1_1	PWM channel COUT71
CCU7	COUT71	out -->	GPIO	P0_7	PWM channel COUT71
CCU7	COUT72	out -->	GPIO	P0_10	PWM channel COUT72
CCU7	COUT72	out -->	GPIO	P0_8	PWM channel COUT72
CCU7	COUT73	out -->	GPIO	P0_2	PWM channel COUT73
CCU7	COUT73	out -->	GPIO	P1_3	PWM channel COUT73
CCU7	CC70INA	<-- in	GPIO	P1_2	CC70 capture input option A
CCU7	CC70INB	<-- in	GPIO	P2_2	CC70 capture input option B
CCU7	CC70INC	<-- in	GPIO	P0_4	CC70 capture input option C
CCU7	CC71INA	<-- in	GPIO	P1_0	CC71 capture input option A
CCU7	CC71INB	<-- in	GPIO	P2_3	CC71 capture input option B
CCU7	CC71INC	<-- in	GPIO	P0_6	CC71 capture input option C
CCU7	CC72INA	<-- in	GPIO	P0_1	CC72 capture input option A
CCU7	CC72INB	<-- in	GPIO	P2_4	CC72 capture input option B
CCU7	CC72INC	<-- in	GPIO	P0_9	CC72 capture input option C
CCU7	CCPOS0A	<-- in	GPIO	P1_2	CCPOS0 input option A via GPIO
CCU7	CCPOS0B	<-- in	GPIO	P2_3	CCPOS0 input option B via GPIO
CCU7	CCPOS1A	<-- in	GPIO	P1_1	CCPOS1 input option A via GPIO
CCU7	CCPOS1B	<-- in	GPIO	P2_4	CCPOS1 input option B via GPIO
CCU7	CCPOS2A	<-- in	GPIO	P1_0	CCPOS2 input option A via GPIO
CCU7	CCPOS2B	<-- in	GPIO	P2_2	CCPOS2 input option B via GPIO
CCU7	T12HRA	<-- in	GPIO	P0_0	Start T12 option A via GPIO
CCU7	T12HRB	<-- in	GPIO	P2_0	Start T12 option B via GPIO
CCU7	T13HRA	<-- in	GPIO	P0_2	Start T13 option A via GPIO
CCU7	T13HRB	<-- in	GPIO	P2_1	Start T13 option B via GPIO
CCU7	CTRAPA	<-- in	GPIO	P1_1	Trap input option A via GPIO
CCU7	CTRAPB	<-- in	GPIO	P2_6	Trap input option B via GPIO

Table 38 CCU7 alternate GPIO functions (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
CCU7	T14HRA	<-- in	GPIO	P0_4	Start T14 option A via GPIO
CCU7	T14HRB	<-- in	GPIO	P0_7	Start T14 option B via GPIO
CCU7	T15HRA	<-- in	GPIO	P0_5	Start T15 option A via GPIO
CCU7	T15HRB	<-- in	GPIO	P0_8	Start T15 option B via GPIO
CCU7	T16HRA	<-- in	GPIO	P0_6	Start T16 option A via GPIO
CCU7	T16HRB	<-- in	GPIO	P0_9	Start T16 option B via GPIO

3.10.21 BDRV interconnections

Table 39 BDRV interconnects

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
BDRV	INA1	<-- in	CCU7	CC70	Gate driver INA1 input
BDRV	INA2	<-- in	CCU7	CC71	Gate driver INA2 input
BDRV	INA3	<-- in	CCU7	CC72	Gate driver INA3 input
BDRV	INB1	<-- in	CCU7	COUT70	Gate driver INB1 input
BDRV	INB2	<-- in	CCU7	COUT71	Gate driver INB2 input
BDRV	INB3	<-- in	CCU7	COUT72	Gate driver INB3 input

Table 40 BDRV fail-safe

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
BDRV	SAFE_ENABLE	<-- in	PMU	SSONOUT	Safe enable of BDRV
BDRV	SAFE_SHUTDOWN	<-- in	PMU	SSOOUT	Safe shutdown event
BDRV	SSO_HCDIS	<-- in	PMU	FASTDIS	Safe shutdown pre-discharge
BDRV	VSD_OV	out -->	PMU	WAKEVSDOV	Emergency wake event
BDRV	BDRV_SD	<-- in	SCU	BDRV_SD	Shutdown request from SCU to BDRV
BDRV	VCP_LOTH1	<-- in	ADC2	CMPLO2	VCP monitoring for undervoltage detection
BDRV	VCP_UPTH	<-- in	ADC2	CMPUP2	VCP monitoring for overvoltage detection
BDRV	VSD_LOTH	<-- in	ADC2	CMPLO1	VSD monitoring for undervoltage detection
BDRV	VSD_UPTH	<-- in	ADC2	CMPUP0	VSD monitoring for overvoltage detection

Table 40 BDRV fail-safe (cont'd)

From/to module	From/to signal	Direction	From/to module	From/to signal	Description
BDRV	CP_OTSD	<-- in	ADC2	CMPUP4	Temperature monitoring for heat detection
BDRV	VSD_CP1ST	<-- in	ADC2	CMPLO0	VSD monitoring for automatic stage selection of charge pump

3.11 Special function register information

The special function registers (SFRs) contain following information, see example in **Figure 11**:

- Register layout
  - SFR name
  - Bitfield name
  - SFR offset address: this is the SFRs offset address in the address map, which is added to the modules base address (see module base address)
  - Reset values: either there is a value or a link to a table
  - Bit position
  - Access type
- Register description
  - Bitfield name
  - Bit position
  - Access type
  - Bitfield description
- Reset value
  - RESET_TYPE_X: this gives the reset type at which the SFR is reset, please refer to the PMU (**Chapter 5.6.5**)
  - Access type: this describes the access method for the CPU and defines how the bit behaves in case a hardware action is happening
  - Trim type: upon startup the BootROM writes the factory trimming values to some SFRs which may differ from the reset value (TRIM_1, TRIM_2). The user will see these trim values after boot up. The factory trim values (TRIM_1 and TRIM_2) cannot be influenced. Some trim values are user changeable, e. g. configuration for P0.10 as GPIO or RESET pin (TRIM_100_TP). The TRIM_100_TP can be configured via config sector setting.
- Module base address
  - This is the module base address. It can be found at the beginning of each SFR module chapter

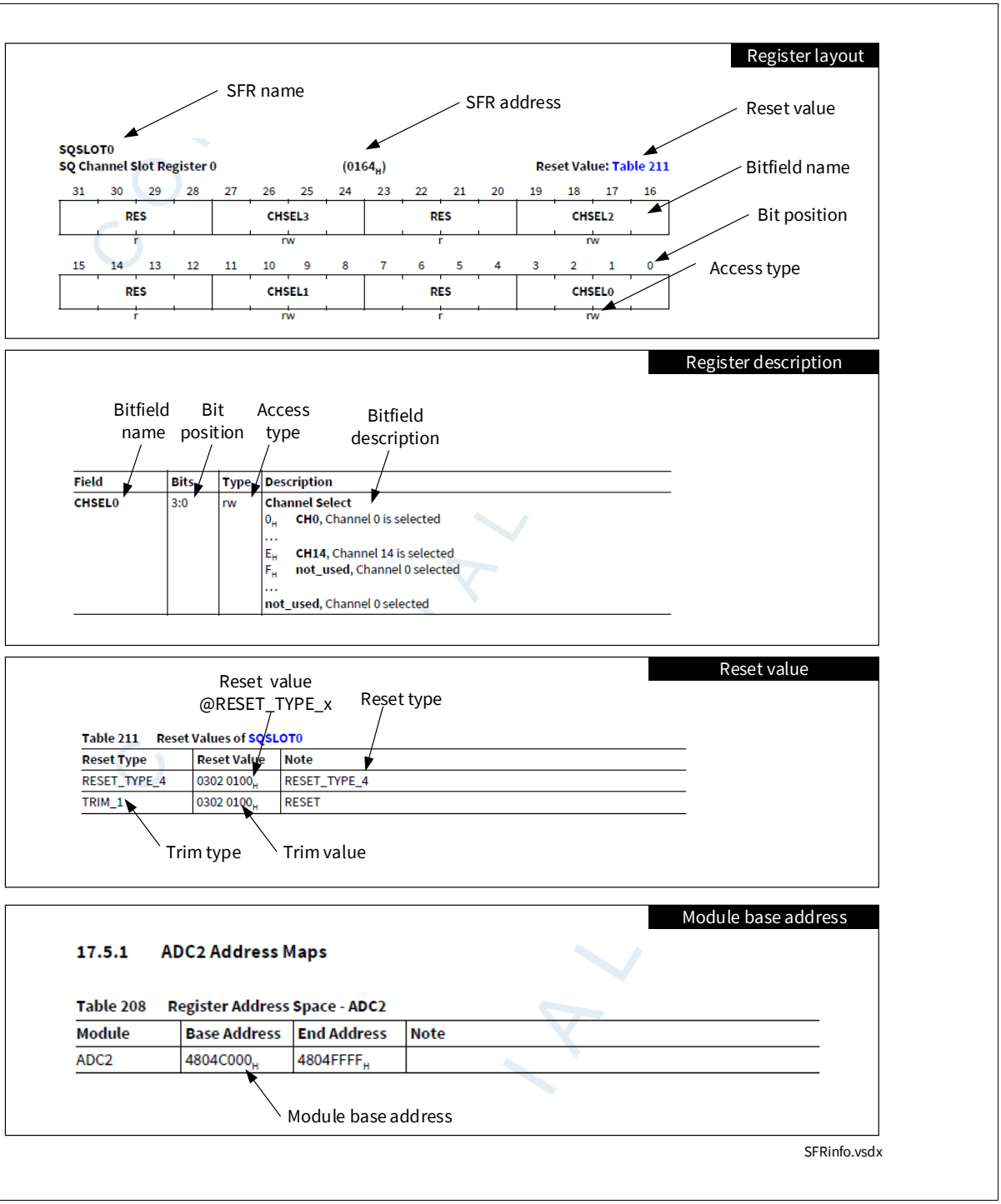


Figure 11 SFR information, example SQLOT0

The following register types are used within this device:

Table 41 Register access types

Bit-type	Meaning	Read access from SW (bus)	Write access from SW (bus)	Bit-set by HW	Bit-clear by HW	Bit-clear from SW
r	read-only of HW signal from SW, no register	yes	n/a	n/a	n/a	n/a
rc rh	bit set by HW, sticky, clear on read by SW	yes	no	yes	no	yes, by software read
rw	read-write from SW, not influenced by HW	yes	yes	no	no	(covered by SW write)
rwh	written by HW, read by SW, cleared by writing 0	yes	no	yes	yes	yes, by software write 0
rhxre	read by SW / set by HW on rising edge SW can set and clear the register by another register	yes	no	yes - by rising edge	no	yes, by external register
rhxr	read by SW / set by HW SW can set and clear the register by another register	yes	no	yes - by level	no	yes, by external register
rwph rwpht	rwph: read / write with write access protection from hardware (protection bit) rwpht: as rwph (testmode write access is not relevant in user mode)	yes	yes (when unprotected)	no	no	(covered by SW write)
rwc	read / write clear. Writing any value will clear the register	yes	yes - for cleaning	yes	yes	yes, by software write any value
rwx	read / write toggle. By writing (to another register) the register value is toggled	yes	yes - for toggling	no	no	(covered by SW write)
rwput	read / write with write access protection for usermode and testmode	yes	(not in user mode)	no	no	(not in user mode)
rwv	read / write virtual behavior defined in bitfield description	(see bitfield)	(see bitfield)	(see bitfield)	(see bitfield)	(see bitfield)
rwhisv	rw from SW, can be set by HW	yes	yes	yes	no	(covered by SW write)

Table 41 Register access types (cont'd)

Bit-type	Meaning	Read access from SW (bus)	Write access from SW (bus)	Bit-set by HW	Bit-clear by HW	Bit-clear from SW
rwpt	read-only in user mode (only writable in testmode and by firmware)	yes	(not in user mode)	no	no	(not in user mode)
w	clear on write 1, for interrupts (interrupt status clear bit) and sticky status registers (status clear)	no	no	no	no	yes, by software write 1

Write access to protected bits (rwp, rwpt, rwph, rwpht, rwput, r, rhxr, rhxre) may result in a Hardfault with following exceptions:

- In case the protection is inactive, no Hardfault is generated
- In case a protected bit is written together with an unprotected bit, no Hardfault is generated

Note: The bus access type WORD, HALFWORD and BYTE have to be respected accordingly.

3.12 Memory map

3.12.1 Memory map overview

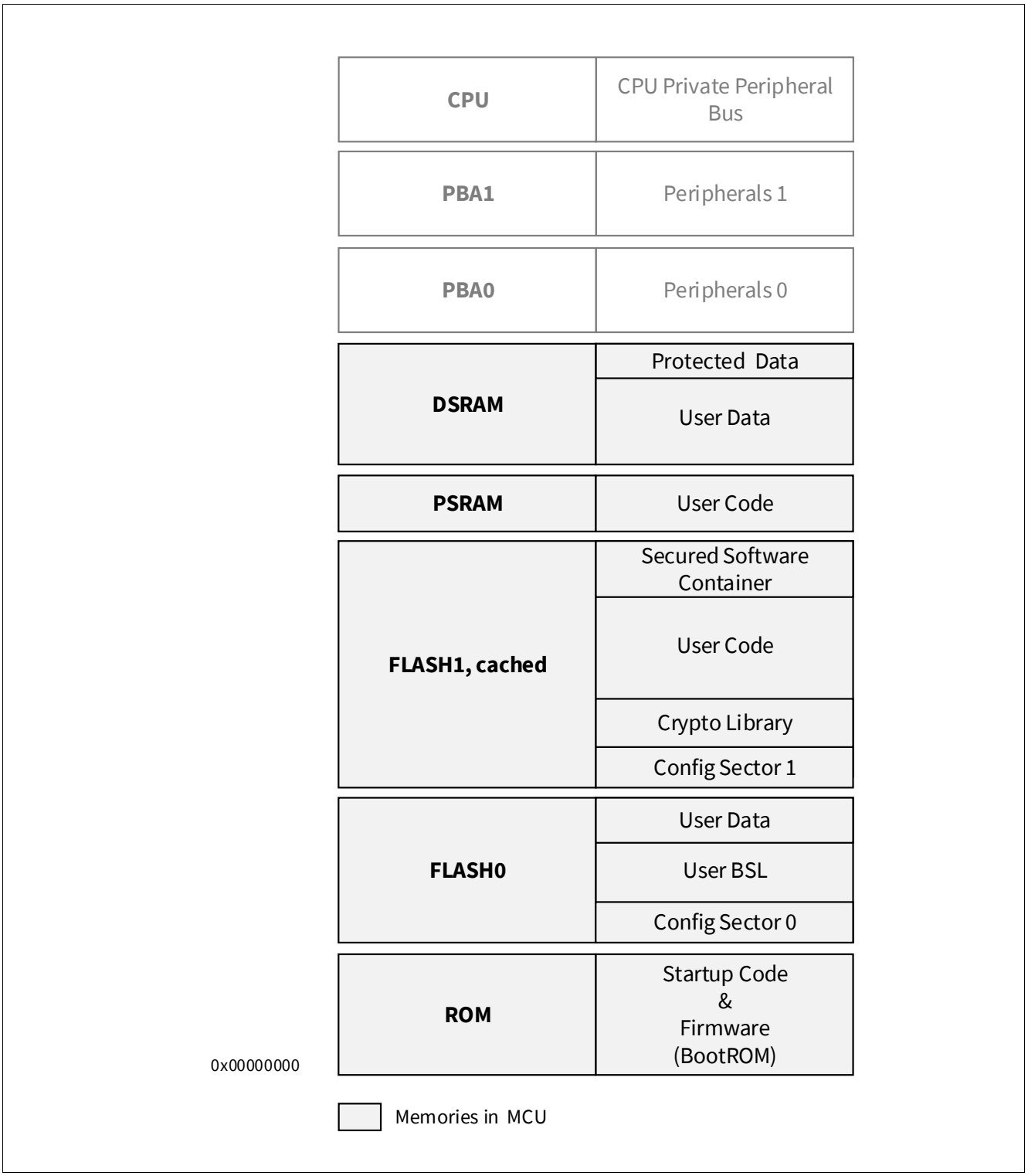


Figure 12 Memory map overview

3.12.2 Variants with 256 KB FLASH1, 32 KB RAM

Table 42 Bus matrix slave address mapping (256 KB FLASH1, 32 KB RAM)

Address range start	Address range end	Memory/Module in MCU	Description
0xE00F_F000	0xE00F_FFFF	CPU Private Peripheral Bus	ROM Table
0xE000_F000	0xE00F_EFFF		reserved
0xE000_E000	0xE00E_FFFF		NVIC
0xE000_3000	0xE00D_FFFF		reserved
0xE000_2000	0xE002_FFFF		FPB
0xE000_1000	0xE001_FFFF		DWT
0xE000_0000	0xE000_FFFF		reserved
0x6000_0000	0xDFFF_FFFF	reserved	n.a.
0x4800_0000	0x5FFF_FFFF	PBA1	Peripheral Bus 1
0x4000_0000	0x47FF_FFFF	PBA0	Peripheral Bus 0
0x1800_8000	0x3FFF_FFFF	reserved	n.a.
0x1800_7C00	0x1800_7FFF	DSRAM	Protected DSRAM Secure (1 KB)
0x1800_2000	0x1800_7BFF		User Data (23 KB)
0x1800_0000	0x1800_1FFF	PSRAM	Program memory, reserved for Secured Software Container if present
0x1204_0000	0x17FF_FFFF	reserved	n.a.
0x1200_2000	0x1203_FFFF	FLASH1	[1)]Secured Software Container (size is configurable)
			User Code (up to 248 KB)
0x1200_0000	0x1200_1FFF		[2)]Crypto Library
0x11FF_F000	0x11FF_FFFF		[3)]Config Sector of FLASH1 Key Storage
0x1100_8000	0x11FF_EFFF	reserved	n.a.
0x1100_0000 + UBSL_size	0x1100_7FFF	FLASH0	[4)]User data (EEPROM emulation up to 28 KB)
0x1100_0000	0x1100_0000 + UBSL_size - 1		[5)]User BSL (4 KB up to 32 KB)
0x10FF_F000	0x10FF_FFFF		[6)]Config Sector FLASH0 100TP
0x0000_8000	0x10FF_EFFF	reserved	n.a.
0x0000_0000	0x0000_7FFF	ROM	BootROM firmware

1) Default size is 0 KB. The Secured Software Container only available on devices with security features enabled.
2) No direct access from User Code, function calls provided with dedicated firmware APIs. The Crypto Library only available on devices with security features enabled.
3) No direct access from User Code, usage with with dedicated firmware APIs.

4) Default User Data size is 8 KB.
5) Default User BSL size is 24 KB.
6) No direct access from User Code, read/write access provided with dedicated firmware APIs.

3.12.3 Variants with 128 KB FLASH1, 16 KB RAM

Table 43 Busmatrix slave address mapping (128 KB FLASH1, 16 KB RAM)

Address range start	Address range end	Memory/Module in MCU	Description
0xE00F_F000	0xE00F_FFFF	CPU Private Peripheral Bus	ROM Table
0xE000_F000	0xE00F_EFFF		reserved
0xE000_E000	0xE00E_FFFF		NVIC
0xE000_3000	0xE00D_FFFF		reserved
0xE000_2000	0xE002_FFFF		FPB
0xE000_1000	0xE001_FFFF		DWT
0xE000_0000	0xE000_FFFF		reserved
0x6000_0000	0xDFFF_FFFF	reserved	n.a.
0x4800_0000	0x5FFF_FFFF	PBA1	Peripheral Bus 1
0x4000_0000	0x47FF_FFFF	PBA0	Peripheral Bus 0
0x1800_8000	0x3FFF_FFFF	reserved	n.a.
0x1800_7C00	0x1800_7FFF	DSRAM	Protected DSRAM Secure (1 KB)
0x1800_4000	0x1800_7BFF		reserved
0x1800_2000	0x1800_3FFF		User Data (23 KB)
0x1800_0000	0x1800_1FFF	PSRAM	Program memory, reserved for Secured Software Container if present
0x1202_0000	0x17FF_FFFF	reserved	n.a.
0x1200_2000	0x1201_FFFF	FLASH1	[1)]Secured Software Container (size is configurable)
			User Code (up to 120 KB)
0x1200_0000	0x1200_1FFF		[2)]Crypto Library
0x11FF_F000	0x11FF_FFFF		[3)]Config Sector of FLASH1 Key Storage
0x1100_8000	0x11FF_EFFF	reserved	n.a.
0x1100_0000 + UBSL_size	0x1100_7FFF	FLASH0	[4)]User data (EEPROM emulation up to 28 KB)
0x1100_0000	0x1100_0000 + UBSL_size - 1		[5)]User BSL (4 KB up to 32 KB)
0x10FF_F000	0x10FF_FFFF		[6)]Config Sector FLASH0 100TP

Table 43 Busmatrix slave address mapping (128 KB FLASH1, 16 KB RAM) (cont'd)

Address range start	Address range end	Memory/Module in MCU	Description
0x0000_8000	0x10FF_EFFF	reserved	n.a.
0x0000_0000	0x0000_7FFF	ROM	BootROM firmware

1) Default size is 0 KB. The Secured Software Container only available on devices with security features enabled.
2) No direct access from User Code, function calls provided with dedicated firmware APIs. The Crypto Library only available on devices with security features enabled.
3) No direct access from User Code, usage with with dedicated firmware APIs.
4) Default User Data size is 8 KB.
5) Default User BSL size is 24 KB.
6) No direct access from User Code, read/write access provided with dedicated firmware APIs.

3.12.4 PBA0 address mapping

The **Table 44** lists a superset of peripheral modules connected to PBA0 peripheral bus and their corresponding address mappings. The availability of the peripherals is product variant specific.

Table 44 PBA0 address map

Address range start	Address range end	Peripheral
0x4002_2000	0x47FF_FFFF	reserved (default slave)
0x4001_C000	0x4001_FFFF	LS (reserved for future use; default slave)
0x4001_8000	0x4001_BFFF	HS (reserved for future use; default slave)
0x4001_4000	0x4001_7FFF	GPT12
0x4001_0000	0x4001_3FFF	CSA
0x4000_C000	0x4000_FFFF	Bridge Driver
0x4000_8000	0x4000_BFFF	CCU7
0x4000_4000	0x4000_7FFF	SDADC
0x4000_0000	0x4000_3FFF	ADC1

3.12.5 PBA1 address mapping

The **Table 45** lists a superset of peripheral modules connected to PBA1 peripheral bus and their corresponding address mappings. The availability of the peripherals is product variant specific.

Table 45 PBA1 address map

Address range start	Address range end	Peripheral
0xF000_0000	0xF000_0FFF	ROM Table
0x4805_4000	0x5FFF_FFFF	reserved (default slave)
0x4805_0000	0x4805_3FFF	Math Div / Cordic (reserved for future use)
0x4804_C000	0x4804_FFFF	ADC2
0x4804_8000	0x4804_BFFF	Cache SFR
0x4804_4000	0x4804_7FFF	MemCtrl
0x4804_0000	0x4804_3FFF	FLASH1 SFR Register
0x4803_C000	0x4803_FFFF	FLASH0 SFR Register
0x4803_8000	0x4803_BFFF	ARVG

Table 45 PBA1 address map (cont'd)

Address range start	Address range end	Peripheral
0x4803_4000	0x4803_7FFF	DMA
0x4803_0000	0x4803_3FFF	GPIO
0x4802_C000	0x4802_FFFF	Timer21
0x4802_8000	0x4802_BFFF	Timer2
0x4802_4000	0x4802_7FFF	SSC1
0x4802_0000	0x4802_3FFF	SSC0
0x4801_C000	0x4801_FFFF	UART1
0x4801_8000	0x4801_BFFF	UART0
0x4801_4000	0x4801_7FFF	LIN
0x4801_0000	0x4801_3FFF	MultiCAN
0x4800_C000	0x4800_FFFF	CAN
0x4800_8000	0x4800_BFFF	PLL
0x4800_4000	0x4800_7FFF	SCU
0x4800_0000	0x4800_3FFF	PMU Failsafe

3.12.6 100TP address mapping

The 100TP contains device specific data that can be customized by the user. Each page of the 100TP can be programmed up to 100 times. during lifetime of the device. For details on the individual configuration entries in **Table 46** please refer to the Firmware User's Manual.

Table 46 100TP address mapping

Address range start	Address range end	Description
0x10FF_FC00	0x10FF_FC7F	[1)]100TP, page #0, contains PMU.START_CONFIG register configuration value
0x10FF_FC80	0x10FF_FCFF	[2)]100TP , page #1, contains configuration variable NVM_SA_WITH_PROT, the Service Algorithm configuration value
0x10FF_FD00	0x10FF_FD7F	100TP, page #2
0x10FF_FD80	0x10FF_FDFF	100TP, page #3
0x10FF_FE00	0x10FF_FE7F	100TP, page #4
0x10FF_FE80	0x10FF_FEFF	100TP, page #5
0x10FF_FF00	0x10FF_FF7F	100TP, page #6
0x10FF_FF80	0x10FF_FFFF	100TP, page #7

1) The first four bytes of page 0 contain user configuration of the PMU.START_CONFIG register.

2) Byte with offset 0xFB contains one byte of Service Algorithm configuration variable NVM_SA_WITH_PROT. By default NVM_SA_WITH_PROT is set to 0xA5. Any other value deactivates the Service Algorithm at startup if User Data section write protection is set.

4 BLDC driver application information

Figure 13 shows the TLE989x/TLE988x in an electric drive application setup controlling a BLDC motor.

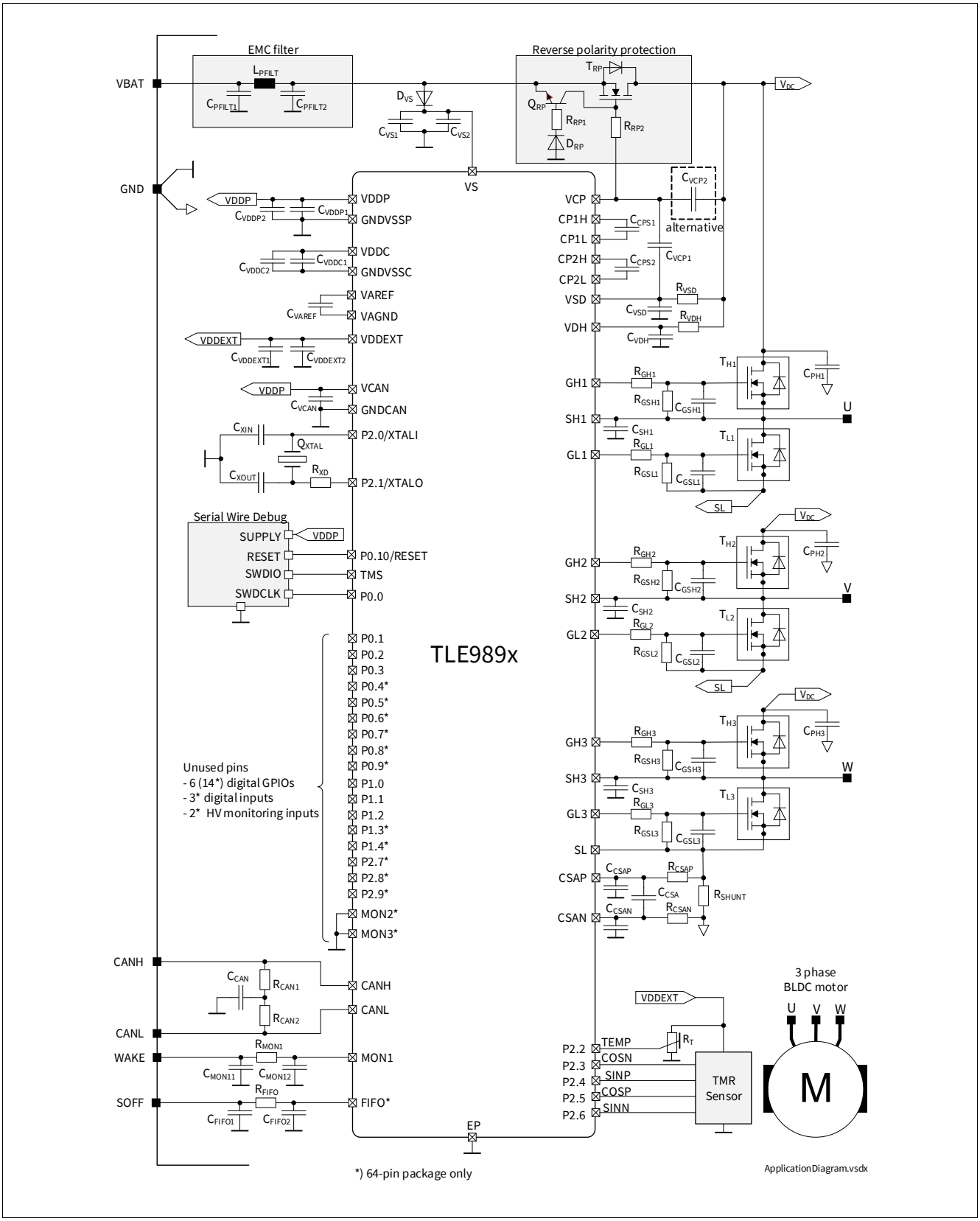


Figure 13 Simplified application diagram example for a BLDC system

Note: The following information is given as a hint for the implementation of the device only and shall not be regarded as a description or warranty of a certain functionality, condition or quality of the device. This is a very simplified example of an application circuit and bill of material. The function must be verified in the actual application.

Table 47 External components (BOM)

Symbol	Function	Component
D_{VS}	Reverse polarity protection diode	e.g. BAS52-02V
C_{VS1}	Decoupling capacitor at VS pin	see P_PMU_01_06
C_{VS2}	Buffer capacitor at VS pin	see P_PMU_01_07
C_{VDDP1}	Decoupling capacitor at VDDP pin	see P_PMU_03_22
C_{VDDP2}	Stability capacitor at VDDP pin	see P_PMU_03_22
$C_{VDDEXT1}$	Decoupling capacitor at VDDEXT pin	see P_PMU_05_13
$C_{VDDEXT2}$	Stability capacitor at VDDEXT pin	see P_PMU_05_13
C_{VCAN}	Decoupling capacitor at VCAN pin	see P_PMU_03_23
C_{VDDC1}	Decoupling capacitor at VDDC pin	see P_PMU_04_21
C_{VDDC2}	Decoupling capacitor at VDDC pin	see P_PMU_04_21
C_{VAREF}	Stability capacitor at VAREF pin	see P_ARVG_03_03
C_{CPS1}	Charge pump flying capacitor stage 1	application dependent, min. 100 nF
C_{CP2S}	Charge pump flying capacitor stage 2	application dependent, min. 100 nF
C_{VCP1}	Charge pump storage capacitor (placing option 1)	application dependent, min. 220 nF
C_{VCP2}	Charge pump storage capacitor (placing option 2)	application dependent, min. 220 nF
R_{MON1}	Resistor at MON1 pin for ISO pulses	application dependent, e. g. 1 kΩ
C_{MON11}	Pi Filter Capacitor at MONx	application dependent, e. g. 10 nF
C_{MON12}	Pi Filter Capacitor at MONx	application dependent, e. g. 1 nF
R_{FIFO}	Resistor at FIFO pin for ISO pulses	application dependent, e. g. 1 kΩ
C_{FIFO1}	Pi Filter Capacitor at FIFO pin	application dependent, e. g. 10 nF
C_{FIFO2}	Pi Filter Capacitor at FIFO pin	application dependent, e. g. 1 nF
C_{PH1}	DC link capacitor	application dependent, e. g. 680 µF
C_{PH2}	DC link capacitor	application dependent, e. g. 680 µF
C_{PH3}	DC link capacitor	application dependent, e. g. 680 µF
C_{CSA}	Filter capacitor	application dependent, e. g. 1 nF
R_{CSAN}	Filter resistor (optional)	application dependent, e. g. 12 Ω
R_{CSAP}	Filter resistor (optional)	application dependent, e. g. 12 Ω
C_{CSAN}	Filter capacitor (optional)	application dependent, e. g. 1 nF
C_{CSAP}	Filter capacitor (optional)	application dependent, e. g. 1 nF
R_{SHUNT}	Shunt resistor	application dependent, e. g. 5 mΩ
R_{VSD}	Limitation of reverse current due to transient (-2 V, 8 ms)	2 Ω

Table 47 External components (BOM) (cont'd)

Symbol	Function	Component
C_{VSD}	Capacitor	1 µF
R_{VDH}	Filter resistor	optional, e. g. 1 kΩ
C_{VDH}	Filter capacitor	optional, e. g. 100 nF
$R_{GH1/2/3}$	Resistor	optional, 2 Ω
$R_{GL1/2/3}$	Resistor	optional, 2 Ω
$R_{GSH1/2/3}$	Resistor	optional, 100 kΩ
$R_{GSL1/2/3}$	Resistor	optional, 100 kΩ
$C_{GSH1/2/3}$	Capacitor	optional, 4.7 nF
$C_{GSL1/2/3}$	Capacitor	optional, 4.7 nF
$T_{H1/2/3}$	N-channel MOSFET	e.g. IPC70N04S5-4R6
$T_{L1/2/3}$	N-channel MOSFET	e.g. IPC70N04S5-4R6
T_{RP}	Reverse polarity protection MOSFET	e.g. IPC70N04S5-4R6
Q_{RP}	Reverse polarity protection transistor	e.g. BC817
R_{RP1}	Reverse polarity protection resistor 1	10 kΩ
R_{RP2}	Reverse polarity protection resistor 2	3.3 kΩ
D_{RP}	Reverse polarity protection circuit diode	e.g. BAS52-02V
L_{PFILT}	EMC filter coil	e.g. 4.7 µH
$C_{PFILT1/2}$	EMC filter capacitor	e.g. 22 µF
$C_{SH1/2/3}$	Capacitor	optional
R_{SH2}	Resistor	optional
R_{SH3}	Resistor	optional
Q_{XTAL}	Crystal or ceramic resonator	optional , e. g. NG3225GA, 16 MHz
R_{XD}	Damping resistor	optional, e. g. 330 Ω
C_{XIN}	Capacitor	optional, e. g. 0 Ω
C_{XOUT}	Capacitor	optional, e. g. 4.7 pF
C_{CAN}	Capacitor	optional, e. g. 4.7 nF
$R_{CAN1/2}$	Resistor	optional, e. g. 62 Ω
R_T	Thermal resistor (e. g. NTC)	optional
TMR	TMR sensor	optional, e. g. TLE5501

4.1 Further application information

- Please contact Infineon Technologies for information regarding the pins FMEA and Safety Manual
- For further information, please follow the link: **https://www.infineon.com/motixmcu**

5 Power Management Unit (PMU)

5.1 Features overview

The Power Management Unit (PMU) manages all functions related to the device power supply and its supervision. The PMU controls all operating mode transitions and ensures a fail-safe behavior.

The PMU provides following features:

- State control
  - Operating state machine (Start-up, Active, Stop, Sleep and Fail-Sleep)
  - Voltage regulator control
  - Master clock generation (MCLK) acting as PMU clock
  - Reset management controlling the reset behavior of the entire device
  - Bi-directional reset pin (P0.10/RESET) as reset input and reset output indicating an internally generated reset
  - Wake-up control for wake-up in Stop/Sleep modes via MON, CAN, BDRV, GPIOs, cyclic timer
- Voltage regulators
  - Linear voltage regulators (VMSUP) for internal supply of the device
  - Linear voltage regulator (VDDP, 5 V typ.) for GPIO and CAN transceiver supply
  - Linear voltage regulator (VDDC, 1.5 V typ.) for internal digital logic supply
  - Reference voltage generation (VAREFSUP)
  - Linear voltage regulator (VDDEXT, 5 V typ.) for external sensors supply
- Fail-safe supervision
  - System monitor, monitoring of fail-safe relevant signals
  - Supply monitor, monitoring of fail-safe relevant voltages
  - Safe reference clock (REF_CLK) and clock watchdog for monitoring of the MCLK
  - Fail-safe input/output (FIFO pin) for external safe shutdown request or indication
  - Fail-safe window watchdog (FS_WDT) for monitoring the CPU execution timing
  - Safe shutdown mechanism to bring the bridge driver (BDRV) into a safe off-state
- Retention memory (GPUDATA with 96 bits) for data storage in Sleep and Fail-sleep modes

5.2 Block diagram

The PMU module consists of the following major functional parts:

- State control
- Voltage regulators
- Fail-safe supervision
- Retention memory

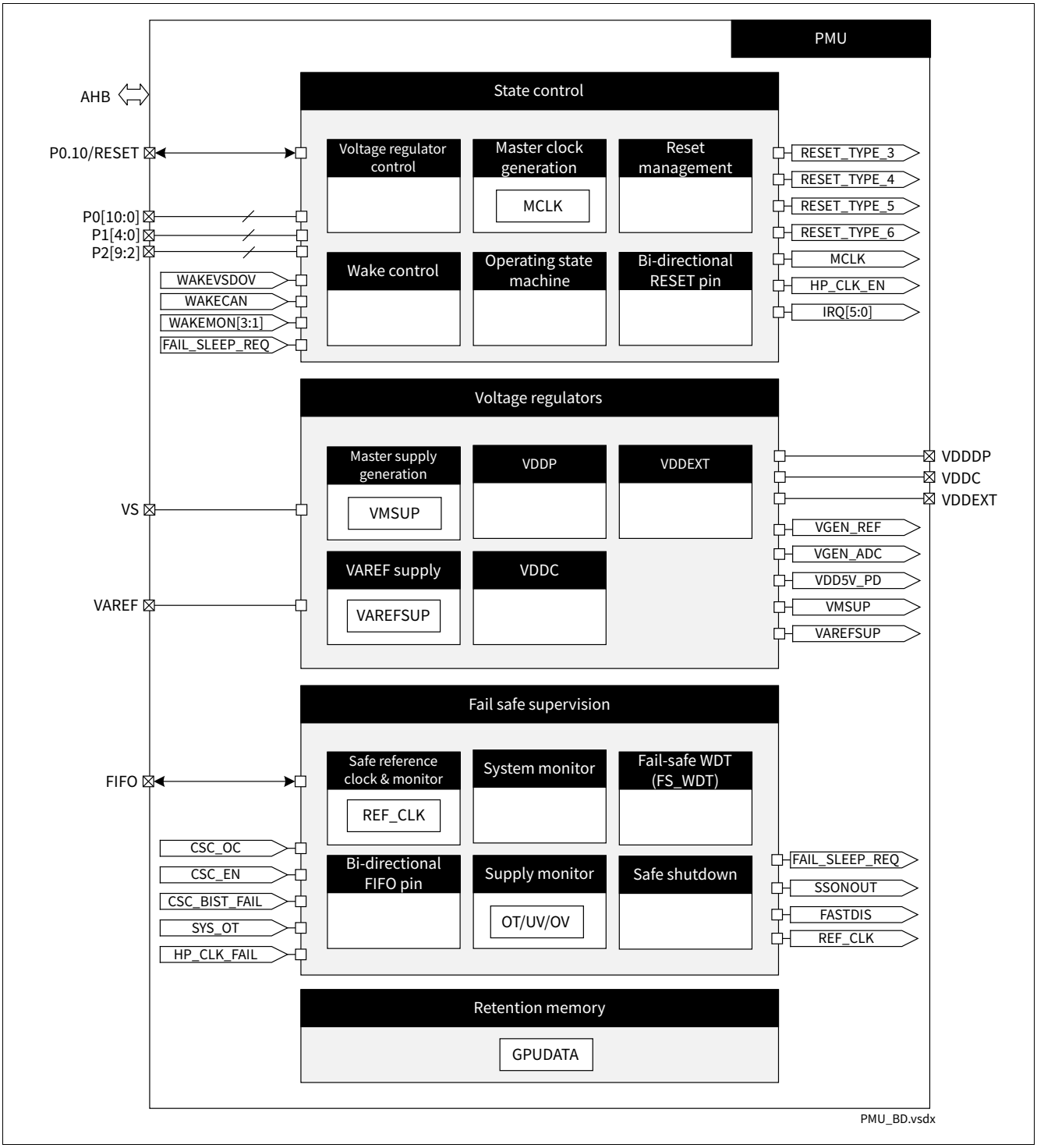


Figure 14 Block diagram PMU

5.3 Toplevel signals

The PMU provides voltage, clocks, reset and control signals from/to different modules. For more details, please refer to the following tables.

Table 48 PMU power domain

Signal	Direction	Description
VS	Power	Input supply voltage (at pin VS)
VDDP	Power	5 V I/O domain supply (at pin VDDP)
VDDC	Power	1.5 V core domain supply (at pin VDDC)
VGEN_REF	Power	Reference voltage for voltage regulators (internal reference)
VDD5V_PD	Power	5 V peripherals always-on supply (internal supply)
VMSUP	Power	1.5 V digital peripherals always-on supply (internal supply)
VAREFSUP	Power	5.5 V ARVG supply
VDDEXT	Power	5 V external sensor supply (at pin VDDEXT)

Table 49 PMU clock and reset domain signals

Signal	Direction	Description	From/To
P0.10/RESET	Input/Output	Bi-directional external reset signal	GPIO
RESET_TYPE_[6:3]	Output	Internal module reset signals	SCU
MCLK	Output	Master clock (20 MHz typ.)	SCU, CSA, CSC, CANTRX
REF_CLK	Output	Reference clock (100 kHz typ.)	SCU

Table 50 PMU bus interface and interrupt request signals

Signal	Direction	Description	From/To
IRQ[5:0]	Output	Interrupt service request signals	SCU
AHB slave bus I/F	Input/Output	Slave AHB-lite bus interface	MCU

Table 51 PMU fail-safe control signals

Signal	Direction	Description	From/To
FIFO	Input/Output	Fail-safe request input/output flag	I/O
SSONOUT	Output	Bridge driver safe enable signal	BDRV
FASTDIS	Output	Bridge driver safe-shutdown pre-discharge signal	BDRV
HP_CLK_EN	Output	High-precision clock enable signal	SCU
HP_CLK_FAIL	Input	High-precision clock monitoring signal	SCU
CSC_EN	Input	CSC comparator enable signal after self-test	CSA, CSC
CSC_OC	Input	Current sense monitor overcurrent signal	CSA, CSC
CSC_BIST_FAIL	Input	Current sense monitor BIST fail signal	CSA, CSC
SYS_OT	Input	Sensor overtemperature signal	ADC2
VGEN_REF	Reference	Central PMU bandgap reference voltage	ARVG

Table 51 PMU fail-safe control signals (cont'd)

Signal	Direction	Description	From/To
VGEN_ADC	Output	Central PMU bandgap reference voltage measured at ADC2	ADC2
VAREF	Input	Analog reference voltage	VAREF

Table 52 PMU wake request signals

Signal	Direction	Description	From/To
WAKEVSDOV	Input	VSD wake event	BDRV
WAKECAN	Input	CAN wake event	CANTRX
WAKEMON3	Input	MON3 wake event	MON
WAKEMON2	Input	MON2 wake event	MON
WAKEMON1	Input	MON1 wake event	MON
P[10:0]	Input	GPIO P0.x wake event	GPIO
P1[4:0]	Input	GPIO P1.x wake event	GPIO
P2[9:2]	Input	GPIO P2.x wake event	GPIO

5.4 Interrupts

Events

The PMU has six interrupt request events:

- VDDP undervoltage warning (VDDP_STS.UVWARN_IS)
- VDDP overvoltage (VDDP_STS.OV_IS)
- VDDC undervoltage warning (VDDC_STS.UVWARN_IS)
- VDDC overvoltage (VDDC_STS.OV_IS)
- VDDEXT undervoltage (VDDEXT_STS.UV_IS)
- VDDEXT overtemperature (VDDEXT_STS.OT_IS)

Interrupts

The interrupt request event is indicated in corresponding status bit *_STS.<EVENT>_IS.

The interrupt request can be set alternatively by software via *_STS_SET.<EVENT>_SET.

The interrupt request can be cleared by software via *_STS_CLR.<EVENT>_CLR.

The interrupt request can be enabled via *_IRQEN.<EVENT>_IEN.

The PMU interrupt requests are ored and propagated to the interrupt node pointer in the SCU.

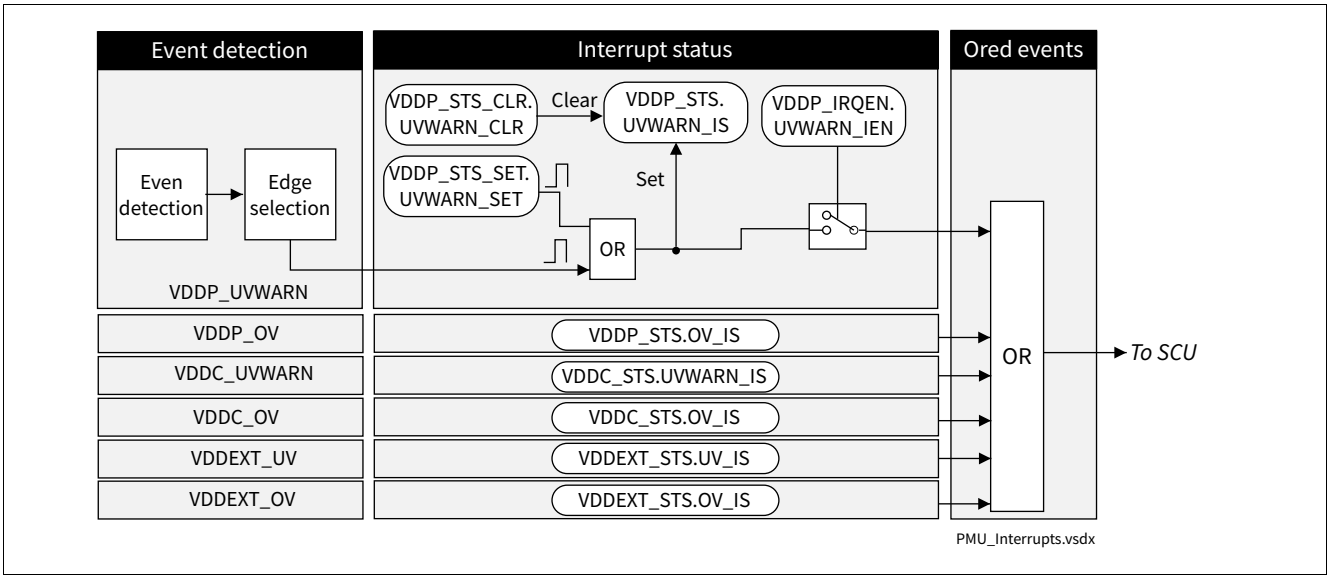


Figure 15 Interrupt and status registers

5.5 Signal flow

Event signal flow

The signal flow for events which are generated within the PMU is shown in the example figure below.

- A circuit (here Regulator x) generates an output voltage VDDx. Optionally the circuit has a local monitor which indicates a status in STS and IS
- The Supply monitor monitors VDDx and detects EVENT_x (e.g. undervoltage). The status is indicated in FS_STS
- The System monitor requests the transition to Fail-sleep mode via FAIL_SLEEP_REQ in case of a severe error on the supply VDDx. The status is indicated WAKE_FAIL_STS
- The Reset management generates the RESET_Types according to EVENT_x. The reset status is indicated in RESET_STS
- The Wake control generates the wake request via WAKE_x according to EVENT_s. The wake status is indicated in WAKE_STS
- The Safe shutdown logic activates the shutdown according to FS_STS via the signals SSONOUT and FASTDIS
- The Operating state machine changes the operating state according to the signals FAIL_SLEEP_REQ, EVENTx and WAKEx

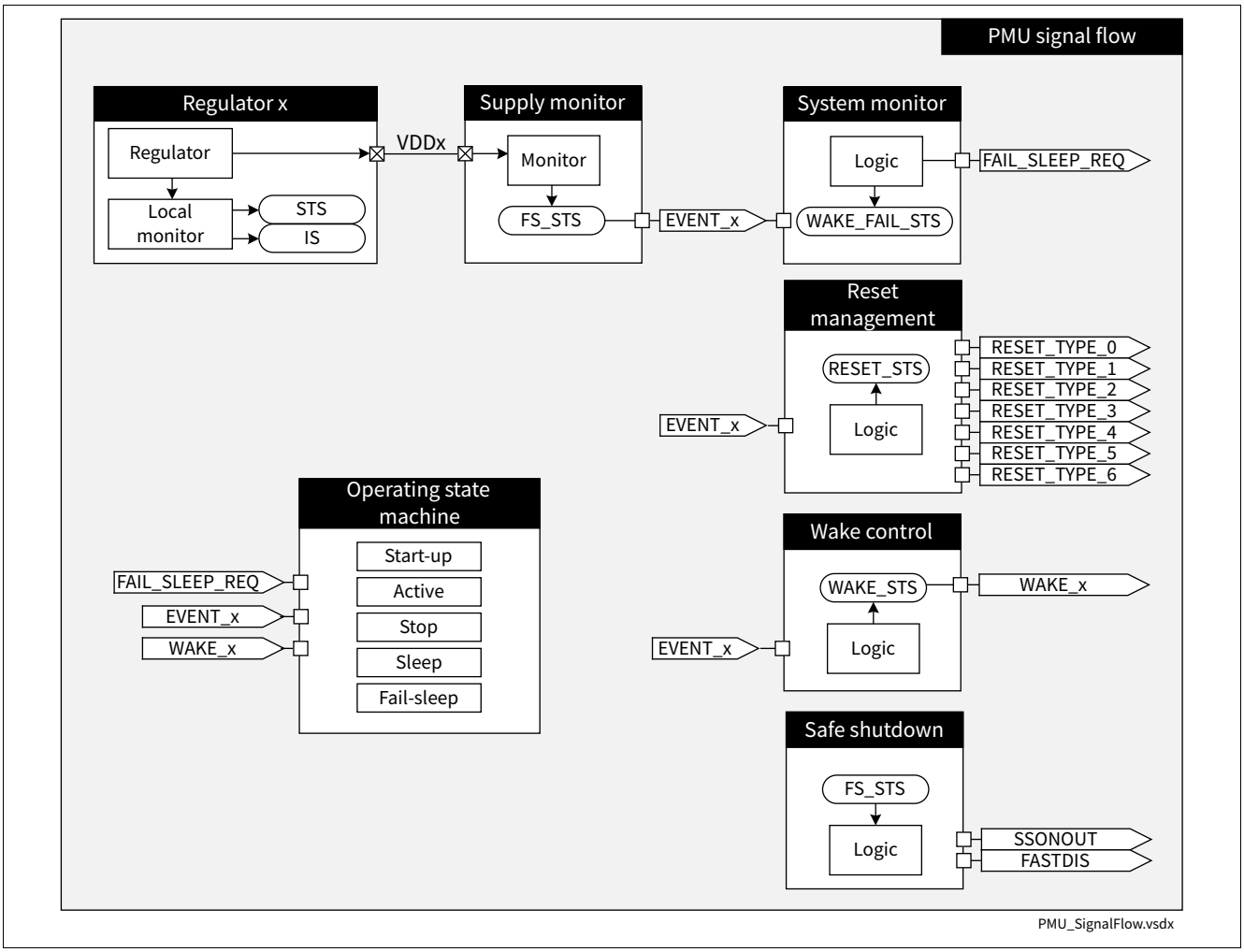


Figure 16 PMU signal flow

Interrupt request signal flow

The signal flow for interrupt requests (e.g. undervoltage warning) which are generated within the PMU is shown in the figure below.

The interrupt status from the Local monitor is indicated in IS and request and interrupt IRQ_x. The interrupt node pointer in the SCU routes the request to the NVIC and CPU accordingly.

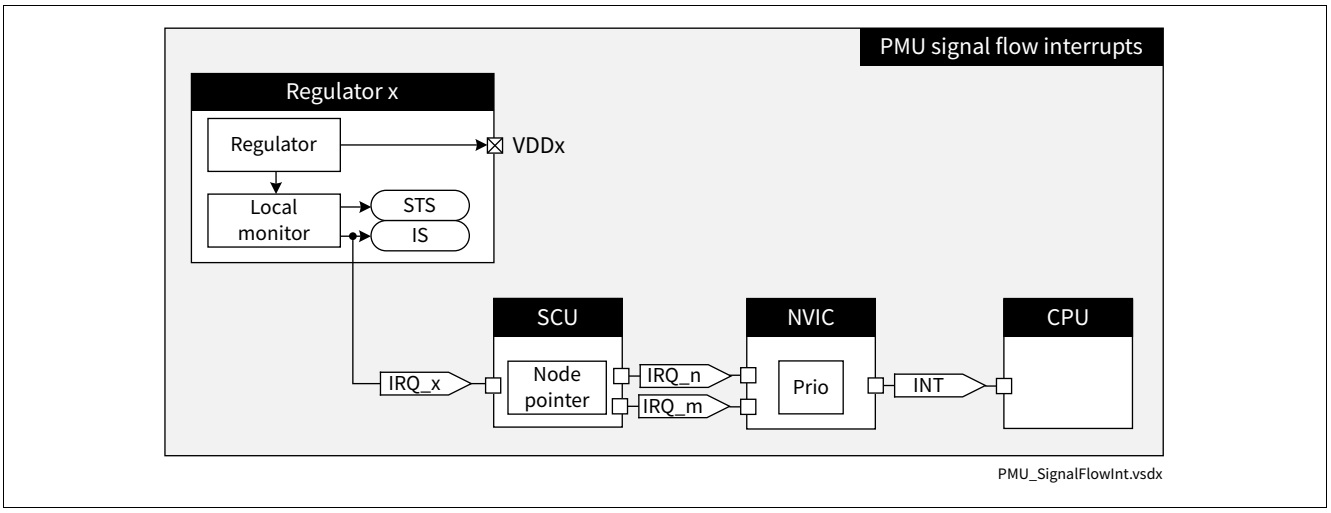


Figure 17 PMU signal flow for interrupt requests

5.6 State control

5.6.1 Operating state machine

The following state diagram shows the PMU operating states.

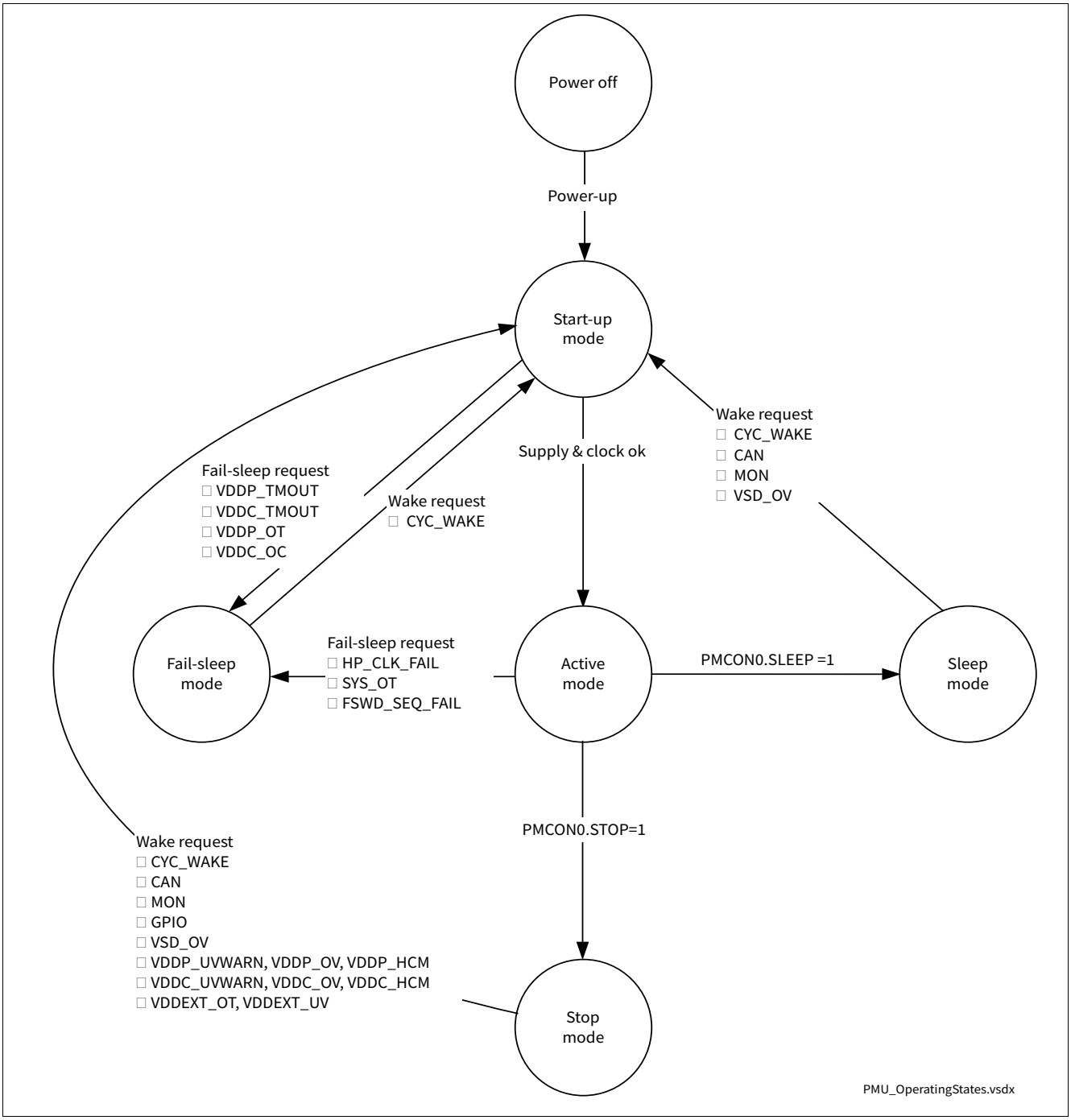


Figure 18 PMU operating state machine

5.6.1.1 Start-up mode

Start-up mode is an intermediate transition phase from Power off to Active. The PMU controls the Start-up mode:

- The VDDP and VDDC ramp up
- The master clock (MCLK), the safe reference clock (REF_CLK) and the clock watchdog (FS_MCLK_WD) start
- The associated monitoring functions are active
- The safe shutdown and the FO output are active
- The reset management releases the RESET_TYPE_3/4/5/6

The transition from Start-up to Active mode is done by the PMU automatically. The startup time is defined in parameter $t_{startup}$.

The transition from Start-up to Fail-sleep is done in case the PMU cannot startup the system properly and a Fail-sleep request event has occurred:

- VDDP_TMOUT
- VDDC_TMOUT
- VDDP_OT
- VDDC_OC

5.6.1.2 Active mode

In Active mode the device is fully operational. The current consumption at VS input is specified as I_{Vs_act}, $I_{Vs\_act\_CAN}$. The PMU controls the Active mode:

- The PMU monitors the input voltage (VS), the output voltages/currents and supervises the master clock (MCLK) and the Fail-safe status
- The start-up firmware in BootROM does the initial device settings and jumps to the user code, see the chapter “Device startup” in the “**Product definitions**” chapter
- The user software takes control of the system
  - The Fail-safe WDT (FS_WDT) has to be initialized, see “**Fail-safe watchdog (FS_WDT)**”
  - The Safe shutdown is activated until it is released by the user software, see “**Safe shutdown**”

The transition from Active to Stop mode is initiated by setting bit SCU.PMCON0.STOP.

The transition from Active to Sleep mode is initiated by setting bit SCU.PMCON0.SLEEP.

The transition from Active to Fail-sleep is done in case a severe system failure is detected and a Fail-sleep request event has occurred:

- HP_CLK_FAIL
- SYS_OT
- FSWD_SEQ_FAIL

5.6.1.3 Stop mode

The Stop mode is a power-saving mode with a reduced current consumption (refer to I_{Vs_stp}). In Stop mode some peripherals are un-powered and lose the context, the MCU sub-system is supplied but clock is stopped (see the chapter “Operation mode transition handling” in the “SCU” chapter).

The PMU controls the Stop mode:

- The VDDP regulator is active
- The VDDC regulator is active and its output voltage can be scaled down in Stop mode for reduced leakage current
- The overtemperature detection is disabled
- The undervoltage monitors are enabled
- The master clock (MCLK) is stopped
- The safe reference clock (REF_CLK) is enabled in case the cyclic wake/sense is enabled otherwise it is stopped
- The FS_WDT is stopped and reset
- The GPU data are retained
- The RAM is supplied and data are retained
- The GPIOs are supplied and the programmed configuration is kept
- The FIFO pin is active
- The Safe shutdown is active
- The WAKE requests from the enabled wake-up sources are handled

The transition from Stop to Start-up and then to Active mode is initiated by a wake event (WAKE request) within the wake-up time (t_{stpex}). The wake-up sources can be configured, refer to chapter “**Wake control**”.

Active to Stop mode

- The Sleep mode entry sequence is triggered via SCU.PMCON0.STOP=1

Stop to Active mode

- The wake-up is triggered by an enabled wake event, see “**Wake control**”

Programmer’s guide for Stop entry and wake-up

The Stop mode entry sequence is defined in **Table 53**.

In case the PLL is locked on an external crystal, the clock system has to be re-configured before entry and recovered after entry.

After wake-up the device starts from the program counters entry location. The steps for the recovery from Stop mode can be found in **Table 54**.

Table 53 Stop entry sequence

Step	Description	Code sequence
1	Chose RAM reduction (optional)	PMU.WAKE_CTRL.VDDC_RED_EN = 1;
2	Clear all wake-up status flags	PMU.WAKE_STS_CLR = 0xFFFF FFFF;
3	Configure the wake-up source accordingly	WAKE_CTRL.<XX>_WAKE_EN = <user_param>; CYC_CTRL.CYC_SENSE_EN= <user_param>;

Table 53 Stop entry sequence (cont'd)

Step	Description	Code sequence
4	Select HP_CLK as system clock	CLKSEL.SELSYS0 = 00b; CLKSEL.SELSYS1 = 00b;
5	Disable XTAL_OSC	XTALCON.XWDGEN=0; XTALCON.XPD =1;
6	Disable all interrupts Disable NMIPLLs	NVIC_ICER = 0xFFFF FFFF;
7	Configure GPIOs to a default state where no current is drawn, e.g. input with pull-up/pull-down enabled	P0_PUD = 0xXXXX; /* X = user value */ P0_DIR = 0x0000; P1_PUD = 0xXXXX; / *X = user value */ P1_DIR = 0x0000; P2_PUD = 0xXXXX; / *X = user value */ P2_DIR = 0x0000;
8	Request Stop mode	SCU.PMCON0.STOP = 1;
9	Free up pipeline	__NOP(); __NOP(); __NOP();
10	Set send event (handshake SCU-CPU)	__SEV();
11	Clear send event (handshake SCU-CPU)	__WFE();
12	Enter CPU DEEPSLEEP	__WFE();

Table 54 Stop wake-up sequence

Step	Description	Code sequence
1	Give CPU something simple	__NOP(); __NOP(); __NOP();
2	Check device status and act accordingly	check WAKE_FAIL_STS check RESET_STS check FS_STS (optional)
3	Check wake status and act accordingly	check WAKE_STS
4	Reconfigure clock system	configure CLKSEL configure XTAL configure PLL lock PLL
5	Reconfigure NVIC as before entry	NVIC_ICER = value_before_entry;
6	Reconfigure GPIOs as before entry	P0_PUD = 0xXXXX; / *X = user value */ P0_DIR = 0x0000; P1_PUD = 0xXXXX; / *X = user value */ P1_DIR = 0x0000; P2_PUD = 0xXXXX; / *X = user value */ P2_DIR = 0x0000;

Table 54 Stop wake-up sequence (cont'd)

Step	Description	Code sequence
7	Initialize FS_WDT	see "**Fail-safe watchdog (FS_WDT)**"
8	Initialize SSO path	see "**Safe shutdown**", Release the safe shutdown mechanism

5.6.1.4 Sleep mode

The Sleep mode is the power-saving mode with the lowest power consumption (refer to the I_{Vs_slp}). All peripherals and the MCU-subsystem are un-powered and lose the context, see the chapter "Operation mode transition handling" in the "SCU" chapter.

The PMU controls the Sleep mode:

- The main supply (VMSUP and VDD5V_PD) is active
- Overtemperature detection is disabled
- VMSUP under-, overvoltage monitors are enabled
- The master clock (MCLK) is stopped
- The safe reference clock (REF_CLK) is enabled in case the cyclic wake/sense is enabled otherwise it is stopped
- FS_WDT is stopped and reset
- The RAM is un-supplied and data are lost
- The GPU data are retained
- The GPIOs are un-supplied and floating
- The Safe shutdown is active
- The FIFO pin is active
- The WAKE requests from the enabled wake-up sources are handled

The transition from Sleep to Start-up and then to Active mode is initiated by a WAKE request within the wake-up time (t_{slpex}). The wake-up sources can be configured, refer to chapter "**Wake control**".

Active to Sleep mode

- The Sleep mode entry sequence is triggered via SCU.PMCON0.SLEEP =1

Sleep to Active mode

- The wake-up is triggered by an enabled wake event, see "**Wake control**"

Programmer's guide for Sleep mode entry and wake-up

The Sleep mode entry sequence is defined in **Table 55**.

The steps for the recovery from Sleep mode can be found in **Table 56**.

Table 55 Sleep entry sequence

Step	Description	Code sequence
1	Configure the wake-up source accordingly	WAKE_CTRL.<XX>_WAKE_EN = <user_param>; CYC_CTRL.CYC_SENSE_EN= <user_param>;
2	Clear all wake-up status flags	WAKE_STS_CLR = 0xFFFF FFFF;

Table 55 Sleep entry sequence (cont'd)

Step	Description	Code sequence
3	Configure a wake-up source, e.g. wake-up over CAN	e.g. WAKE_CTRL.CAN_WAKE_EN;
4	Disable all interrupts	NVIC_ISER = 0x00000000;
5	Optional: configure GPIOs to input with pull-up/pull-down arrangement accordingly to avoid spikes when GPIOs are powered off slowly	
6	Optional: write wake-up indicator to retention memory	GPUDATAx = <user_param>;
7	Request Sleep mode	SCU.PMCON0.Sleep =1;
8	Wait for Sleep entry	while(1);

Table 56 Sleep recovery sequence

Step	Description	Code sequence
1	Check device status and act accordingly	check WAKE_FAIL_STS check RESET_STS check FS_STS (optional)
2	Check wake status and act accordingly	check PMU.WAKE_STS
3	Optional: check wake-up indicator	Read GPUDATAx and branch accordingly
4	Execute user defined bootup sequence at reset vector	User defined code and device init

5.6.1.5 Fail-sleep mode

The Fail-sleep mode is intended to protect the device against failures that cause high power dissipation and can potentially lead to device destruction (e.g. overtemperature). All peripherals and the MCU-subsystem are un-powered and lose the context.

The PMU controls the Fail-Sleep mode:

- The main supply (VMSUP and VDD5V_PD) is active
- Overtemperature detection is disabled
- VMSUP under-, overvoltage monitors are enabled
- The master clock (MCLK) is stopped
- The safe reference clock (REF_CLK) is enabled
- FS_WDT is reset
- The RAM is un-supplied and data are lost
- The GPU data are retained
- The GPIOs are un-supplied and floating
- The Safe shutdown is active
- The FIFO pin is active is active
- The WAKE request CYC_WAKE is handled

The transition from Fail-sleep to Start-up is triggered by the cyclic wake request from the cyclic timer (CYC_WAKE). The off time is fixed to approximately 1024 ms. The PMU restarts the device periodically until the

causing failure condition disappears. After a successful restart, the causing Fail-sleep condition is indicated WAKE_FAIL_STS, see "**System monitor**".

Programmer's guide for Fail-sleep mode recovery sequence

The steps for the recovery from Sleep mode can be found in following table.

Table 57 Fail-sleep recovery sequence

Step	Description	Code sequence
1	Check device status and act accordingly	check WAKE_FAIL_STS check RESET_STS check FS_STS (optional)
2	Check wake status and act accordingly	check WAKE_STS
3	Optional: check wake-up indicator	Read GPUDATAx and branch accordingly
4	Execute user defined bootup sequence at reset vector	User defined code and device init

5.6.2 Voltage regulator control

The operating state machine controls the voltage regulators accordingly. See following table.

Table 58 Supply generation control in operating modes

Operating Mode	VMSUP	VDDP	VDDC	VDDEXT
Power-off	OFF	OFF	OFF	OFF
Start-up	ON	starting up	starting up	OFF
Active	ON	ON	ON	ON/OFF (configurable)
Stop	ON	ON	ON	ON in cyclic sense (configurable)
Sleep	ON	OFF	OFF	OFF
Fail-sleep	ON	OFF	OFF	OFF

5.6.3 Master clock generation

The master clock (MCLK, f_{MCLK}) is the base clock for the operating state machine. It is also the clock source for the FS_WDT.

The operating state machine controls the master clock generation accordingly. See following table.

Table 59 Master clock in different operating modes

Operating mode	MCLK
Power-off	OFF
Start-up	ON
Active	ON
Stop	OFF
Sleep	OFF
Fail-sleep	OFF

5.6.4 Bi-directional RESET pin

The RESET pin is a bi-directional GPIO and acts as a reset input (input path) as well as a reset indication output (output path).

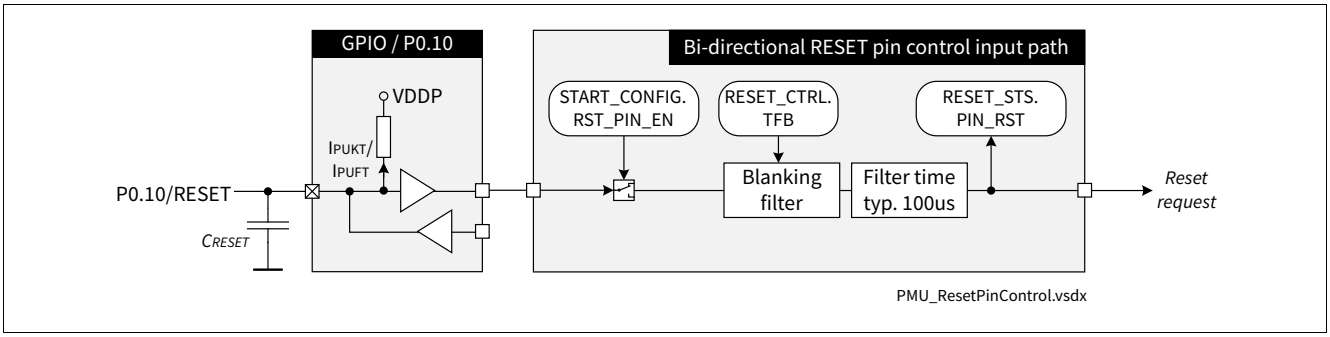


Figure 19 Bi-directional reset pin control input path

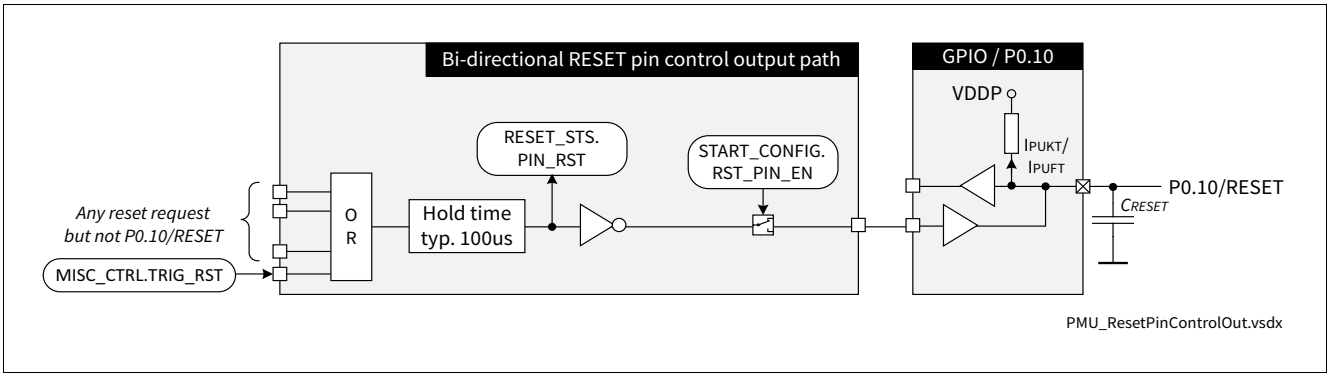


Figure 20 Bi-directional reset pin control output path

Configuration

By default, the RESET pin is configured as a push-pull GPIO port. For the reset functionality the bit START_CONFIG.RST_PIN_EN has to be set. The start-up firmware can set this bit already before the user code starts if the 100TP memory is configured accordingly.

When configured as RESET pin, the GPIO is configured in open drain mode with pull-up device enabled.

In order to avoid a reset deadlock situation due to a slow rising edge (caused by the external capacitor C_{RESET}), a blanking time can be programmed via RST_CTRL.TFB.

Control

A RESET pin output pulse is generated by setting bit MISC_CTRL.TRIG_RST.

Status

A RESET pin event is indicated in RESET_STS.PIN_RST. The status has to be cleared by software via RESET_STS_CLR.PIN_RST_CLR.

RESET pin behavior as input

A high to low edge at P0.10/RESET starts the blanking filter. A fixed filter time (t_{filt_RESET}) follows the blanking time. A new P0.10/RESET pulse is ignored while the filter time is active.

RESET pin behavior as output

Any reset request event (except an externally applied P0.10/RESET event itself) triggers a low pulse at P0.10/RESET for a hold time of typ. 100 us.

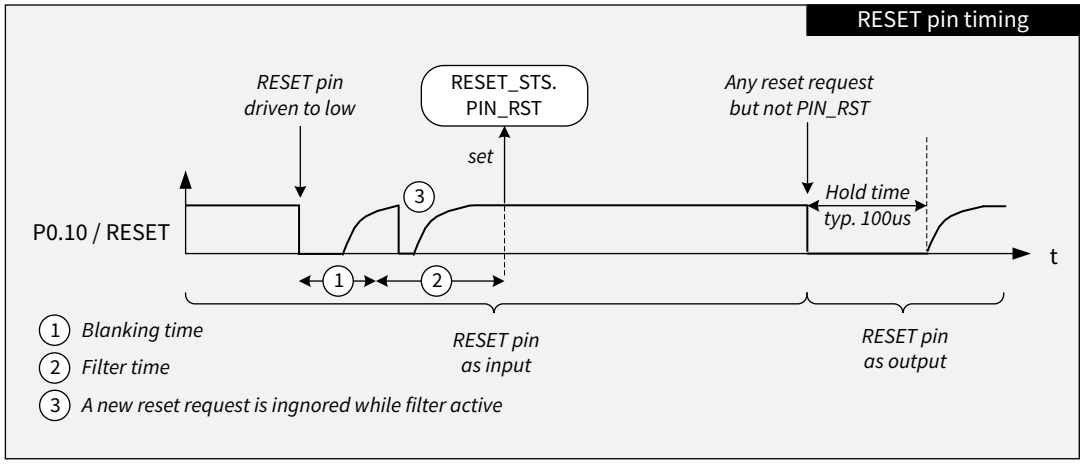


Figure 21 RESET pin timing

5.6.5 Reset management

The Reset management controls the reset behavior of the entire device.

The logic of the device is controlled via SFRs. SFRs need to be reset when the operating mode changes or an exceptional event occurs. Depending on the power-domain and function of the SFR, different RESET_TYPEs and a reset-release scheme are necessary. The assignment of reset source events to one or more RESET_TYPEs is handled by the reset management.

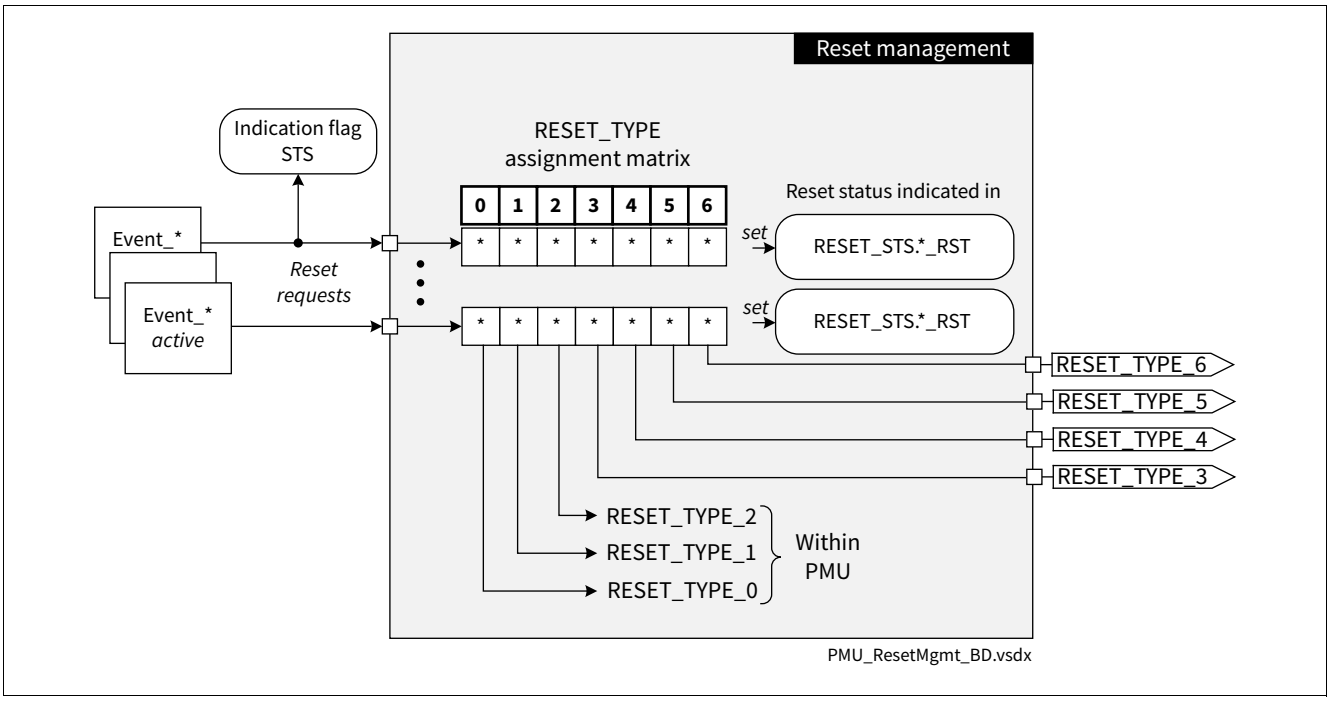


Figure 22 Reset management block diagram

Reset request

A reset source event can request a reset. Usually the reset source event is indicated in a corresponding status flag outside the Reset management (for example VDDC undervoltage event in FS_STS.VDDC_UV_STS). Some reset source events have no corresponding indication flag (for example CPU lockup event).

RESET_TYPE assignment

The Reset management assigns one or more RESET_TYPEs to the reset source according to a defined matrix.

Reset indication

The reset status is indicated in the corresponding RESET_STS.<EVENT>_RTS bit (<EVENT> as event name). The reset status bits can be set or cleared by software via RESET_SET.<EVENT>_SET or RESET_CLR.<EVENT>_CLR.

Reset category

The BootROM firmware uses a different classification of resets: cold, warm, hot reset (see **Figure 23**).

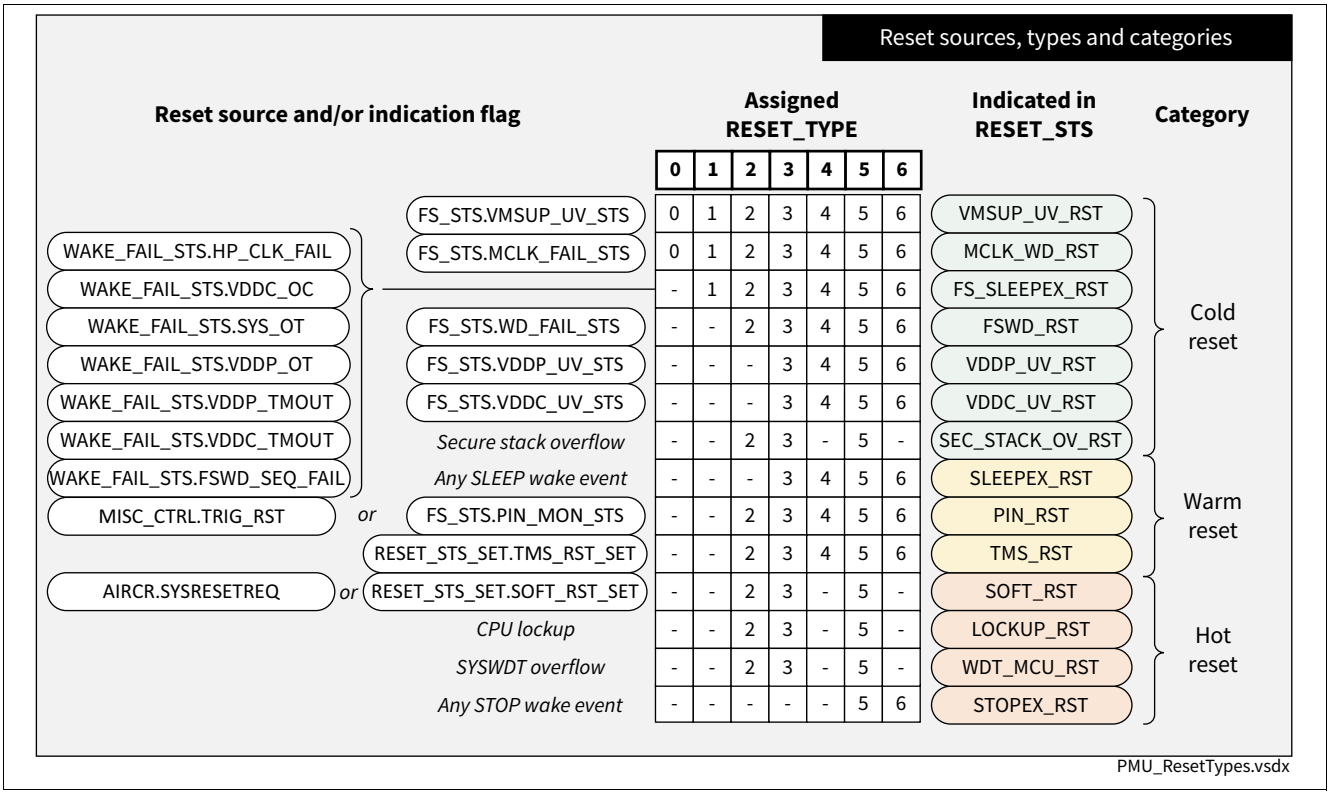


Figure 23 RESET_TYPE assignment matrix

Reset example scenarios

- Power-up of VS: nearly all SFRs are reset to their reset value; the PMU controls the timing and releases the RESET_TYPEs accordingly; the Power-up event is indicated in RESET_STS.VMSUP_UV_RST which has RESET_TYPE_0 to RESET_TYPE_6
- RESET pin event: the PMU related SFRs with RESET_TYPE_0/1 are not affected, all other SFRs which are assigned to RESET_TYPE_2/3/4/5/6 are reset; before the user software runs, the start-up firmware in BootROM does the necessary settings
- Wake-up from Stop-mode: SFRs with RESET_TYPE_5/6 are reset, which is the majority of the peripherals. The CPU related SFRs are not reset. For instance, the reset vector start address (VTOR), the program counter (PC) and stack pointer (SP) keep their values, thus the start-up firmware does not run and the user program continues at the address where the Stop mode was entered
- Wake-up from Sleep mode: all SFRs with RESET_3/4/5/6 are reset; before the user software runs, the start-up firmware in BootROM does the necessary settings
- WD_FAIL event: same scenario as the RESET pin event; the watchdog period is reset but not the watchdog fail counter
- FSWD_SEQ_FAIL event: this causes a Fail-sleep mode and affects VDDP, VDDC power domains; all configuration and control registers are reset except the status registers FS_STS and RESET_STS which indicate the causing event for diagnosis
- Soft reset event: SFRs with RESET_TYPE_0/1/4 /6 are not reset. This means that the power and clock settings are untouched, as well the registers which contain calibration data; the start-up firmware in BootROM runs and does the necessary settings

Overview SFRs and RESET_TYPEs

The following tables give an overview of the SFRs with their assigned RESET_TYPE:

Table 60 Overview SFRs with RESET_TYPE_0 to RESET_TYPE_2

Modules	SFR	0	1	2	3	4	5	6
PMU	Wake fail status (WAKE_FAIL_STS) Wake fail status clear (WAKE_FAIL_CLR) Wake fail status set (WAKE_FAIL_SET) Reset status (RESET_STS) General purpose user data (GPUDATAx) Functional safety status (FS_STS) Functional safety status clear (FS_STS_CLR) Functional safety status set (FS_STS_SET) Safe state status (FS_SSD) Safe state status clear (FS_SSD_CLR) Safe state status set (FS_SSD_SET)	X						
PMU	Reset pin control (RST_CTRL) Reset status clear (RESET_STS_CLR) Reset status set (RESET_STS_SET) Wake status (WAKE_STS) Wake status clear (WAKE_STS_CLR) Wake status set (WAKE_STS_SET) Start configuration control (START_CONFIG)		X					
PMU	VDDEXT voltage regulator control (VDDEXT_CTRL) Wake-up filter control (WAKE_FILT_CTRL) Cyclic sense / cyclic wake control (CYC_CTRL) GPIO wake control (WAKE_GPIO_CTRLx) MON input control (MON_CTRLx) MON input status (MON_STS) Wake control (WAKE_CTRL) Miscellaneous control (MISC_CTRL)			X				

Table 61 Overview SFRs with RESET_TYPE_3 and RESET_TYPE_4

Modules	SFR	0	1	2	3	4	5	6
SCU	Interrupt node mapping (INPx) NMI control (NMICON) NMI status clear (NMISRC) NMI status set (NMISRS) MON interrupt enable (MONIEN) MON interrupt status clear (MONISC) MON interrupt status set (MONISS) MON interrupt configuration (MONCON) External interrupt enable (EXTIEN) External interrupt status clear (EXTISC) External interrupt status set (EXTISS) External interrupt config (EXTCON) GPT12 interrupt enable (GPTIEN) GPT12 interrupt status clear (GPTISC) GPT12 interrupt status set (GPTISS) DMA interrupt enable (DMAIEN) DMA interrupt status clear (DMAISC) DMA interrupt status set (DMAISS) DMA channel select CCU7 (DMAP_CCU7) DMA channel select ADCs (DMAP_ADC) DMA channel select Timer (DMAP_TIM) DMA channel select COM modules (DMAP_COM) Peripheral management control (PMCON) Module suspend control (SUSCTR) DMA control (DMACTRL)				X			
GPIO	General settings				X			
PMU	Watchdog control (WD_CTRL)					X		
ADC1	Calibration settings (CALAIx)					X		

Table 61 Overview SFRs with RESET_TYPE_3 and RESET_TYPE_4 (cont'd)

Modules	SFR	0	1	2	3	4	5	6
ADC2	Calibration settings (CALAIx) Temperature sense control (TSCTR) SQ channel slot (SQSLOTx) Channel configuration (CONVCFGx) Compare channel control (CMPCFGx) Filter configuration (FILTCFG) Internal configuration (INTCFG)					X		
SCU	Peripheral clock prescaler (CLKCON) Peripheral clock enable (CLKEN) XTAL control (XTALCON) XTAL status clear (XTALSTATC) XTAL status set (XTALSTATS) NMI status (NMISR) MON interrupt status (MONIS) External interrupt status (EXTIS) GPT12 interrupt status (GPTIS) DMA interrupt status (DMAIS) Power mode control (PMCON0) PCU control (PCU_CTRL) CPU LOCKUP config (LOCKUPCFG)					X		

Table 62 Overview with RESET_TYPE_5 and RESET_TYPE_6

Modules	SFR	0	1	2	3	4	5	6
ADC1	General settings						X	
ADC2	General settings						X	
ARVG	General settings						X	
BDRV	General settings						X	
CANCONTR	General settings						X	
CANTRX	General settings						X	
CCU7	General settings						X	
CSACSC	General settings						X	
GPT12	General settings						X	
SDADC	General settings						X	
SSC	General settings						X	
T2	General settings						X	
UART	General settings						X	
PLL	General settings						X	X
SCU	System clock select (CLKSEL) XTAL status (XTALSTAT)							X

Power-on reset sequence

The power-on reset sequence is described in the following figure:

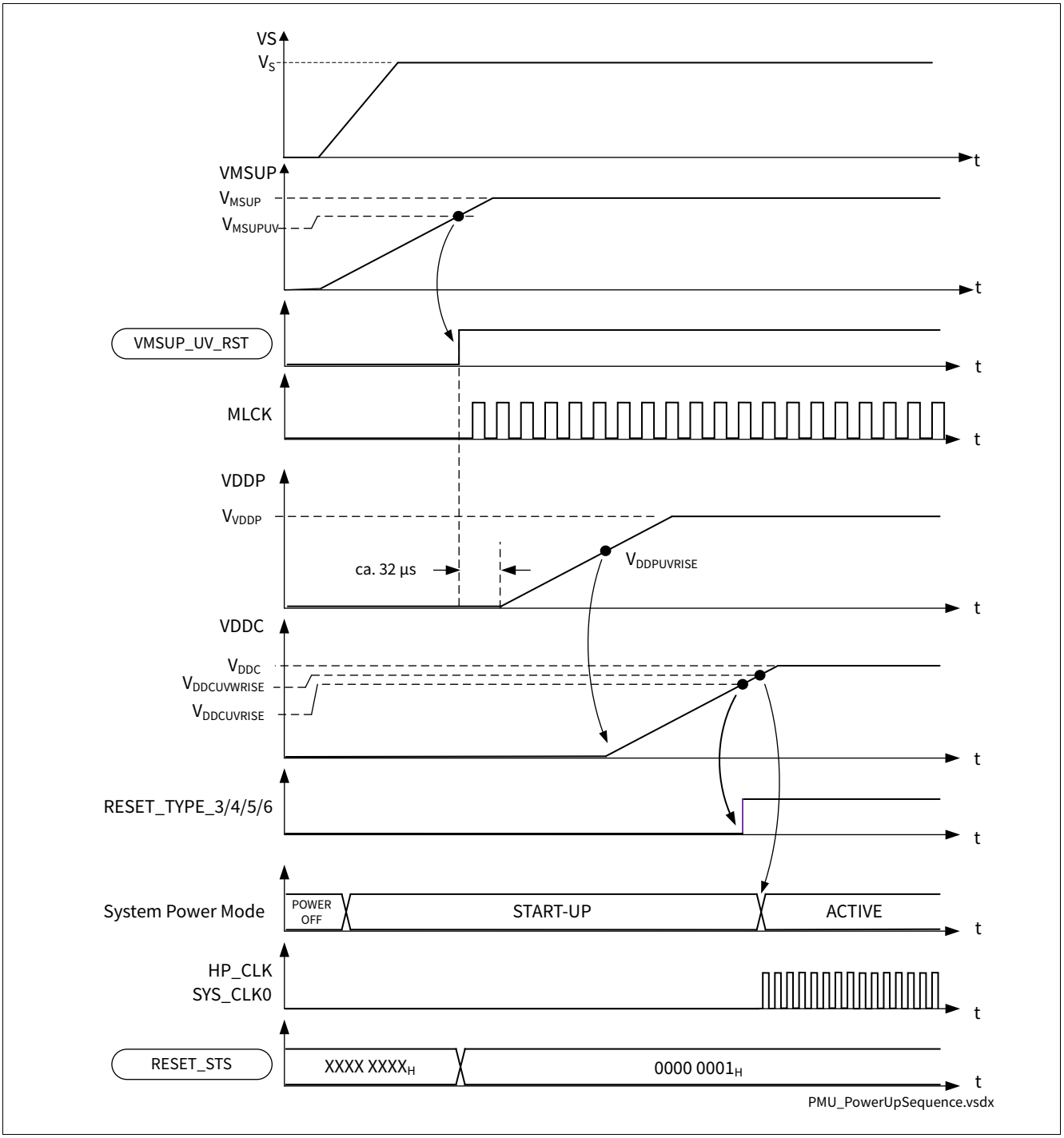


Figure 24 Power-on reset sequence

5.6.6 Wake control

The wake control unit handles the wake-up events and sends a “WAKE request” to the operating state machine which transits to Start-up state.

Following wake-up sources can be selected:

- Wake-up over GPIOs
- Wake-up over MON1 - MON3
- Wake-up over CAN
- Wake-up on VSD OV
- Wake-up on VDDP, VDDC and VDDEXT failures
- Wake-up over cyclic timer (cyclic wake, cyclic sense, cyclic sense with VDDEXT)

5.6.6.1 Wake-up over GPIO Port 0, Port 1 and Port 2

Up to six wake-up events can be selected out of all GPIOs (P0_x, P1_x, P2_x). The cyclic sense can be combined with the wake-up over GPIO. Also refer to the “GPIO” chapter for configuration of the GPIOs.

The wake-up path with the involved SFR can be taken from the figure below:

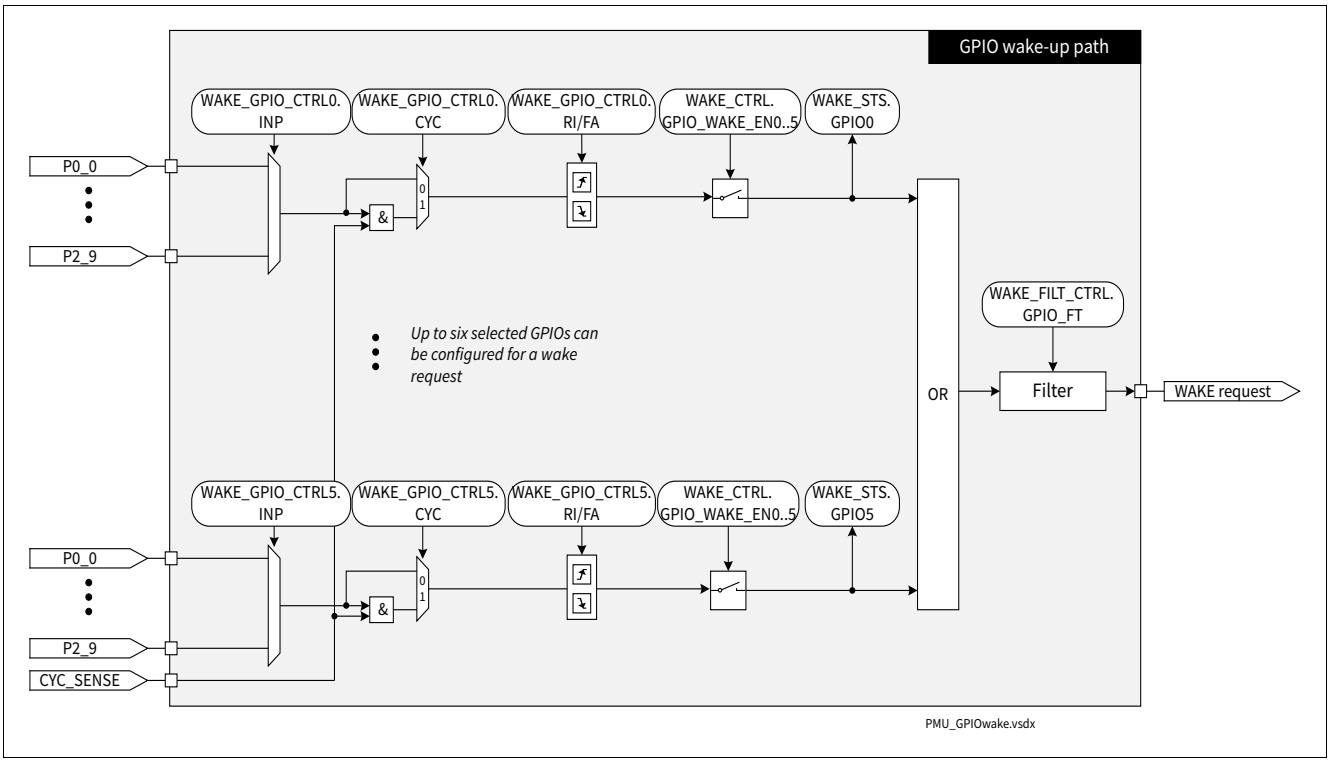


Figure 25 Wake-up over GPIO

5.6.6.2 Wake-up over MON

Up to three wake-up events (WAKEMON1/2/3) can be selected. The cyclic sense can be combined with the wake-up over MON. Also refer to “MON” chapter for configuration of the MON pins.

The wake-up with the involved SFR can be taken from the figure below:

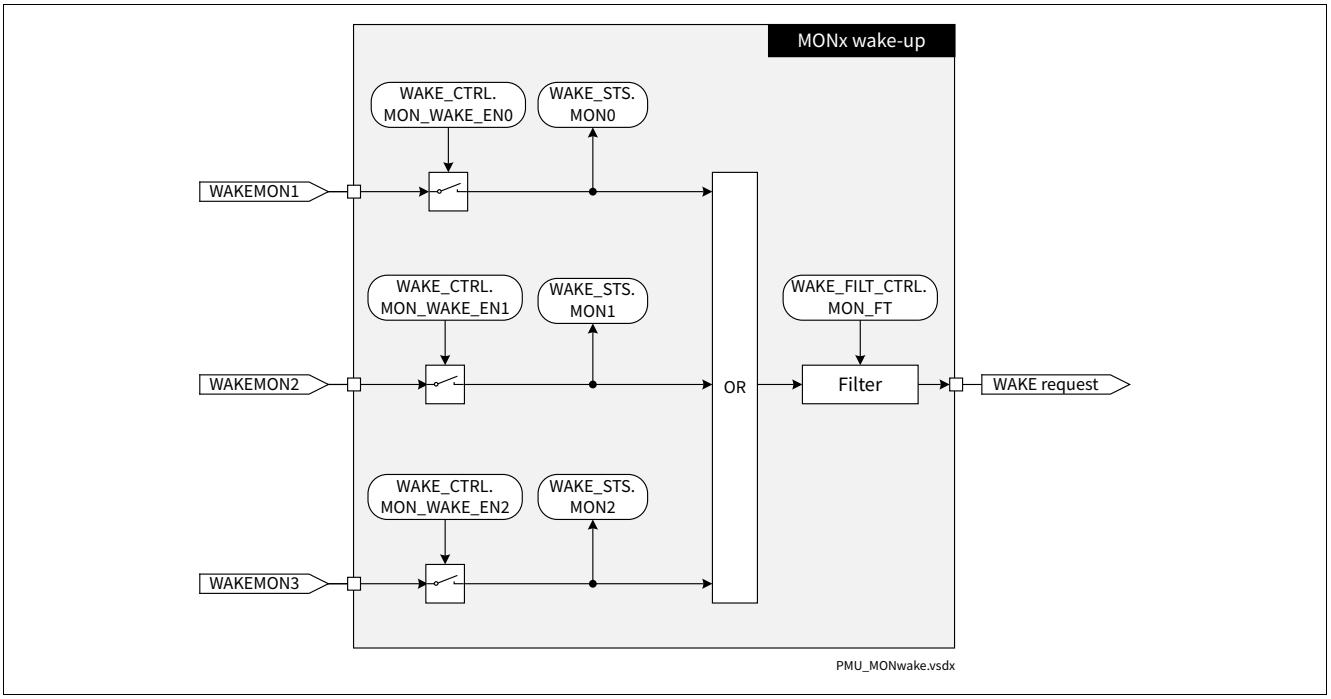


Figure 26 Wake-up over MON

5.6.6.3 Wake-up over CAN

The wake-up event over CAN (CAN bus activity) is an interconnect from the CANTRX (WAKECAN).

The wake-up over CAN can be enabled via WAKE_CTRL.CAN_WAKE_EN.

After a wake-up the status is visible in WAKE_STS.CAN. The status can be cleared via WAKE_STS_CLR.CAN_CLR.

Also refer to chapter CANTRX for its configuration options.

5.6.6.4 Wake-up over VSD overvoltage

The wake-up event over VSD overvoltage (WAKEVSDOV) is an interconnect from the BDRV (VSD_OV).

The wake-up over VSD overvoltage can be enabled via WAKE_CTRL.VSDOV_WAKE_EN.

After a wake-up the status is visible in WAKE_STS.VSD_OV. The status can be cleared via WAKE_STS_CLR.VSD_OV_CLR.

Also refer to chapter BDRV for its configuration options.

5.6.6.5 Wake-up on VDDP, VDDC and VDDEXT events

The following events of the voltage regulators can request a wake-up (see following table):

Table 63 Wake-up on VDDP, VDDC and VDDEX events

Module	SFR wake enable	SFR wake status
VDDP	WAKE_CTRL.VDDP_UVWARN_WAKE_EN	WAKE_STS.VDDP_UVWARN
	WAKE_CTRL.VDDP_OV_WAKE_EN	WAKE_STS.VDDP_OV
	WAKE_CTRL.VDDP_HCM_WAKE_EN	WAKE_STS.VDDP_HCM
VDDC	WAKE_CTRL.VDDC_UVWARN_WAKE_EN	WAKE_STS.VDDC_UVWARN
	WAKE_CTRL.VDDC_OV_WAKE_EN	WAKE_STS.VDDC_OV
	WAKE_CTRL.VDDC_HCM_WAKE_EN	WAKE_STS.VDDC_HCM
VDDEXT	WAKE_CTRL.VDDEXT_OT_WAKE_EN	WAKE_STS.VDDEXT_OT
	WAKE_CTRL.VDDEXT_UV_WAKE_EN	WAKE_STS.VDDEXT_UV

Also refer to chapter "**Voltage regulator control**".

5.6.6.6 Wake-up over cyclic timer (cyclic wake, cyclic sense)

The cyclic timer can generate two cyclic signals which can lead to wake-up requests:

- CYC_WAKE is a periodic event with a programmable on- and off-time which can act as a wake request
- CYC_SENSE is a periodic signal with a programmable on- and off-time which can act as a sense-pulse, which samples the selected GPIOs or MON1/2/3 inputs

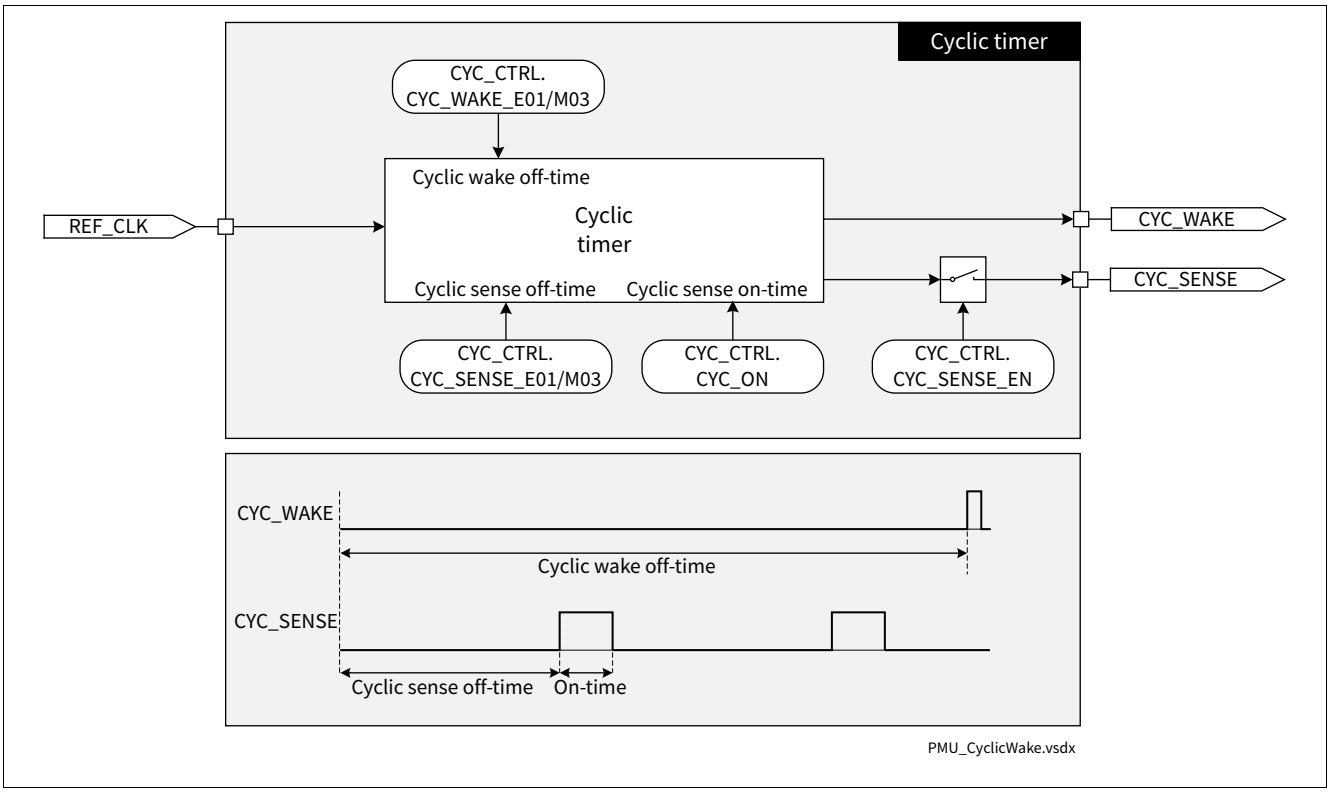


Figure 27 Cyclic timer block diagram

Cyclic wake

The cyclic wake timing can be configured in CYC_CTRL.

The cyclic wake off-time is configurable from 2 ms to 2048 ms typ. (based on REF_CLK). The off-time is calculated by the following formula:

(5.1)

$$T_{off} = 4^{CYC\_WAKE\_E01} * (CYC\_WAKE\_M03+1) * 2ms$$

The cyclic wake can be enabled via WAKE_CTRL.CYC_WAKE_EN. This also enables the REF_CLK in Stop/Sleep mode.

The wake status is visible after wake-up in WAKE_STS.CYC_WAKE.

Cyclic sense

The cyclic sense function samples the selected GPIOs or MON1/2/3 inputs with the on-time of the cyclic sense signal.

The cyclic sense function is available in Stop mode.

The cyclic sense timing can be configured in CYC_CTRL.

The cyclic sense off-time is configurable from 2 ms to 2048 ms typ. (based on REF_CLK). The off-time is calculated by the following formula:

(5.2)

$$T_{off} = 4^{CYC\_SENSE\_E01} * (CYC\_SENSE\_M03+1) * 2ms$$

The cyclic wake on-time is configurable via CYC_CTRL.CYC_ON_TIME.

The cyclic sense can be enabled via CYC_CTRL.CYC_SENSE_EN. This also enables the REF_CLK in Stop mode.

The cyclic sense function for GPIOs has to be additionally enabled in WAKE_GPIO_CTRLx.CYC (x = 0 to 5).

The cyclic sense function for MON1/2/3 has to be additionally enabled in MON_CTRLx.CYC_SENSE_EN (x = 0 to 2).

The wake status is visible after wake-up in the corresponding flags of MON and GPIO in WAKE_STS.

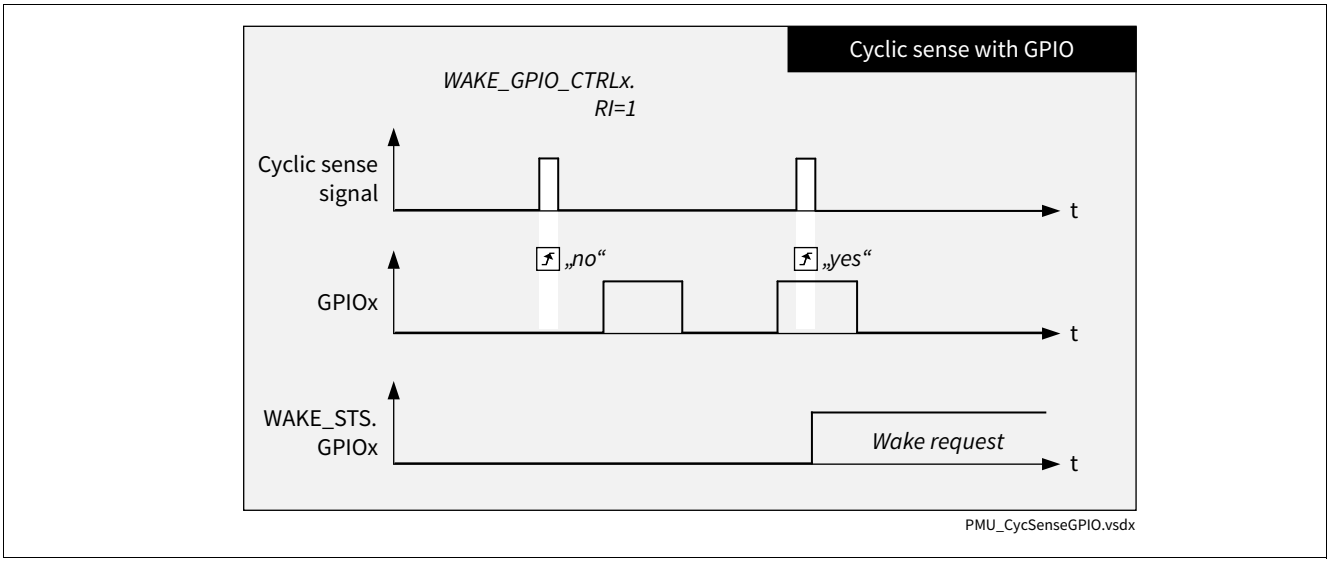


Figure 28 Cyclic sense wake example

Cyclic sense with VDDEXT

Additionally the VDDEXT regulator can be cyclically enabled with the on-time of the sense signal. This can be used to supply external circuitry in the context of cyclic sense operation.

The function has to be additionally enabled via VDDEXT_CTRL.CYC_EN.

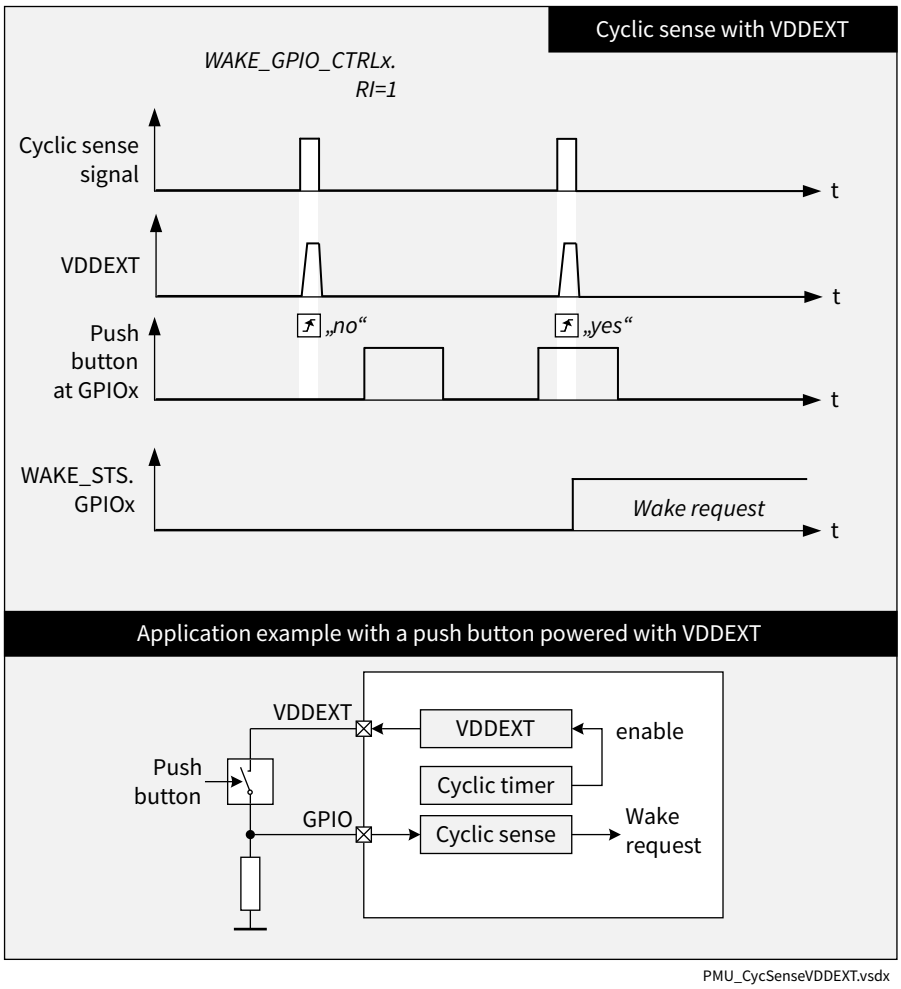


Figure 29 Cyclic timer block diagram

5.6.6.7 Wake control unit initialization

The following steps should be considered to initialize properly the wake control unit:

Table 64 Basic configuration steps

Step 1	WAKE_FILT_CTRL register	Configure GPIO and MON filter time
Step 2	WAKE_CTRL_GPIOx register	Required in case of wake-up via GPIO ports: 1. Configure wake-up event (rising and/or falling edge) 2. Select GPIO port 3. Enable/disable GPIO cyclic sensing
Step 3	MON_CTRLx register	Required in case of wake-up via MON ports: 1. Configure wake-up event (rising and/or falling edge) 2. Configure MON port pull-up/-down current source 3. Enable/disable MON cyclic sensing 4. Enable MON input
Step 4	CYC_CTRL register	Configure cyclic wake and cyclic sense wake timing option (optional): 1. Configure cyclic wake interval (CYC_WAKE_M03, CYC_WAKE_E01) 2. Configure cyclic sense on-time (CYC_ON_TIME) 3. Configure cyclic sense off-time (CYC_SENSE_M01, CYC_SENSE_E01) 4. Enable cyclic sense timer (CYC_SENSE_EN)
Step 5	WAKE_CTRL register	1. Select the wake-up sources (CAN, Cyclic Wake, GPIO, MON, VDDP UV Warning, VDDP OV, VDDP HCM, VDDC UV Warning, VDDC OV, VDDC HCM, VDDEXT OT, VDDEXT UV, VSD OV) 2. Enable/disable VDDC output voltage reduction *Note: To ensure a proper system wake-up, please ensure that one or more wake-up source(s) are enabled.*
Step 6	WAKE_STS register MON_STS register	Check wake and MON (optional) status register to identify active wake-up source(s)

5.7 Voltage regulators

5.7.1 Master supply generation

The VMSUP regulator consists of the following sub-modules:

- Master supply pre-regulator which generates an internal ‘always-on’ rail VDD5V_PD (5V typ.)
- Master supply regulator is supplied by VDD5V_PD and generates an internal ‘always-on’ rail VMSUP (1.5 V typ.)
- Master supply reference generates a reference voltage (VGEN_REF) which is used by various sub-modules and voltage regulators within the PMU and is measured at ADC2 (VGEN_ADC)

The master supply is monitored by the Supply monitor described in chapter “**Fail-safe supervision**”.

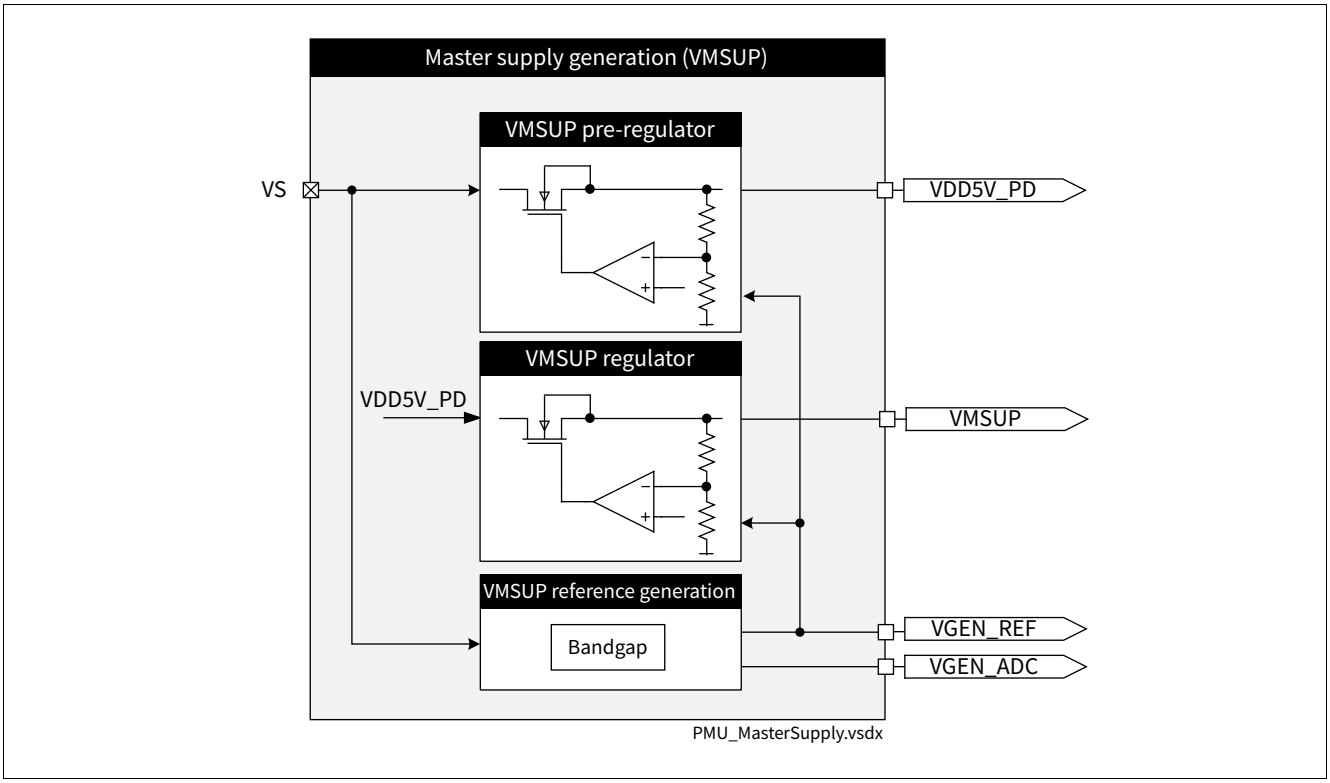


Figure 30 Master supply generation block diagram

5.7.2 VDDP

The VDDP linear regulator is used to supply the GPIOs, several analog on-chip components (e.g. CAN transceiver via VDDP-VCAN connection, ADCs) and can be used as off-chip on board component supply. It provides the following feature set:

- The VDDP regulator is supplied from the VS input
- The VDDP regulator output is available at pin VDDP with electrical parameter V_{DDP} and has to be buffered with C_{VDDP}
- The output current is limited ($I_{DDPILIM}$)
- The output voltage is monitored (see "**Supply monitor**")
- The temperature is monitored (VDDP_T, see "**Supply monitor**")
- The regulator is disabled in Sleep and Fail-sleep modes and the output is pulled-down
- A power-up sequence fail is detected and leads to Fail-sleep mode (see "**Start-up mode**")

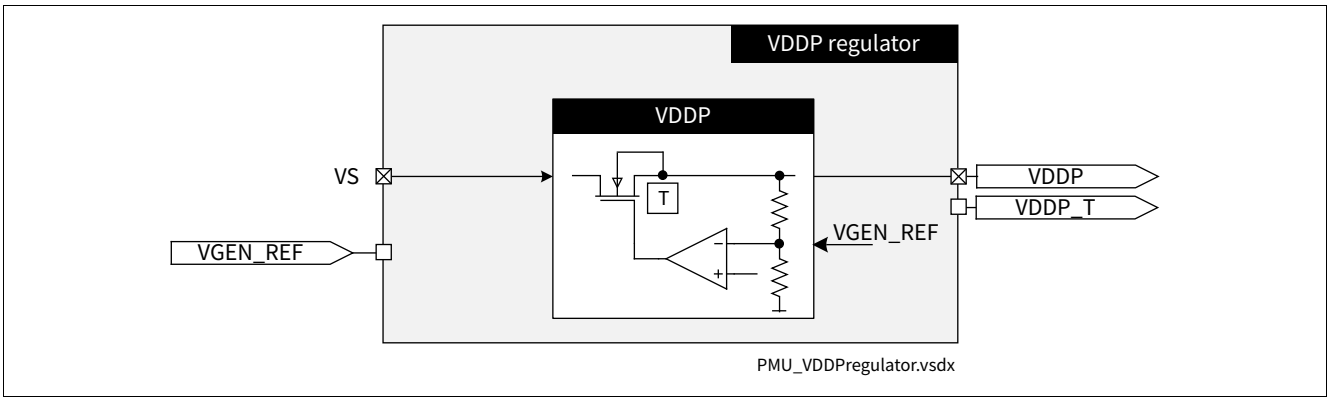


Figure 31 VDDP regulator block diagram

5.7.2.1 VDDP configuration

The VDDP regulator has three operating modes, configurable via VDDP_CTRL.MCNFSTOP:

- High-current mode (HCM):
  The regulator is capable of supporting the full load current range. This mode of operation consumes the highest quiescent current
- Low-current mode non-adaptive (LCMN):
  In this mode the regulator quiescent current is minimum. To maintain proper regulation, the output current shall be limited to less than I_{DDLCM1} (VDDP_CTRL.ITH_SEL bit = 0) or to less than I_{DDLCM2} (VDDP_CTRL.ITH_SEL bit = 1)
- Low-current mode adaptive (LCMA - default):
  In this mode the regulator quiescent current is minimum while the load current is less than I_{DDLCM1} (VDDP_CTRL.ITH_SEL bit = 0) or less than I_{DDLCM2} (VDDP_CTRL.ITH_SEL bit = 1). The regulator transitions automatically into high-current mode (HCM) as the load current exceeds these thresholds

5.7.2.2 VDDP status flags

Following status flags are available:

- Undervoltage warning interrupt status VDDP_STS.UVWARN_IS
- Undervoltage warning status VDDP_STS.UVWARN_STS
- Overvoltage interrupt status VDDP_STS.OV_IS
- High current mode status VDDP_STS.HCM_STS
- Current limitation status VDDP_STS.ILIM_STS

VDDP fail-sleep requests

Following VDDP events can request a fail-sleep mode and a reset (see "**Fail-sleep mode**" and "**Reset management**"):

- VDDP start-up timeout (VDDP_TMOUT), indicated in WAKE_FAIL_STS.VDDP_TMOUT
- VDDP overtemperature (VDDP_OT), indicated in WAKE_FAIL_STS.VDDP_OT

VDDP reset requests

Following VDDP event can request a reset (see "**Reset management**"):

- VDDP undervoltage (VDDP_UV), indicated in RESET_STS.VDDP_UV_RST

VDDP safe shutdown requests

Following VDDP events can request a safe shutdown (see "**Safe shutdown**"):

- VDDP undervoltage (VDDP_UV), indicated in FS_STS.VDDP_UV_STS
- VDDP overvoltage (VDDP_OV), indicated in FS_STS.VDDP_OV_STS
- VDDP overtemperature (VDDP_OT), indicated in FS_STS.VDDP_OT_STS

VDDP wake requests

Following VDDP events can request a wake-up if enabled (see "**Wake control**"):

- VDDP undervoltage warning (VDDP_UVWARN), indicated in WAKE_STS.VDDP_UVWARN
- VDDP overvoltage (VDDP_OV), indicated in WAKE_STS.VDDP_OV
- VDDP high current mode (VDDP_HCM), indicated in WAKE_STS.VDDP_HCM

VDDP interrupt requests

Following VDDP events can request an interrupt if enabled:

- VDDP undervoltage warning (VDDP_UVWARN), indicated in VDDP_STS.UVWARN_IS
- VDDP overvoltage (VDDP_OV), indicated in VDDP_STS.OV_IS

Table 65 Overview VDDP events, the requested reaction and the indication

Event	Interrupt	Wake-up	Safe shutdown	Reset	Fail-sleep mode
VDDP_TMOUT	–	–	yes, due to fail-sleep	yes, due to fail-sleep	WAKE_FAIL_STS. VDDP_TMOUT
VDDP_OT	–	–	FS_STS. VDDP_OT	yes, due to fail-sleep	WAKE_FAIL_STS. VDDP_OT
VDDP_UV	–	–	FS_STS. VDDP_UV_STS	RESET_STS. VDDP_UV_RST	–

Table 65 Overview VDDP events, the requested reaction and the indication (cont'd)

Event	Interrupt	Wake-up	Safe shutdown	Reset	Fail-sleep mode
VDDP_UVWARN	VDDP_STS. UVWARN_IS	WAKE_STS. VDDP_UVWARN	–	–	–
VDDP_OV	VDDP_STS. OV_IS	WAKE_STS. VDDP_OV	FS_STS. VDDP_OV_STS	–	–
VDDP_HCM	–	WAKE_STS. VDDP_HCM	–	–	–
VDDP_ILIM	–	–	–	–	–

5.7.2.3 VDDP monitoring timing diagram

The VDDP regulator monitoring is according to following timing diagram:

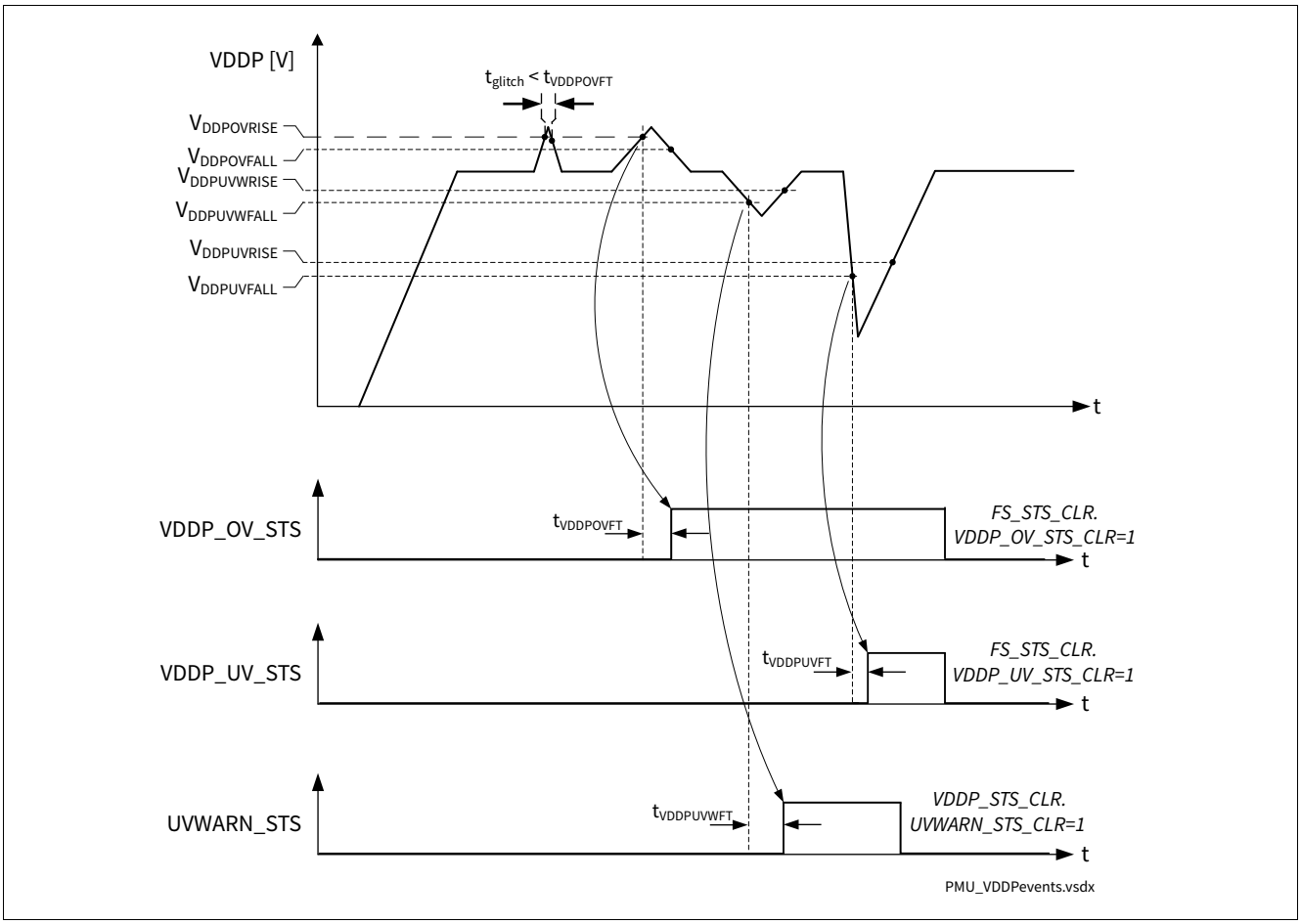


Figure 32 VDDP monitoring timing diagram

5.7.3 VDDC

The VDDC linear regulator is used to supply the device logic, it provides the following feature set:

- The VDDC regulator is supplied from the VDDP supply
- The VDDC regulator output is available at pin VDDC with electrical parameter V_{DDC} and has to be buffered with C_{VDDC}
- The output current is limited ($I_{VDDCILIM}$)
- The output voltage is monitored (see "**Supply monitor**")
- The output voltage can be reduced down in Stop mode for reduced leakage current
- The regulator is disabled in Sleep and Fail-sleep modes and the output is pulled-down
- A power-up sequence fail is detected and leads to Fail-sleep mode (see "**Start-up mode**")

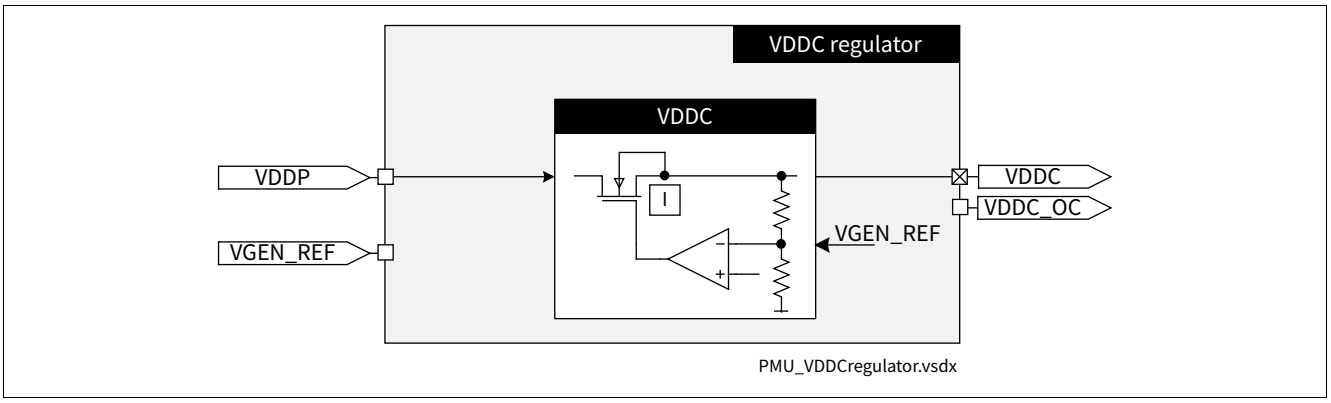


Figure 33 VDDC regulator block diagram

5.7.3.1 VDDC configuration

The VDDC regulator has following configuration options.

- Supports three operating modes, configurable via VDDC_CTRL.MCNFSTOP:
  - High-current mode (HCM):
    The regulator is capable of supporting the full load current range. This mode of operation consumes the highest quiescent current
  - Low-current mode non-adaptive (LCMN):
    In this mode the regulator quiescent current is minimum. To maintain proper regulation, the output current shall be less than I_{DDCLCM}
  - Low-current mode adaptive (LCMA - default):
    In this mode the regulator quiescent current is minimum while the load current is less than I_{DDCLCM}. The regulator transitions automatically into high-current mode (HCM) as the load current exceeds this threshold
- VDDC output voltage reduction to 0.9 V typ., configurable via WAKE_CTRL.VDDC_RED_EN

5.7.3.2 VDDC status flags

Following status flags are available in register VDDC_STS:

- Undervoltage warning interrupt status VDDC_STS.UVWARN_IS
- Undervoltage warning status VDDC_STS.UVWARN_STS
- Overvoltage interrupt status VDDC_STS.OV_IS
- High current mode status VDDC_STS.HCM_STS

VDDC Fail -sleep requests

Following VDDC events can request a Fail-sleep mode and a reset (see “**Fail-sleep mode**” and “**Reset management**”):

- VDDC start-up timeout (VDDC_TMOUT), indicated in WAKE_FAIL_STS.VDDC_TMOUT
- VDDC overcurrent (VDDC_OC), indicated in WAKE_FAIL_STS.VDDC_OC

VDDC Reset requests

Following VDDC event can request a reset (see “**Reset management**”):

- VDDC undervoltage (VDDC_UV), indicated in RESET_STS.VDDC_UV_RST

VDDC safe shutdown requests

Following VDDC events can request a safe shutdown (see “**Safe shutdown**”):

- VDDC undervoltage (VDDC_UV), indicated in FS_STS.VDDC_UV_STS
- VDDC overvoltage (VDDC_OV), indicated in FS_STS.VDDC_OV_STS

VDDC Wake requests

Following VDDC events can request a wake-up if enabled (see “**Wake control**”):

- VDDC undervoltage warning (VDDC_UVWARN), indicated in WAKE_STS.VDDC_UVWARN
- VDDC overvoltage (VDDC_OV), indicated in WAKE_STS.VDDC_OV
- VDDC high current mode (VDDC_HCM), indicated in WAKE_STS.VDDC_HCM

VDDC Interrupt requests

Following VDDC events can request an interrupt if enabled:

- VDDC undervoltage warning (VDDC_UVWARN), indicated in VDDC_STS.UVWARN_IS
- VDDC overvoltage (VDDC_OV), indicated in VDDC_STS.OV_IS

Table 66 Overview VDDC events, the requested reaction and the indication

Event	Interrupt	Wake-up	Safe shutdown	Reset	Fail-sleep mode
VDDC_TMOUT	–	–	yes, due to fail-sleep	yes, due to fail-sleep	WAKE_FAIL_STS.VDDC_TMOUT
VDDC_OC	–	–	yes, due to fail-sleep	yes, due to fail-sleep	WAKE_FAIL_STS.VDDC_OC
VDDC_UV	–	–	FS_STS.VDDC_UV_STS	RESET_STS.VDDC_UV_RST	–
VDDC_UVWARN	VDDC_STS.UVWARN_IS	WAKE_STS.VDDC_UVWARN	–	–	–
VDDC_OV	VDDC_STS.OV_IS	WAKE_STS.VDDC_OV	FS_STS.VDDC_OV_STS	–	–
VDDC_HCM	–	WAKE_STS.VDDC_HCM	–	–	–

5.7.3.3 VDDC monitoring timing diagram

The VDDC regulator monitoring is according to following timing diagram:

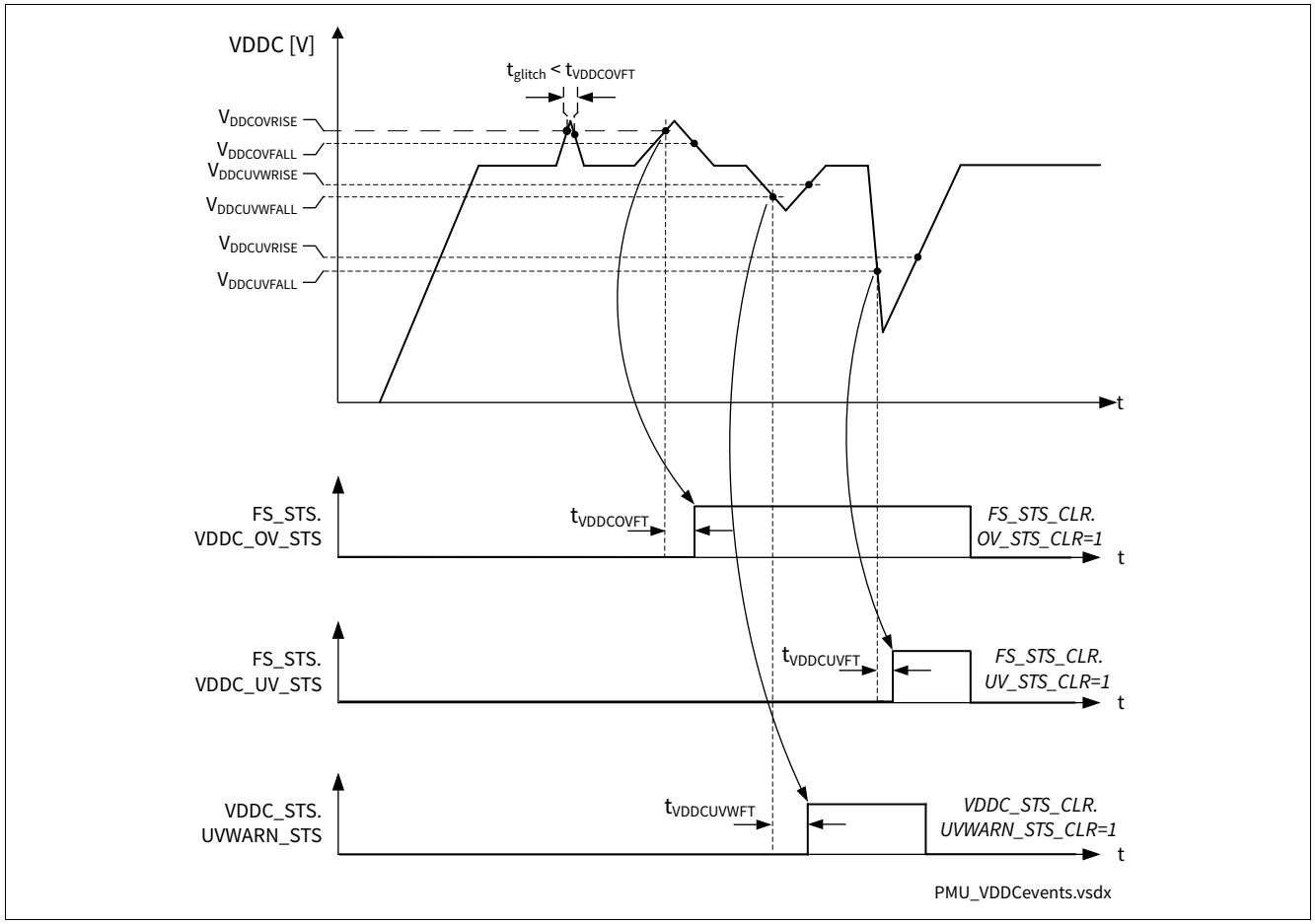


Figure 34 VDDC monitoring timing diagram

5.7.4 VAREF supply

The VAREF supply generates the supply for the VAREF regulator (VAREFSUP, see the “ARVG” chapter).

5.7.5 VDDEXT

The VDDEXT linear regulator is used to supply external sensors, LEDs or potentiometers, etc. The regulator provides the following feature set:

- The VDDEXT regulator is supplied from the VS input
- The VDDEXT regulator output is available at pin VDDEXT and has to be buffered with C_{VDDEXT}
- The output voltage is 5.0 V typ. (V_{DDEXT})
- The output current capability is up to 40 mA typ. (I_{DDEXT})
- The output is protected against output short-circuit to VS and GND and the current is limited ($I_{DDEXTILIM}$)
- The VDDEXT has an undervoltage (UV) and overtemperature monitoring (OT) with interrupt request
- The regulator can be enabled/disabled
- The regulator is disabled in Stop, Sleep and in Fail-Sleep mode
- The VDDEXT supports cyclic sense operation in Stop and Sleep mode (see “**Wake-up over cyclic timer (cyclic wake, cyclic sense)**”)

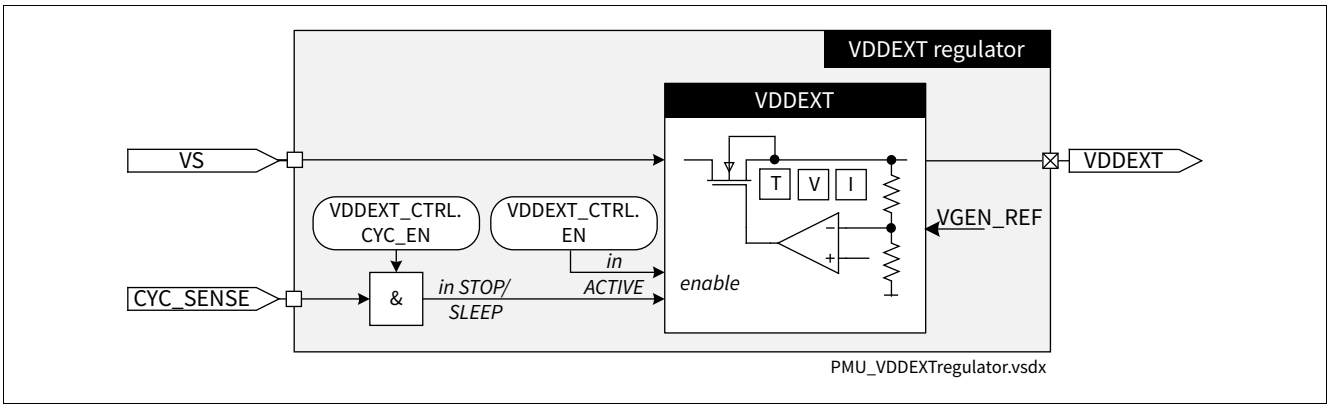


Figure 35 VDDEXT regulator block diagram

5.7.5.1 VDDEXT configuration

The VDDEXT regulator has following configuration options

- VDDEXT can be enabled via VDDEXT_CTRL.EN, when disabled the output is discharged via a resistive path
- Cyclic sense can be enabled via VDDEXT_CTRL.CYC_EN (see “**Wake-up over cyclic timer (cyclic wake, cyclic sense)**”)

5.7.5.2 VDDEXT status flags

Following status flags are available in register VDDEXT_STS:

- Undervoltage status VDDEXT_STS.UV_STS
- Undervoltage interrupt status VDDEXT_STS.UV_IS
- Overtemperature status VDDEXT_STS.OT_STS
- Overtemperature status VDDEXT_STS.OT_IS

VDDEXT Wake requests

Following VDDEXT events can request a wake-up if enabled (see “**Wake control**”):

- VDDEXT undervoltage (VDDEXT_UV), indicated in WAKE_STS.VDDEXT_UV
- VDDEXT overcurrent (VDDEXT_OT), indicated in WAKE_STS.VDDEXT_OT

Table 67 Overview VDDEXT events, the requested reaction and the indication

Event	Interrupt	Wake-up	Safe shutdown	Reset	Fail-sleep mode
VDDEXT_UV	VDDEXT_STS. UV_STS	WAKE_STS. VDDEXT_UV	–	–	–
VDDEXT_OT	VDDEXT_STS. OT_IS	WAKE_STS. VDDEXT_OT	–	–	–

5.7.5.3 VDDEXT control

In case of an overtemperature event, the VDDEXT_STS.OT_STS status flag is set and the regulator is disabled. To resume, the causing condition has to disappear and the OT_STS has to be cleared via VDDEXT_STS_CLR.OT_STS_CLR=1.

5.7.5.4 VDDEXT monitoring timing diagram

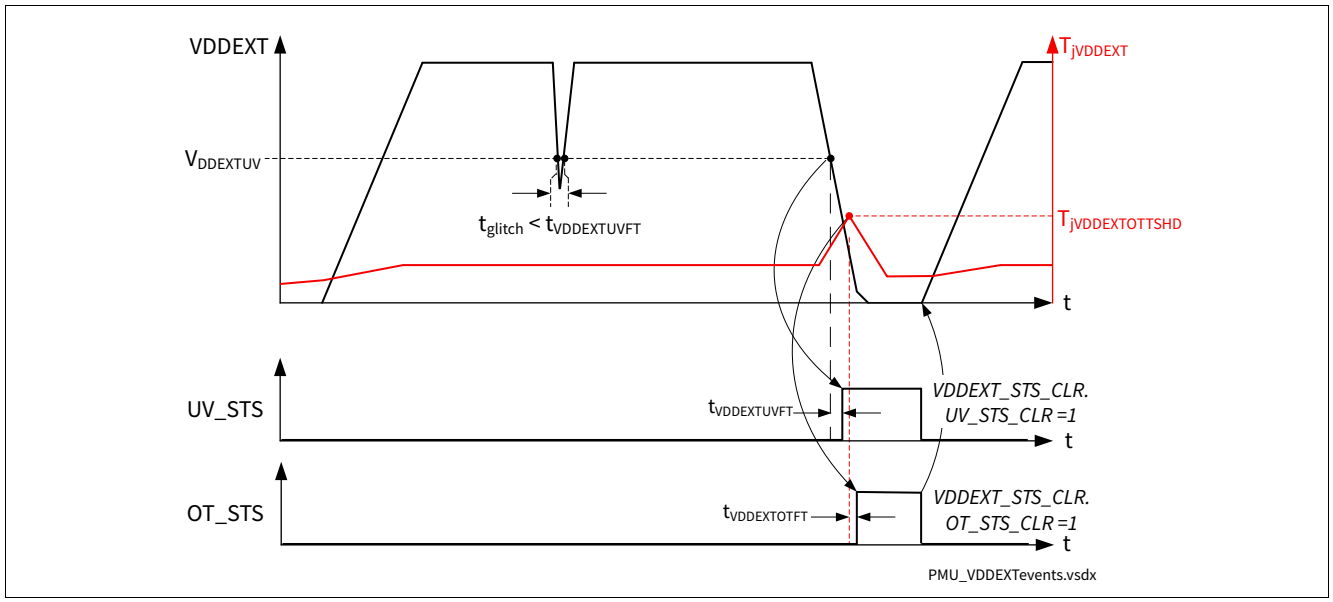


Figure 36 VDDEXT monitoring timing diagram

5.8 Fail-safe supervision

5.8.1 System monitor

The system monitor generates the Fail-sleep request to the operating state machine (see “**State control**”) and indicates the causing event in the register WAKE_FAIL_STS.

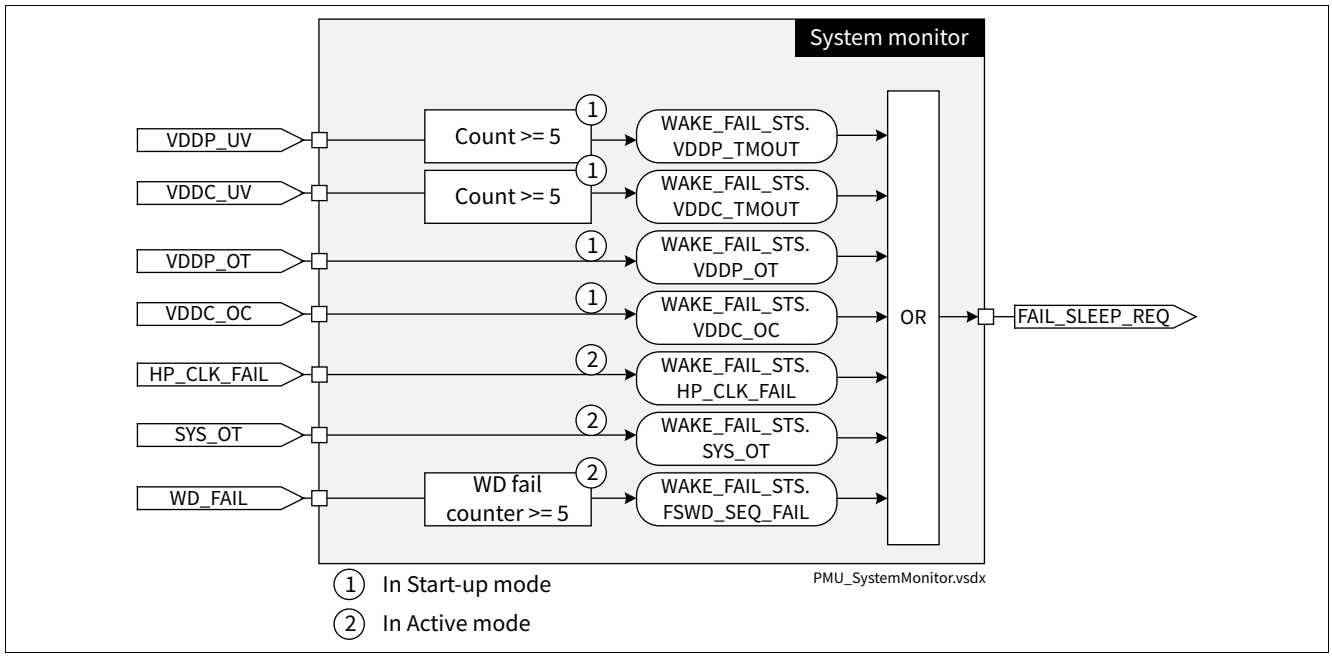


Figure 37 Supply monitor block diagram

Causing conditions for a Fail-sleep request (FAIL_SLEEP_REQ)

- Five times VDDP undervoltage detected at Start-up (VDDP_TMOUT)
- Five times VDDC undervoltage detected at Start-up (VDDC_TMOUT)
- VDDP overtemperature at Start-up (VDDP_OT)
- VDDC overcurrent at Start-up (VDDC_OC)
- Clock fail detection at the HP_CLK watchdog (HPWDG) in SCU (HP_CLK_FAIL)
- Overtemperature at the system temperature sensor measured by ADC2 (SYS_OT)
- Five consecutive FS_WDT fail events (WD_FAIL to FSWD_SEQ_FAIL)

The FS_WDT fail events are counted by the WD fail counter. The WD fail counter acts according to following rules:

- The WD fail counter is incremented with a wrong or missing FS_WDT service
- The WD fail counter is cleared with a correct FS_WDT service in Normal Window (NW) state
- In case the WD fail counter reaches a value of five, the FSWD_SEQ_FAIL is set

System monitor status flags

- VDDP timeout status WAKE_FAIL_STS.VDDP_TMOUT
- VDDC timeout status WAKE_FAIL_STS.VDDC_TMOUT
- VDDP overtemperature status WAKE_FAIL_STS.VDDP_OT
- VDDC overcurrent status WAKE_FAIL_STS.VDDC_OC

- HP_CLK_FAIL status WAKE_FAIL_STS.HP_CLK_FAIL
- SYS_OT status WAKE_FAIL_STS.SYS_OT
- WD_FAIL status WAKE_FAIL_STS.WD_SEQ_FAIL

5.8.2 Supply monitor

The supply monitor indicates fail-safe relevant events in the supply voltage generation. The events can request:

- A reset (see "**Reset management**")
- A safe shutdown (see "**Safe shutdown**")

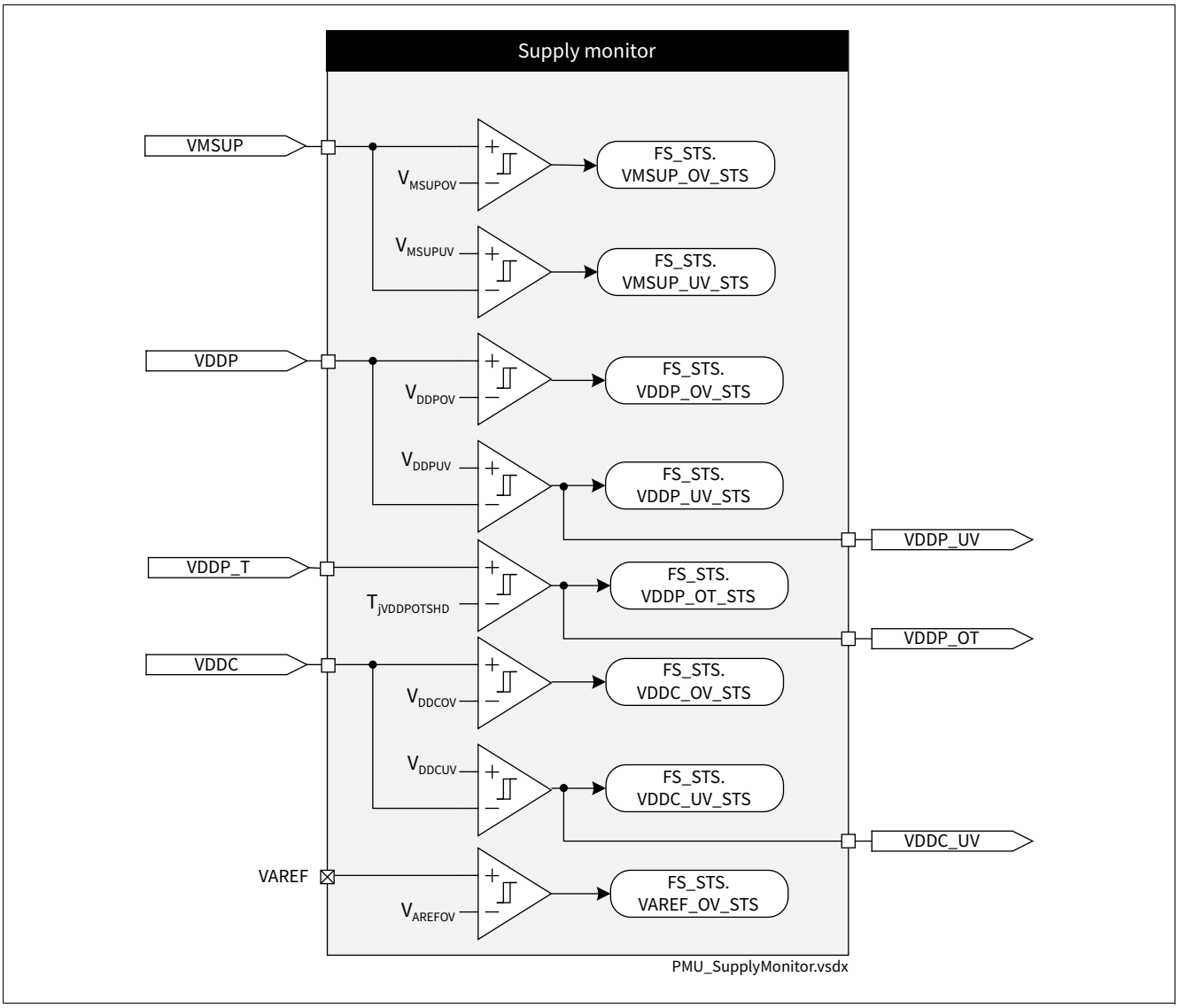


Figure 38 Supply monitor block diagram

Supply monitor status flags

Following status flags are available:

- Master supply overvoltage status FS_STS.VMSUP_OV_STS
- Master supply undervoltage status FS_STS.VMSUP_UV_STS
- VDDP overvoltage status FS_STS.VDDP_OV_STS

- VDDP undervoltage status FS_STS.VDDP_UV_STS
- VDDP overtemperature status FS_STS.VDDP_OT_STS
- VDDC overvoltage status FS_STS.VDDC_OV_STS
- VDDC undervoltage status FS_STS.VDDC_UV_STS
- VAREF overvoltage status FS_STS.VAREF_OV_STS

5.8.3 Safe reference clock and monitor

The safe reference clock (REF_CLK, $f_{SAFERCLK}$) is used to monitor the master clock (MCLK) and is clock source for the cyclic wake.

In case the clock monitor detects an error, the MCLK_WD_FAIL signal is activated and FS_STS.MCLK_WD_STS is set. This requests a safe-shutdown and a reset, see "**Safe shutdown**" and "**Reset management**".

This MCLK fail event is indicated in FS_STS.MCLK_WD_STS and in RESET_STS.MCLK_WD_RST.

Table 68 Safe reference clock in different operating modes

Operating mode	REF_CLK
Power-off	OFF
Start-up	ON
Active	ON
Stop	OFF if no cyclic wake configured ON if cyclic wake configured
Sleep	OFF if no cyclic wake configured ON if cyclic wake configured
Fail-sleep	ON

5.8.4 Bi-directional FIFO pin

The FIFO pin monitoring is shown in the figure below:

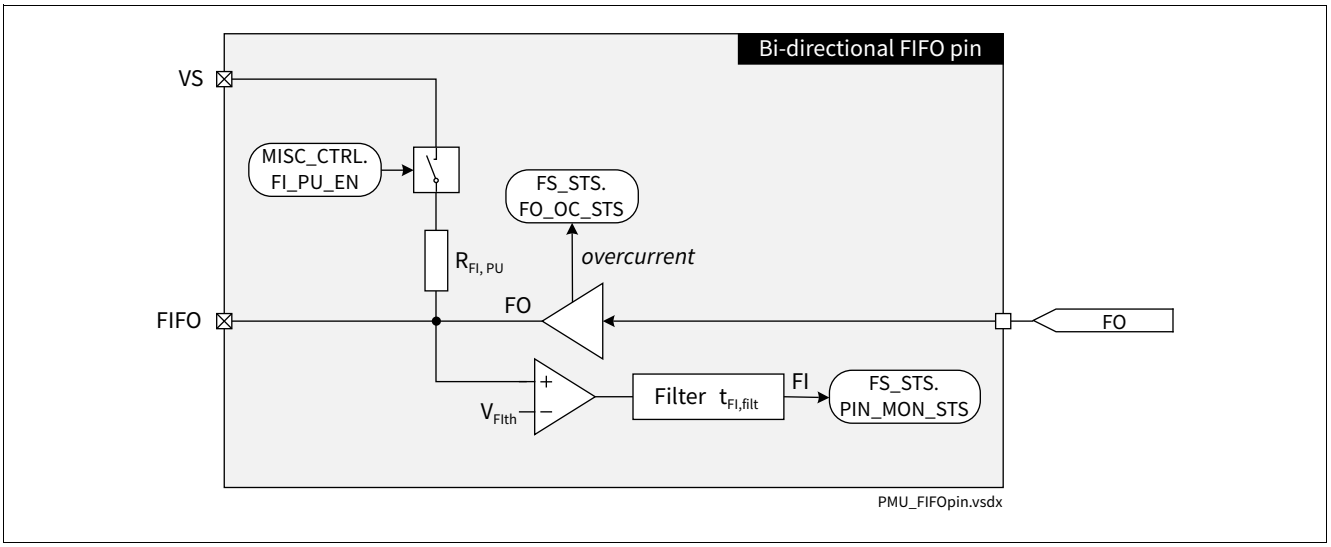


Figure 39 FIFO pin monitoring

The FIFO pin is a bi-directional high-voltage capable input/output pin. It acts as a fail input function (FI) as well as a fail output function (FO).

Fail Input function (FI)

The FIFO pin is monitored by an input comparator. If a low state is detected on the FIFO pin (for longer than the filter time), the flag FS_STS.PIN_MON_STS is set. This flag requests a safe shutdown (see “**Fail-safe supervision**”).

The FIFO is an open-drain pull-down circuit.

Fail Output function (FO)

An internal safe-shutdown state (FS_SSD.FO_STS=1) is indicated to the outside via a low level at the FIFO pin. The Fail-safe supervision activates the FO signal which enables the FO driver.

The FIFO circuitry is protected against external short-circuits (e.g. to VS). This failure will be reported in the FS_STS.FO_OC_STS bit.

The FIFO circuitry is supplied from VDD5V_PD thereby allowing Sleep mode operation. The FO is active in Stop, Sleep and Fail-Sleep modes. FIFO is actively driven low

Configurable pull-up

To prevent floating input on the FIFO pin, the device has a pull-up resistor to VS. The resistor can be enabled/disabled via software (MISC_CTRL.FI_PU_EN bit). An external pull-up resistor at the FIFO pin is not recommended.

In order to save current, the FI pull-up enable (FI_PU_EN bit) is automatically disabled in following cases:

- The FO signal is active
- The device is in Sleep and Fail-Sleep state

5.8.5 Fail-safe watchdog (FS_WDT)

Features

- The FS_WDT is a window watchdog running on a different clock source than the MCU sub-system
- The FS_WDT period can be configured from nom. 16 ms to nom. 1008 ms
- The FS_WDT has a special mode for deterministic software timing operations, e.g. flash programming
- The FS_WDT has to be serviced correctly within its open window
- An incorrect service requests a reset, see “**Reset management**”
- Five consecutive incorrect services request a transition to Fail-sleep mode, see “**System monitor**”

5.8.5.1 Block diagram

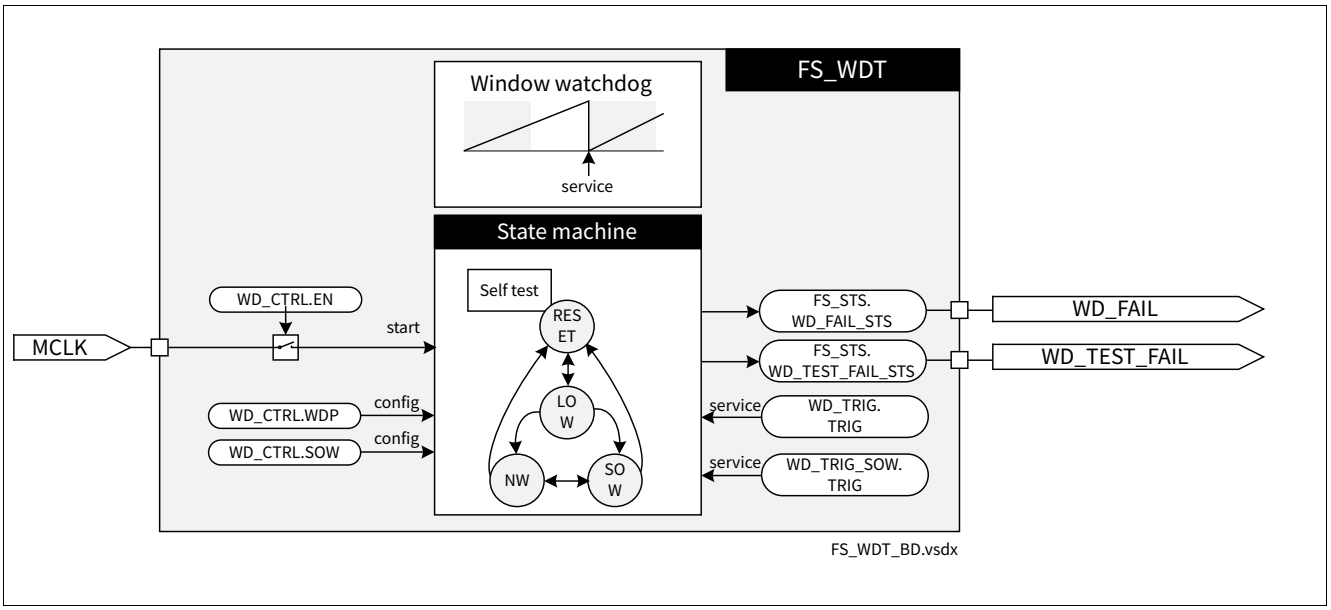


Figure 40 FS_WDT block diagram

5.8.5.2 Toplevel signals

The FS_WDT has interconnects as specified in **Table 69**.

Table 69 FS_WDT connections

Signal	Direction	Description	From/To
MCLK	Input	Clock source	PMU
WD_FAIL	Output	Watchdog fail indication	PMU
WD_TEST_FAIL	Output	Watchdog self-test indication	PMU

5.8.5.3 Operation mode behavior

Table 70 Operation mode behavior FS_WDT

Power-up	• A self-test is performed • With the release of RESET_TYPE_5 the FS_WDT is enabled and running
Active mode	• The FS_WDT allows a one-time configuration • User software has to service the FS_WDT periodically and correctly otherwise a WD_FAIL event is generated which requests a reset
Stop mode	• The FS_WDT counter is reset and stopped • The FS_WDT starts in LOW
Sleep mode	• The FS_WDT configuration is reset, the timer is stopped • After a wake-up from Sleep the FS_WDT starts like from Power-up
Fail-sleep mode	• The FS_WDT configuration is reset, the timer is stopped • After a wake-up from Fail-sleep the FS_WDT starts like from Power-up
Debug mode	• The FS_WDT is disabled • Enabling the FS_WDT triggers a self-test which causes a WD_FAIL reset
BSL mode	• The FS_WDT is disabled • Enabling the FS_WDT triggers a self-test which causes a WD_FAIL reset

5.8.5.4 FS_WDT state machine

The FS_WDT state machine has following states:

- RESET with self-test and fail counter
- Long Open Window (LOW)
- Normal Window (NW)
- Short Open Window (SOW)

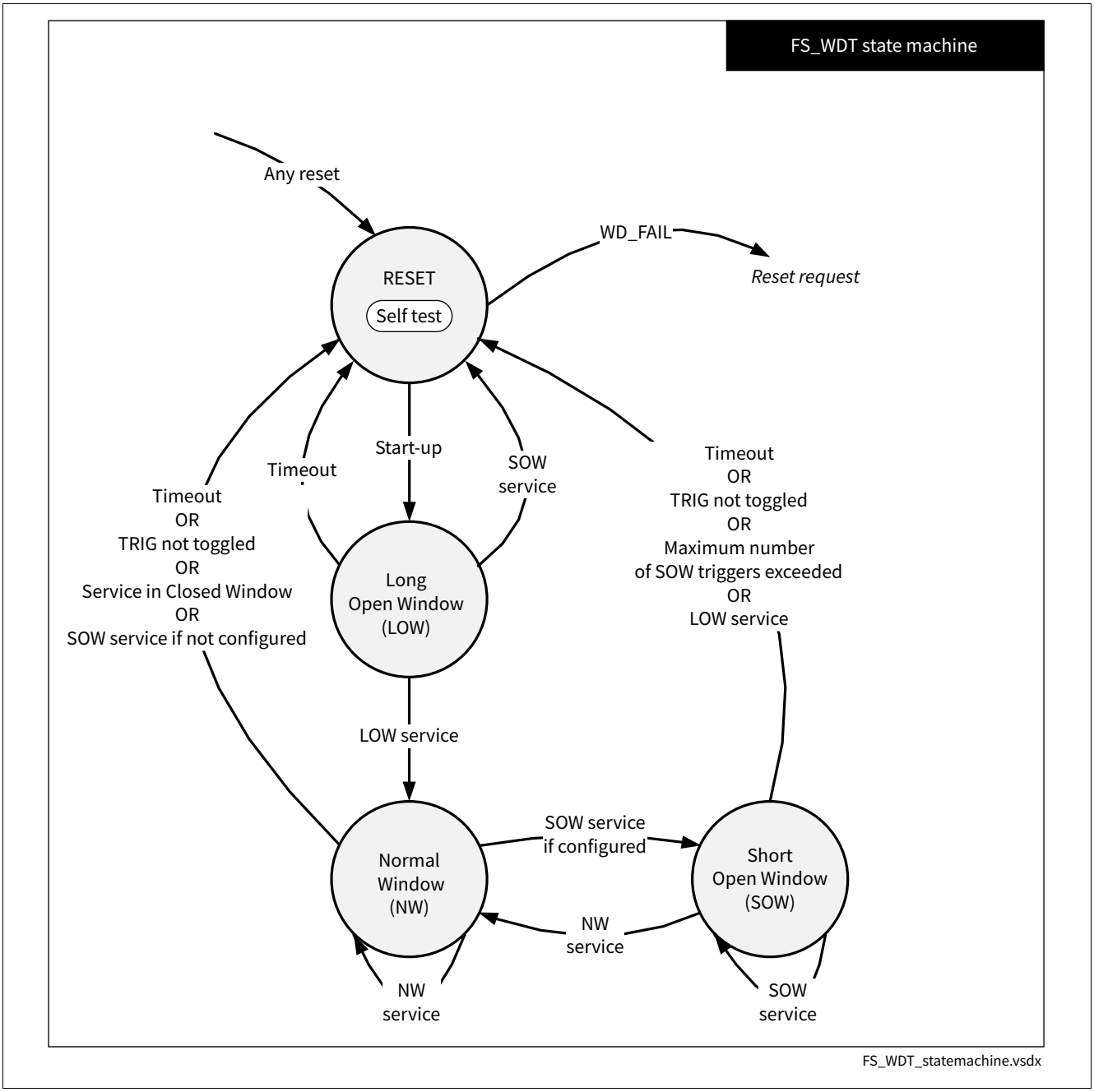


Figure 41 Fail-safe watchdog state machine

5.8.5.4.1 Reset state with self-test

After reset is released the FS_WDT is enabled and running. The LOW state is entered.

In case of a wrong or missing FS_WDT service (WD fail) in LOW, NW or SOW states the RESET state is entered and a WD_FAIL event is generated.

A WD fail event is indicated in FS_STS.WD_FAIL_STS and additionally in RESET_STS.FSWD_RST. Both flags have to be cleared by user accordingly.

FS_WDT self-test

First a self-test is performed while the device is executing the boot-up firmware from BootROM. The self-test starts the FS_WDT in LOW with a counter value close to its period and lets the FS_WDT overflow. The self-test runs only once. The boot-up firmware hands over the device to user with the FS_WDT in LOW mode.

A successful self-test sets FS_STS.WD_FAIL_STS. The flag must be cleared by user software in order to release the safe shutdown.

In case the self-test is not successful the bit FS_STS.WD_TEST_FAIL_STS is set. A reset shall be requested by user software via MISC_CTRL.TRIG_RST=1.

5.8.5.4.2 Long Open Window (LOW)

The FS_WDT starts with a Long Open Window after the FS_WDT RESET state.

The LOW has a fixed length of T_{LOW} = 3.600.000 /f_{MCLK} (180 ms at nom. f_{MCLK}).

FS_WDT initialization

The user software has to configure FS_WDT in the Long Open Window state. The configuration is locked and cannot be changed in Normal or Short Open Window states

Following configuration is possible:

- Watchdog period via WD_CTRL. WDP: T_{NW} = WDP * 320.000 /f_{MCLK} (WDP * 16 ms at nom. f_{MCLK})
- Allowed number of consecutive Short Open Windows via WD_CTRL.SOW

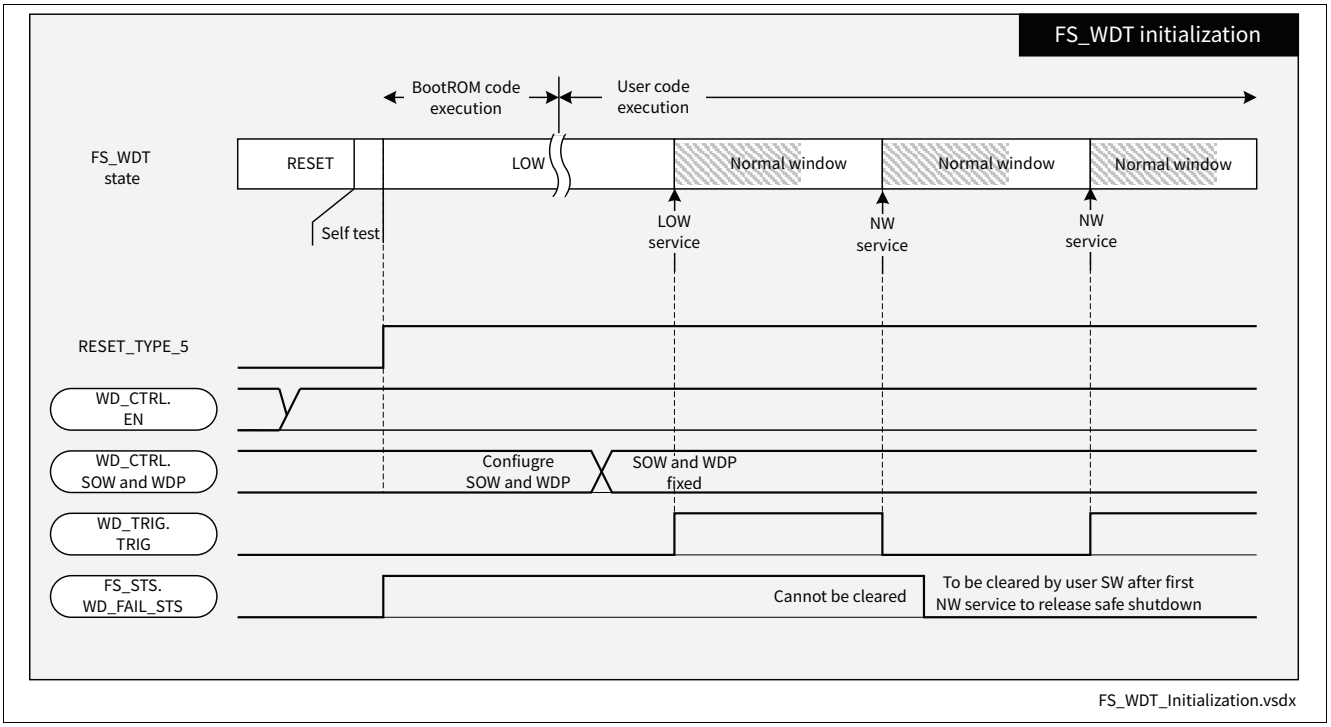


Figure 42 Fail-safe watchdog initialization sequence

FS_WDT service in LOW

- Correct LOW service:
  Set bit WD_TRIG.TRIG. This first service triggers the state change to the Normal Open Window (NW).
- Wrong LOW service:
  A wrong FS_WDT service (WD fail) causes a transition to RESET state. A wrong LOW service can be caused by following:
  - A missing service within the Long Open Window (FS_WDT overflow)
  - Wrong WD_TRIG.TRIG level
  - Servicing via the SOW service method (WD_TRIG_SOW.TRIG) if not configured

*Note: The boot-up time (firmware execution time in BootROM) consumes time of the Long Open Window and must be considered. The boot-up time can vary depending on user setting. See the chapter "Device startup" in the "**Product definitions**" chapter.*

5.8.5.4.3 Normal Window (NW) state

The Normal Window has a first half which is the Closed Window and a second half which is the Open Window. The user software has to service the FS_WDT periodically in Normal Window while the window is open.

The FS_WDT has to be serviced via a NW service method:

- Correct NW service:
  Only a service within the Open Window is a correct service. The service has to toggle bit WD_TRIG.TRIG with respect to the previous service. A correct FS_WDT service starts a new NW period
- Wrong NW service:
  A wrong FS_WDT service (WD fail) causes a transition to RESET state. A wrong NW service can be caused by following:
  - A missing service within the Normal Open Window (FS_WDT overflow)

- Not toggling bit WD_TRIG.TRIG with respect to the previous service
- A service within the Closed Window
- Servicing via the SOW service method (WD_TRIG_SOW.TRIG)

Effective Open Window

Calculating the correct trigger point for the FS_WDT service in Normal Window has to consider that the clock sources for FS_WDT and the CPU/peripherals are different:

- The FS_WDT clock source is the master clock (MCLK, f_{MCLK}) with tolerances
- The CPU and its peripherals clock source is the system clock (SYS0_CLK) with tolerances

The different clock sources and its tolerances (TOL) result in an effectively usable Open Window which is shorter than 50% of the watchdog period. **Figure 43** shows these dependencies.

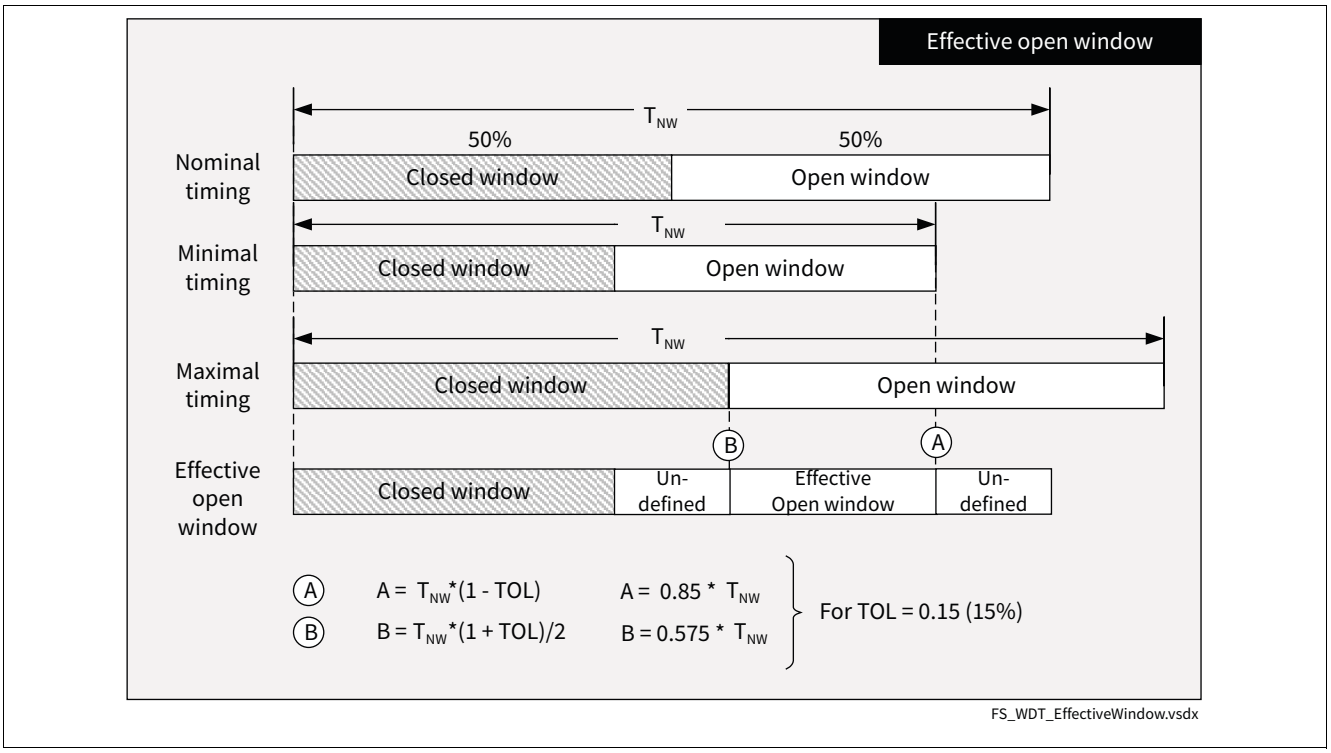


Figure 43 Effective open window

5.8.5.4.4 Short Open Window (SOW) state

The Short Open Window mechanism gives software some timing freedom for the FS_WDT service. This is helpful for example to start a NVM write/erase operation which requires a deterministic run-time. The SOW allows the software to service the FS_WDT right before it starts the NVM write/erase operation.

The SOW is a “short but always open” window.

The SOW has a fixed length of T_{LOW} = 600.000 * f_{MCLK} (30 ms at nom. f_{MCLK}).

The number of allowed consecutive SOWs (SOW sequence) is configurable via WD_CTRL.SOW. After a SOW sequence there must be at least one Normal Window. The number of SOW sequences is not restricted.

The SOW can only be configured once in the Long Open Window and is then fixed.

The FS_WDT has to be serviced via a SOW service method:

- Correct SOW service:
  The service has to toggle bit WD_TRIG_SOW.TRIG with respect to the previous service. A correct FS_WDT SOW service starts a new SOW. A correct FS_WDT NW service starts a NW
- Wrong SOW service:
  A wrong FS_WDT service (WD fail) causes a transition to RESET state. A wrong SOW service can be caused by following:
  - A missing service within the Short Open Window (FS_WDT overflow)
  - Not toggling bit WD_TRIG_SOW.TRIG with respect to the previous TRIG state
  - More SOW services than configured in WD_CTRL.SOW

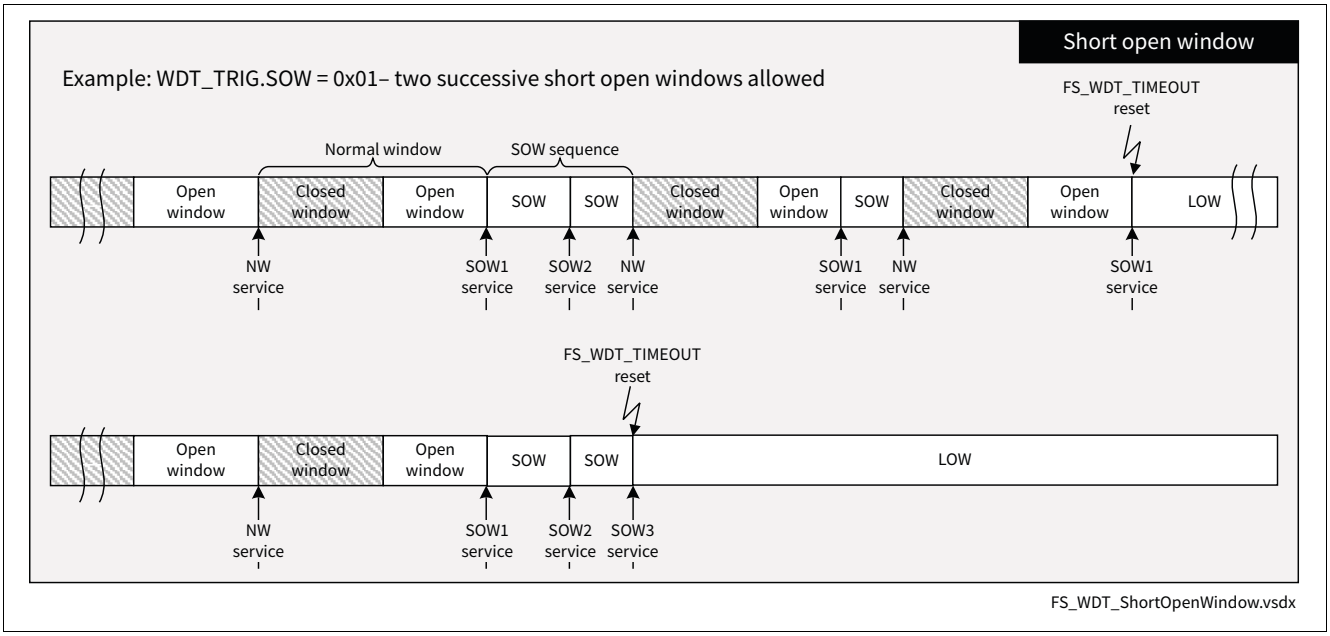


Figure 44 Short open window (SOW)

5.8.6 Safe shutdown

The safe shutdown mechanism brings the bridge driver (BDRV) into a safe off-state and signalize the safe-state to the outside world via the FIFO pin. The inputs to the safe shutdown are the status bits in FS_STS (see **Figure 45**).

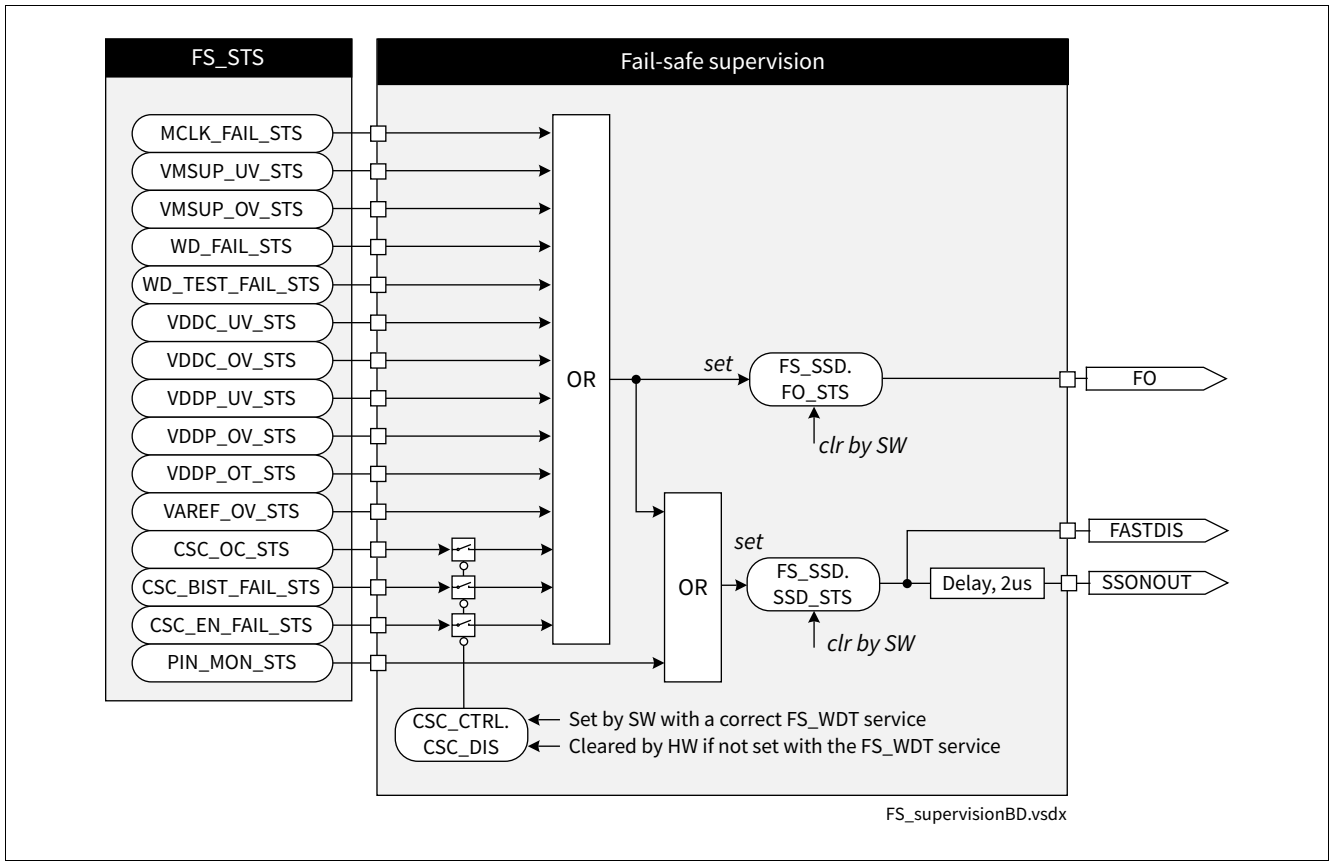


Figure 45 Safe shutdown block diagram

Safe shutdown indication

A safe shutdown request is indicated in FS_SSD.SSD_STS and FS_SSD_FO_STS.

Bit FS_SSD.SSD_STS is an OR combination of bits [14:0] in FS_STS.

Bit FS_SSD.FO_STS is an OR combination of bits [13:0] in FS_STS.

Safe shutdown outputs

The safe shutdown output signals are:

- FO is set with FS_SSD.FO_STS=1 and indicates the safe-shutdown via FO pin
- FASTDIS is set with FS_SSD.SSD_STS=1 and initiates the BDRV to do a high-current discharge of the external MOSFET gates
- SSONOUT is set with a short delay after FS_SSD.SSD_STS=1 and enables the passive pull-down mode of the BDRV

CSC fail reaction disable

A dedicated mechanism allows to disable the current sense comparator (CSC) as safe shutdown source. The disable mechanism has to be activated periodically every FS_WDT period. The disable mechanism works like following:

- Set bit CSC_CTRL.CSC_DIS before a correct Normal Window service of the FS_WDT
- Repeat the disabling if necessary in the next FS_WST period

Note: CSC_OC interrupts can still be generated while the CSC_OC safe-shutdown path is disabled.

Release the safe shutdown mechanism

To release the safe shutdown mechanism following steps are necessary:

- The causing condition must be inactive (e.g. no undervoltage, FIFO pin level high, CSC_OC inactive)
- The corresponding status flag(s) in FS_STS register have to be cleared (via FS_STS_CLR)
- The SSD_STS status flags FS_SSD.SSD_STS and FS_SSD.FO_STS have to be cleared (via FS_SSD_CLR)

The following figure shows the release sequence after device startup. Here the WD_FAIL_STS can only be cleared after the first NW service. For clearing the CSC related fail status bits, the CSA/CSC self-test has to be performed, see the chapter “Programmer’s guide” in the “CSC” chapter.

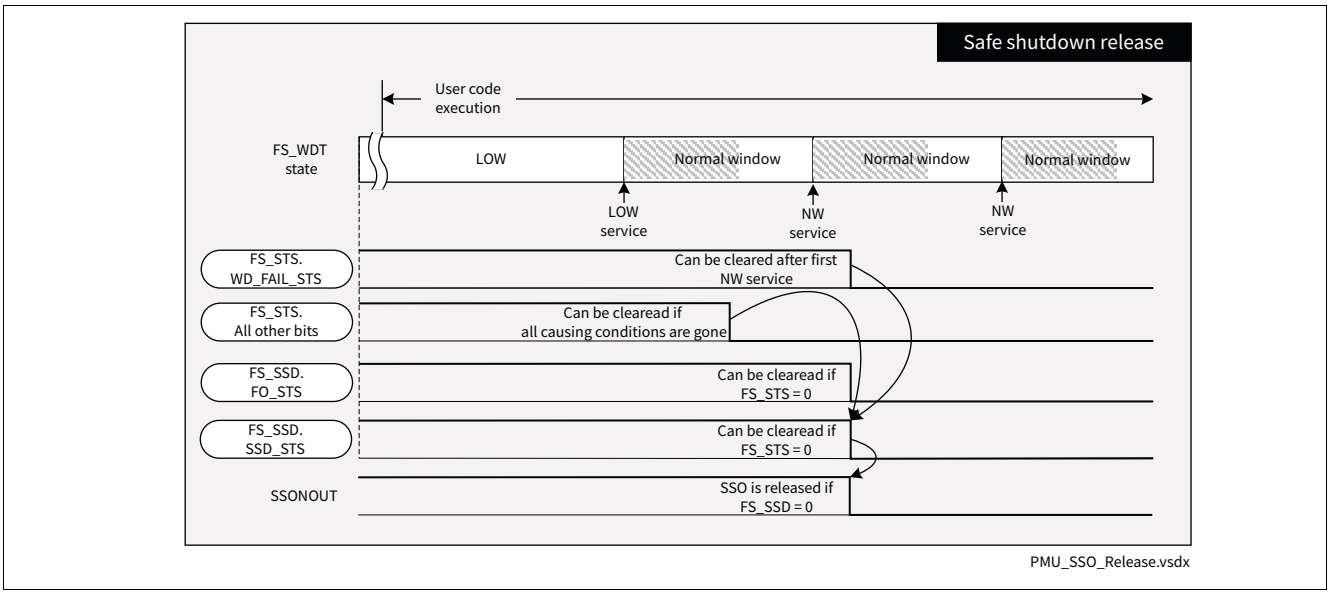


Figure 46 Safe shutdown release

5.9 Retention memory

The PMU provides a data retention memory of 96 bits. This memory is implemented as SFRs. These SFRs retain their content when VDDP and VDDC voltage is off (Sleep and Fail-sleep mode).

The retention memory is accessible in SFRs GPUDATA0/1/2.

5.10 Programmer's guide

5.10.1 PMU initialization

The following steps should b e considered to initialize properly the power management unit.

Table 71 Basic configuration steps for PMU

Step 1	MISC_CTRL	Configure FIFO pull-up resistor, see MISC_CTRL.FI_PU_EN bit
Step 2a	RESET_STS	Check the reset status • Normal start-up (i.e. VS ramp-up), RESET_STS = 0000 0001_H • Battery brownout: RESET_STS.VMSUP_UV_RST bit = 1 • Stop mode exit: RESET_STS.STOPEX_RST bit = 1 • Sleep mode exit: RESET_STS.SLEEPEX_RST bit = 1 • check other possible reset sources where user software has branches Clear RESET_STS flags accordingly and consider appropriate actions
Step 2b	WAKE_FAIL_STS	Check start-up failures (after Fail-sleep mode wake) • VDDP, VDDC start-up failure • VDDP overtemperature • VDDC overcurrent • HP clock failure • System overtemperature failure • FS_WDT sequential failure Clear WAKE_FAIL_STS flags accordingly and consider appropriate actions
Step 2c	WAKE_STS	Check wake-up status • CANTRX, GPIO, MON • VDDP power integrity • VDDC power integrity • VDDEXT power integrity, overtemperature • VSD overvoltage • Cyclic wake Clear WAKE_STS flags accordingly and consider appropriate actions
Step 3	FS_WDT	Configure the fail-safe watchdog (period and number of SOWs)
Step 4	FS_WDT	Perform first NW service
Step 5	SSO	Release safe shutdown mechanism
Step 6	VDDEXT	Enable VDDEXT regulator (optional)
Step 7	Wake control	Configure the wake control (optional)

5.11 Register description PMU

5.11.1 PMU Address Maps

Table 72 Register Address Space - PMU

Module	Base Address	End Address	Note
PMU	48000000_H	$48003FFF_H$	

Table 73 Register Overview - PMU (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
VDDP_CTRL	VDDP voltage regulator control register	0000_H	133
VDDP_IRQEN	VDDP interrupt enable register	0004_H	133
VDDP_STS	VDDP regulator status register	0008_H	134
VDDP_STS_CLR	VDDP regulator status clear register	$000C_H$	135
VDDP_STS_SET	VDDP regulator status set register	0010_H	136
VDDC_CTRL	VDDC voltage regulator control register	0014_H	137
VDDC_IRQEN	VDDC interrupt enable register	0018_H	137
VDDC_STS	VDDC regulator status register	$001C_H$	138
VDDC_STS_CLR	VDDC regulator status clear register	0020_H	139
VDDC_STS_SET	VDDC regulator status set register	0024_H	140
VDDEXT_CTRL	VDDEXT voltage regulator control register	0028_H	140
VDDEXT_IRQEN	VDDEXT interrupt enable register	$002C_H$	141
VDDEXT_STS	VDDEXT regulator status register	0030_H	141
VDDEXT_STS_CLR	VDDEXT regulator status register clear	0034_H	142
VDDEXT_STS_SET	VDDEXT regulator status register set	0038_H	143
WAKE_FAIL_STS	Wake fail status register	$003C_H$	144
WAKE_FAIL_CLR	Wake fail status clear register	0040_H	145
WAKE_FAIL_SET	Wake fail status set register	0044_H	146
RST_CTRL	RESET pin control register	0048_H	147
RESET_STS	Reset status register	$004C_H$	147
RESET_STS_CLR	Reset status clear register	0050_H	149
RESET_STS_SET	Reset status set register	0054_H	150
WAKE_FILT_CTRL	Wake up filter control register	0058_H	152
CYC_CTRL	Cyclic sense / Cyclic wake control register	$005C_H$	152
WAKE_GPIO_CTRLx	GPIO wake control register	0060_H+x*4	154
MON_CTRLx	MON input control register	0078_H+x*4	155
MON_STS	MON input status register	0084_H	156

Table 73 Register Overview - PMU (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Page Number
WAKE_CTRL	Wake control register	0088_H	157
WAKE_STS	Wake status register	$008C_H$	158
WAKE_STS_CLR	Wake status clear register	0090_H	160
WAKE_STS_SET	Wake status set register	0094_H	161
GPUDATAx	General purpose user data register	0098_H+x*4	163
MISC_CTRL	Miscellaneous control register	$00D8_H$	163
START_CONFIG	Start configuration control register	$00E4_H$	164
WD_CTRL	Watchdog control register	0700_H	164
WD_TRIG	Watchdog trigger register	0704_H	165
WD_TRIG_SOW	Watchdog SOW trigger register	0708_H	165
FS_STS	Functional safety status register	$070C_H$	166
FS_STS_CLR	Functional safety status clear register	0710_H	168
FS_STS_SET	Functional safety status set register	0714_H	170
FS_SSD	Safe state status register	0718_H	171
FS_SSD_CLR	Safe state status clear register	$071C_H$	172
FS_SSD_SET	Safe state status set register	0720_H	173
CSC_CTRL	Current sense comparator control register	0724_H	173

5.11.2 PMU Registers

VDDP voltage regulator control register

VDDP_CTRL
VDDP voltage regulator control register (0000$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

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

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

Field	Bits	Type	Description
MCNFSTOP	1:0	rw	**Stop mode configuration of VDDP regulator** 00_B **LCMA**, Regulator is configured to adaptive mode in Stop mode 01_B **LCMN**, Regulator is configured to low current mode in Stop mode 10_B **LCMN**, Regulator is configured to low current mode in Stop mode 11_B **HCM**, Regulator is configured to high current mode in Stop mode
ITH_SEL	2	rw	**Select current threshold for LCM/HCM mode** 0_B **SEL_0**, low threshold 1_B **SEL_1**, high threshold
RES	31:3	r	**Reserved**

VDDP interrupt enable register

VDDP_IRQEN
VDDP interrupt enable register (0004$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

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

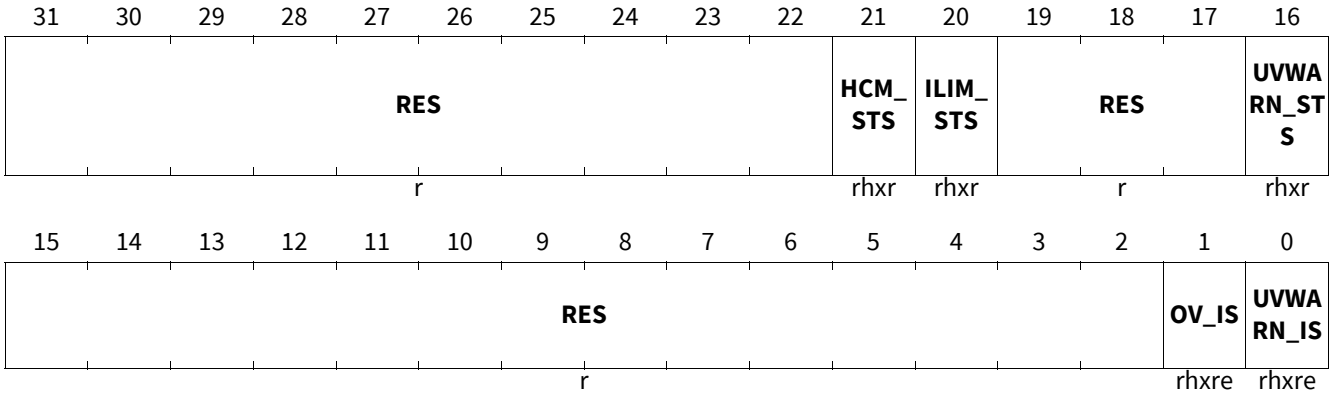
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														OV_IEN	UVWARN_IEN
r														rw	rw

Field	Bits	Type	Description
UVWARN_IEN	0	rw	**VDDP undervoltage warning interrupt enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled

Field	Bits	Type	Description
OV_IEN	1	rw	**VDDP overvoltage interrupt enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
RES	31:2	r	**Reserved**

VDDP regulator status register

VDDP_STS
VDDP regulator status register **(0008_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
UVWARN_IS	0	rhxre	**VDDP undervoltage warning interrupt status** 0_B **INACTIVE**, No undervoltage warning occurred 1_B **ACTIVE**, Undervoltage warning occurred
OV_IS	1	rhxre	**VDDP overvoltage interrupt status** 0_B **INACTIVE**, No overvoltage occurred 1_B **ACTIVE**, Overvoltage occurred
RES	15:2, 19:17, 31:22	r	**Reserved**
UVWARN_STS	16	rhxr	**VDDP undervoltage warning status** 0_B **INACTIVE**, No undervoltage warning occurred 1_B **ACTIVE**, Undervoltage warning occurred
ILIM_STS	20	rhxr	**VDDP current limitation status** 0_B **INACTIVE**, No current limitation occurred 1_B **ACTIVE**, Current limitation occurred
HCM_STS	21	rhxr	**VDDP high current mode status** 0_B **INACTIVE**, No high current mode occurred 1_B **ACTIVE**, High current mode occurred

VDDP regulator status clear register

VDDP_STS_CLR
VDDP regulator status clear register (000C$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										HCM_STS_CLR	ILIM_STS_CLR	RES			UVWARN_STS_CLR
				r						w	w		r		w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														OV_IS_CLR	UVWARN_IS_CLR
						r								w	w

Field	Bits	Type	Description
UVWARN_IS_CLR	0	w	**VDDP undervoltage warning interrupt status clear** 0_B **Not_Cleared**, Undervoltage warning status not cleared 1_B **Cleared**, Undervoltage warning status cleared
OV_IS_CLR	1	w	**VDDP overvoltage interrupt status clear** 0_B **Not_Cleared**, Overvoltage status not cleared 1_B **Cleared**, Overvoltage status cleared
RES	15:2, 19:17, 31:22	r	**Reserved**
UVWARN_STS_CLR	16	w	**VDDP undervoltage warning status clear** 0_B **Not_Cleared**, Undervoltage status warning not cleared 1_B **Cleared**, Undervoltage status warning cleared
ILIM_STS_CLR	20	w	**VDDP current limitation status clear** 0_B **Not_Cleared**, Current limitation status not cleared 1_B **Cleared**, Current limitation status cleared
HCM_STS_CLR	21	w	**VDDP high current mode status clear** 0_B **Not_Cleared**, High current mode status not cleared 1_B **Cleared**, High current mode status cleared

VDDP regulator status set register

VDDP_STS_SET
VDDP regulator status set register **(0010_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										HCM_STS_SET	ILIM_STS_SET	RES			UVWARN_STS_SET
				r						w	w		r		w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														OV_IS_SET	UVWARN_IS_SET
						r								w	w

Field	Bits	Type	Description
UVWARN_IS_SET	0	w	**VDDP undervoltage warning interrupt status set** 0_B **Not_Set**, Undervoltage warning status not set 1_B **Set**, Undervoltage warning status set
OV_IS_SET	1	w	**VDDP overvoltage interrupt status set** 0_B **Not_Set**, Overvoltage status not set 1_B **Set**, Overvoltage status set
RES	15:2, 19:17, 31:22	r	**Reserved**
UVWARN_STS_SET	16	w	**VDDP undervoltage warning status set** 0_B **Not_Set**, Undervoltage warning status not set 1_B **Set**, Undervoltage warning status set
ILIM_STS_SET	20	w	**VDDP Ilimit status set** 0_B **Not_Set**, Ilimit status not set 1_B **Set**, Ilimit status set
HCM_STS_SET	21	w	**VDDP high current mode status set** 0_B **Not_Set**, High current mode status not set 1_B **Set**, High current mode status set

Register description PMU

VDDC voltage regulator control register

VDDC_CTRL
VDDC voltage regulator control register (0014$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

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

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

Field	Bits	Type	Description
MCNFSTOP	1:0	rw	**Stop mode configuration of VDDC regulator** 00_B **LCMA**, Regulator is configured to adaptive mode in Stop mode 01_B **LCMN**, Regulator is configured to low current mode in Stop mode 10_B **LCMN**, Regulator is configured to low current mode in Stop mode 11_B **HCM**, Regulator is configured to high current mode in Stop mode
RES	31:2	r	**Reserved**

VDDC interrupt enable register

VDDC_IRQEN
VDDC interrupt enable register (0018$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

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

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

Field	Bits	Type	Description
UVWARN_IEN	0	rw	**VDDC undervoltage warning interrupt enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
OV_IEN	1	rw	**VDDC overvoltage interrupt enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
RES	31:2	r	**Reserved**

Register description PMU

VDDC regulator status register

VDDC_STS
VDDC regulator status register **($001C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

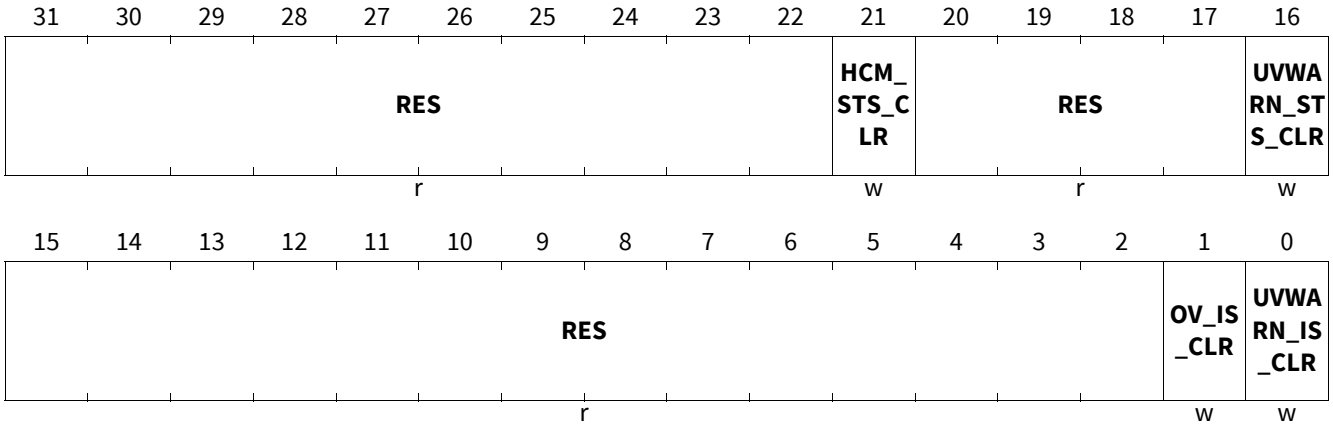
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										HCM_STS	RES				UVWARN_STS
r										rhxr	r				rhxr

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														OV_IS	UVWARN_IS
r														rhxre	rhxre

Field	Bits	Type	Description
UVWARN_IS	0	rhxre	**VDDC undervoltage warning interrupt status** 0_B **INACTIVE**, No undervoltage warning occurred 1_B **ACTIVE**, Undervoltage warning occurred
OV_IS	1	rhxre	**VDDC overvoltage interrupt status** 0_B **INACTIVE**, No overvoltage occurred 1_B **ACTIVE**, Overvoltage occurred
RES	15:2, 20:17, 31:22	r	**Reserved**
UVWARN_STS	16	rhxr	**VDDC undervoltage warning status** 0_B **INACTIVE**, No undervoltage warning occurred 1_B **ACTIVE**, Undervoltage warning occurred
HCM_STS	21	rhxr	**VDDC high current mode status** 0_B **INACTIVE**, No high current mode occurred 1_B **ACTIVE**, High current mode occurred

VDDC regulator status clear register

VDDC_STS_CLR
VDDC regulator status clear register **(0020$_H$)** **RESET_TYPE_3 Value: 0000 0000$_H$**



Field	Bits	Type	Description
UVWARN_IS_CLR	0	w	**VDDC undervoltage warning interrupt status clear** 0_B **Not_Cleared**, Undervoltage warning status not cleared 1_B **Cleared**, Undervoltage warning status cleared
OV_IS_CLR	1	w	**Overvoltage interrupt status clear** 0_B **Not_Cleared**, Overvoltage status not cleared 1_B **Cleared**, Overvoltage status cleared
RES	15:2, 20:17, 31:22	r	**Reserved**
UVWARN_STS_CLR	16	w	**VDDC undervoltage warning status clear** 0_B **Not_Cleared**, Undervoltage warning status not cleared 1_B **Cleared**, Undervoltage warning status cleared
HCM_STS_CLR	21	w	**VDDC high current mode status clear** 0_B **Not_Cleared**, High current mode status not cleared 1_B **Cleared**, High current mode status cleared

VDDC regulator status set register

VDDC_STS_SET

VDDC regulator status set register (0024$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										HCM_STS_SET	RES				UVWARN_STS_SET
r										w	r				w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														OV_IS_SET	UVWARN_IS_SET
r														w	w

Field	Bits	Type	Description
UVWARN_IS_SET	0	w	**VDDC undervoltage warning interrupt status set** 0_B **Not_Set**, Undervoltage warning status not set 1_B **Set**, Undervoltage warning status set
OV_IS_SET	1	w	**VDDC overvoltage interrupt status set** 0_B **Not_Set**, Overvoltage status not set 1_B **Set**, Overvoltage status set
RES	15:2, 20:17, 31:22	r	**Reserved**
UVWARN_STS_SET	16	w	**VDDC undervoltage warning status set** 0_B **Not_Set**, Undervoltage warning status not set 1_B **Set**, Undervoltage warning status set
HCM_STS_SET	21	w	**VDDC high current mode status set** 0_B **Not_Set**, High current mode status not set 1_B **Set**, High current mode status set

VDDEXT voltage regulator control register

VDDEXT_CTRL

VDDEXT voltage regulator control register (0028$_H$) RESET_TYPE_2 Value: 0000 0000$_H$

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

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

Field	Bits	Type	Description
EN	0	rw	**VDDEXT enable** 0_B **Disable**, VDDEXT regulator disabled 1_B **Enable**, VDDEXT regulator enabled
CYC_EN	1	rw	**VDDEXT cyclic sense enable** 0_B **Disable**, VDDEXT regulator with cyclic sense disabled 1_B **Enable**, VDDEXT regulator with cyclic sense enabled
RES	31:2	r	**Reserved**

VDDEXT interrupt enable register

VDDEXT_IRQEN
VDDEXT interrupt enable register **($002C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														**OT_IEN**	**UV_IEN**
r														rw	rw

Field	Bits	Type	Description
UV_IEN	0	rw	**VDDEXT undervoltage interrupt enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
OT_IEN	1	rw	**VDDEXT overtemperature interrupt enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
RES	31:2	r	**Reserved**

VDDEXT regulator status register

VDDEXT_STS
VDDEXT regulator status register **(0030_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES														**OT_STS**	**UV_STS**
r														rhxr	rhxr

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														**OT_IS**	**UV_IS**
r														rhxre	rhxre

Field	Bits	Type	Description
UV_IS	0	rhxre	**VDDEXT undervoltage interrupt status** 0_B **INACTIVE**, No undervoltage occurred 1_B **ACTIVE**, Undervoltage occurred
OT_IS	1	rhxre	**VDDEXT overtemperature interrupt status** 0_B **INACTIVE**, No overtemperature occurred 1_B **ACTIVE**, Overtemperature occurred
RES	15:2, 31:18	r	**Reserved**
UV_STS	16	rhxr	**VDDEXT undervoltage status** 0_B **INACTIVE**, No undervoltage occurred 1_B **ACTIVE**, Undervoltage occurred
OT_STS	17	rhxr	**VDDEXT overtemperature status** 0_B **INACTIVE**, No overtemperature occurred 1_B **ACTIVE**, Overtemperature occurred

VDDEXT regulator status register clear

VDDEXT_STS_CLR
VDDEXT regulator status register clear **(0034_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES														OT_STS_CLR	UV_STS_CLR
r														w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														OT_IS_CLR	UV_IS_CLR
r														w	w

Field	Bits	Type	Description
UV_IS_CLR	0	w	**VDDEXT undervoltage interrupt status clear** 0_B **Not_Cleared**, Undervoltage warning status not cleared 1_B **Cleared**, Undervoltage warning status cleared
OT_IS_CLR	1	w	**VDDEXT overtemperature interrupt status clear** 0_B **Not_Cleared**, Overtemperature status not cleared 1_B **Cleared**, Overtemperature status cleared
RES	15:2, 31:18	r	**Reserved**
UV_STS_CLR	16	w	**VDDEXT undervoltage status clear** 0_B **Not_Cleared**, Undervoltage warning status not cleared 1_B **Cleared**, Undervoltage warning status cleared

Field	Bits	Type	Description
OT_STS_CLR	17	w	**VDDEXT overtemperature status clear** 0_B **Not_Cleared**, Overtemperature status not cleared 1_B **Cleared**, Overtemperature status cleared

VDDEXT regulator status register set

VDDEXT_STS_SET

VDDEXT regulator status register set **(0038_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES														OT_STS_SET	UV_STS_SET
r														w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														OT_IS_SET	UV_IS_SET
r														w	w

Field	Bits	Type	Description
UV_IS_SET	0	w	**VDDEXT undervoltage interrupt status set** 0_B **Not_Set**, Undervoltage warning status not set 1_B **Set**, Undervoltage warning status set
OT_IS_SET	1	w	**VDDEXT overtemperature interrupt status set** 0_B **Not_Set**, Overtemperature status not set 1_B **Set**, Overtemperature status set
RES	15:2, 31:18	r	**Reserved**
UV_STS_SET	16	w	**VDDEXT undervoltage status set** 0_B **Not_Set**, Undervoltage warning status not set 1_B **Set**, Undervoltage warning status set
OT_STS_SET	17	w	**VDDEXT overtemperature status set** 0_B **Not_Set**, Overtemperature status not set 1_B **Set**, Overtemperature status set

Register description PMU

Wake fail status register

WAKE_FAIL_STS
Wake fail status register **($003C_H$)** **RESET_TYPE_0 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES									VDDC_OC	VDDP_OT	FSWD_SEQ_FAIL	SYS_OT	HPCLK_FAIL	VDDC_TMOUT	VDDP_TMOUT
r									rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr

Field	Bits	Type	Description
VDDP_TMOUT	0	rhxr	**VDDP regulator timeout** VDDP startup failed 5 times in sequence 0_B **INACTIVE**, No timeout occurred 1_B **ACTIVE**, Timeout occurred
VDDC_TMOUT	1	rhxr	**VDDP regulator timeout** VDDC startup failed 5 times in sequence 0_B **INACTIVE**, No timeout occurred 1_B **ACTIVE**, Timeout occurred
HPCLK_FAIL	2	rhxr	**HP clock fail status** 0_B **INACTIVE**, No HPCLK fail occurred 1_B **ACTIVE**, HPCLK fail occurred
SYS_OT	3	rhxr	**System overtemperature status** 0_B **INACTIVE**, No system overtemperature occurred 1_B **ACTIVE**, System overtemperature occurred
FSWD_SEQ_FAIL	4	rhxr	**Fail safe watchdog sequential fail status** Watchdog failed 5 times in sequence 0_B **INACTIVE**, No sequential watchdog fail occurred 1_B **ACTIVE**, Sequential watchdog fail occurred
VDDP_OT	5	rhxr	**VDDP regulator overtemperature status** 0_B **INACTIVE**, No VDDP overtemperature occurred 1_B **ACTIVE**, VDDP overtemperature occurred
VDDC_OC	6	rhxr	**VDDC overcurrent Status** 0_B **INACTIVE**, No overcurrent occurred 1_B **ACTIVE**, Overcurrent occurred
RES	31:7	r	**Reserved**

Wake fail status clear register

WAKE_FAIL_CLR
Wake fail status clear register **(0040_H)** **RESET_TYPE_0 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES									VDDC_OC_CLR	VDDP_OT_CLR	FSWD_SEQ_FAIL_CLR	SYS_OT_CLR	HPCLK_FAIL_CLR	VDDC_TMOUT_CLR	VDDP_TMOUT_CLR
r									w	w	w	w	w	w	w

Field	Bits	Type	Description
VDDP_TMOUT_CLR	0	w	**VDDP Regulator timeout Status Clear** 0_B **Not_Cleared**, VDDP timeout status not cleared 1_B **Cleared**, VDDP timeout status cleared
VDDC_TMOUT_CLR	1	w	**VDDP regulator timeout status clear** 0_B **Not_Cleared**, VDDC timeout status not cleared 1_B **Cleared**, VDDC timeout status cleared
HPCLK_FAIL_CLR	2	w	**HP clock fail status clear** 0_B **Not_Cleared**, HPCLK status not cleared 1_B **Cleared**, HPCLK status cleared
SYS_OT_CLR	3	w	**System overtemperature status clear** 0_B **Not_Cleared**, System overtemperature status not cleared 1_B **Cleared**, System overtemperature status cleared
FSWD_SEQ_FAIL_CLR	4	w	**Fail safe watchdog sequential fail status clear** 0_B **Not_Cleared**, Watchdog sequential fail status not cleared 1_B **Cleared**, Watchdog sequential fail status cleared
VDDP_OT_CLR	5	w	**VDDP regulator overtemperature status clear** 0_B **Not_Cleared**, VDDP overtemperature status not cleared 1_B **Cleared**, VDDP overtemperature status cleared
VDDC_OC_CLR	6	w	**VDDC overcurrent status clear** 0_B **Not_Cleared**, Overcurrent status not cleared 1_B **Cleared**, Overcurrent status cleared
RES	31:7	r	**Reserved**

Register description PMU

Wake fail status set register

WAKE_FAIL_SET
Wake fail status set register **(0044_H)** **RESET_TYPE_0 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES									VDDC_OC_SET	VDDP_OT_SET	FSWD_SEQ_FAIL_SET	SYS_OT_SET	HPCLK_FAIL_SET	VDDC_TMOUT_SET	VDDP_TMOUT_SET
r									w	w	w	w	w	w	w

Field	Bits	Type	Description
VDDP_TMOUT_SET	0	w	**VDDP regulator timeout status set** 0_B **Not_Set**, VDDP timeout status not set 1_B **Set**, VDDP timeout status set
VDDC_TMOUT_SET	1	w	**VDDP regulator timeout status set** 0_B **Not_Set**, VDDC timeout status not set 1_B **Set**, VDDC timeout status set
HPCLK_FAIL_SET	2	w	**HP clock fail status set** 0_B **Not_Set**, HPCLK fail status not set 1_B **Set**, HPCLK fail status set
SYS_OT_SET	3	w	**System overtemperature status set** 0_B **Not_Set**, System overtemperature status not set 1_B **Set**, System overtemperature status set
FSWD_SEQ_FAIL_SET	4	w	**Fails safe watchdog sequential fail status set** 0_B **Not_Set**, Watchdog sequential fail status not set 1_B **Set**, Watchdog sequential fail status set
VDDP_OT_SET	5	w	**VDDP regulator overtemperature status set** 0_B **Not_Set**, VDDP overtemperature status not set 1_B **Set**, VDDP overtemperature status set
VDDC_OC_SET	6	w	**VDDC overcurrent status set** 0_B **Not_Set**, Overcurrent status not set 1_B **Set**, Overcurrent status set
RES	31:7	r	**Reserved**

Register description PMU

RESET pin control register

RST_CTRL
RESET pin control register **(0048$_H$)** **RESET_TYPE_1 Value: 0000 0007$_H$**

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

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

Field	Bits	Type	Description
TFB	2:0	rw	**Reset blind time selection** 000_B **TFB_0**, 8 us 001_B **TFB_1**, 16 us 010_B **TFB_2**, 31 us 011_B **TFB_3**, 63 us 100_B **TFB_4**, 127 us 101_B **TFB_5**, 255 us 110_B **TFB_6**, 511 us 111_B **TFB_7**, 1023 us
RES	31:3	r	**Reserved**

Reset status register

RESET_STS
Reset status register **(004C$_H$)** **RESET_TYPE_0 Value: 0000 0001$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		TMS_RST	SEC_STACK_RST	VDDC_UV_RST	VDDP_UV_RST	LOCKUP_RST	SOFT_RST	WDT_MCU_RST	FSWD_RST	PIN_RST	STOPEX_RST	SLEEPEX_RST	FS_SLEEPEX_RST	MCLK_WD_RST	VMSUP_UV_RST
r		rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr

Field	Bits	Type	Description
VMSUP_UV_RST	0	rhxr	**Master supply undervoltage reset status** 0_B **INACTIVE**, No undervoltage reset occurred 1_B **ACTIVE**, undervoltage reset occurred
MCLK_WD_RST	1	rhxr	**Master clock watchdog reset status** 0_B **INACTIVE**, No master clock watchdog reset occurred 1_B **ACTIVE**, Master clock watchdog reset occurred

Field	Bits	Type	Description
FS_SLEEPEX_RST	2	rhxr	**Fail Sleep mode exit reset status** 0_B **INACTIVE**, No Fail Sleep exit reset occurred 1_B **ACTIVE**, Fail Sleep exit reset occurred
SLEEPEX_RST	3	rhxr	**Sleep mode exit reset status** 0_B **INACTIVE**, No Sleep Exit reset occurred 1_B **ACTIVE**, Sleep exit reset occurred
STOPEX_RST	4	rhxr	**Stop mode exit reset status** 0_B **INACTIVE**, No stop exit reset occurred 1_B **ACTIVE**, Stop exit reset occurred
PIN_RST	5	rhxr	**Pin-Reset reset status** 0_B **INACTIVE**, No pin reset occurred 1_B **ACTIVE**, Pin reset occurred
FSWD_RST	6	rhxr	**Fail safe watchdog reset status** 0_B **INACTIVE**, No watchdog reset occurred 1_B **ACTIVE**, Watchdog reset occurred
WDT_MCU_RST	7	rhxr	**MCU watchdog timer reset status** 0_B **INACTIVE**, No watchdog timer reset occurred 1_B **ACTIVE**, Watchdog timer reset occurred
SOFT_RST	8	rhxr	**Soft reset status** 0_B **INACTIVE**, No soft reset occurred 1_B **ACTIVE**, Soft reset occurred
LOCKUP_RST	9	rhxr	**ARM core lockup reset status** 0_B **INACTIVE**, No lockup reset occurred 1_B **ACTIVE**, Lockup reset occurred
VDDP_UV_RST	10	rhxr	**VDDP undervoltage reset status** 0_B **INACTIVE**, No undervoltage reset occurred 1_B **ACTIVE**, undervoltage reset occurred
VDDC_UV_RST	11	rhxr	**VDDC undervoltage reset status** 0_B **INACTIVE**, No undervoltage reset occurred 1_B **ACTIVE**, undervoltage reset occurred
SEC_STACK_RST	12	rhxr	**Secure stack overflow reset status** 0_B **INACTIVE**, No secure stack overflow occurred 1_B **ACTIVE**, Secure stack overflow occurred
TMS_RST	13	rhxr	**TMS reset status** 0_B **INACTIVE**, No TMS reset occurred 1_B **ACTIVE**, TMS reset occurred
RES	31:14	r	**Reserved**

Register description PMU

Reset status clear register

RESET_STS_CLR
Reset status clear register **(0050_H)** **RESET_TYPE_1 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		TMS_RST_CLR	SEC_STACK_RST_CLR	VDDC_UV_RST_CLR	VDDP_UV_RST_CLR	LOCKUP_RST_CLR	SOFT_RST_CLR	WDT_MCU_RST_CLR	FSWD_RST_CLR	PIN_RST_CLR	STOPEX_RST_CLR	SLEEPEX_RST_CLR	FS_SLEEPEX_RST_CLR	MCLK_WD_RST_CLR	VMSUP_UV_RST_CLR
r		w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
VMSUP_UV_RST_CLR	0	w	**Master supply undervoltage reset status clear** 0_B **Not_Cleared**, Master supply undervoltage reset status not cleared 1_B **Cleared**, Master supply undervoltage reset status cleared
MCLK_WD_RST_CLR	1	w	**Master clock watchdog reset status clear** 0_B **Not_Cleared**, Master clock watchdog reset status not cleared 1_B **Cleared**, Master clock watchdog reset status cleared
FS_SLEEPEX_RST_CLR	2	w	**Fail Sleep mode exit reset status clear** 0_B **Not_Cleared**, Fail Sleep exit reset status not cleared 1_B **Cleared**, Fail Sleep exit reset status cleared
SLEEPEX_RST_CLR	3	w	**Sleep mode exit reset status clear** 0_B **Not_Cleared**, Sleep exit reset status not cleared 1_B **Cleared**, Sleep exit reset status cleared
STOPEX_RST_CLR	4	w	**Stop mode exit reset status clear** 0_B **Not_Cleared**, Stop exit reset status not cleared 1_B **Cleared**, Stop exit reset status cleared
PIN_RST_CLR	5	w	**Pin reset Status Clear** 0_B **Not_Cleared**, Pin reset status not cleared 1_B **Cleared**, Pin reset status cleared
FSWD_RST_CLR	6	w	**Fail safe watchdog reset status clear** 0_B **Not_Cleared**, Watchdog fail reset status not cleared 1_B **Cleared**, Watchdog fail reset status cleared
WDT_MCU_RST_CLR	7	w	**MCU watchdog timer reset status clear** 0_B **Not_Cleared**, Watchdog timer reset status not cleared 1_B **Cleared**, Watchdog timer reset status cleared
SOFT_RST_CLR	8	w	**Soft reset status clear** 0_B **Not_Cleared**, Soft reset status not cleared 1_B **Cleared**, Soft reset status cleared

Register description PMU

Field	Bits	Type	Description
LOCKUP_RST_CLR	9	w	**ARM core lockup reset status** 0_B **Not_Cleared**, Lockup reset status not cleared 1_B **Cleared**, Lockup reset status cleared
VDDP_UV_RST_CLR	10	w	**VDDP undervoltage reset status clear** 0_B **Not_Cleared**, VDDP undervoltage reset status not cleared 1_B **Cleared**, VDDP undervoltage reset status cleared
VDDC_UV_RST_CLR	11	w	**VDDC undervoltage reset status clear** 0_B **Not_Cleared**, VDDC undervoltage reset status not cleared 1_B **Cleared**, VDDC undervoltage reset status cleared
SEC_STACK_RST_CLR	12	w	**Secure stack overflow reset status clear** 0_B **Not_Cleared**, Secure stack overflow reset status not cleared 1_B **Cleared**, Secure stack overflow reset status cleared
TMS_RST_CLR	13	w	**TMS reset Status Clear** 0_B **Not_Cleared**, TMS reset status not cleared 1_B **Cleared**, TMS reset status cleared
RES	31:14	r	**Reserved**

Reset status set register

RESET_STS_SET
Reset status set register **(0054_H)** **RESET_TYPE_1 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		TMS_RST_SET	SEC_STACK_RST_SET	VDDC_UV_RST_SET	VDDP_UV_RST_SET	LOCKUP_RST_SET	SOFT_RST_SET	WDT_MCU_RST_SET	FSWD_RST_SET	PIN_RST_SET	STOPEX_RST_SET	SLEEPEX_RST_SET	FS_SLEEPEX_RST_SET	MCLK_WD_RST_SET	VMSUP_UV_RST_SET
r		w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
VMSUP_UV_RST_SET	0	w	**Master supply undervoltage reset status set** 0_B **Not_Set**, Master supply undervoltage status not set 1_B **Set**, Master supply undervoltage status set
MCLK_WD_RST_SET	1	w	**Master clock watchdog reset status set** 0_B **Not_Set**, Master clock watchdog reset status not set 1_B **Set**, Master clock watchdog reset status set
FS_SLEEPEX_RST_SET	2	w	**Fail Sleep mode exit reset status set** 0_B **Not_Set**, Fail Sleep exit reset status not set 1_B **Set**, Fail Sleep exit reset status set

Field	Bits	Type	Description
SLEEPEX_RST_SET	3	w	**Sleep mode exit reset status set** 0_B **Not_Set**, Sleep exit reset status not set 1_B **Set**, Sleep exit reset status set
STOPEX_RST_SET	4	w	**Stop mode exit reset status set** 0_B **Not_Set**, Stop exit reset status not set 1_B **Set**, Stop exit reset status set
PIN_RST_SET	5	w	**Pin reset status set** 0_B **Not_Set**, Pin reset status not set 1_B **Set**, Pin reset status set
FSWD_RST_SET	6	w	**Fail safe watchdog reset status set** 0_B **Not_Set**, Watchdog fail reset status not set 1_B **Set**, Watchdog fail reset status set
WDT_MCU_RST_SET	7	w	**MCU watchdog timer reset status set** 0_B **Not_Set**, Watchdog timer reset status not set 1_B **Set**, Watchdog timer reset status set
SOFT_RST_SET	8	w	**Soft reset status set** 0_B **Not_Set**, Soft reset status not set 1_B **Set**, Soft reset status set
LOCKUP_RST_SET	9	w	**ARM core lockup reset status set** 0_B **Not_Set**, Lockup reset status not set 1_B **Set**, Lockup reset status set
VDDP_UV_RST_SET	10	w	**VDDP undervoltage reset status set** 0_B **Not_Set**, VDDP undervoltage reset status not set 1_B **Set**, VDDP undervoltage reset status set
VDDC_UV_RST_SET	11	w	**VDDC undervoltage reset status set** 0_B **Not_Set**, VDDC undervoltage reset status not set 1_B **Set**, VDDC undervoltage reset status set
SEC_STACK_RST_SET	12	w	**Secure stack overflow reset status set** 0_B **Not_Set**, Secure stack overflow status not set 1_B **Set**, Secure stack overflow reset status set
TMS_RST_SET	13	w	**TNS reset status set** 0_B **Not_Set**, TMS reset status not set 1_B **Set**, TMS reset status set
RES	31:14	r	**Reserved**

Wake up filter control register

WAKE_FILT_CTRL
Wake up filter control register **(0058$_H$)** **RESET_TYPE_2 Value: 0000 0000$_H$**

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

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

Field	Bits	Type	Description
RES	0, 6:4, 7, 31:8	r	**Reserved**
MON_FT	1	rw	**MON wake up filter time selection** 0_B **FILT_0**, 16 us 1_B **FILT_1**, 32 us
GPIO_FT	3:2	rw	**GPIO wake up filter time selection** 00_B **FILT_0**, 4 us 01_B **FILT_1**, 8us 10_B **FILT_2**, 16 us 11_B **FILT_3**, 32 us

Cyclic sense / Cyclic wake control register

CYC_CTRL
Cyclic sense / Cyclic wake control register **(005C$_H$)** **RESET_TYPE_2 Value: 3700 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		CYC_WAKE_E01		CYC_WAKE_M03				RES		CYC_SENSE_E01		CYC_SENSE_M03			
r		rw		rw				r		rw		rw			

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

Field	Bits	Type	Description
CYC_SENSE_EN	0	rw	**Cyclic sense enable** 0_B **Disable**, Cyclic sense disabled 1_B **Enable**, Cyclic sense enabled
RES	3:1, 15:7, 23:22, 31:30	r	**Reserved**
CYC_ON_TIME	6:4	rw	**On time in cyclic sense mode** 000_B **ON_0**, 150 us 001_B **ON_1**, 250 us 010_B **ON_2**, 500 us 011_B **ON_3**, 1 ms 100_B **ON_4**, 2 ms 101_B **ON_5**, 5 ms 110_B **ON_6**, 10 ms 111_B **ON_7**, 20 ms
CYC_SENSE_M03	19:16	rw	**Cyclic sense timer mantissa** 0_H **MANT_1**, Mantissa value is 1 … F_H **MANT_16**, Mantissa value is 16
CYC_SENSE_E01	21:20	rw	**Cyclic sense timer exponent** 00_B **EXP_0**, Exponent value is 0 01_B **EXP_1**, Exponent value is 1 10_B **EXP_2**, Exponent value is 2 11_B **EXP_3**, Exponent value is 3
CYC_WAKE_M03	27:24	rw	**Cyclic wake timer mantissa** 0_H **MANT_1**, Mantissa value is 1 … F_H **MANT_16**, Mantissa value is 16
CYC_WAKE_E01	29:28	rw	**Cyclic wake timer exponent** 00_B **EXP_0**, Exponent value is 0 01_B **EXP_1**, Exponent value is 1 10_B **EXP_2**, Exponent value is 2 11_B **EXP_3**, Exponent value is 3

Register description PMU

GPIO wake control register

WAKE_GPIO_CTRLx (x=0-5)
GPIO wake control register **(0060_H+x*4)** **RESET_TYPE_2 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES			INP					RES					CYC	FA	RI
r			rw					r					rw	rw	rw

Field	Bits	Type	Description
RI	0	rw	**Rising edge wake enable** 0_B **Disable**, Rising edge wake disabled 1_B **Enable**, Rising edge wake enabled
FA	1	rw	**Falling edge wake enable** 0_B **Disable**, Falling edge wake disabled 1_B **Enable**, Falling edge wake enabled
CYC	2	rw	**Cyclic sense enable** 0_B **Disable**, Cyclic sense disabled 1_B **Enable**, Cyclic sense enabled
RES	7:3, 31:13	r	**Reserved**

Field	Bits	Type	Description
INP	12:8	rw	**GPIO input pointer** 00_H **P0_0**, P0.0 selected 01_H **P0_1**, P0.1 selected 02_H **P0_2**, P0.2 selected 03_H **P0_3**, P0.3 selected 04_H **P0_4**, P0.4 selected 05_H **P0_5**, P0.5 selected 06_H **P0_6**, P0.6 selected 07_H **P0_7**, P0.7 selected 08_H **P0_8**, P0.8 selected 09_H **P0_9**, P0.9 selected $0A_H$ **P1_0**, P1.0 selected $0B_H$ **P1_1**, P1.1 selected $0C_H$ **P1_2**, P1.2 selected $0D_H$ **P1_3**, P1.3 selected $0E_H$ **P1_4**, P1.4 selected $0F_H$ **P2_0**, P2.0 selected 10_H **P2_1**, P2.1 selected 11_H **P2_2**, P2.2 selected 12_H **P2_3**, P2.3 selected 13_H **P2_4**, P2.4 selected 14_H **P2_5**, P2.5 selected 15_H **P2_6**, P2.6 selected 16_H **P2_7**, P2.7 selected 17_H **P2_8**, P2.8 selected 18_H **P2_9**, P2.9 selected

MON input control register

MON_CTRLx (x=0-2)
MON input control register **(0078_H+x*4)** **RESET_TYPE_2 Value: 0000 0007_H**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										PD	PU	CYC_SENSE_EN	WAKE_FALL	WAKE_RISE	EN
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
EN	0	rw	**MON input enable** 0_B **Disable**, MON disabled 1_B **Enable**, MON enabled

Register description PMU

Field	Bits	Type	Description
WAKE_RISE	1	rw	**Rising edge wake enable** 0_B **Disable**, Rising edge wake disabled 1_B **Enable**, Rising edge wake enabled
WAKE_FALL	2	rw	**Falling edge wake enable** 0_B **Disable**, Falling edge wake disabled 1_B **Enable**, Falling edge wake enabled
CYC_SENSE_EN	3	rw	**Cyclic sense enable** 0_B **Disable**, Cyclic sense disabled 1_B **Enable**, Cyclic sense enabled
PU	4	rw	**Input pull up current Source enable** 0_B **Disable**, Pull up current source disabled 1_B **Enable**, Pull up current source enabled
PD	5	rw	**Input pull down current source enable** 0_B **Disable**, Pull down current source disabled 1_B **Enable**, Pull down current source enabled
RES	31:6	r	**Reserved**

MON input status register

MON_STS

MON input status register **(0084_H)** **RESET_TYPE_2 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
STS	2:0	r	**MON input status** Each bit represents the corresponding MON input status 0: LOW - MON input low 1: HIGH - MON input high
RES	31:3	r	**Reserved**

Register description PMU

Wake control register

WAKE_CTRL
Wake control register **(0088_H)** **RESET_TYPE_2 Value: $2000\ 0004_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		VDDC_RED_EN	RES	RES			VSDOV_WAKE_EN	VDDEXT_UV_WAKE_EN	VDDEXT_OT_WAKE_EN	VDDC_HCM_WAKE_EN	VDDC_OV_WAKE_EN	VDDC_UVWARN_WAKE	VDDP_HCM_WAKE_EN	VDDP_OV_WAKE_EN	VDDP_UVWARN_WAKE
r		rw	r	r			rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	MON_WAKE_EN			RES		GPIO_WAKE_EN						RES	CYC_WAKE_EN	RES	CAN_WAKE_EN
r	rw			r		rw						r	rw	r	rw

Field	Bits	Type	Description
CAN_WAKE_EN	0	rw	**CAN wake enable** 0_B **Disable**, CAN wake disabled 1_B **Enable**, CAN wake enabled
RES	1, 3, 11:10, 15, 27:25, 28, 31:30	r	**Reserved**
CYC_WAKE_EN	2	rw	**Cyclic wake enable** 0_B **Disable**, Cyclic wake disabled 1_B **Enable**, Cyclic wake enabled
GPIO_WAKE_EN	9:4	rw	**GPIO wake enable** Each bit represents the corresponding GPIO wake enable 0: Disable - GPIO wake disabled 1: Enable - GPIO wake enabled
MON_WAKE_EN	14:12	rw	**MON wake enable** Each bit represents the corresponding MON wake enable 0: Disable - MON wake disabled 1: Enable - MON wake enabled
VDDP_UVWARN_WAKE_EN	16	rw	**Stop mode VDDP undervoltage warning wake enable** 0_B **Disable**, VDDP undervoltage warning wake disabled 1_B **Enable**, VDDP undervoltage warning wake enabled
VDDP_OV_WAKE_EN	17	rw	**Stop mode VDDP overvoltage wake enable** 0_B **Disable**, VDDP overvoltage wake disabled 1_B **Enable**, VDDP overvoltage wake enabled

Field	Bits	Type	Description
VDDP_HCM_WAKE_EN	18	rw	**Stop mode VDDP high current mode wake enable** 0_B **Disable**, VDDP high current mode wake disabled 1_B **Enable**, VDDP high current mode wake enabled
VDDC_UVWARN_WAKE_EN	19	rw	**Stop mode VDDC undervoltage warning wake enable** 0_B **Disable**, VDDC undervoltage warning wake disabled 1_B **Enable**, VDDC undervoltage warning wake enabled
VDDC_OV_WAKE_EN	20	rw	**Stop mode VDDC overvoltage wake enable** 0_B **Disable**, VDDC overvoltage wake disabled 1_B **Enable**, VDDC overvoltage wake enabled
VDDC_HCM_WAKE_EN	21	rw	**Stop mode VDDC high current mode wake enable** 0_B **Disable**, VDDC high current mode wake disabled 1_B **Enable**, VDDC high current mode wake enabled
VDDEXT_OT_WAKE_EN	22	rw	**Stop mode VDDEXT overtemperature wake enable** 0_B **Disable**, VDDEXT overtemperature wake disabled 1_B **Enable**, VDDEXT overtemperature wake enabled
VDDEXT_UV_WAKE_EN	23	rw	**Stop mode VDDEXT undervoltage wake enable** 0_B **Disable**, VDDEXT undervoltage wake disabled 1_B **Enable**, VDDEXT undervoltage wake enabled
VSDOV_WAKE_EN	24	rw	**VSD overvoltage wake enable** 0_B **Disable**, VSD overvoltage wake disabled 1_B **Enable**, VSD overvoltage wake enabled
VDDC_RED_EN	29	rw	**Stop mode - VDDC output voltage reduction enable** 0_B **Disable**, VDDC is not reduced in Stop mode 1_B **Enable**, VDDC is reduced in Stop mode

Wake status register

WAKE_STS
Wake status register **(008C$_H$)** **RESET_TYPE_1 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES							VSD_OV	VDDEXT_UV	VDDEXT_OT	VDDC_HCM	VDDC_OV	VDDC_UVWARN	VDDP_HCM	VDDP_OV	VDDP_UVWARN
r							rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	MON			RES		GPIO						RES	CYC_WAKE	RES	CAN
r	rhxr			r		rhxr						r	rhxr	r	rhxr

Field	Bits	Type	Description
CAN	0	rhxr	**CAN wake status** 0_B **INACTIVE**, No CAN wake occurred 1_B **ACTIVE**, CAN wake occurred

Field	Bits	Type	Description
RES	1, 3, 11:10, 15, 31:25	r	**Reserved**
CYC_WAKE	2	rhxr	**Cyclic wake status** 0_B **INACTIVE**, No cyclic wake occurred 1_B **ACTIVE**, Cyclic wake occurred
GPIO	9:4	rhxr	**GPIO wake status** Each bit represents the corresponding GPIO wake status 0: INACTIVE - no GPIO wake occurred 1: ACTIVE - GPIO wake occurred
MON	14:12	rhxr	**MON wake status** Each bit represents the corresponding MON wake status 0: INACTIVE - no MON wake occured 1: ACTIVE - MON wake occured
VDDP_UVWARN	16	rhxre	**VDDP undervoltage warning wake status** 0_B **INACTIVE**, No VDDP undervoltage warning wake occurred 1_B **ACTIVE**, VDDP undervoltage warning wake occurred
VDDP_OV	17	rhxre	**VDDP overvoltage wake status** 0_B **INACTIVE**, No VDDP overvoltage wake occurred 1_B **ACTIVE**, VDDP overvoltage wake occurred
VDDP_HCM	18	rhxre	**VDDP high current mode wake Status** 0_B **INACTIVE**, No VDDP high current mode wake occurred 1_B **ACTIVE**, VDDP high current mode wake occurred
VDDC_UVWARN	19	rhxre	**VDDC undervoltage warning wake status** 0_B **INACTIVE**, No VDDC undervoltage warning wake occurred 1_B **ACTIVE**, VDDC undervoltage warning wake occurred
VDDC_OV	20	rhxre	**VDDC overvoltage wake status** 0_B **INACTIVE**, No VDDC overvoltage wake occurred 1_B **ACTIVE**, VDDC overvoltage wake occurred
VDDC_HCM	21	rhxre	**VDDC high current mode wake status** 0_B **INACTIVE**, No VDDC high current mode wake occurred 1_B **ACTIVE**, VDDC high current mode wake occurred
VDDEXT_OT	22	rhxre	**VDDEXT overtemperature wake status** 0_B **INACTIVE**, No VDDEXT overtemperature wake occurred 1_B **ACTIVE**, VDDEXT overtemperature wake occurred
VDDEXT_UV	23	rhxre	**VDDEXT undervoltage wake status** 0_B **INACTIVE**, No VDDEXT undervoltage wake occurred 1_B **ACTIVE**, VDDEXT undervoltage wake occurred
VSD_OV	24	rhxre	**VSD overvoltage wake status** 0_B **INACTIVE**, No VSD overvoltage wake occurred 1_B **ACTIVE**, VSD overvoltage wake occurred

Register description PMU

Wake status clear register

WAKE_STS_CLR
Wake status clear register **(0090$_H$)** **RESET_TYPE_1 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES							VSD_OV_CLR	VDDEXT_UV_CLR	VDDEXT_OT_CLR	VDDC_HCM_CLR	VDDC_OV_CLR	VDDC_UVWARN_CLR	VDDP_HCM_CLR	VDDP_OV_CLR	VDDP_UVWARN_CLR
r							w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	MON_CLR			RES		GPIO_CLR						RES	CYC_WAKE_CLR	RES	CAN_CLR
r	w			r		w						r	w	r	w

Field	Bits	Type	Description
CAN_CLR	0	w	**CAN wake status clear** 0_B **Not_Cleared**, CAN wake status not cleared 1_B **Cleared**, CAN wake status cleared
RES	1, 3, 11:10, 15, 31:25	r	**Reserved**
CYC_WAKE_CLR	2	w	**Cyclic wake status** 0_B **Not_Cleared**, Cyclic wake status not cleared 1_B **Cleared**, Cyclic wake status cleared
GPIO_CLR	9:4	w	**GPIO wake status** Each bit represents the corresponding GPIO wake status clear 0: Not cleared - GPIO wake status not cleared 1: Cleared - GPIO wake status cleared
MON_CLR	14:12	w	**MON wake status clear** Each bit represents the corresponding MON wake status clear 0: Not cleared - MON wake status not cleared 1: Cleared - MON wake status cleared
VDDP_UVWARN_CLR	16	w	**VDDP undervoltage warning wake status clear** 0_B **Not_Cleared**, VDDP undervoltage warning wake status not cleared 1_B **Cleared**, VDDP undervoltage warning wake status cleared
VDDP_OV_CLR	17	w	**VDDP overvoltage wake status clear** 0_B **Not_Cleared**, VDDP overvoltage wake status not cleared 1_B **Cleared**, VDDP overvoltage wake status cleared

Field	Bits	Type	Description
VDDP_HCM_CLR	18	w	**VDDP high current mode wake status clear** 0_B **Not_Cleared**, VDDP high current mode wake status not cleared 1_B **Cleared**, VDDP high current mode wake status cleared
VDDC_UVWARN_CLR	19	w	**VDDC undervoltage warning wake status clear** 0_B **Not_Cleared**, VDDC undervoltage warning wake status not cleared 1_B **Cleared**, VDDC undervoltage warning wake status cleared
VDDC_OV_CLR	20	w	**VDDC overvoltage wake status clear** 0_B **Not_Cleared**, VDDC overvoltage wake status not cleared 1_B **Cleared**, VDDC overvoltage wake status cleared
VDDC_HCM_CLR	21	w	**VDDC high current mode wake status clear** 0_B **Not_Cleared**, VDDC high current mode wake status not cleared 1_B **Cleared**, VDDP high current mode wake status cleared
VDDEXT_OT_CLR	22	w	**VDDEXT overtemperature wake status clear** 0_B **Not_Cleared**, VDDEXT overtemperature wake status not cleared 1_B **Cleared**, VDDEXT overtemperature wake status cleared
VDDEXT_UV_CLR	23	w	**VDDEXT undervoltage wake status clear** 0_B **Not_Cleared**, VDDEXT undervoltage wake status not cleared 1_B **Cleared**, VDDEXT undervoltage wake status cleared
VSD_OV_CLR	24	w	**VSD undervoltage wake status clear** 0_B **Not_Cleared**, VSD overvoltage wake status not cleared 1_B **Cleared**, VSD overvoltage wake status cleared

Wake status set register

WAKE_STS_SET
Wake status set register **(0094_H)** **RESET_TYPE_1 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES							VSD_OV_SET	VDDEXT_UV_SET	VDDEXT_OT_SET	VDDC_HCM_SET	VDDC_OV_SET	VDDC_UVWARN_SET	VDDP_HCM_SET	VDDP_OV_SET	VDDP_UVWARN_SET
			r				w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	MON_SET			RES		GPIO_SET						RES	CYC_WAKE_SET	RES	CAN_SET
r	w			r		w						r	w	r	w

Field	Bits	Type	Description
CAN_SET	0	w	**CAN wake status set** 0_B **Not_Set**, CAN wake status not set 1_B **Set**, CAN wake status set
RES	1, 3, 11:10, 15, 31:25	r	**Reserved**
CYC_WAKE_SET	2	w	**Cyclic wake status set** 0_B **Not_Set**, Cyclic wake status not set 1_B **Set**, Cyclic wake status set
GPIO_SET	9:4	w	**GPIO wake status set** Each bit represents the corresponding GPIO wake status set 0: Not set - GPIO wake status not set 1: Set- GPIO wake status set
MON_SET	14:12	w	**MON wake status set** Each bit represents the corresponding MON wake status set 0: Not set - MON wake status not set 1: Set- MON wake status set
VDDP_UVWARN_SET	16	w	**VDDP undervoltage warning wake status set** 0_B **Not_Set**, VDDP undervoltage warning wake status not set 1_B **Set**, VDDP undervoltage warning wake status set
VDDP_OV_SET	17	w	**VDDP overvoltage wake status set** 0_B **Not_Set**, VDDP overvoltage wake status not set 1_B **Set**, VDDP overvoltage wake status set
VDDP_HCM_SET	18	w	**VDDP high current mode wake status set** 0_B **Not_Set**, VDDP high current mode wake status not set 1_B **Set**, VDDP high current mode wake status set
VDDC_UVWARN_SET	19	w	**VDDC undervoltage warning wake status set** 0_B **Not_Set**, VDDC undervoltage warning wake status not set 1_B **Set**, VDDC undervoltage warning wake status set
VDDC_OV_SET	20	w	**VDDC overvoltage wake status set** 0_B **Not_Set**, VDDC overvoltage wake status not set 1_B **Set**, VDDC overvoltage wake status set
VDDC_HCM_SET	21	w	**VDDC high current mode wake status set** 0_B **Not_Set**, VDDC high current mode wake status not set 1_B **Set**, VDDC high current mode wake status set
VDDEXT_OT_SET	22	w	**VDDEXT overtemperature wake status set** 0_B **Not_Set**, VDDEXT overtemperature wake status not set 1_B **Set**, VDDEXT overtemperature wake status set
VDDEXT_UV_SET	23	w	**VDDEXT undervoltage wake status set** 0_B **Not_Set**, VDDEXT undervoltage wake status not set 1_B **Set**, VDDEXT undervoltage wake status set

Register description PMU

Field	Bits	Type	Description
VSD_OV_SET	24	w	**VSD overvoltage wake status set** 0_B **Not_Set**, VSD overvoltage wake status not set 1_B **Set**, VSD overvoltage wake status set

General purpose user data register

GPUDATAx (x=0-2)
General purpose user data register **(0098_H+x*4)** **RESET_TYPE_0 Value: 0000 0000_H**

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

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

Field	Bits	Type	Description
DATA	31:0	rw	**Data storage**

Miscellaneous control register

MISC_CTRL
Miscellaneous control register **($00D8_H$)** **RESET_TYPE_2 Value: 0000 0003_H**

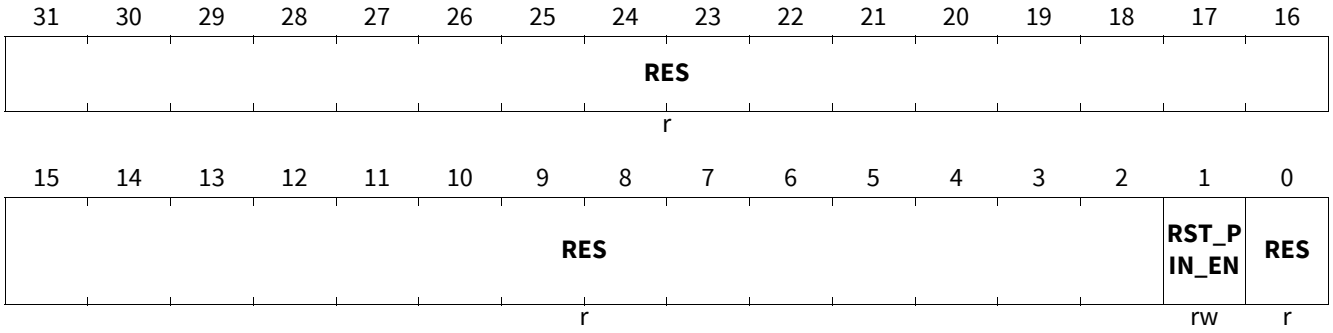
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															TRIG_RST
r															rw

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

Field	Bits	Type	Description
FI_PU_EN	0	rw	**Failure input pull up enable** 0_B **Disable**, Failure input pull up disabled 1_B **Enable**, Failure Input pull up enabled
RES	15:1, 31:17	r	**Reserved**
TRIG_RST	16	rw	**Trigger Pin Reset** 0_B **INACTIVE**, No Pin Reset trigger 1_B **ACTIVE**, Trigger Pin Reset

Start configuration control register

START_CONFIG
Start configuration control register **($00E4_H$)** **Reset Value: Table 74**



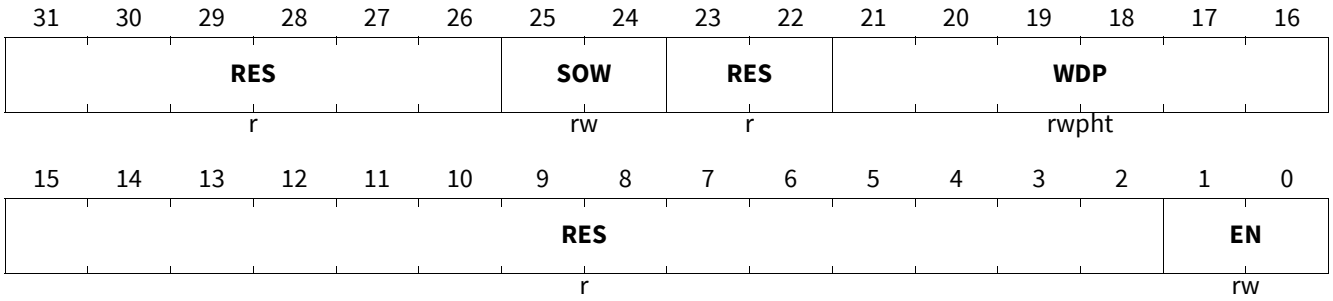
Field	Bits	Type	Description
RES	0, 31:2	r	**Reserved**
RST_PIN_EN	1	rw	**Reset PIN enable** 0_B **Disable**, Pin P0.10 used as GPIO 1_B **Enable**, Pin P0.10 used as RESET pin

Table 74 Reset Values of START_CONFIG

Reset Type	Reset Value	Note
RESET_TYPE_1	$0000\ 0000_H$	RESET_TYPE_1
TRIM_100_TP	$0000\ 0000_H$	RESET

Watchdog control register

WD_CTRL
Watchdog control register **(0700_H)** **RESET_TYPE_4 Value: $0000\ 0003_H$**



Field	Bits	Type	Description
EN	1:0	rw	**Watchdog enable** 00_B **Disable**, Watchdog disabled 01_B **Enable**, Watchdog enabled (do not use) 10_B **Enable**, Watchdog enabled (do not use) 11_B **Enable**, Watchdog enabled

Register description PMU

Field	Bits	Type	Description
RES	15:2, 23:22, 31:26	r	**Reserved**
WDP	21:16	rwpht	**Watchdog period selection** The watchdog period is only programable as long as the watchdog is in LONG OPEN window 00_H **WP_0**, Watchdog period 16 ms 01_H **WP_1**, Watchdog period = 1 * 16 ms … $3F_H$ **WP_63**, Watchdog period = 63 * 16 ms
SOW	25:24	rw	**Short open window selection - Can only be configured in LOW** 00_B **Disabled**, 01_B **SOW1**, two successive short open window allowed 10_B **SOW2**, three successive short open windows allowed 11_B **SOW3**, four successive short open windows allowed

Watchdog trigger register

WD_TRIG
Watchdog trigger register **(0704_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
TRIG	0	rw	**Trigger bit for long open- /window Mode** This bit needs to be toggled in order to provide a proper trigger
RES	31:1	r	**Reserved**

Watchdog SOW trigger register

WD_TRIG_SOW
Watchdog SOW trigger register **(0708_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
TRIG	0	rw	**Trigger bit for short open window mode** This bit needs to be toggled in order to provide a proper trigger
RES	31:1	r	**Reserved**

Functional safety status register

FS_STS

Functional safety status register **(070C$_H$)** **RESET_TYPE_0 Value: 0000 0002$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															FO_OC_STS
r															rhxr

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	PIN_MON_STS	CSC_EN_FAIL_STS	CSC_BIST_FAIL_STS	CSC_OC_STS	VAREF_OV_STS	VDDP_OT_STS	VDDP_OV_STS	VDDP_UV_STS	VDDC_OV_STS	VDDC_UV_STS	WD_TEST_FAIL_STS	WD_FAIL_STS	VMSUP_OV_STS	VMSUP_UV_STS	MCLK_FAIL_STS
r	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr	rhxr

Field	Bits	Type	Description
MCLK_FAIL_STS	0	rhxr	**Master clock watchdog fail status** 0_B **INACTIVE**, No master clock watchdog fail occurred 1_B **ACTIVE**, Master clock watchdog fail occurred
VMSUP_UV_STS	1	rhxr	**Master supply undervoltage status** 0_B **INACTIVE**, No undervoltage occurred 1_B **ACTIVE**, undervoltage occurred
VMSUP_OV_STS	2	rhxr	**Master supply overvoltage status** 0_B **INACTIVE**, No overvoltage occurred 1_B **ACTIVE**, overvoltage occurred
WD_FAIL_STS	3	rhxr	**Watchdog Fail status** 0_B **INACTIVE**, No watchdog fail occurred 1_B **ACTIVE**, watchdog fail occurred
WD_TEST_FAIL_STS	4	rhxr	**Watchdog self-test fail status** 0_B **INACTIVE**, No watchdog self-test fail occurred 1_B **ACTIVE**, watchdog self-test fail occurred
VDDC_UV_STS	5	rhxr	**VDDC undervoltage status** 0_B **INACTIVE**, No undervoltage occurred 1_B **ACTIVE**, undervoltage occurred
VDDC_OV_STS	6	rhxr	**VDDC overvoltage status** 0_B **INACTIVE**, No overvoltage occurred 1_B **ACTIVE**, overrvoltage occurred

Field	Bits	Type	Description
VDDP_UV_STS	7	rhxr	**VDDP undervoltage status** 0_B **INACTIVE**, No undervoltage occurred 1_B **ACTIVE**, undervoltage occurred
VDDP_OV_STS	8	rhxr	**VDDP overvoltage status** 0_B **INACTIVE**, No overvoltage occurred 1_B **ACTIVE**, overrvoltage occurred
VDDP_OT_STS	9	rhxr	**VDDP overtemperature status** 0_B **INACTIVE**, No VDDP overtemperature occurred 1_B **ACTIVE**, VDDP overtemperature occurred
VAREF_OV_STS	10	rhxr	**VAREF overvoltage status** 0_B **INACTIVE**, No overvoltage occurred 1_B **ACTIVE**, overrvoltage occurred
CSC_OC_STS	11	rhxr	**Current Sense comparator overcurrent status** 0_B **INACTIVE**, No overcurrent occurred 1_B **ACTIVE**, overcurrent occurred
CSC_BIST_FAIL_STS	12	rhxr	**Current sense comparator self-test fail status** 0_B **INACTIVE**, No current sense comparator self-test fail occurred 1_B **ACTIVE**, Current sense comparator self-test fail occurred
CSC_EN_FAIL_STS	13	rhxr	**Current sense comparator enabling fail status** 0_B **INACTIVE**, No current sense comparator enable fail occurred 1_B **ACTIVE**, Bridge driver is enabled without having CSC and VAREF enabled
PIN_MON_STS	14	rhxr	**Pin monitor fail status (FI)** 0_B **INACTIVE**, No pin monitor fail occurred 1_B **ACTIVE**, Pin monitor fail occurred
RES	15, 31:17	r	**Reserved**
FO_OC_STS	16	rhxr	**FO overcurrent status (FO)** 0_B **INACTIVE**, No FO overcurrent occurred 1_B **ACTIVE**, FO overcurrent occurred

Functional safety status clear register

FS_STS_CLR
Functional safety status clear register **(0710_H)** **RESET_TYPE_0 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															FO_OC_STS_CLR
							r								w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	PIN_MON_STS_CLR	CSC_EN_FAIL_STS_CLR	CSC_BIST_FAIL_STS_CL	CSC_OC_STS_CLR	VAREF_OV_STS_CLR	VDDP_OT_STS_CLR	VDDP_OV_STS_CLR	VDDP_UV_STS_CLR	VDDC_OV_STS_CLR	VDDC_UV_STS_CLR	WD_TEST_FAIL_STS_CL	WD_FAIL_STS_CLR	VMSUP_OV_STS_CLR	VMSUP_UV_STS_CLR	MCLK_FAIL_STS_CLR
r	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
MCLK_FAIL_STS_CLR	0	w	**Master clock watchdog fail status** 0_B **Not_Cleared**, Master clock fail status not cleared 1_B **Cleared**, Master clock fail status cleared
VMSUP_UV_STS_CLR	1	w	**Master supply undervoltage status** 0_B **Not_Cleared**, Master supply undervoltage status not cleared 1_B **Cleared**, Master supply undervoltage status cleared
VMSUP_OV_STS_CLR	2	w	**Master supply overvoltage status** 0_B **Not_Cleared**, Master supply overvoltage status not cleared 1_B **Cleared**, Master supply overvoltage status cleared
WD_FAIL_STS_CLR	3	w	**Watchdog fail status** 0_B **Not_Cleared**, Watchdog fail status not cleared 1_B **Cleared**, Watchdog fail status cleared
WD_TEST_FAIL_STS_CLR	4	w	**Watchdog self test fail status** 0_B **Not_Cleared**, Watchdog self test fail status not cleared 1_B **Cleared**, Watchdog self test fail status cleared
VDDC_UV_STS_CLR	5	w	**VDDC undervoltage status** 0_B **Not_Cleared**, VDDC undervoltage status not cleared 1_B **Cleared**, VDDC undervoltage status cleared
VDDC_OV_STS_CLR	6	w	**VDDC overvoltage status** 0_B **Not_Cleared**, VDDC overvoltage status not cleared 1_B **Cleared**, VDDC overvoltage status cleared
VDDP_UV_STS_CLR	7	w	**VDDP undervoltage status** 0_B **Not_Cleared**, VDDP undervoltage status not cleared 1_B **Cleared**, VDDP undervoltage status cleared
VDDP_OV_STS_CLR	8	w	**VDDP overvoltage status** 0_B **Not_Cleared**, VDDP overvoltage status not cleared 1_B **Cleared**, VDDP overvoltage status cleared

Field	Bits	Type	Description
VDDP_OT_STS_CLR	9	w	**VDDP overtemperature status** 0_B **Not_Cleared**, VDDP overtemperature status not cleared 1_B **Cleared**, VDDP overtemperature status cleared
VAREF_OV_STS_CLR	10	w	**VAREF overvoltage status** 0_B **Not_Cleared**, VAREF overvoltage status not cleared 1_B **Cleared**, VAREF overvoltage status cleared
CSC_OC_STS_CLR	11	w	**Current sense comparator overcurrent status** 0_B **Not_Cleared**, Current sense comparator overcurrent status not cleared 1_B **Cleared**, Current sense comparator overcurrent status cleared
CSC_BIST_FAIL_STS_CLR	12	w	**Current sense comparator self-test fail status** 0_B **Not_Cleared**, Current sense comparator self-test status not cleared 1_B **Cleared**, Current sense comparator self-test status cleared
CSC_EN_FAIL_STS_CLR	13	w	**Current sense comparator enable fail status** 0_B **Not_Cleared**, Current sense comparator enable fail status not cleared 1_B **Cleared**, Current sense comparator enable fail status cleared
PIN_MON_STS_CLR	14	w	**Pin monitor fail status (FI)** 0_B **Not_Cleared**, Pin monitoring status not cleared 1_B **Cleared**, Pin monitoring status cleared
RES	15, 31:17	r	**Reserved**
FO_OC_STS_CLR	16	w	**FO overcurrent status clear (FO)** 0_B **Not_Cleared**, FO overcurrent status not cleared 1_B **Cleared**, FO overcurrent status cleared

Functional safety status set register

FS_STS_SET
Functional safety status set register **(0714$_H$)** **RESET_TYPE_0 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															FO_OC_STS_SET
							r								w

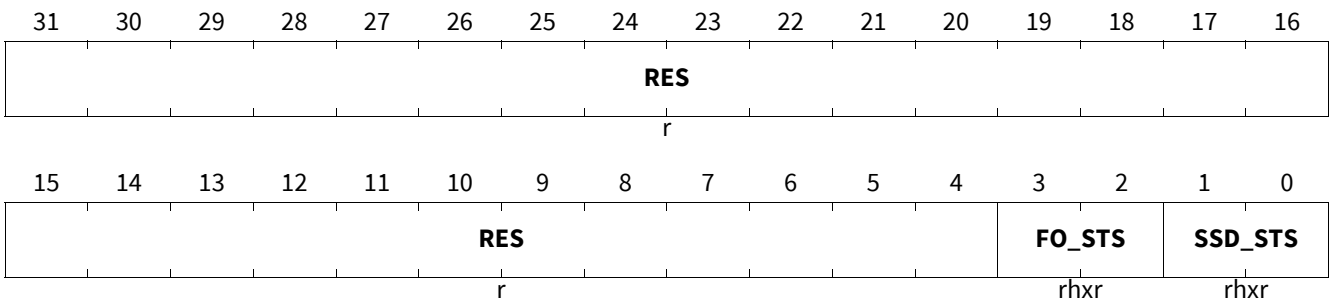
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	PIN_MON_STS_SET	CSC_EN_FAIL_STS_SET	CSC_BIST_FAIL_STS_SE	CSC_OC_STS_SET	VAREF_OV_STS_SET	VDDP_OT_STS_SET	VDDP_OV_STS_SET	VDDP_UV_STS_SET	VDDC_OV_STS_SET	VDDC_UV_STS_SET	WD_TEST_FAIL_STS_SE	WD_FAIL_STS_SET	VMSUP_OV_STS_SET	VMSUP_UV_STS_SET	MCLK_FAIL_STS_SET
r	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
MCLK_FAIL_STS_SET	0	w	**Master clock watchdog fail status** 0_B **Not_Set**, Master clock fail status not set 1_B **Set**, Master clock fail status set
VMSUP_UV_STS_SET	1	w	**Master supply undervoltage status** 0_B **Not_Set**, Master supply undervoltage status not set 1_B **Set**, Master supply undervoltage status set
VMSUP_OV_STS_SET	2	w	**Master supply overvoltage status** 0_B **Not_Set**, Master supply overvoltage status not set 1_B **Set**, Master supply overvoltage status set
WD_FAIL_STS_SET	3	w	**Watchdog fail status** 0_B **Not_Set**, Watchdog fail status not set 1_B **Set**, Watchdog fail status set
WD_TEST_FAIL_STS_SET	4	w	**Watchdog self test fail status** 0_B **Not_Set**, Watchdog self test fail status not set 1_B **Set**, Watchdog self test fail status set
VDDC_UV_STS_SET	5	w	**VDDC undervoltage status** 0_B **Not_Set**, VDDC undervoltage status not set 1_B **Set**, VDDC undervoltage status set
VDDC_OV_STS_SET	6	w	**VDDC overvoltage status** 0_B **Not_Set**, VDDC overvoltage status not set 1_B **Set**, VDDC overvoltage status set
VDDP_UV_STS_SET	7	w	**VDDP undervoltage status** 0_B **Not_Set**, VDDP undervoltage status not set 1_B **Set**, VDDP undervoltage status set
VDDP_OV_STS_SET	8	w	**VDDP overvoltage status** 0_B **Not_Set**, VDDP Overvoltage status not set 1_B **Set**, VDDP Overvoltage status set

Field	Bits	Type	Description
VDDP_OT_STS_SET	9	w	**VDDP overtemperature status** 0_B **Not_Set**, VDDP overtemperature status not set 1_B **Set**, VDDP overtemperature status set
VAREF_OV_STS_SET	10	w	**VAREF overvoltage status** 0_B **Not_Set**, VAREF overvoltage status not set 1_B **Set**, VAREF overvoltage status set
CSC_OC_STS_SET	11	w	**Current sense comparator overcurrent status** 0_B **Not_Set**, Current sense comparator overcurrent status not set 1_B **Set**, Current sense comparator overcurrent status set
CSC_BIST_FAIL_STS_SET	12	w	**Current sense comparator self test fail status** 0_B **Not_Set**, Current sense comparator self test status not set 1_B **Set**, Current sense comparator selftest status set
CSC_EN_FAIL_STS_SET	13	w	**Current sense comparator enable fail status** 0_B **Not_Set**, Current sense comparator enable fail status not set 1_B **Set**, Current sense comparator enable fail status set
PIN_MON_STS_SET	14	w	**Pin monitor fail status (FI)** 0_B **Not_Set**, Pin monitoring status not set 1_B **Set**, Pin monitoring status set
RES	15, 31:17	r	**Reserved**
FO_OC_STS_SET	16	w	**FO overcurrent status set (FO)** 0_B **Not_Set**, FO overcurrent status not set 1_B **Set**, FO overcurrent status set

Safe state status register

FS_SSD
Safe state status register **(0718_H)** **RESET_TYPE_0 Value: 0000 000F$_H$**



Field	Bits	Type	Description
SSD_STS	1:0	rhxr	**Safe shutdown status** 00_B **INACTIVE**, Safe shutdown is inactive 01_B **ACTIVE**, Safe shutdown is active … 11_B **ACTIVE**, Safe shutdown is active
FO_STS	3:2	rhxr	**Fail output status (FO)** 00_B **INACTIVE**, Fail output is inactive 01_B **ACTIVE**, Fail output is active … 11_B **ACTIVE**, Fail output is active
RES	31:4	r	**Reserved**

Safe state status clear register

FS_SSD_CLR

Safe state status clear register **($071C_H$)** **RESET_TYPE_0 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														FO_STS_CLR	SSD_STS_CLR
r														w	w

Field	Bits	Type	Description
SSD_STS_CLR	0	w	**Safe shutdown status clear** 0_B **Not_Cleared**, Safe shutdown status not cleared 1_B **Cleared**, Safe shutdown status cleared
FO_STS_CLR	1	w	**Fail output status clear (FO)** 0_B **Not_Cleared**, Fail output status not cleared 1_B **Cleared**, Fail output status cleared
RES	31:2	r	**Reserved**

Register description PMU

Safe state status set register

FS_SSD_SET

Safe state status set register **(0720_H)** **RESET_TYPE_0 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														FO_STS_SET	SSD_STS_SET
r														w	w

Field	Bits	Type	Description
SSD_STS_SET	0	w	**Safe shutdown status set** 0_B **Not_Set**, Safe shutdown status not set 1_B **Set**, Safe shutdown status set
FO_STS_SET	1	w	**Fail output status set (FO)** 0_B **Not_Set**, Fail output status not set 1_B **Set**, Fail output status set
RES	31:2	r	**Reserved**

Current sense comparator control register

CSC_CTRL

Current sense comparator control register **(0724_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															CSC_DIS
r															w

Field	Bits	Type	Description
CSC_DIS	0	w	**Current sense comparator disable** 0_B **Enable**, CSC enabled 1_B **Disable**, CSC disabled
RES	31:1	r	**Reserved**

6 System Control Unit (SCU)

6.1 Features overview

The SCU provides following features:

- Flexible clock management with different clock sources and prescaler options. This allows a high flexibility for the operation modes and ensures a fail-safe behavior in case of a clock failure
- Flexible peripheral management when enabling and disabling peripherals, when switching the system states and when debugging. The SCU supports the shutdown for some peripherals and the whole system in case of a critical system state
- The assignment of interrupt and exception request events to the NVIC and DMA request events to the DMA module is done inside the SCU

6.2 Block diagram

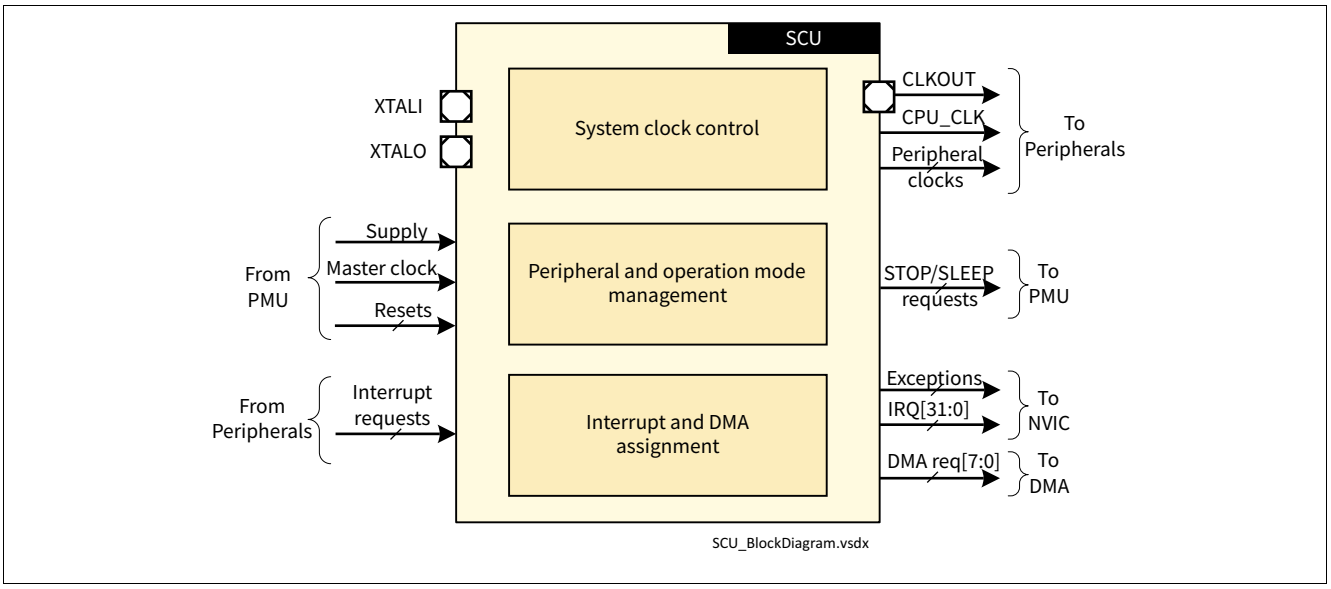


Figure 47 Block diagram SCU

6.3 Toplevel signals

For SCU interconnects refer to the **Product definitions**, **SCU interconnections**.

6.4 Interrupts

Interrupt nodes

The SCU routes the peripheral requests via 8 shared interrupt nodes SCU.INP[7:0]_IRQ0/1 of the 32 NVIC lines, see **Product definitions**, **Interrupt request mapping**.

There is one shared exception node (NMI) with multiple individual peripheral requests, see **Product definitions**, **Exception request mapping**.

DMA connections

The multiple individual peripheral DMA requests are connected to the 8 DMA channels, see **Product definitions**, **DMA request mapping**.

6.5 Operation mode behavior

Table 75 Operation mode behavior SCU

Reset	• The SCU has RESET_TYPE_3/4/5/6. An activated RESET has following effect: – All SFRs of the SCU are reset according to its reset type – The generated events show reset value
Power-up / Power-down	• The SCU is kept in reset state as long as supply is not in specified operating range • The SCU is released from reset state when supply is in specified operating range
Active mode	• The SCU is automatically enabled
Stop mode	• Some functions of the SCU are powered in STOP mode, some are not powered and lose the configuration
Sleep mode	• All functions are unpowered in SLEEP mode and lose the configuration
Fail-sleep mode	• All functions are unpowered in SLEEP mode and lose the configuration

6.6 System clock control

- Various clock prescaler options allow to balance the system performance and power consumption
- An independent adjustment of peripheral, filter and bus clocks allows a flexible scaling for optimum tailoring of the chip resources to the application needs
- The clock tree configuration offers an independent scaling of actuator peripherals (e.g. ADC, timers) from communication peripherals which have to ensure a constant baudrate (e.g. UART, CAN)
- A backup clock in case of a clock failure ensures a fail-safe behavior, i.e. allows the user to bring a started process to a defined end

6.6.1 System clock control block diagram

The system clock control consists of following blocks (see **Figure 48**):

- Clock sources
  - Internal HPOSC with HP_CLK and HPWDG
  - XTALOSC for external crystal or resonator at XTALI/O with XTAL_CLK and XTALWDG

  - Two PLLs PLL0 and PLL1 with lock detection LCKDET0/1
  - MCLK and REF_CLK are independent clocks coming from PMU
- Clock tree configuration
  - Clock selection and distribution for system clock SYS0_CLK
  - Clock selection and distribution for system clock SYS1_CLK
  - Clock selection for GPIO alternate function CLKOUT

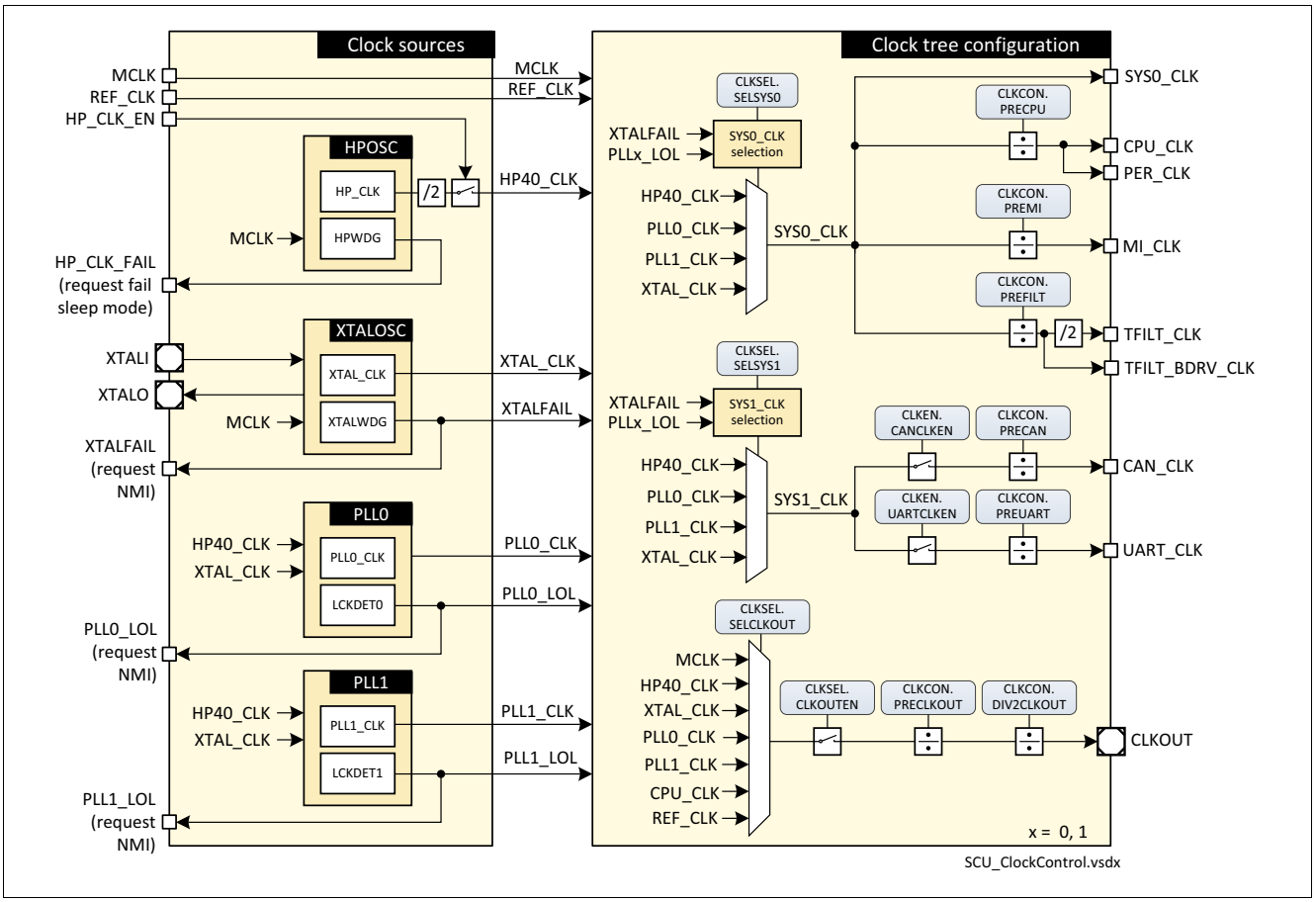


Figure 48 System clock control block diagram

6.6.2 Reset and interrupt events

Reset event

- The HPWDG event HP_CLK_FAIL requests a fail sleep mode (see WAKE_FAIL_STS.HPCLK_FAIL in the PMU chapter (**Chapter 5**)

Interrupt events

The following events are connected to the shared NMI exception node:

- XTAL oscillator watchdog event XTALFAIL triggers NMISR.NMIXTAL, see **Figure 49**
- PLL0 unlock event PLL0_LOL triggers NMISR.NMIPLL0, see **Figure 50**
- PLL1 unlock event PLL1_LOL triggers NMISR.NMIPLL1, see **Figure 50**

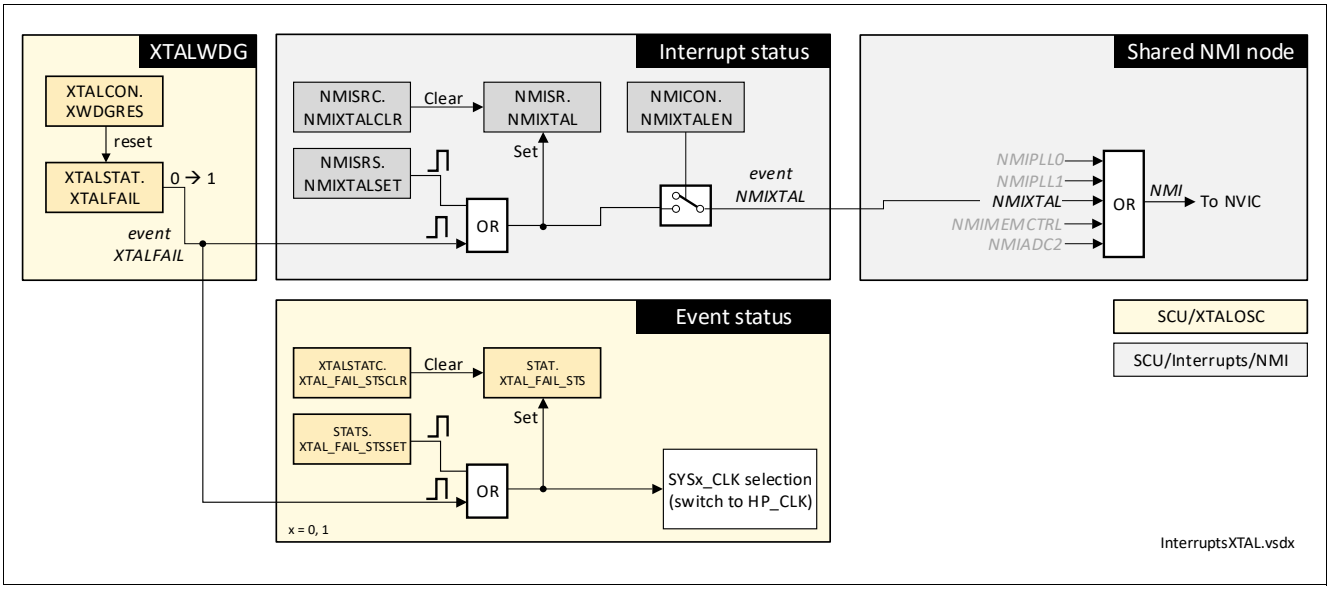


Figure 49 XTALOSC NMI

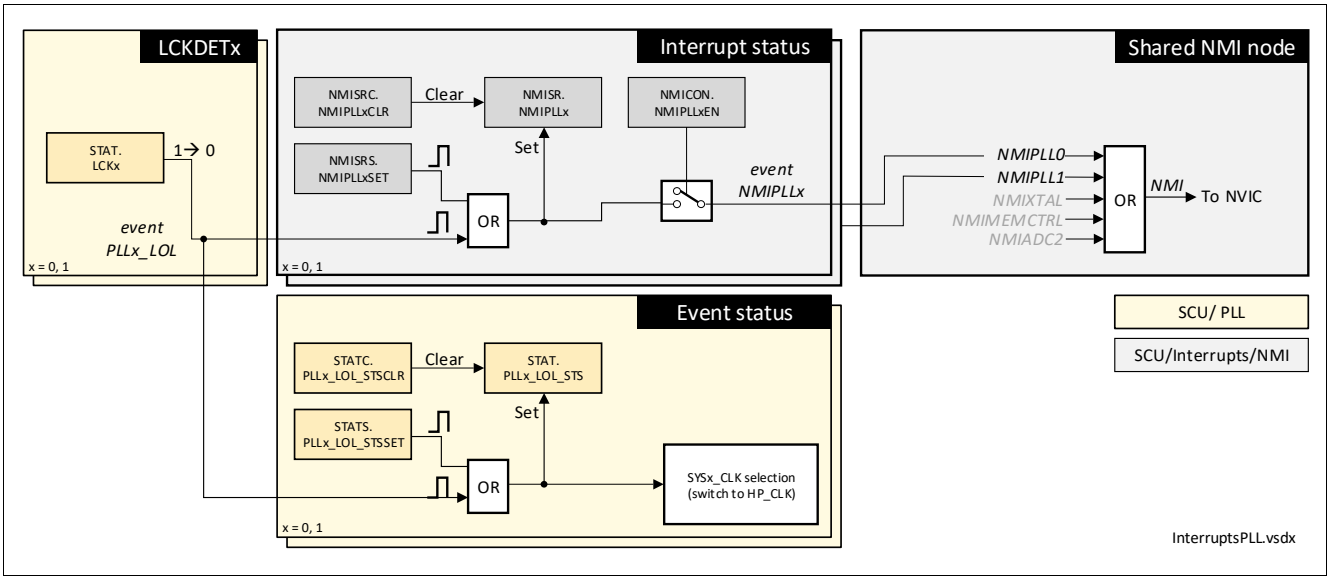


Figure 50 PLL NMIs

6.6.3 High precision oscillator (HPOSC)

The HPOSC consists of two blocks, the HP_CLK and the HPWDG.

6.6.3.1 HP_CLK

- The HPOSC is a high precision oscillator circuit with HP_CLK as the system's default and fallback clock
- The HP_CLK characteristics can be found in the datasheet
- The HP_CLK starts automatically and is always on in Active mode, controlled by the PMU signal PMU.HP_CLK_EN

6.6.3.2 HPWDG

The HPWDG is meant to protect the system from a critical situation caused by a too fast or too slow clock which is out of the operating range.

- The HPWDG is the clock watchdog for HP_CLK. It compares the HP_CLK against the MCLK
- The HPOSC has a clock watchdog HPWDG which detects if the HP_CLK deviates for more than ±65% from its nominal value
- The event HP_CLK_FAIL triggers a fail sleep mode. The event in stored inside the PMU, see WAKE_FAIL_STS.HPCLK_FAIL in the PMU chapter (**Chapter 5**)

6.6.3.3 HP_CLK and HPWDG behavior

The HP_CLK and HPWDG behave in the operation modes according to **Table 76**.

Table 76 HP_CLK and HPWDG behavior

Reset	• VDDC_UV resets/releases the HP_CLK and HPWDG
Power-up / Power-down	• The HP40_CLK and HPWDG are enabled/disabled with PMU.HP_CLK_EN
Active mode	• The HP_CLK and HPWDG are enabled automatically and cannot be disabled by user
Stop mode	• The HP_CLK and HPWDG are disabled automatically when Stop mode is requested • The HP_CLK and HPWDG are enabled automatically with the stop-to-active transition
Sleep mode	• The HP_CLK and HPWDG are disabled automatically when Sleep mode is requested • The HP_CLK and HPWDG are enabled automatically with the sleep-to-active transition
Fail-sleep mode	• In case of the HPWDG detects a fail, the Fail-sleep mode is entered

6.6.4 Crystal oscillator (XTALOSC)

The XTALOSC consists of two blocks, the XTAL_CLK and the XTALWDG.

6.6.4.1 XTAL_CLK

- The XTAL_CLK (fxtal) provides a very accurate clock to the system (e.g. for CAN baudrate) based on an external clock source. Following clock sources can be used:
  - A digital oscillator connected to XTALI pin, e.g. from another microcontroller or an oscillator circuit, see **Figure 51**. The input frequency range is defined by f_{XTALI}.
  - An analog oscillator connected to XTALI and XTALO pins, see **Figure 51**. The input frequency range is defined by f_{XTAL}.
- The oscillator circuit can be enabled via XTALCON.XPD=0. Once enabled the oscillation starts based on the external crystal or resonator.
- There is a shaper circuit which allows to shape the signal asymmetry and filters too fast oscillations or too low amplitudes. This can be useful to filter low amplitudes at the startup oscillation phase of a crystal oscillator or to balance the duty cycle of an external digital clock, see **Figure 52**. The effect of the shaper can be analyzed via selecting XTAL_CLK at the GPIO alternate function CLKOUT (CLKSEL.SELCLKOUT). The shaper settings can be found in **Table 77**.

Note: The oscillation startup time depends on the characteristics of the external components.

Table 77 Shaper configuration

XTALHSEN	XTALHYS	Hysteresis (typical value)	LP Filter (typical value)	Note
0	Don’t care	< 100 mV	Off	At high frequent clock input (typ. above 24 MHz) the hysteresis shall be switched off
1	00b	~ 350 mV	~ 24 ns	Recommended default setting for 16 MHz
1	01b	~ 300 mV	~ 18 ns	
1	10b	~ 200 mV	~ 6 ns	
1	11b	~ 150 mV	Off	

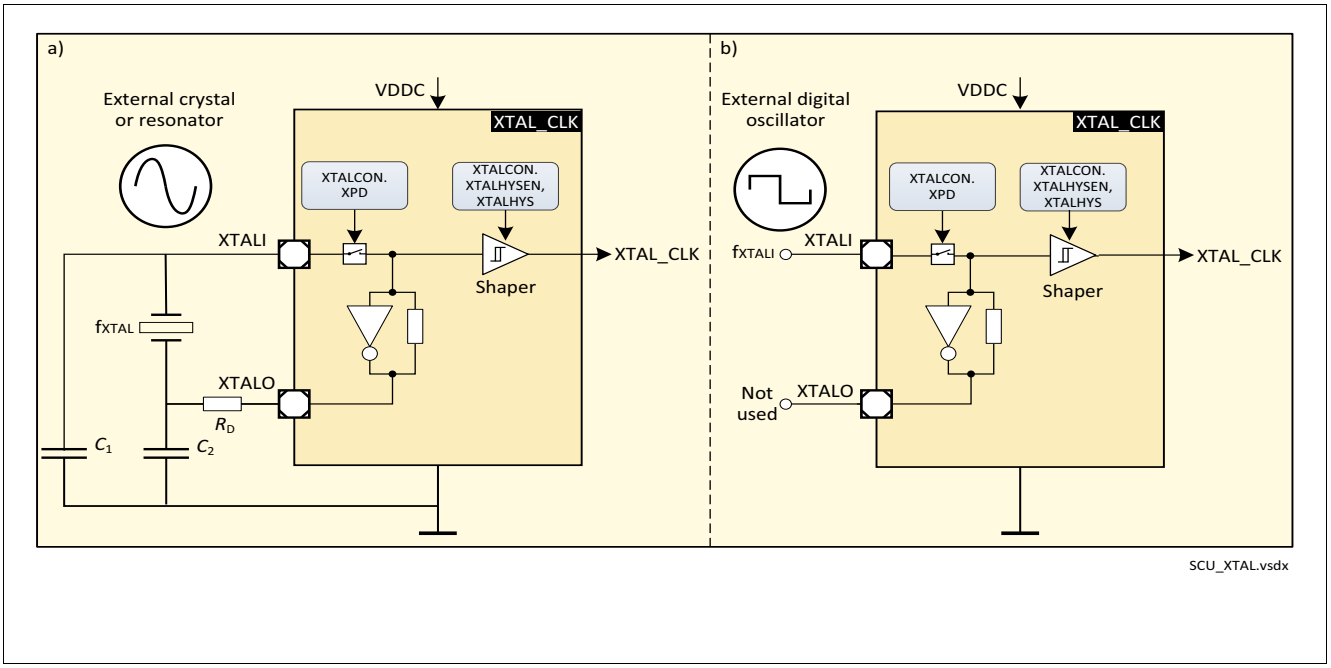


Figure 51 XTAL_CLK with a) external crystal/resonator and b) external digital clock

6.6.4.2 XTALWDG

The XTALWDG is meant to protect the system from a critical situation caused by a too fast or too slow clock which is out of the operating range.

- The XTALOSC has a clock watchdog XTALWDG which detects missing clock or too fast clock.
  - The XTALWDG has to be enabled via XTALCON.XWDGEN
  - The status can be monitored in XTALSTAT.XTALFAIL and is registered in bit XTALSTAT.XTAL_FAIL_STS
  - The event XTALFAIL triggers an exception NMISR.NMIXTAL
  - The clock selection for SYS0_CLK and SYS1_CLK (CLKSEL.SELSYS0/SELSYS1) is switched to HP_CLK automatically in case XTAL_CLK is selected
  - The XTALWDG counter can be reset and restarted via XTALCON.XWDGRES

6.6.4.3 XTAL_CLK and XTALWDG behavior

The XTAL_CLK behaves in the operation modes according to **Table 78**.

Table 78 XTAL_CLK behavior

Reset	• RESET_TYPE_4 and RESET_TYPE_6 control the configuration and status SFRs • At active reset, the oscillation stops
Power-up / Power-down	• At power-up the XTAL_CLK and XTALWDG are disabled. • At power-down the XTAL_CLK is reset by RESET_TYPE_4/6
Active mode	• The XTAL_CLK and XTALWDG can be enabled by user
Stop mode	• The XTAL_CLK configuration is reset when Stop mode is entered. The oscillation stops
Sleep mode	• The XTAL_CLK configuration is reset when Sleep mode is entered. The oscillation stops

Table 78 XTAL_CLK behavior (cont'd)

Fail sleep mode	• None
fail-safe	• XTALFAIL switches the clock source for the system clock (SYS0_CLK, SYS1_CLK) to HP_CLK automatically in case XTAL_CLK is selected; a NMIXTAL is requested

6.6.4.4 XTALOSC programmer's guide

The XTALOSC startup sequence is described in **Figure 52**.

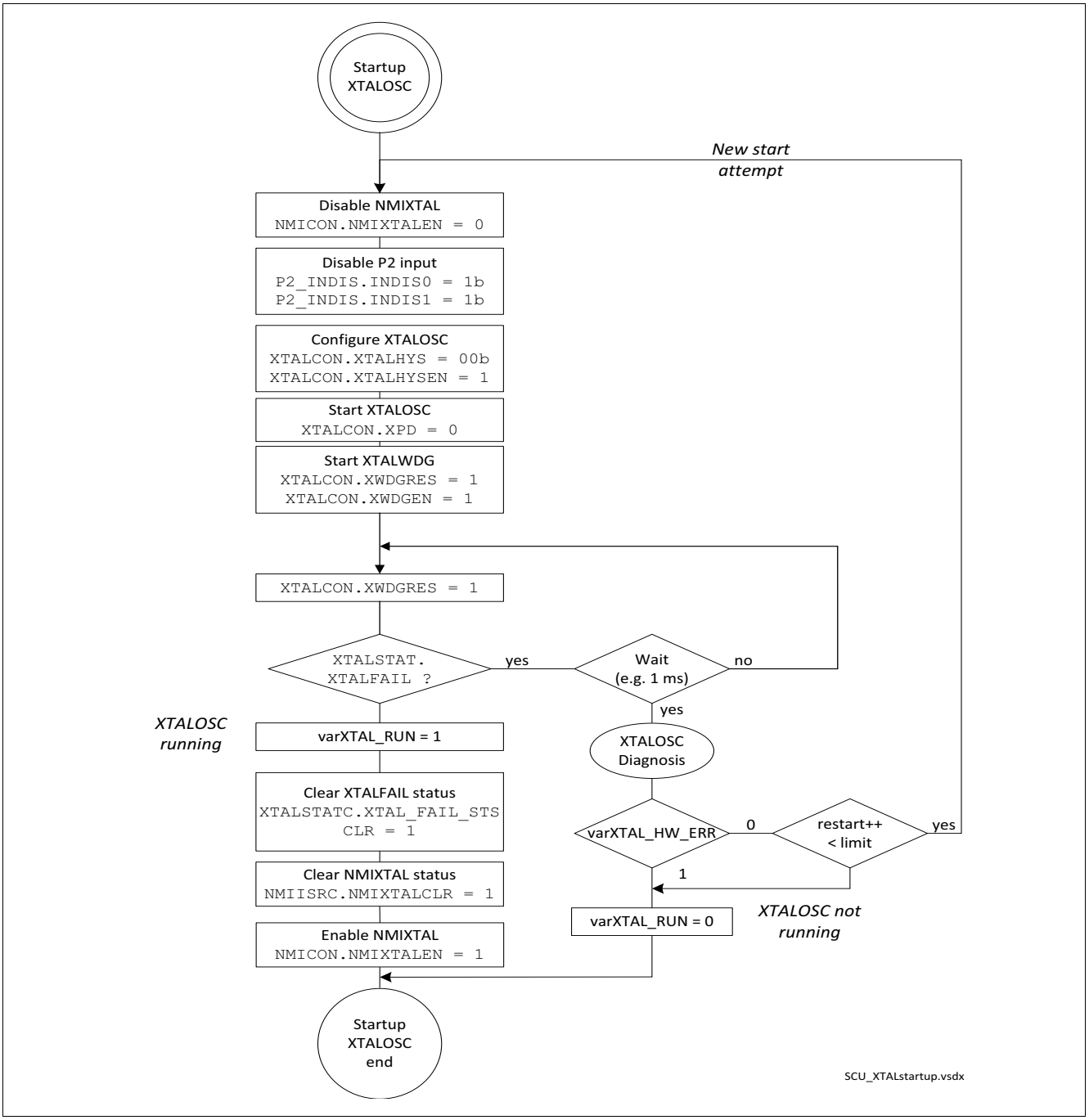


Figure 52 XTALOSC startup sequence

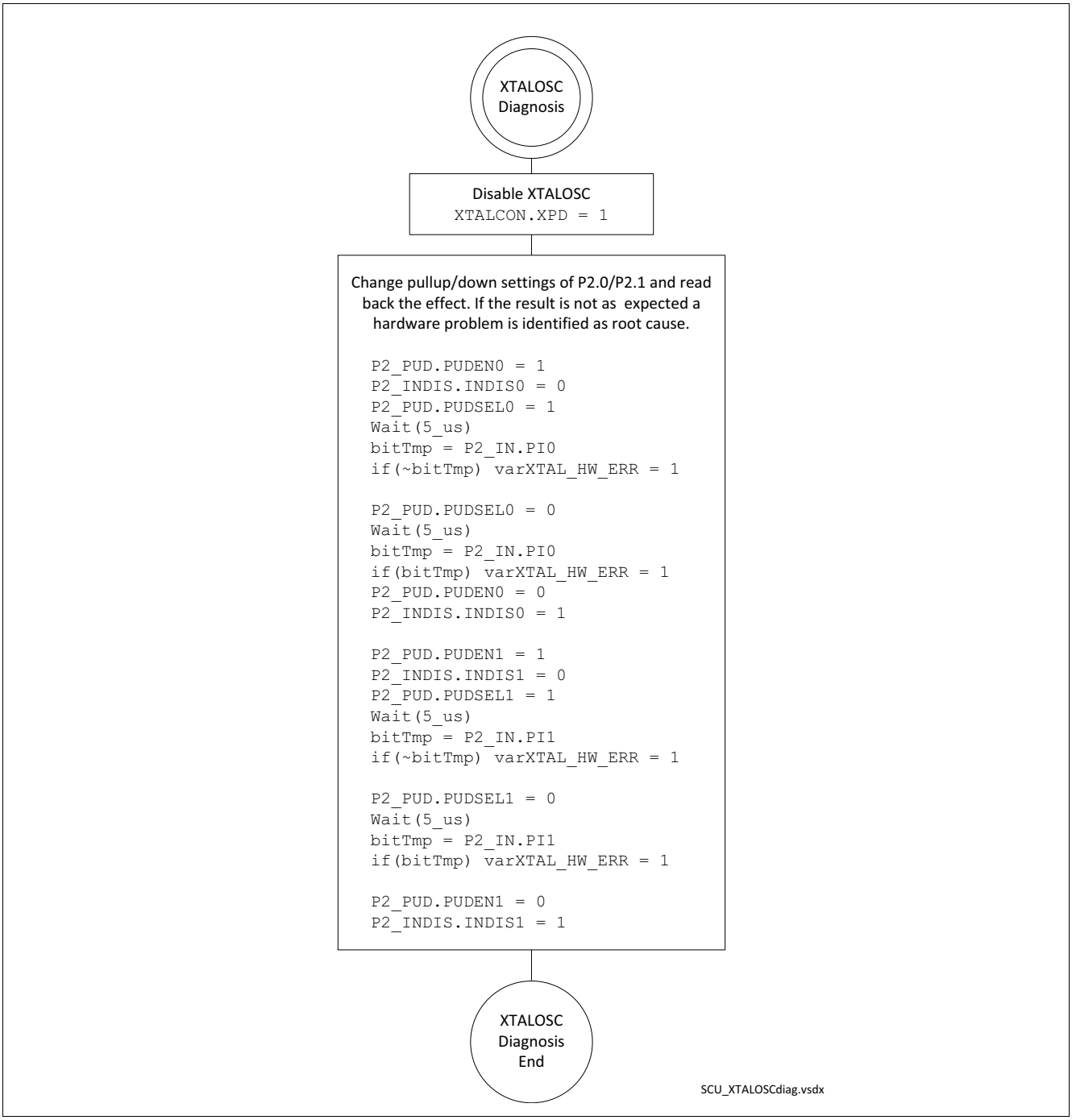


Figure 53 XTALOSC hardware diagnosis

6.6.4.5 XTALOSC error handling

The root cause for an improper startup of the XTALOSC can be the wrong selection of the passive components (load caps and damping resistor, see **Figure 51**). It must be ensured that XTALOSC starts up over the full temperature and voltage range.

A NMIXTAL event signalizes a severe malfunction of the system. It can be caused by a single EMI or ESD event or a permanent problem at the PCB. For diagnosis a permanent issue, the P2 pull-up/pull-down arrangement can be used.

It is recommended to re-run a complete XTALOSC startup sequence, see **Figure 52** and **Figure 53**.

6.6.5 Phase lock loop (PLL0 and PLL1)

6.6.5.1 PLL features

There are two identical PLLs (PLL0 and PLL1, in the following named PLL or PLLx, x = 0, 1). The PLLs allow a very flexible clock management with a wide input clock range (e.g. internal or external clock sources from 4 MHz to 40 MHz) and a programmable output clock range for the system clock (e.g. fsys0 from 5 MHz to 60 MHz and fsys1 from 5 MHz to 80 MHz).

The two PLLs allow to separate the system clock for MCU and peripherals (fsys0 with max. 60 MHz) from the asynchronous interfaces CAN and UART (fsys1 with max. 80 MHz).

Both PLLs have an optional spread spectrum feature which spreads and lowers the EMI spectrum for EMI sensitive applications.

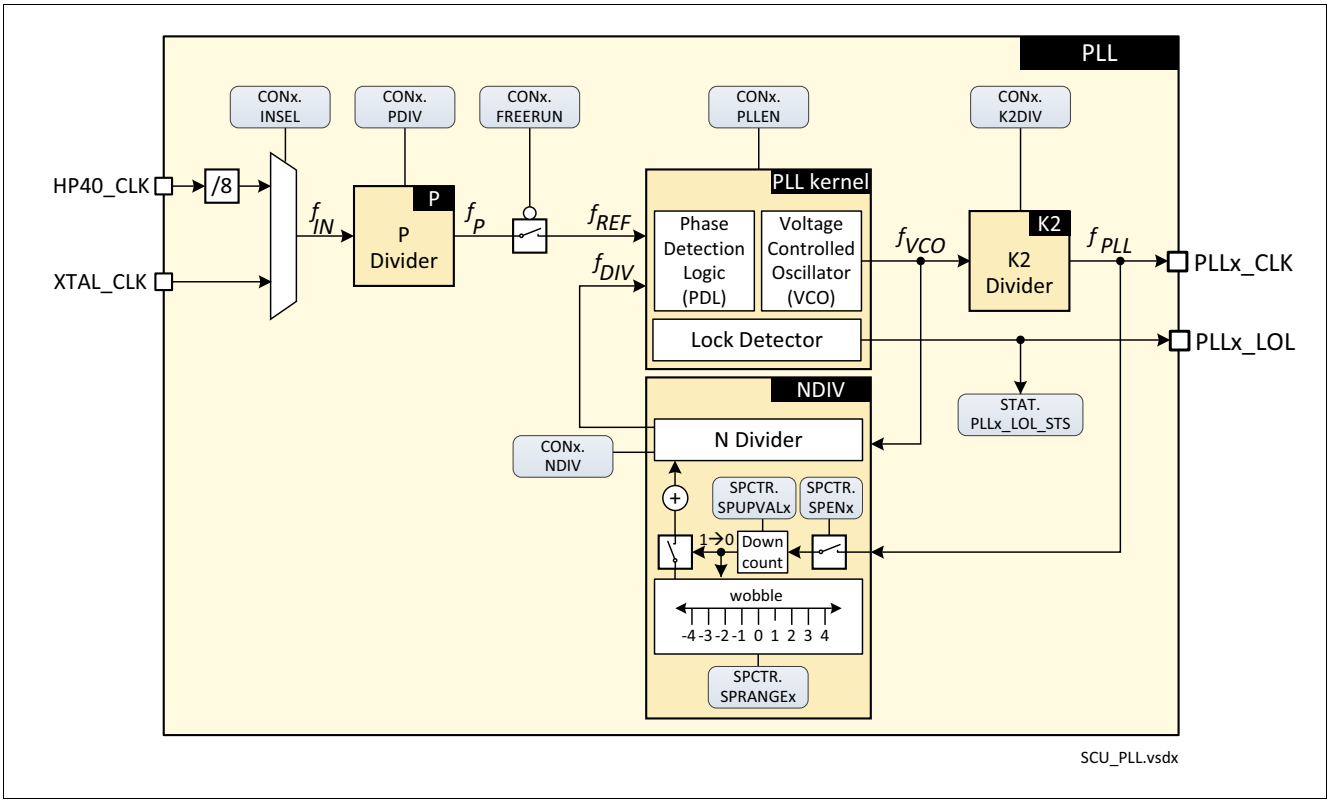


Figure 54 PLL0/1 block diagram

Both PLLs consist of following sub blocks, see **Figure 54**:

- An input clock selected via CONx.INSEL divided by the 6-bit P-divider (CONx.PDIV) and connected to the PLL kernel as reference clock f_{REF} via CONx.FREERUN (=0)

  Note: The PLL's reference clock must fulfill the limits defined in the EC parameter f_{REF0}, f_{REF1}. Therefore the PDIV has to be set accordingly. The default selection is HP40_CLK divided by 8, resulting in PDIV=5, see **Equation (6.1)**.

- The PLL kernel with:
  - A Phase Detection Logic (PDL) which compares f_{REF} against f_{DIV} an tunes the VCO frequency until f_{REF} matches with f_{DIV} ("PLL locked")
  - A Voltage Controlled Oscillator (VCO) which is tuned by the PDL and outputs f_{VCO}

- A Lock Detector which controls the PDL and outputs the “loss of lock” status STAT.PLLx_LOL. The PLL0 unlock event PLLx_LOL triggers NMISR.NMIPLLx and the clock selection for SYSx_CLK (CLKSEL.SELSYS0/SELSYS1) is switched to HP_CLK automatically in case PLLx_CLK was selected

Note: The VCO’s output clock must fulfill the limits defined in the EC parameter f_{VCO0}, f_{VCO1}. Therefore the NDIV has to be set accordingly.

- The NDIV with:
  - A programmable 8-bit N Divider (via CONx.NDIV) with f_{VCO} as input an f_{DIV} as output
  - An optional spread spectrum feature which wobbles the f_{VCO} with a programmable frequency (via SPCTR.SPRANGEx) step at a programmable wobble frequency (via SPCTR.SPUPVALx). In case spread spectrum feature is used the maximum system frequency limits ($f_{SYS0/1}$) have to be respected, i.e. the NDIV value must be reduced by the SPRANGE value, see example in **Figure 55**
- The 3-bit K2DIV as final divider for f_{VCO} to generate PLLx_CLK (f_{PLLx})

Note: The PLLx must fulfill the limits defined in EC parameters f_{PLL0}, f_{PLL1}.

The PLL kernel has following operation modes:

- The free running mode. This is the default mode after PLL is enabled via CONx.PLLEN and when the PLL is not yet locked to the reference clock (f_{REF}). The free running frequency is defined in $f_{VCOfreex}$ (typ. 21.5 MHz)
- The normal mode. After the PLL is locked to the reference clock, the output clock f_{VCO} is stable. The frequency f_{PLL} (PLLx_CLK) is defined according to **Equation (6.2)**
- The spread spectrum mode wobbles the pre-set NDIV value on-the-fly with a defined update rate and width:
  - The spread spectrum mode is enabled via SPCTR.SPENx, by default it is disabled
  - If enabled the actual NDIV is incremented or decremented by 1 until the limit value is reached. The limit is defined in SPCTR.SPRANGEx. This results in variation of f_{PLL} according to **Equation (6.3)**
  - The update of the NDIV happens at every 1 to 0 event of a down counter (clocked by $f_{PLL} = f_{VCO}/K2DIV$). The down-counter starts and auto-reloads with the value defined in SPCTR.SPUPVALx. This results in a wobble frequency according to **Equation (6.4)**

Note: The spread spectrum mode is limited by the NDIV boundaries in CONx.NDIV and has no effect in case of STAT.PLLx_LOL_STS is set.

(6.1)

$$f_{REF} = \frac{f_{IN}}{PDIV}$$

(6.2)

$$f_{PLL} = \frac{NDIV}{PDIV \times K2DIV} \times f_{IN}$$

(6.3)

$$\Delta f_{PLL} = \frac{SPRANGEx}{K2DIV} \times f_{REF}$$

(6.4)

$$f_{WOBBLE} = \frac{f_{PLL}}{SPUPVALx}$$

6.6.5.2 PLL behavior

The PLL0/1 behave in the operation modes according to **Table 79**.

Table 79 PLL behavior

Reset	• The PLL configuration and status is reset with RESET_TYPE_5 and RESET_TYPE_6
Power-up / Power-down	• After power-up the PLL0/1 are disabled by default • At power-down the PLL0/1 are reset by RESET_TYPE_5/6
Active mode	• The PLL0/1 can be enabled by the user, following a defined sequence
Stop mode	• The PLL0/1 configuration is reset when Stop mode is entered
Sleep mode	• The PLL0/1 configuration is reset when Sleep mode is entered
Fail-sleep mode	• None
fail-safe	• In case of an unlock, the PLL0/1 switches to the free running mode, the down-counterclock-selection for SYSx_CLK is switched to HP_CLK automatically in case PLLx_CLK was selected and a NMIPLL0/1 is requested

6.6.5.3 PLL programmer's guide

Finding the correct PLL settings, see also example in **Figure 55**:

- PDIV: take the input frequency f_{IN} and divide by a PDIV (4 to 50) to get f_{REF} = ~1 MHz; respect the f_{REF} limits!
- K2DIV: take the targeted f_{SYS} and multiply with K2DIV (2 to 9) to get f_{VCO}; respect f_{SYS} and f_{VCO} limits!
- NDIV: divide f_{VCO} with f_{REF} and get NDIV (39 to 200); respect f_{SYS} and f_{VCO} limits!

PLL settings for fPLL = 60 MHz at different fIN								
fIN [MHz]	**4**	**5**	**6**	**8**	**12**	**16**	**20**	**40**
PDIV	4	5	6	8	12	16	20	40
K2DIV	2	2	2	2	2	2	2	2
NDIV	116	116	116	116	116	116	116	116
SPUPVAL	250	250	250	250	250	250	250	250
SPRANGE	4	4	4	4	4	4	4	4
fREF [MHz]	1	1	1	1	1	1	1	1
fVCO [MHz]	116	116	116	116	116	116	116	116
fPLL [MHz]	**58**	**58**	**58**	**58**	**58**	**58**	**58**	**58**
fWOBBLE [MHz]	0,232	0,232	0,232	0,232	0,232	0,232	0,232	0,232
dfPLL [MHz]	2	2	2	2	2	2	2	2

PLL settings for fPLL = 80 MHz at different fIN, spread spectrum disabled								
fIN [MHz]	**4**	**5**	**6**	**8**	**12**	**16**	**20**	**40**
PDIV	4	5	6	8	12	16	20	40
K2DIV	2	2	2	2	2	2	2	2
NDIV	160	160	160	160	160	160	160	160
fREF [MHz]	1	1	1	1	1	1	1	1
fVCO [MHz]	160	160	160	160	160	160	160	160
fPLL [MHz]	**80**	**80**	**80**	**80**	**80**	**80**	**80**	**80**

Figure 55 Example for correct PLL settings

The PLL initialization and locking sequence can be taken from **Figure 56**.

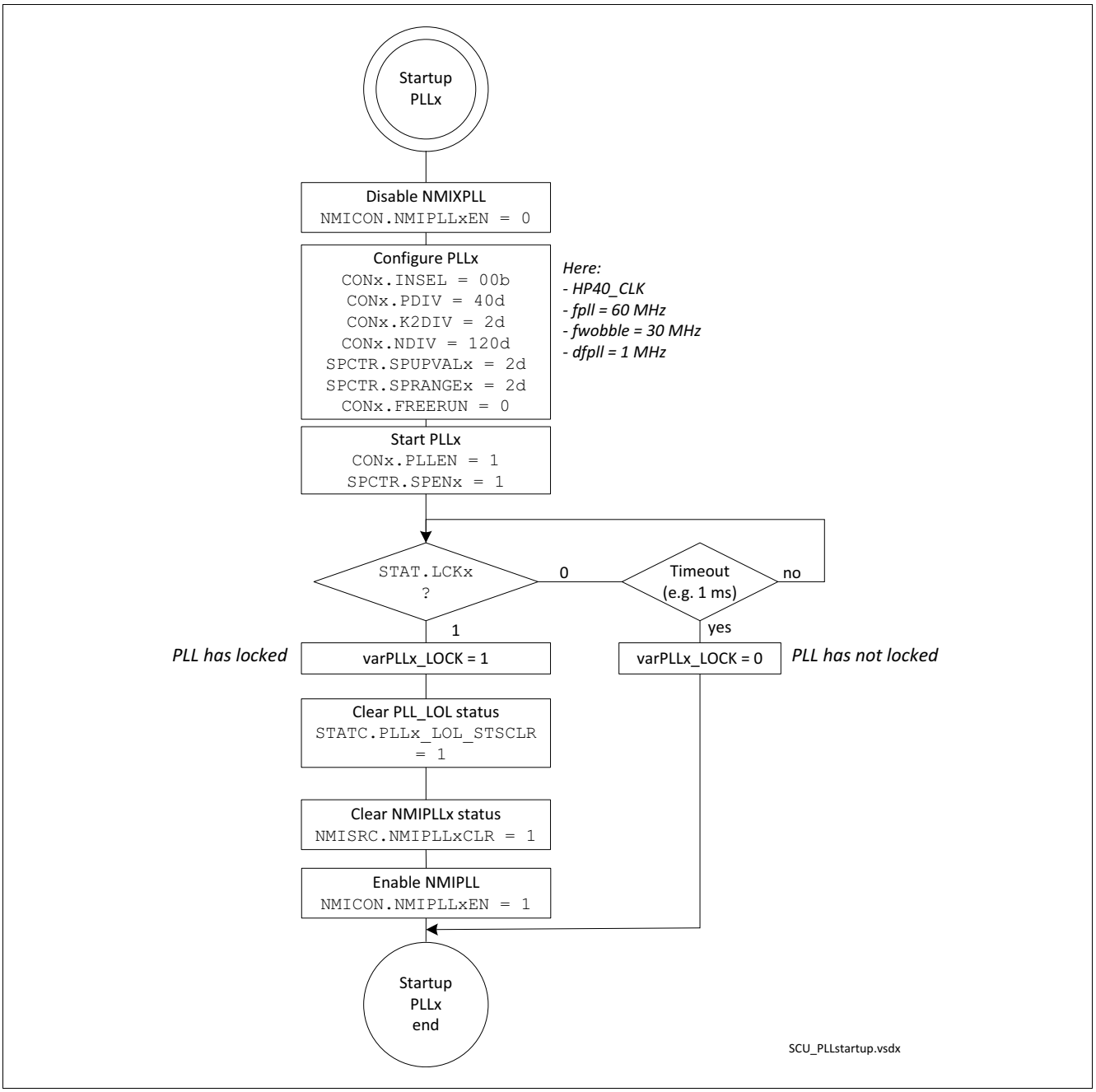


Figure 56 PLL initialization and locking sequence

6.6.5.3.1 PLL error handling

The root cause for an improper startup of the PLL (not locking) can be an instable input clock or a wrong configuration.

A NMIPLL event signalizes a severe malfunction of the system. It can be caused by a single EMI or ESD event or a permanent problem. For diagnosing a permanent issue at the XTALOSC, the P2 pull-up/pull-down arrangement can be used, see **Figure 53**.

In case of a PLL loss of lock event it is recommended to repeat the complete clock tree initialization, i. e. starting with the clock source (e.g. XTALOSC startup).

6.6.6 Clock tree configuration

6.6.6.1 Clock tree features

The clock tree is shown in **Figure 48** and provides following features:

- Reprogramming of the clock tree configuration can be done at any time in Active mode
- The system clock SYS0_CLK is the base clock for the MCU sub-system and all peripherals (except the asynchronous interfaces CAN and UART0/1). It can be derived from clock sources HP40_CLK, PLL0_CLK, PLL1_CLK, XTAL_CLK. The selection is done via CLKSEL.SELSYS0:
  - CLKCON.PRECPU is the prescaler for CPU_CLK and PER_CLK
  - CLKCON.PREMI is the prescaler for MI_CLK
  - CLKCON.PREFILT is the prescaler for TFILT_CLK and TFILT_BDRV_CLK
- The system clock SYS1_CLK is the base clock for the asynchronous interfaces CAN and UART0/1. It can be derived from clock sources HP40_CLK, PLL0_CLK, PLL1_CLK, XTAL_CLK. The selection is done via CLKSEL.SELSYS1:
  - CLKEN.CANCLKEN is the enable for CAN_CLK
  - CLKCON.PRECAN is the prescaler for CAN_CLK
  - CLKEN.UARTCLKEN is the enable for UART_CLK
  - CLKCON.PREUART is the prescaler for UART_CLK
- The alternate function CLKOUT can be assigned to several GPIOs, please refer to Product definitions chapter **SCU interconnections**. The clock source can be selected via CLKSEL.SELCLKOUT:
  - CLKSEL.CLKOUTEN is the enable for the CLKOUT function
  - CLKCON.PRECLKOUT is the prescaler for the CLKOUT function
  - CLKCON.DIV2CLKOUT is an additional prescaler for the CLKOUT function to achieve a duty cycle of 50%

Notes

1. *The system clock range for SYS0_CLK and SYS1_CLK must fulfill the limits given in the EC parameters, see f_{SYS0} and f_{SYS1}.*
2. *The maximum CLKOUT frequency is limited by the GPIOs driver capability and its load. It is recommended to go not far beyond 20 MHz.*

6.6.6.2 Clock tree programmer's guide

The default configuration is HP40_CLK as clock source for SYS0_CLK and SYS1_CLK. For reconfiguration it is recommended to follow **Figure 57**.

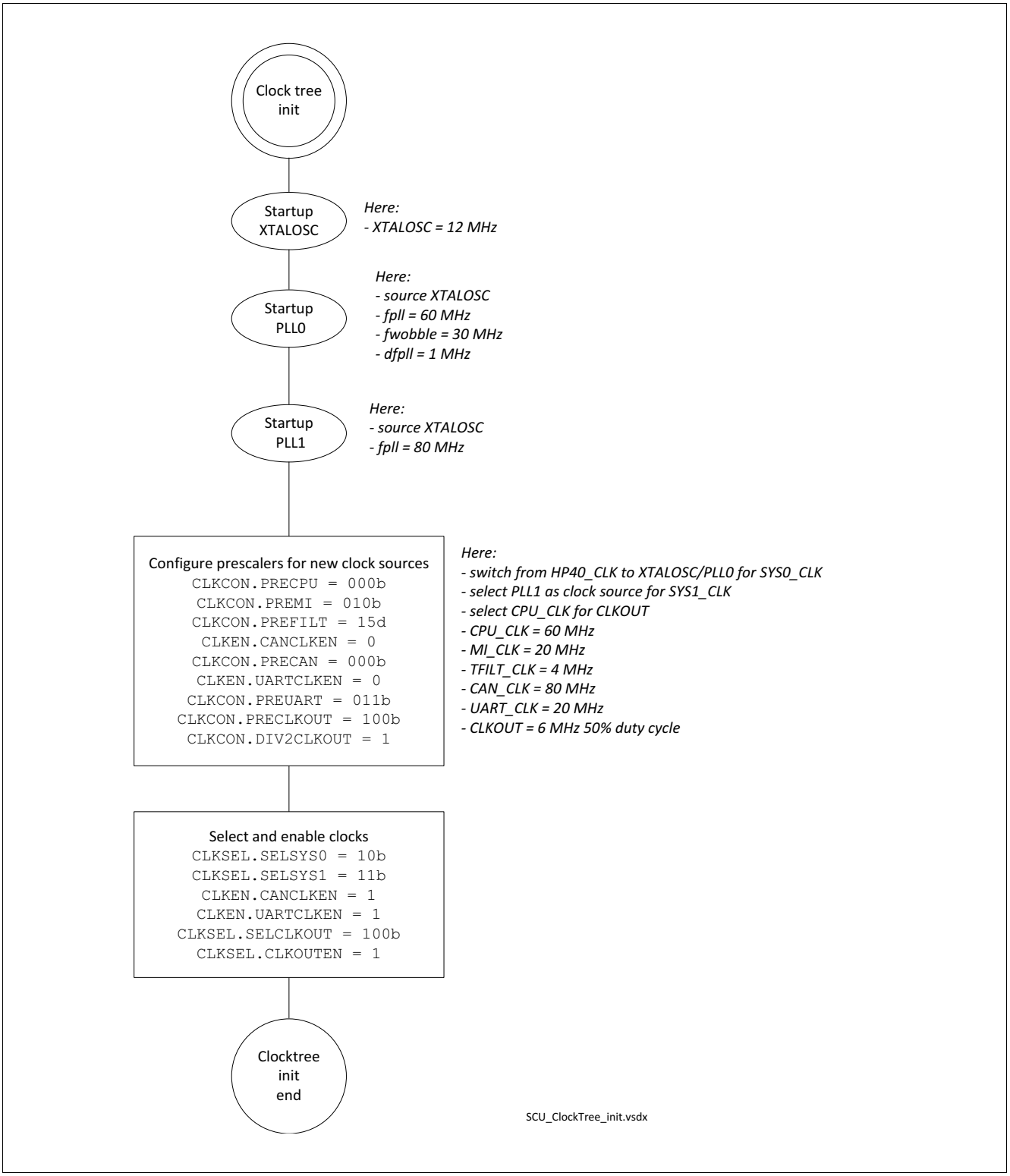


Figure 57 Clock tree initialization

6.7 Peripheral and operation mode management

The peripheral management takes care of the proper sequence and timing of clock and power when enabling and disabling peripherals.

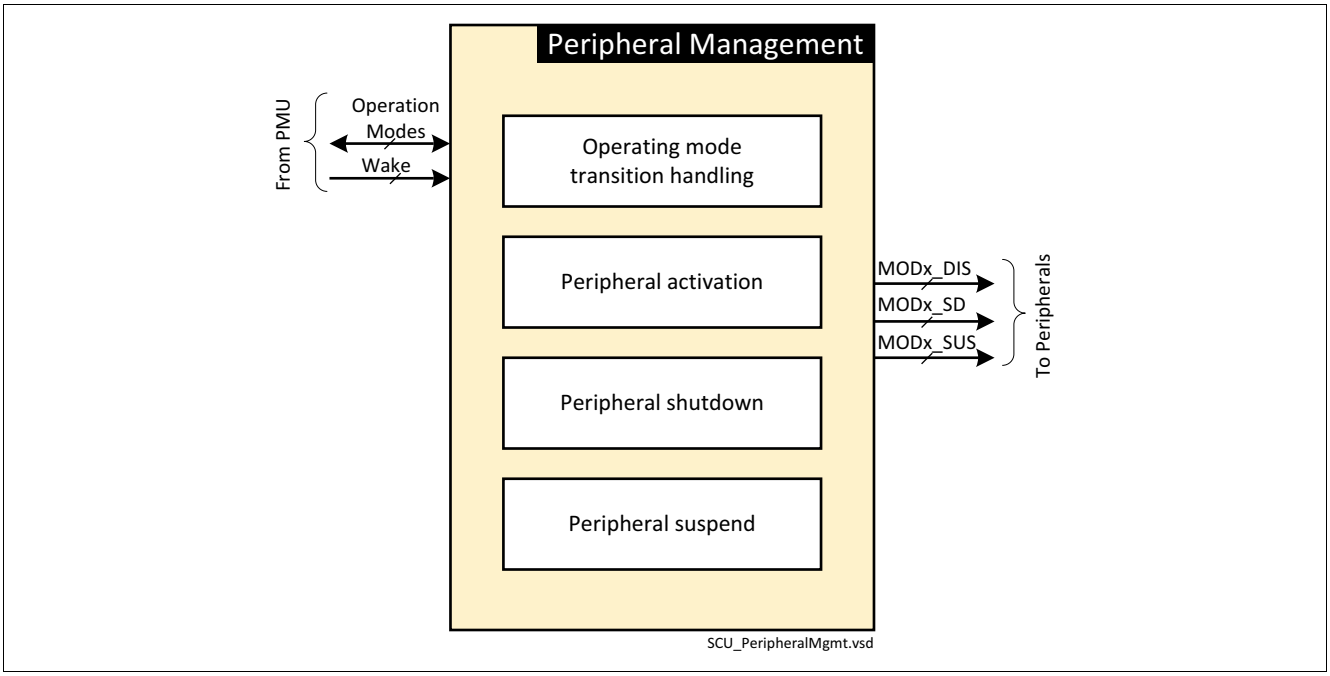


Figure 58 Peripheral management block diagram

6.7.1 Operation mode transition handling

The peripheral management offers a user interface to request the change of the operation mode. All modes can be used for power saving. The different power saving modes allow a flexible adaptation to the application specific use case. The modes allow to balance performance, current consumption and reaction time.

Changing the operation mode is a complex process which triggers state transitions at a deterministic timing. This is controlled by the SCU. The major interactions can be found in **Figure 59**. For Stop and Sleep state transitions changes on power supply (with reset) and clock configuration are necessary.

The respective values for current consumption and mode transition times (entry and exit) can be found in the electrical characteristics.

An overview of all operation modes and their differences can be found in **Figure 60**.

Notes

1. *For Stop and Sleep modes the PMU and the startup firmware in BootROM are involved. Please also refer to the PMU chapter (**Chapter 5**) and Product definitions (**Chapter 3**) for more details.*
2. *The Fail-sleep mode is a reaction on a severe system fault. The wake-up behavior is similar to a wake-up from Sleep mode but additional SFRs are reset because of the different reset type of FS_SLEEPEX_RST. Please also refer to the PMU chapter (**Chapter 5**) for more details.*

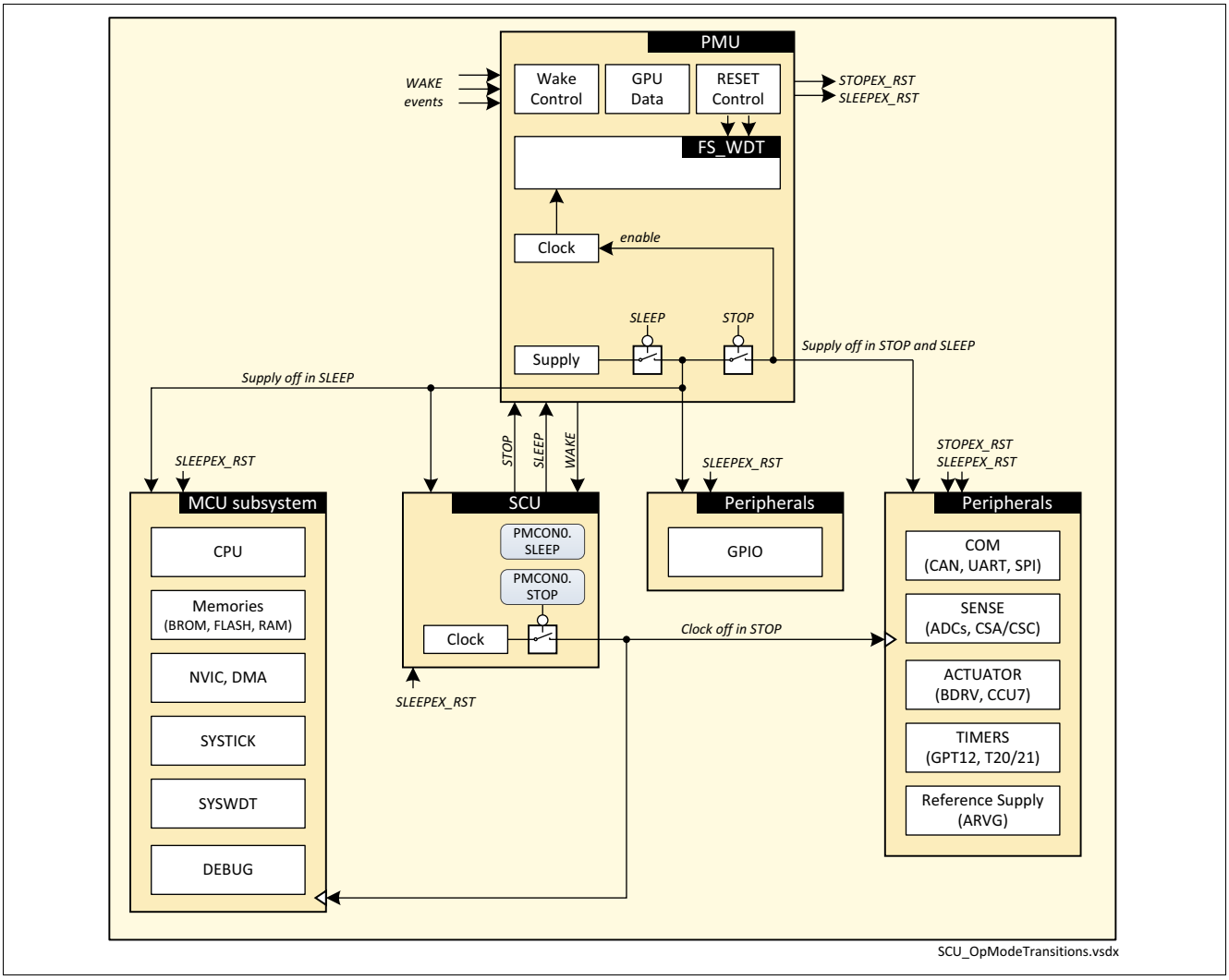


Figure 59 Interaction between SCU, PMU, MCU subsystem and peripherals

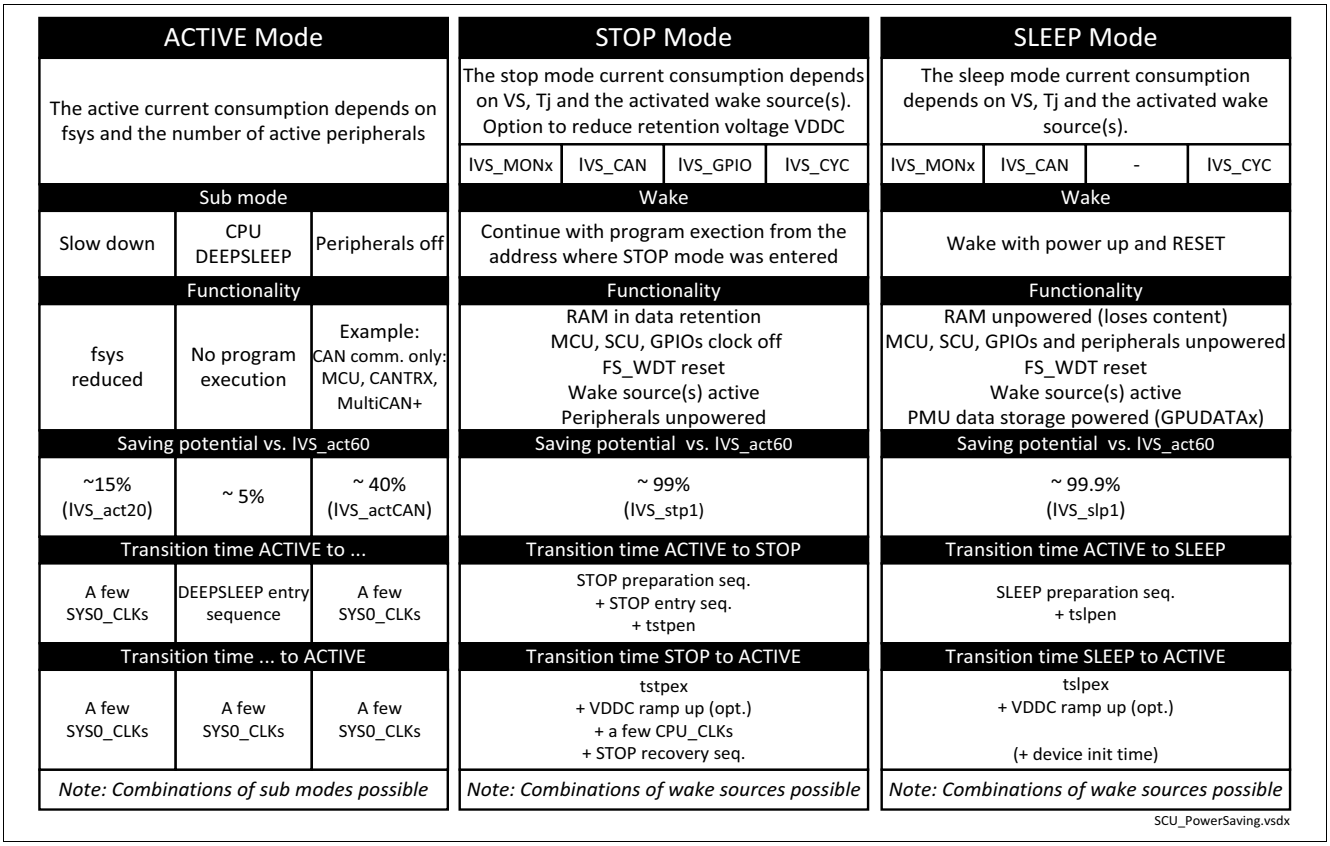


Figure 60 Operation modes in comparison

6.7.1.1 Active mode

The Active mode has three sub modes for power saving.

Slow down mode

This mode can be used for reducing the power consumption when the performance is not needed and immediately switch back to full speed if needed. The clock configuration has to be changed via CLKCON.PRECPU.

A relevant use case is to reduce the CPU_CLK and the peripheral clock to a minimum while waiting for a communication interrupt. Within the interrupt service routine, the CPU_CLK can be scaled up again. The baudrate clock can be left untouched.

Notes

1. *The watchdog timer FS_WDT is not affected by the clock configuration and has to be serviced according to its time base.*
2. *The watchdog timer SYSWDT is affected by the clock configuration, i.e. its time base changes.*

Peripherals off

Peripherals can be selectively disabled if not needed.

A relevant use case is the CAN-only-communication mode. Here major power contributors can be switched off like BDRV, ADC1, CSA/CSC, CCU7.

CPU DEEPSLEEP mode

The CPU is sent to Arm's DEEPSLEEP mode, i.e. program fetching and execution is halted. The CPU registers are "frozen". A special DEEPSLEEP entry sequence has to be executed according to **Table 80**.

A wake-up event can be any interrupt request or exception.

After wake-up the program counter continues with the value at the entry point.

Note: The watchdog timers FS_WDT and SYSWDT are not affected by the CPU DEEPSLEEP entry and have to be serviced accordingly.

Table 80 CPU DEEPSLEEP entry sequence

Step	Description	Call CMSIS intrinsic function
1	Free up pipeline	__NOP(); __NOP(); __NOP();
2	Set send event; (handshake SCU-CPU)	__SEV();
3	Clear send event (handshake SCU-CPU)	__WFE();
4	Enter CPU DEEPSLEEP	__WFE();

6.7.1.2 Stop mode

The Stop mode overview is shown in **Figure 59**, **Figure 60** and in PMU chapter **State control**, **Stop mode**

Active to Stop mode

- The Sleep mode entry sequence is triggered with PMCON0.STOP=1

Stop to Active mode

- The wake-up is triggered by an enabled wake event, see the PMU chapter **Wake control**

Stop mode considerations

- MCU clock system
  - The clock configuration is reset when entering Stop mode. The clock source for SYS0_CLK and SYS1_CLK is HP40_CLK
  - After wake-up an external oscillator has to be started (**Figure 52**) if needed
  - After wake-up the PLLs have to be locked (see **Figure 56**) if needed
- Fail-safe watchdog (FS_WDT)
  - At Stop mode entry: the FS_WDT is stopped, the count value is reset
  - After Stop mode wake-up: the FS_WDT starts in long open window mode (LOW)
- Safe switch off path (SSO)
  - In Stop mode the SSO is activated
  - After a wake-up from Stop mode a safe release sequence has to be executed

6.7.1.3 Sleep mode

The Sleep mode overview is shown in **Figure 59**, **Figure 60** and in PMU chapter **State control**, **Sleep mode**.

Active to Sleep mode

- The Sleep mode entry sequence is triggered with PMCON0.SLEEP =1

Sleep to Active mode

- The wake-up is triggered by an enabled wake event, see the PMU chapter **Wake control**

Sleep mode considerations

- Fail-safe watchdog (FS_WDT)
  - At Stop mode entry: the FS_WDT is stopped and reset
  - After Stop mode wake-up: the FS_WDT starts in long open window mode (LOW)
- Safe switch off path (SSO)
  - In Sleep mode the SSO is activated
  - After a wake-up from Sleep mode a safe release sequence has to be executed
- The GPIOs are unpowered

6.7.1.4 Fail-sleep mode

The Fail-sleep mode is similar to a Sleep mode. The entry and exit conditions are different, see the PMU chapter **Fail-sleep mode**.

There are two SCU events which can cause a Fail-sleep mode entry:

- System over-temperature (SYS_OT via ADC2)
- HP_CLK_FAIL fail (see **Chapter 6.6.3**)

A Fail-sleep mode is indicated in SFR PMU.WAKE_FAIL_STS.

6.7.2 Peripheral activation

For power saving peripherals can be disabled individually. Some peripherals have their enable/disable configuration bit within the peripheral, some have them inside the SCU, some are automatically enabled by SCU/PMU.

Following peripherals have a disable bit within the SCU:

- SSC0 via PMCON.SSC0_DIS
- SSC1 via PMCON.SSC1_DIS
- TIMER20 via PMCON.T2_DIS
- TIMER21 via PMCON.T21_DIS
- GPT12 via GPT12.GPT12_DIS

Note: By default above peripherals are enabled.

6.7.3 Peripheral shutdown

- There are two critical events which can be configured to an automatic shutdown of the BDRV:
  - System overtemperature (SYS_OT_WARN via ADC2) if enabled via PCU_CTRL.OTWARN_ SD_DIS
  - XTAL fail event (XTAL_FAIL_STS) if enabled via PCU_CTRL.XTALWDG_SD_DIS
- An additional gating allows to enable the shutdown:
  - PCU_CTRL.BDRV_SD_EN

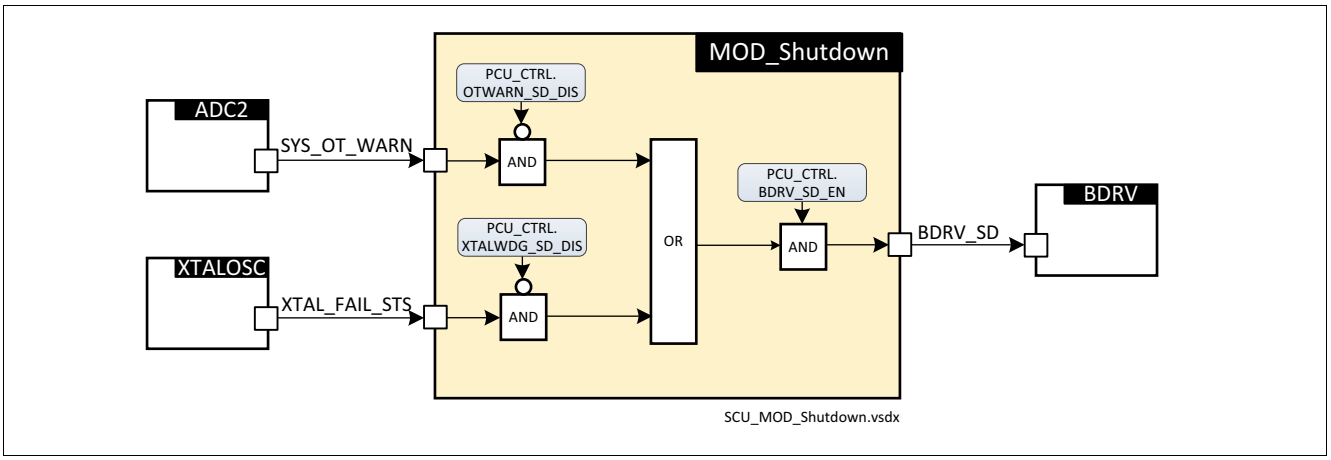


Figure 61 Peripheral shutdown

6.7.4 Peripheral suspend

For debugging some peripherals can be suspended. The suspend configuration for some peripherals is inside the SCU, for other peripherals it is inside the peripheral itself. In suspend mode the peripheral clock is halted but the bus access is still possible and debugger can read the registers.

- The suspend mode can be enabled in SUSCTR for following peripherals:
  - SSC0 and SSC1
  - T20 and T21
  - GPT12
  - SYSWDT

6.8 Interrupts and DMA requests

The SCU collects the interrupt and DMA requests from the modules and forwards them to the NVIC and DMA. For some modules an additional control logic is inserted between the module requests and the NVIC node.

The mapping of the interrupt requests to the NVIC is can be found in the **Interrupt request mapping**.

The mapping of the DMA requests to the DMA controller can be found in the .

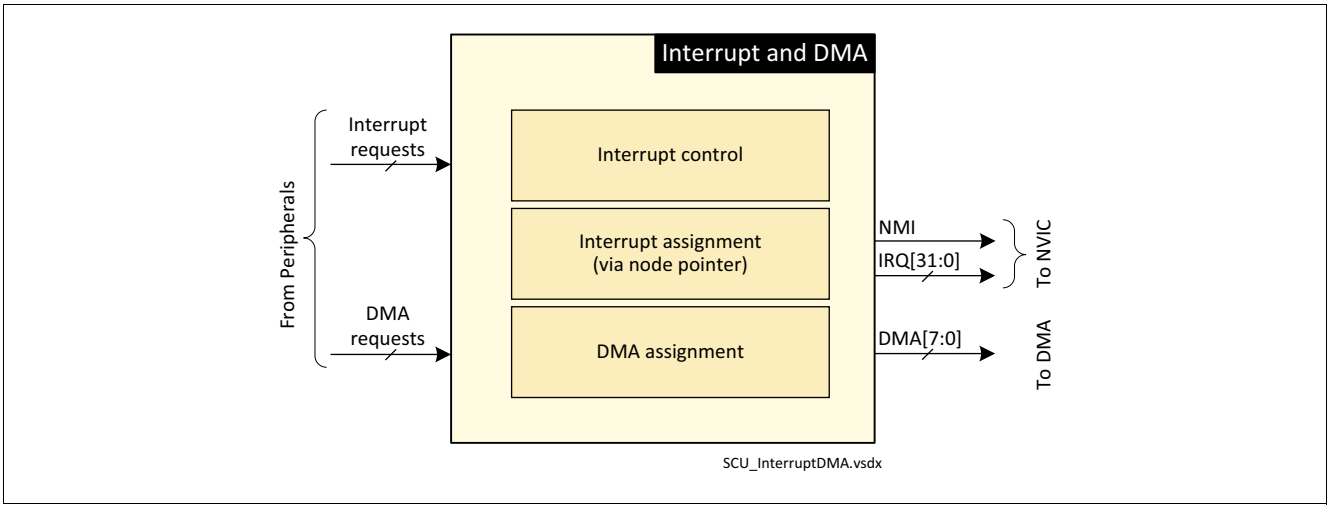


Figure 62 Interrupt and DMA

6.8.1 Interrupt and exception control

As there are more peripheral events than interrupt nodes at the NVIC a “compression” of events is necessary. This is handled with a shared node pointer scheme, see **Figure 63**.

For following peripherals have interrupt node pointer is inside the SCU:

- CANTRX, BDRV, CSC, PMU and ARVG (warning events)
- GPT12
- MON
- BEMFC and SDADC
- EXINT
- UART0/1
- SSC0/1
- DMA (end of DMA and DMA error)

The error events of some peripherals are handled as exceptions via a shared NMI node:

- SCU (clock related faults)
- MCU/MEMCTRL (memory related faults)
- ADC2 (events from the supervisory ADC2)

Shared node pointer scheme

For following peripherals the node pointer is inside the module itself, the assignment is in **Interrupt request mapping**:

- ADC1
- ADC2
- SDADC
- CCU7
- MultiCAN

The concept of the interrupt generation is shown in **Figure 63**.

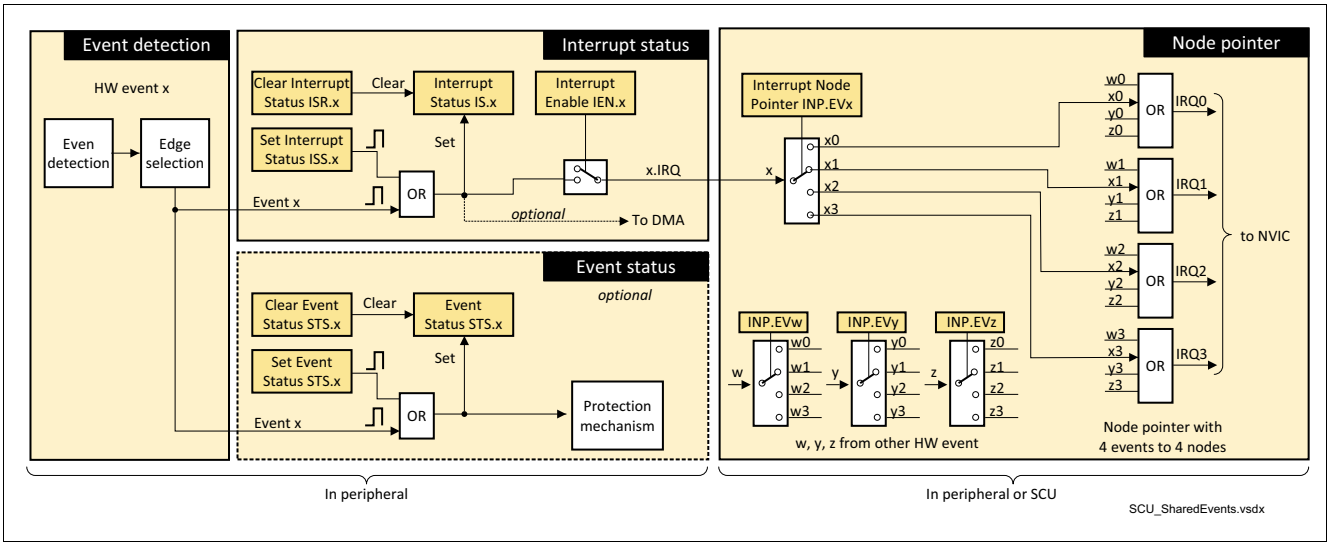


Figure 63 Interrupt generation with shared node pointer scheme

Direct connected interrupts

For following peripherals the node pointer is just an OR gate in the module itself, the assignment is in **Interrupt request mapping**:

- Timer 20/21
- MCU/MEMCTRL

Programmer's guide shared interrupts

In case of individual interrupt handling there is only one causing event which can request an NVIC interrupt. The interrupt service routine can be minimized, i.e. there is no need to check for the causing event and there is no need to clear the causing event because of the edge triggered event scheme (ARM: "pulse interrupt").

In case of shared interrupt handling there are multiple causing events which can request an NVIC interrupt. The causing event must be identified, cleared and handled accordingly within the interrupt service routine (ISR). Special considerations are necessary in case multiple events can request an interrupt asynchronously, i.e. a new event is set (ARM: "set to pending") while the ISR is executed (ARM: "interrupt set to active"). Here the software timing and the event timing may vary, which has to be considered by the ISR handler accordingly. In some corner cases a "dummy interrupt" can happen, see **Figure 64**. This "dummy interrupt" is caused by the fact that the NVIC has still a "PEND" state from an event which is already cleared and handled. Hence the software will enter the ISR, check the events, will not find a valid request and return from the ISR. Therefore the "dummy interrupt" causes only unnecessary CPU load. Because of this effect it is recommended to avoid shared interrupts by selecting the node pointers accordingly. This is possible for the majority of the use cases.

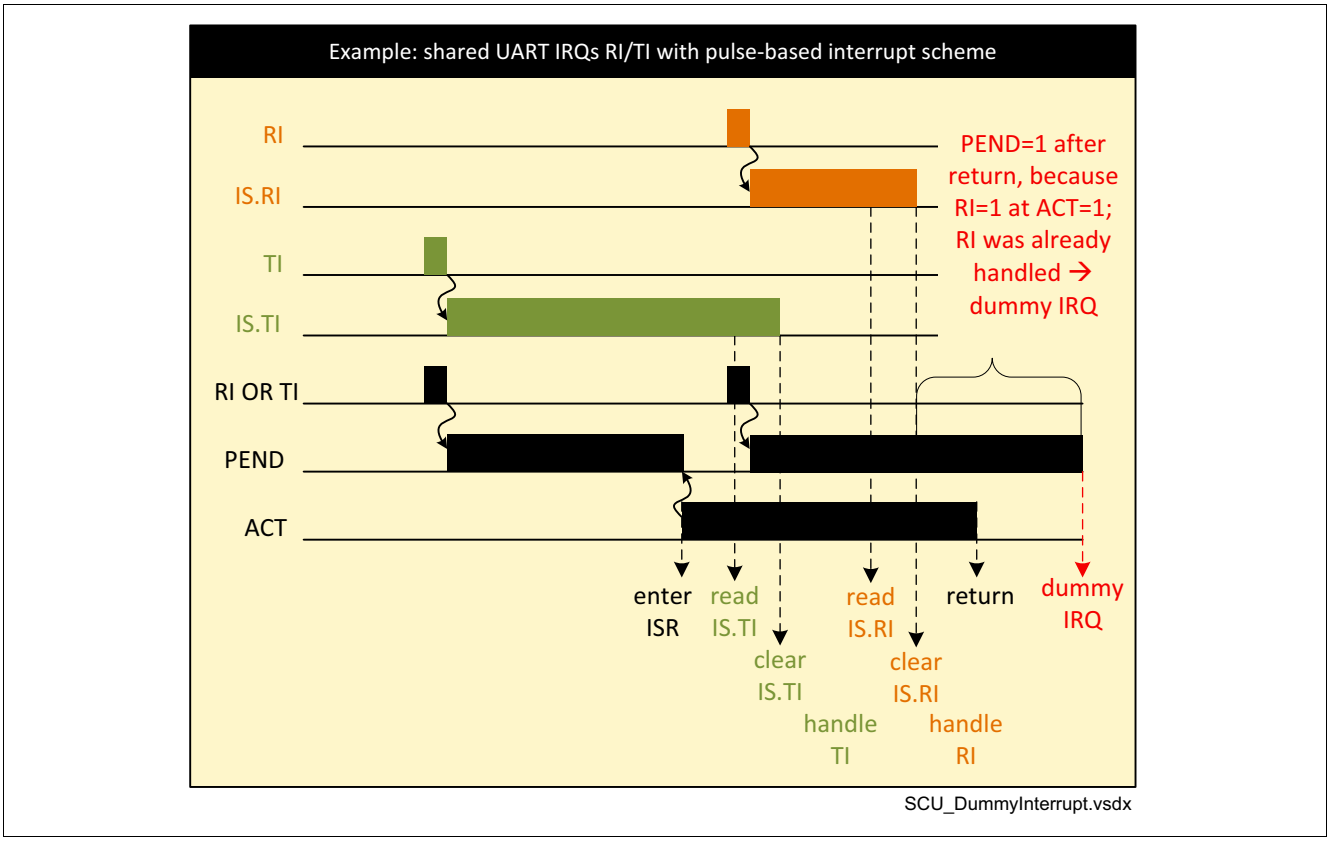


Figure 64 Shared interrupts with dummy interrupt

6.8.1.1 Warning interrupt generation

The interrupt requests from CANTRX, BDRV, CSC, PMU, ARVG have a shared node pointer inside the SCU. The node pointer scheme is described in **Figure 66**.

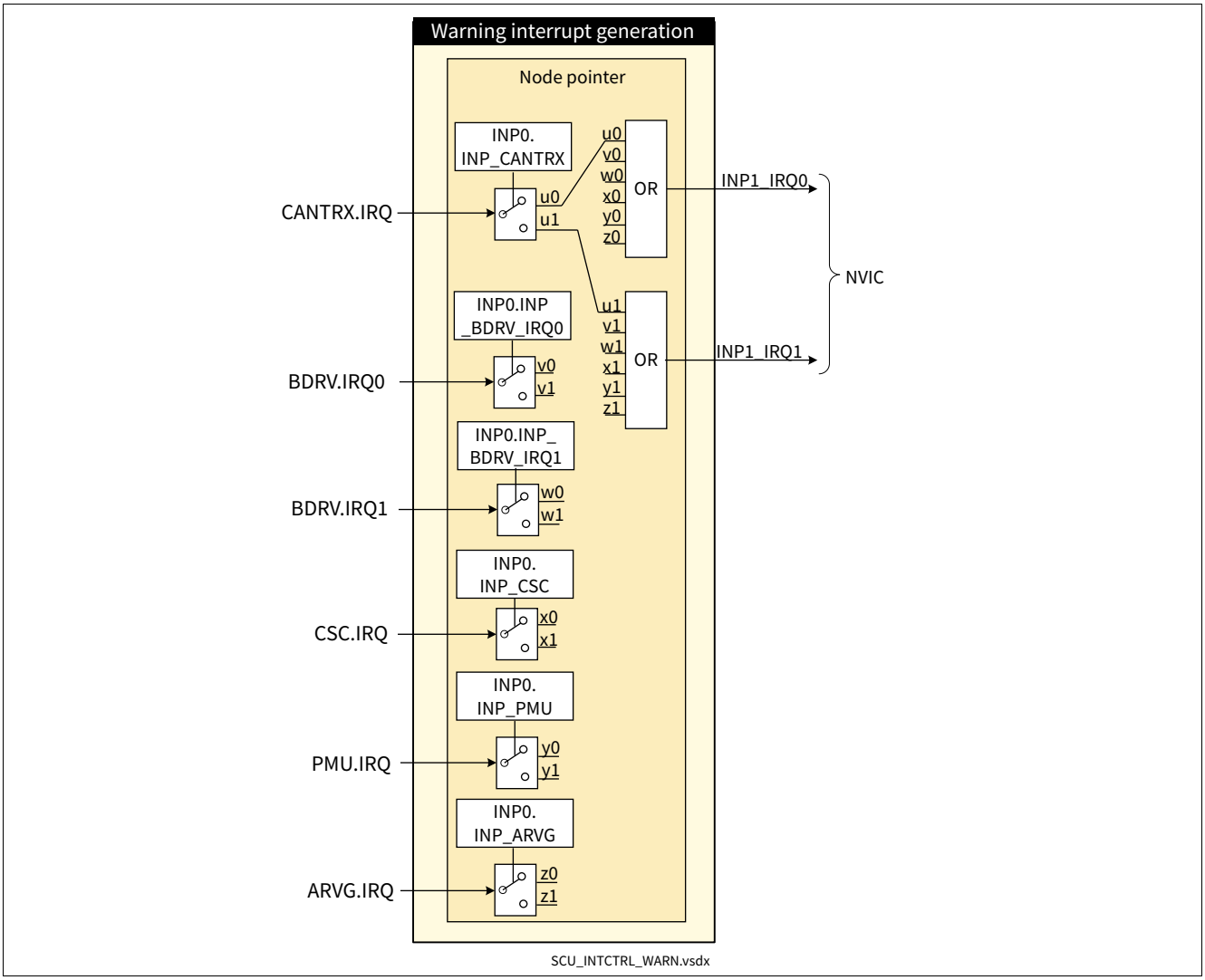


Figure 65 Warning interrupt generation

6.8.1.2 GPT12 interrupt generation

The interrupt requests from the GPT12 module are handled inside the SCU.

The GPT12 interrupt scheme allows following configuration:

- The interrupt status is stored in register GPTIS
- Software can set and clear the interrupt status via GPTISS and GPTISC
- The interrupt request can be enabled via GPTIEN
- The six GPTx.IRQs are mapped to interrupt node pointer INP1 which has two NVIC request lines

The GPT12 interrupt scheme is described in **Figure 66**.

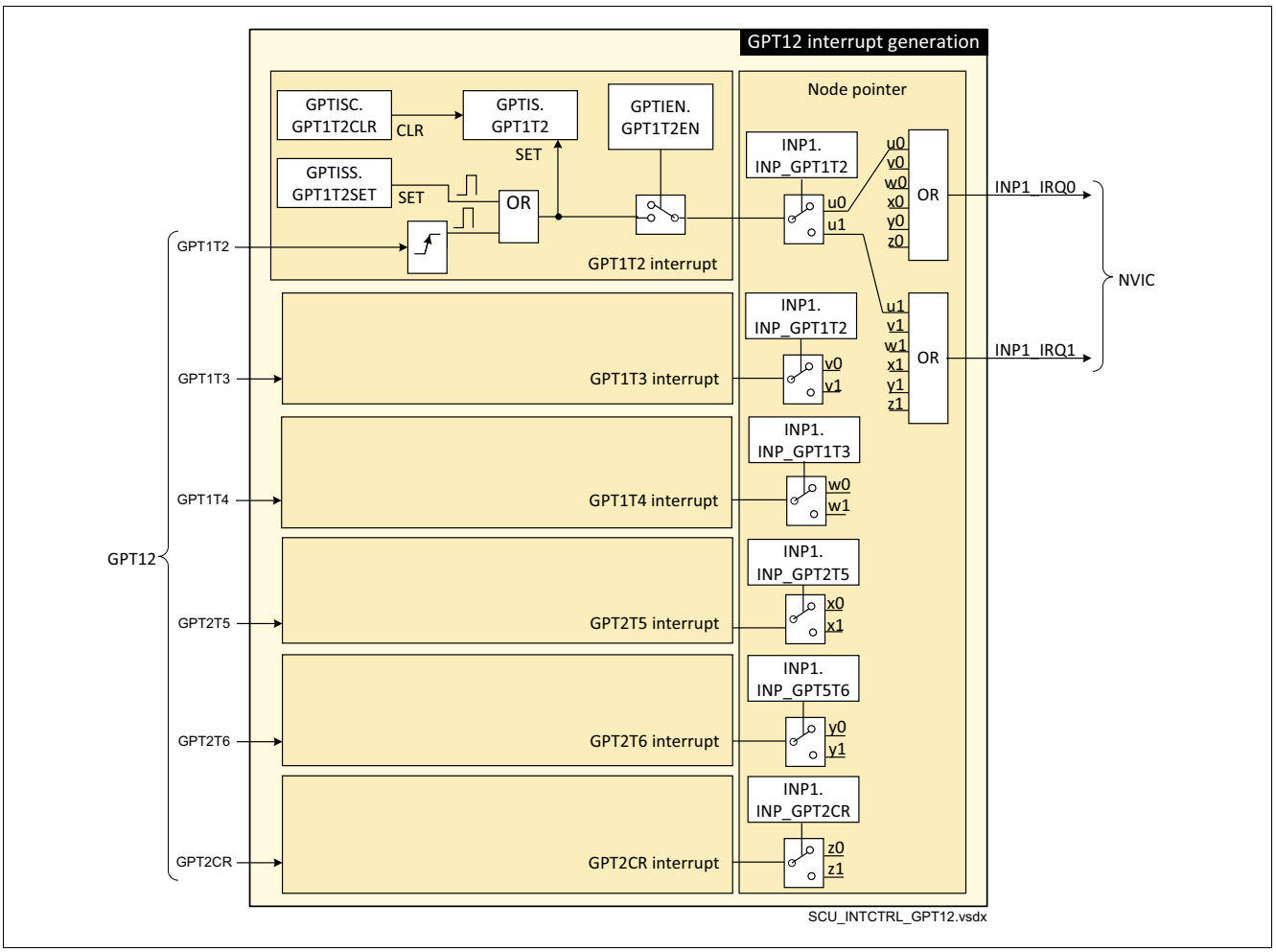


Figure 66 GPT12 interrupt generation

6.8.1.3 MON interrupt generation

The digital output of the high-voltage monitor input can be used to generate interrupts on crossing the voltage threshold. The MONxO output signal of the analog module is connected to the SCU (x = 1, 2, 3).

The MON interrupt scheme allows following configuration:

- The MONxO (x = 1, 2, 3) signal passes a spike filter with a filter time of ~1 us
- The edge can be selected via MONCON.MONxIEV (x = 1, 2, 3)
- The interrupt status is stored in register MONIS.MONxR/F (x = 1, 2, 3)
- Software can set and clear the interrupt status via MONISS.MONxRCLR/MONxFCLR (x = 1, 2, 3) and MONISS.MONxRSET/MONxFSET (x = 1, 2, 3)
- The interrupt request can be enabled via MONIEN.MONxEN (x = 1, 2, 3)
- The three MONx.IRQs are mapped to interrupt node pointer INP2.INP_MONx (x = 1, 2, 3) which has two NVIC request lines

Note: The MON event generation has to be globally enabled via PMU.MON_CTRLx.EN (x = 1, 2, 3).

The MON interrupt scheme is described in **Figure 67**.

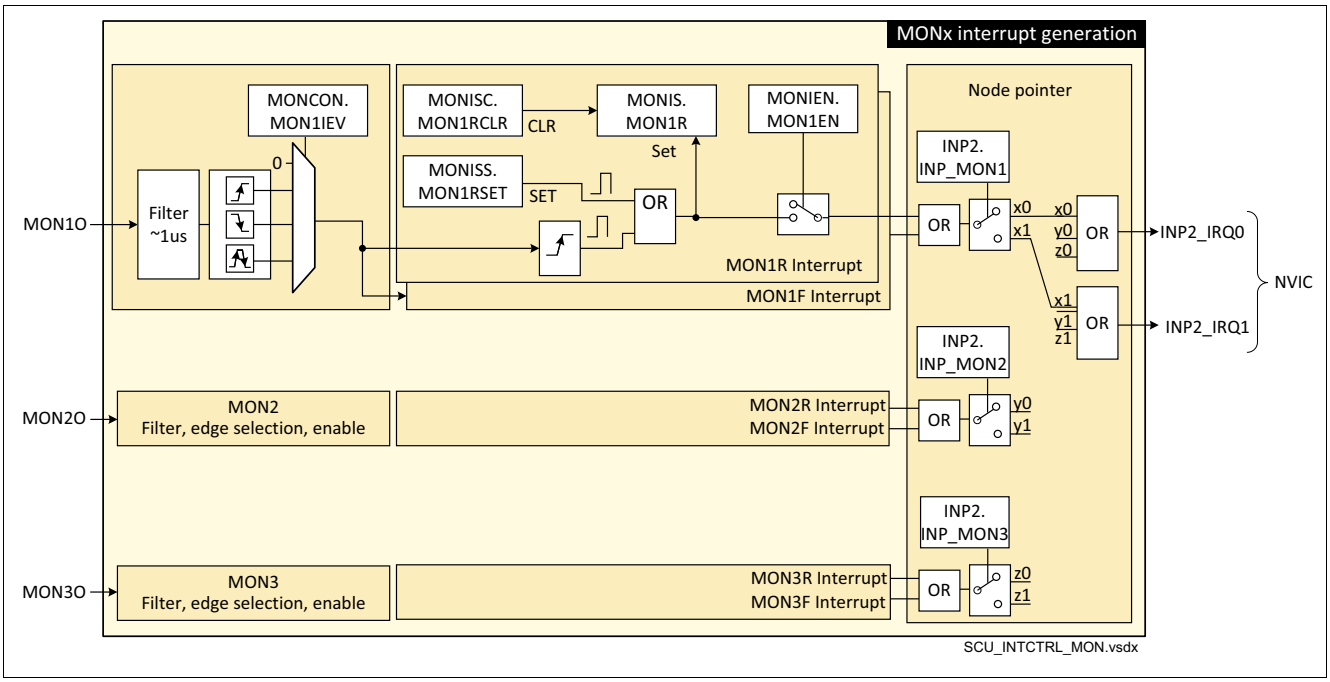


Figure 67 MONx interrupt generation

6.8.1.4 BEMFC and SDADC interrupt generation

The interrupt requests from BEMFC and SDADC have a shared node pointer inside the SCU.

The node pointer scheme is described in **Figure 68**.

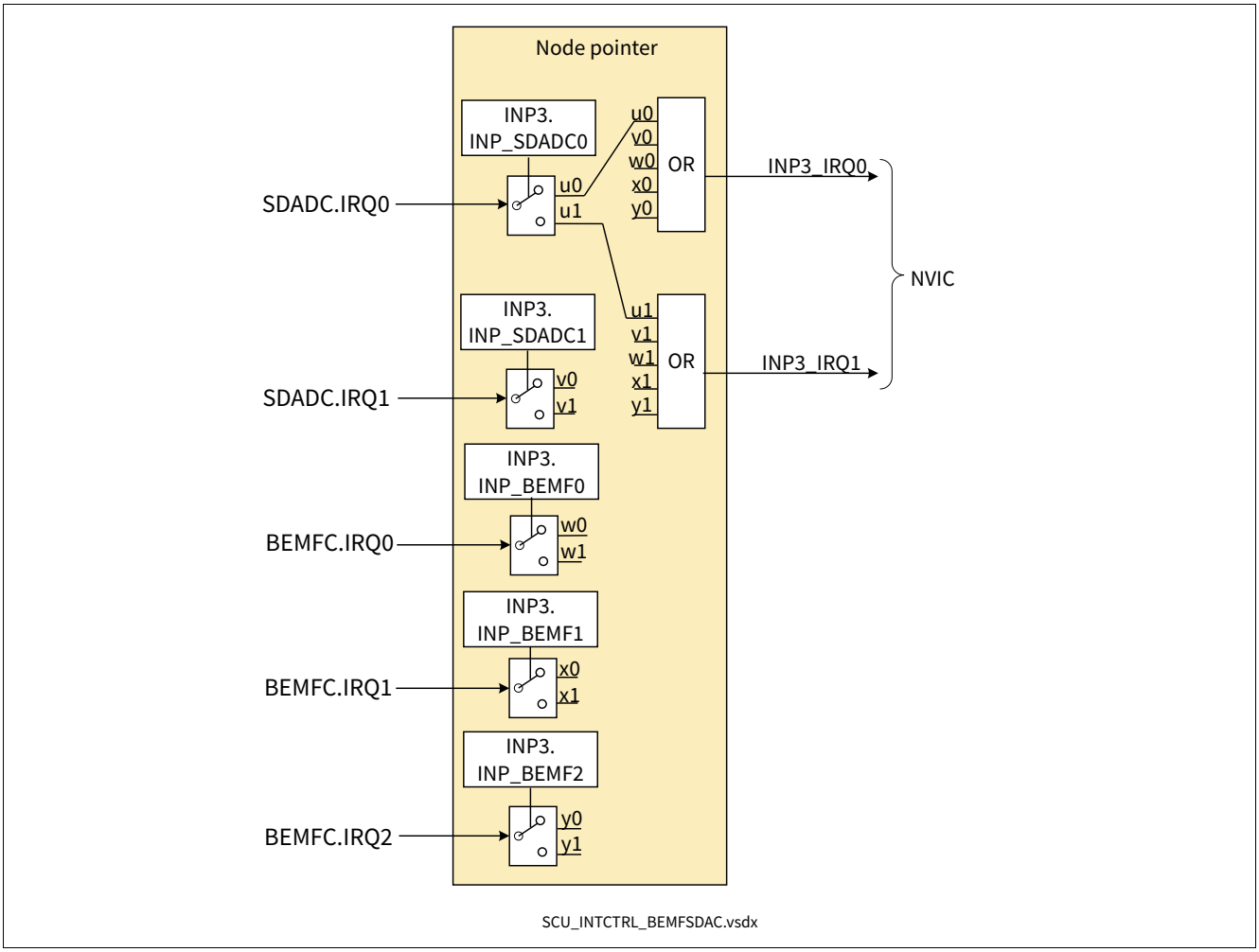


Figure 68 BEMFC/SDADC interrupt generation

6.8.1.5 External interrupt generation (EXINT)

Several GPIOs can request one of four external interrupts EXINTx (x = 0 to 3). The alternate input mapping of the GPIOs can be found in Product definitions chapter **SCU interconnections**.

The EXINT interrupt scheme allows following configuration:

- The GPIO signal EXINTxA/B/C/D has to be selected via EXTCON.EXTINTxINSEL (x = 0 to 3)
- The selected signal passes a spike filter with a filter time of ~1 us
- The edge can be selected via EXTCON.EXTINTxIEV (x = 0 to 3)
- The interrupt status is stored in register EXTIS.EXTINTxR/F (x = 0 to 3)
- Software can set and clear the interrupt status via EXTISS.EXTINTxRCLR/EXTINTxFCLR and EXTISS.EXTINTxRSET/EXTINTxFSET (x = 0 to 3)
- The interrupt request can be enabled via EXTIEN.EXTINTxEN (x = 0 to 3)
- The four EXINTx.IRQs are mapped to interrupt node pointer INP4.INP_EXINTx (x = 0 to 3) which has two NVIC request lines

The EXINT interrupt scheme is described in **Figure 69**.

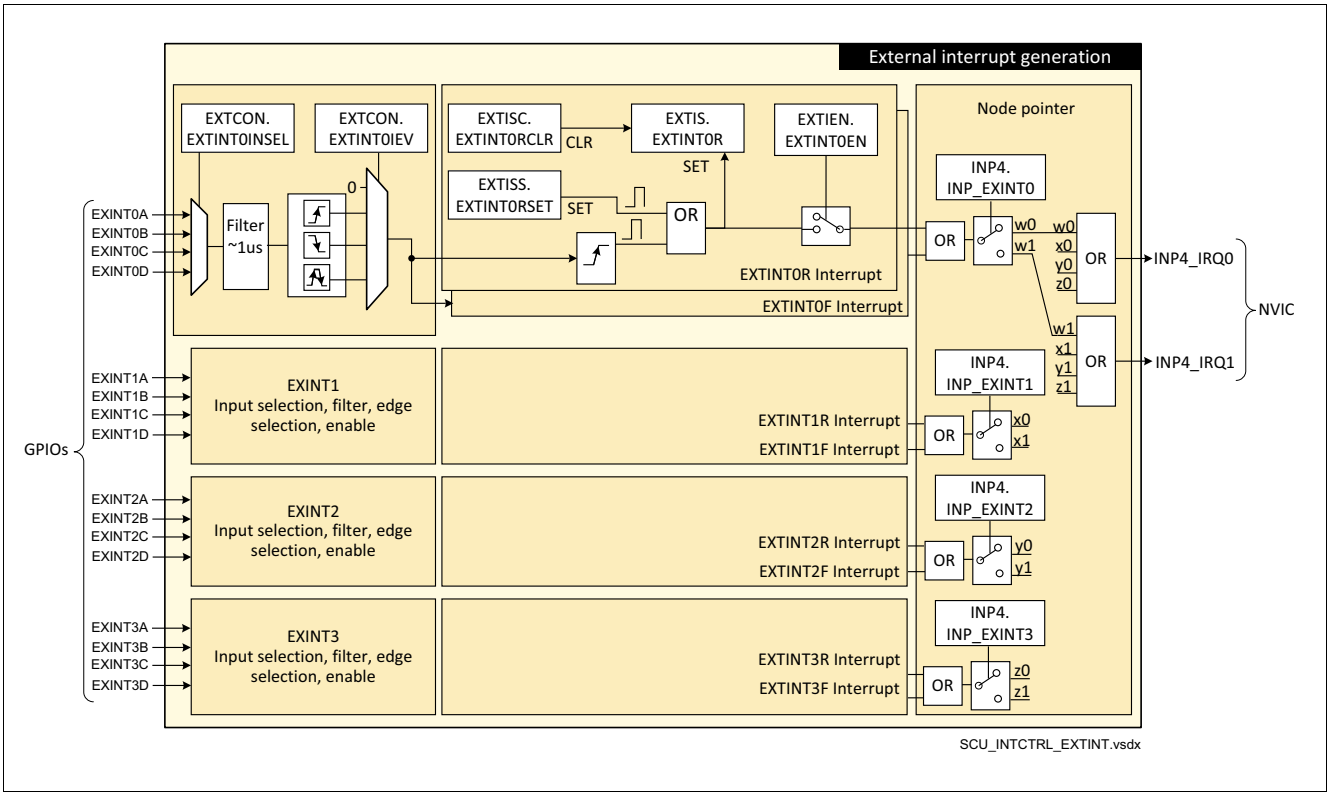


Figure 69 EXINT interrupt generation

6.8.1.6 UART0/1 interrupt generation

The interrupt requests from the UART0/2 modules are handled in the SCU and are propagated to the NVIC.

The UART0/1 interrupt scheme is described in **Figure 70**.

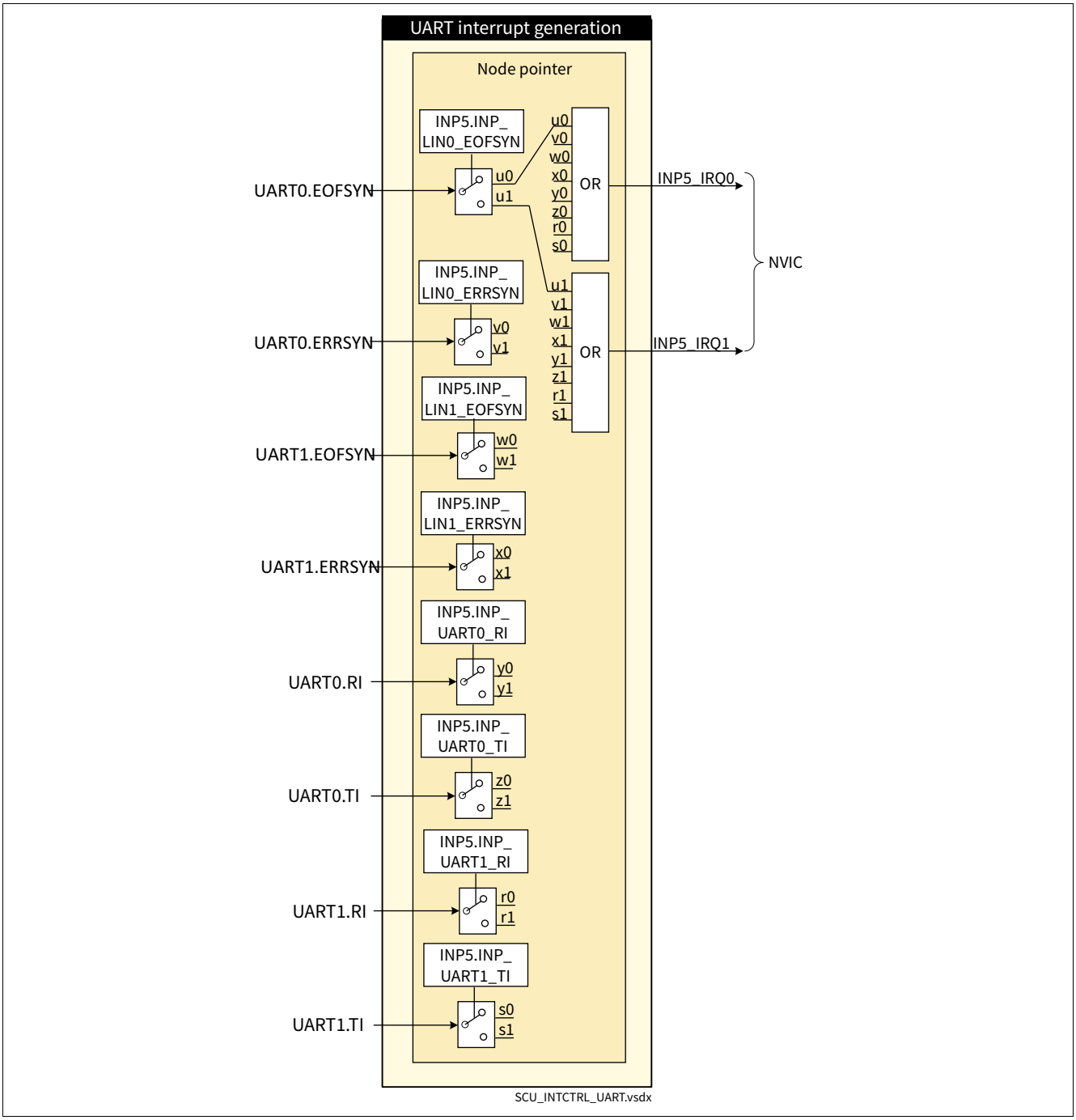


Figure 70 UART0/1 interrupt generation

6.8.1.7 SSC0/1 interrupt generation

The interrupt requests from the SSC0/2 modules are handled in the SCU and are propagated to the NVIC.
The SSC0/1 interrupt scheme is described in **Figure 71**.

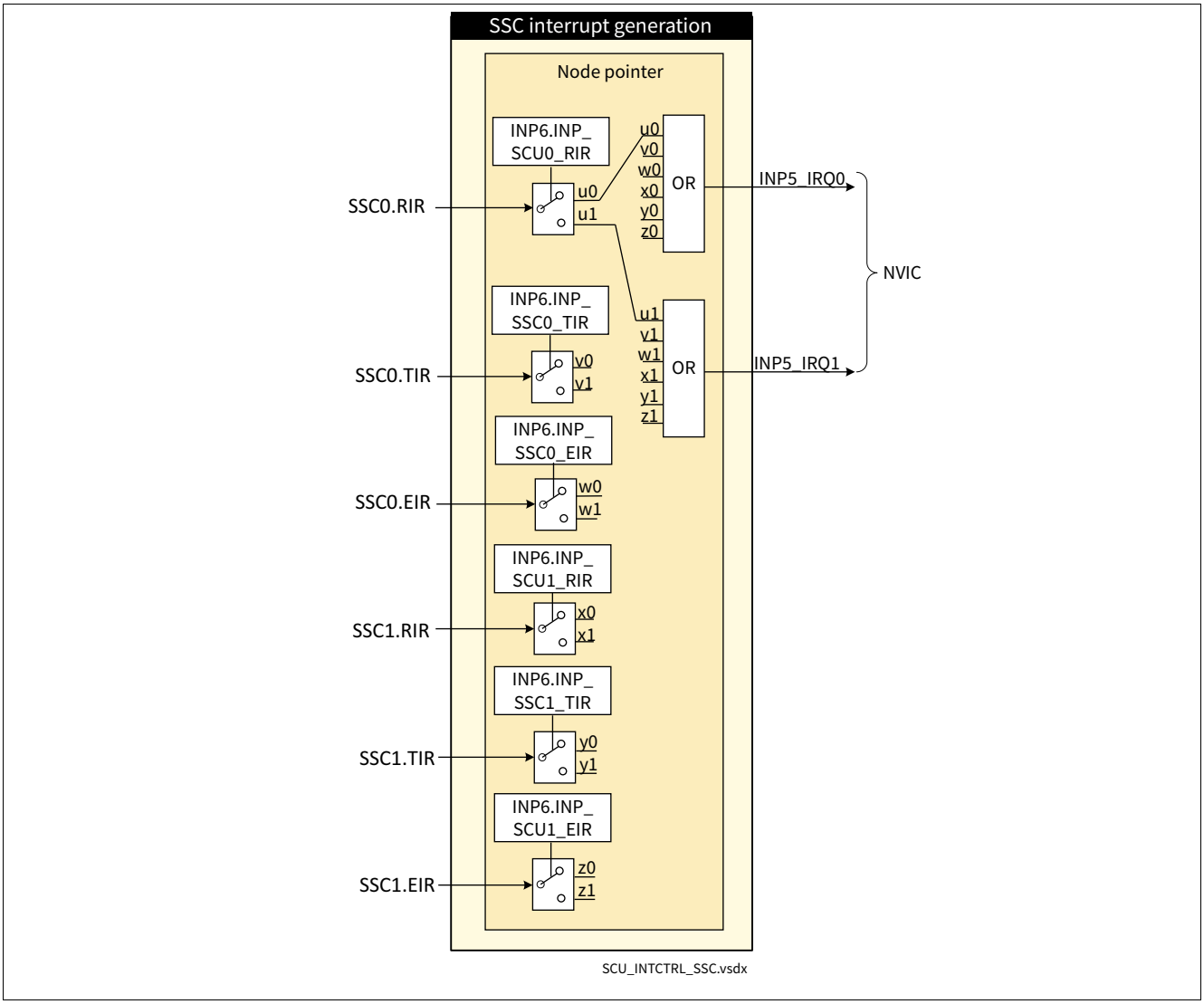


Figure 71 SSC0/1 interrupt generation

6.8.1.8 DMA interrupt generation

The DMA module can request an “end-of-DMA-task” IRQ for each of its eight channels (DMACHx, x = 0 to 7).

The end-of-DMA-task interrupt scheme allows following configuration:

- The interrupt status is stored in register DMAIS.DMACHx (x =0 to 7)
- Software can set and clear the interrupt status via DMAISS.DMACHxSET and DMAISC.DMACHxCLR (x = 0 to 7)
- The interrupt request can be enabled via DMAIEN.DMACHxEN (x = 0 to 7)
- The eight “end-of-DMA-task” IRQs are mapped to interrupt node pointer INP7 which has two NVIC request lines

The DMA module has an additional transmit error interrupt request (DMATRERR):

- The interrupt status is stored in MCU register DMA.ERR_CLRSC.ERR_CLR. Please refer to the MCU chapter **Direct Memory Access (DMA)**
- Software can clear the status in MCU register DMA.ERR_CLRSC.ERR_CLR. Please refer to the MCU chapter **Direct Memory Access (DMA)**
- The interrupt request can be enabled via DMAIEN.DMATRERREN
- The DMATRERR IRQ is mapped to interrupt node pointer INP7 which has two NVIC request lines

The interrupt scheme is described in **Figure 72**.

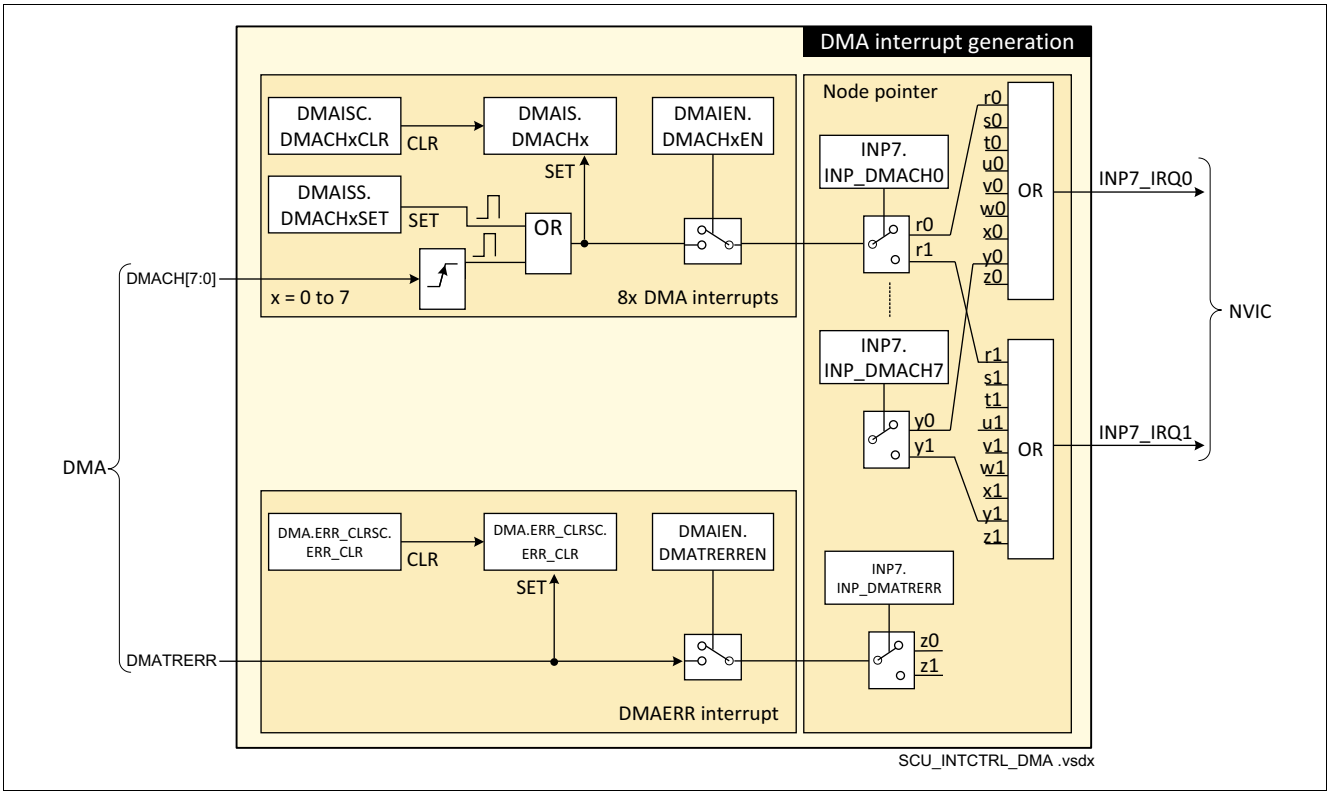


Figure 72 DMA interrupt generation

6.8.1.9 NMI exception

The NMI exception collects inputs from following modules:

- SCU system clock control (CLKNMI)
- MCU memory control (MEMNMI)
- ADC2 device supervisory (ADC2NMI)

The configuration and handling for the SCU system clock control (please also refer to **Chapter 6.6**):

- The NMI status is stored in register NMIS.NMI<src> (<src> = XTAL, PLL0, PLL1)
- Software can set and clear the NMI status via NMISRS.NMI<src>SET, NMISRC.NMI<src>CLR (<src> = XTAL, PLL0, PLL1)
- The NMI request can be enabled via NMICON.NMI<src>EN (<src> = XTAL, PLL0, PLL1)
- The NMIs are routed to the NVIC exception lines, see **Product definitions**, **Exception request mapping**

Note: The configuration and handling SFRs for MEMCTRL and ADC2 are described within these modules.

The interrupt scheme is described in **Figure 73**.

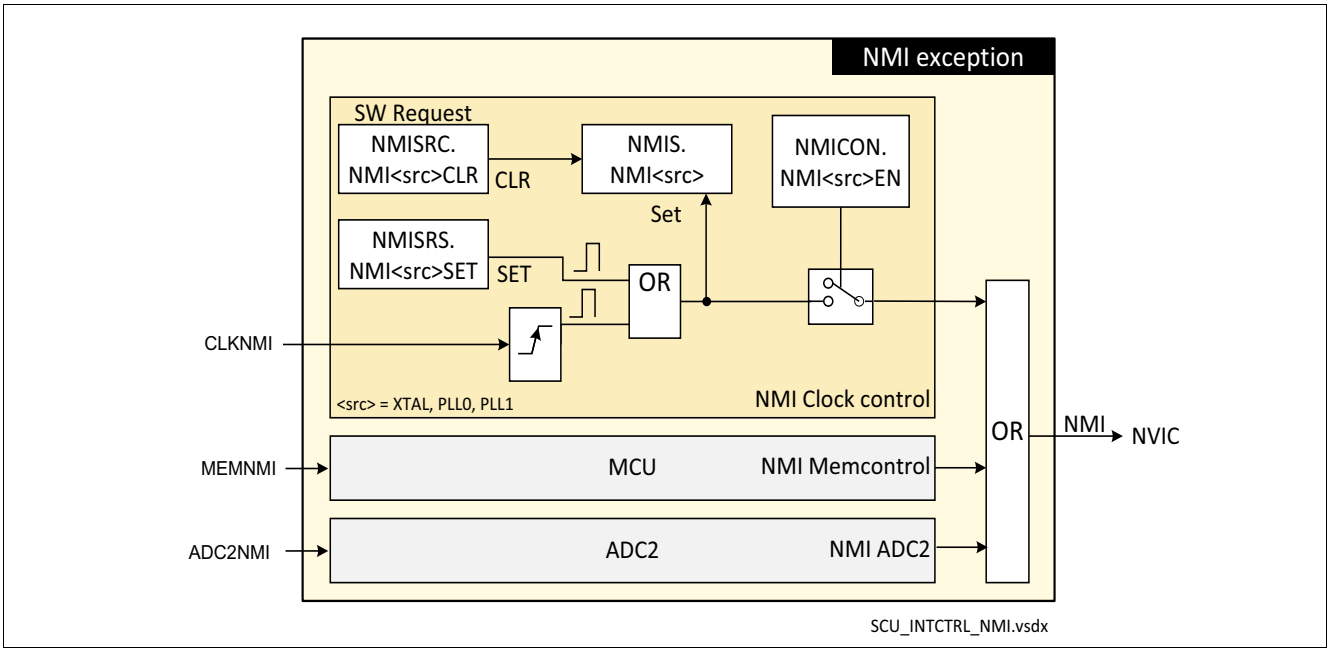


Figure 73 NMI exception

6.8.2 Interrupt assignment

For the interrupt request assignment please refer to **Product definitions**, **Interrupt request mapping**.

6.8.3 Exception request assignment

For exception handling there is one shared node pointer.

For the interrupt request assignment please refer to **Product definitions**, **Exception request mapping**. The exception requests can be individually enabled in SFR NMICON.

6.8.4 DMA request assignment

Many peripheral events can trigger a DMA request. The DMA requests can be assigned to eight DMA channels.

The assignment is done via following configuration, please refer to **Product definitions**, **DMA request mapping**:

- CCU7 events are mapped to one of the eight possible DMA channels via DMAP_CCU7.<CCU7_event>
- ADC1 events are mapped to one of the eight possible DMA channels via DMAP_ADC.<ADC1_event>
- SDADC events are mapped to one of the eight possible DMA channels via DMAP_ADC.<SDADC_event>
- Timer20/21 events are mapped to one of the eight possible DMA channels via DMAP_TIM.<T2x_event>
- GPT12 events are mapped to one of the eight possible DMA channels via DMAP_TIM.<GPT12_event
- SSC0/1 events are mapped to one of the eight possible DMA channels via DMAP_COM.<SSCx_event>
- UART0/1 events are mapped to one of the eight possible DMA channels via DMAP_COM.<UARTx_event>
- MultiCAN events are mapped to one of the eight possible DMA channels via DMAP_COM.<MultiCAN_event>

6.9 Register description SCU

6.9.1 SCU Address Maps

Table 81 Register Address Space - SCUREG

Module	Base Address	End Address	Note
SCU	48004000_H	$48007FFF_H$	

Table 82 Register Address Space - PLLREG

Module	Base Address	End Address	Note
PLL	48008000_H	$4800BFFF_H$	

Table 83 Register Overview - SCUREG (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CLKSEL	System Clock Select Register	0000_H	211
CLKCON	Peripheral Clock Prescaler Register	0004_H	212
CLKEN	Peripheral Clock Enable Register	0008_H	214
XTALCON	XTAL Control Register	0014_H	214
XTALSTAT	XTAL Status Register	0018_H	215
XTALSTATC	XTAL Status Clear Register	$001C_H$	216
XTALSTATS	XTAL Status Set Register	0020_H	217
INP0	Interrupt Node 0 Mapping Register	0024_H	217
INP1	Interrupt Node 1 Mapping Register	0028_H	218
INP2	Interrupt Node 2 Mapping Register	$002C_H$	219
INP3	Interrupt Node 3 Mapping Register	0030_H	219
INP4	Interrupt Node 4 Mapping Register	0034_H	220
INP5	Interrupt Node 5 Mapping Register	0038_H	221
INP6	Interrupt Node 6 Mapping Register	$003C_H$	222
INP7	Interrupt Node 7 Mapping Register	0040_H	223
NMICON	NMI Control Register	0044_H	224
NMISR	NMI Status Register	0048_H	224
NMISRC	NMI Status Clear Register	$004C_H$	225
NMISRS	NMI Status Set Register	0050_H	226
MONIEN	MON Interrupt Enable Register	0054_H	226
MONIS	MON Interrupt Status Register	0058_H	227
MONISC	MON Interrupt Status Clear Register	$005C_H$	228
MONISS	MON Interrupt Status Set Register	0060_H	229

Table 83 Register Overview - SCUREG (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Page Number
MONCON	MON Interrupt Configuration Register	0064_H	230
EXTIEN	External Interrupt Enable Register	0068_H	231
EXTIS	External Interrupt Status Register	$006C_H$	231
EXTISC	External Interrupt Status Clear Register	0070_H	233
EXTISS	External Interrupt Status Set Register	0074_H	234
EXTCON	EXT Interrupt Configuration Register	0078_H	235
GPTIEN	General Purpose Timer 12 Interrupt Enable Register	$007C_H$	236
GPTIS	General Purpose Timer 12 Interrupt Status Register	0080_H	237
GPTISC	General Purpose Timer 12 Interrupt Status Clear Register	0084_H	238
GPTISS	General Purpose Timer 12 Interrupt Status Set Register	0088_H	239
DMAIEN	DMA Interrupt Enable Register	$008C_H$	240
DMAIS	DMA Interrupt Status Register	0090_H	241
DMAISC	DAM Interrupt Status Clear Register	0094_H	242
DMAISS	DMA Interrupt Status Set Register	0098_H	244
DMAP_CCU7	DMA Channel Select Register CCU7	$009C_H$	245
DMAP_ADC	DMA Channel Select Register ADCs	$00A0_H$	247
DMAP_TIM	DMA Channel Select Register Timer	$00A4_H$	249
DMAP_COM	DMA Channel Select Register COM Modules	$00A8_H$	250
PMCON0	Power Mode Control Register	$00AC_H$	251
PMCON	Peripheral Management Control Register	$00B0_H$	252
SUSCTR	Module Suspend Control Register	$00B4_H$	253
PCU_CTRL	PCU Control Register	$00B8_H$	254
DMACTRL	DMA Control Register	$00E0_H$	255
LOCKUPCFG	CPU LOCKUP Config Register	$00E8_H$	256

Table 84 Register Overview - PLLREG (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CON0	PLL0 Control Register	0000_H	258
CON1	PLL1 Control Register	0004_H	259
SPCTR	PLL NDIV Spread Control Register	0008_H	260
STAT	PLL Status Register	$000C_H$	262
STATC	PLL Status Clear Register	0010_H	263
STATS	PLL Status Set Register	0014_H	264

6.9.2 SCUREG Registers

System Clock Select Register

CLKSEL
System Clock Select Register **(0000_H)** **RESET_TYPE_6 Value: $0000\ 0000_H$**

Field	Bits	Type	Description
SELSYS0	1:0	rw	**System Clock fsys0 Select** 00_B **HP40_CLK**, hp40_clk selected 01_B **XTAL_CLK**, xtal_clk selected 10_B **PLL0_CLK**, pll0_clk selected 11_B **PLL1_CLK**, pll1_clk selected
SELSYS1	3:2	rw	**System Clock fsys1 Select** 00_B **HP40_CLK**, hp40_clk selected 01_B **XTAL_CLK**, xtal_clk selected 10_B **PLL0_CLK**, pll0_clk selected 11_B **PLL1_CLK**, pll1_clk selected
RES	15:4, 31:20	r	**Reserved**
SELCLKOUT	18:16	rw	**CLKOUT Selection** 000_B **HP_CLK**, hp_clk selected as CLKOUT 001_B **XTAL_CLK**, xtal_clk selected as CLKOUT 010_B **PLL0_CLK**, pll0_clk selected as CLKOUT 011_B **PLL1_CLK**, pll1_clk selected as CLKOUT 100_B **CPU_CLK**, cpu_clk selected as CLKOUT 101_B **M_CLK**, m_clk selected as CLKOUT 110_B **REF_CLK**, ref_clk selected as CLKOUT 111_B **CAN_RX_CLK_DIV4** , reserved
CLKOUTEN	19	rw	**CLKOUT Enable** 0_B **Disable**, CLKOUT disabled 1_B **Enable**, CLKOUT enabled

Register description SCU

Peripheral Clock Prescaler Register

CLKCON
Peripheral Clock Prescaler Register **(0004_H)** **RESET_TYPE_4 Value: $0000\ 0290_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES				DIV2CLKOUT	PRECLKOUT			RES	PREUART			RES	PRECAN		
r				rw	rw			r	rwpht			r	rwpht		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				PREMI			PREFILT					RES	PRECPU		
r				rw			rw					r	rw		

Field	Bits	Type	Description
PRECPU	2:0	rw	**CPU_CLK Prescaler Setting (based on sys0_clk)** 000_B **DIV1**, clk divided by 1 001_B **DIV2**, clk divided by 2 010_B **DIV3**, clk divided by 3 011_B **DIV4**, clk divided by 4 100_B **DIV5**, clk divided by 5 101_B **DIV8**, clk divided by 8 110_B **DIV16**, clk divided by 16 111_B **DIV20**, clk_divided by 20
RES	3, 15:12, 19, 23, 31:28	r	**Reserved**
PREFILT	8:4	rw	**TFILT_CLK Prescaler Setting (based on sys0_clk)** Shall be set to 4MHz. leading to tfilt_bdrv_clk = 4MHz, tfilt_clk = 2MHz 00_H **DIV_1**, clock divided by 1 … $1F_H$ **DIV_32**, clock divided by 32
PREMI	11:9	rw	**MI_CLK Prescaler Setting (based on sys0_clk)** used as br_drv and can_phy clock, shall be set to 20MHz. 000_B **DIV1**, clk divided by 1 001_B **DIV2**, clk divided by 2 010_B **DIV3**, clk divided by 3 011_B **DIV4**, clk divided by 4 100_B **DIV5**, clk divided by 5 101_B **DIV8**, clk divided by 8 110_B **DIV16**, clk divided by 16 111_B **DIV20**, clk divided by 20

Field	Bits	Type	Description
PRECAN	18:16	rwpht	**CAN_CLK Prescaler Setting (based on sys1_clk)** The bit field can only be written if CLKEN.CANCLKEN = 0 000_B **DIV1**, clk divided by 1 001_B **DIV2**, clk divided by 2 010_B **DIV3**, clk divided by 3 011_B **DIV4**, clk divided by 4 100_B **DIV5**, clk divided by 5 101_B **DIV8**, clk divided by 8 110_B **DIV16**, clk divided by 16 111_B **DIV20**, clk divided by 20
PREUART	22:20	rwpht	**UART_CLK Prescaler Setting (based on sys1_clk)** The bit field can only be written if CLKEN.UARTCLKEN = 0 000_B **DIV1**, clk divided by 1 001_B **DIV2**, clk divided by 2 010_B **DIV3**, clk divided by 3 011_B **DIV4**, clk divided by 4 100_B **DIV5**, clk divided by 5 101_B **DIV8**, clk divided by 8 110_B **DIV16**, clk divided by 16 111_B **DIV20**, clk divided by 20
PRECLKOUT	26:24	rw	**CLKOUT_CLK Prescaler Setting (based on selected clock by SELCLKOUT)** 000_B **DIV1**, clk divided by 1 001_B **DIV2**, clk divided by 2 010_B **DIV3**, clk divided by 3 011_B **DIV4**, clk divided by 4 100_B **DIV5**, clk divided by 5 101_B **DIV8**, clk divided by 8 110_B **DIV16**, clk divided by 16 111_B **DIV20**, clk divided by 20
DIV2CLKOUT	27	rw	**CLKOUT clock divider by2** Divider by2 for CLKOUT with 50/50 duty cycle. 0_B **DIV1**, CLKOUT divided by 1 1_B **DIV2**, CLKOUT divided by 2 (50/50 duty cycle)

Register description SCU

Peripheral Clock Enable Register

CLKEN
Peripheral Clock Enable Register **(0008_H)** **RESET_TYPE_4 Value: 0000 0003$_H$**

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

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

Field	Bits	Type	Description
UARTCLKEN	0	rw	**UART Clock Enable** 0_B **Disable**, Clock disabled 1_B **Enable**, Clock enabled
CANCLKEN	1	rw	**CAN Clock Enable** 0_B **Disable**, Clock disabled 1_B **Enable**, Clock enabled
RES	31:2	r	**Reserved**

XTAL Control Register

XTALCON
XTAL Control Register **(0014_H)** **RESET_TYPE_6 Value: 0000 0033$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES			XWDGRES	RES			XWDGEN	RES							
r			w	r			rw	r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										XTALHYS		RES		XTALHYSEN	XPD
r										rw		r		rw	rw

Field	Bits	Type	Description
XPD	0	rw	**XTAL Power Down Control** Note: When XPD is set, switch of PLLx reference clock source to HP_CLK has to be done asynchronous see PLL.CONx.INSEL (x=0,1) 0_B **Enable**, XTAL is powered 1_B **Disable**, XTAL power down

Field	Bits	Type	Description
XTALHYSEN	1	rw	**XTAL Hysteresis Enable** 0_B **Disable**, lowest hysteresis, filter off 1_B **Enable**, see values according to XTALHYS setting
RES	3:2, 23:6, 27:25, 31:29	r	**Reserved** Always read as 0
XTALHYS	5:4	rw	**XTAL Hysteresis Control** The hysteresis control defines the min pulse width to pass through the shaper in ns Note: These settings apply only if XTALHYSEN = 1 00_B **XHYST_0**, high hysteresis, slow filter 01_B **XHYST_1**, medium hysteresis, medium filter 10_B **XHYST_2**, low hysteresis, fast filter 11_B **XHYST_3**, very low hysteresis, filter off
XWDGEN	24	rw	**XTALWDG Enable** 0_B **Disable**, XTALWDG disabled 1_B **Enable**, XTALWDG enabled
XWDGRES	28	w	**XTALWDG Reset** Note: Resetting XTALWDG will set XTALSTAT.XTAL_FAIL_STS flag. 0_B **NORESET**, XTALWDG not reset 1_B **RESET**, XTALWDG reset

XTAL Status Register

XTALSTAT
XTAL Status Register **(0018_H)** **RESET_TYPE_6 Value: $0000\ 0101_H$**

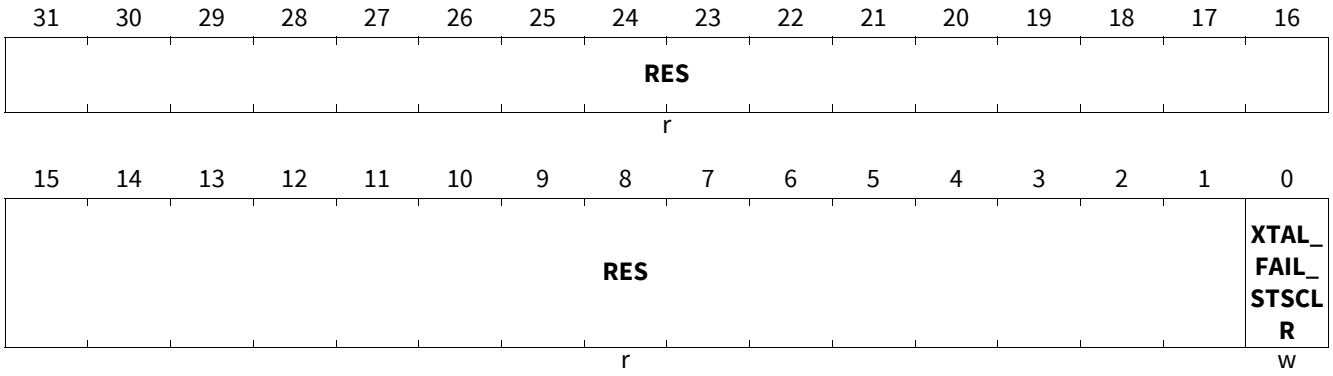
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES							XTALFAIL	RES							XTAL_FAIL_STS
r							r	r							rhxr

Field	Bits	Type	Description
XTAL_FAIL_STS	0	rhxr	**XTAL Watchdog Fail Latched Status** This bit shows the latched XTAL Watchdog Fail status. This bit can only be cleared if XTALFAIL status bit is OK. If this bit is set sysx_clk using xtal_clk is switched to hp_clk. If this bit is cleared sysx_clk is switched back to xtal_clk. This bit is cleared by HW if XTAL_FAIL_STSC bit is written to 1. 0_B **OK**, XTAL Clock is OK 1_B **FAIL**, XTAL Clock Fail
RES	7:1, 31:9	r	**Reserved**
XTALFAIL	8	r	**XTAL Watchdog Fail Current Status** This bit shows the current status of the XTALWDG. Only if this bit is not set XTAL_FAIL_STS shall be cleared. 0_B **OK**, XTAL Clock is OK 1_B **FAIL**, XTAL Clock Fail

XTAL Status Clear Register

XTALSTATC
XTAL Status Clear Register **($001C_H$)** **RESET_TYPE_4 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
XTAL_FAIL_STSCLR	0	w	**XTAL Watchdog Fail Latched Status Clear** Writing this bit clears XTAL_FAIL_STS. This bit is set by software and automatically cleared by hardware. 0_B **NotCleared**, Status not cleared 1_B **Cleared**, Status cleared
RES	31:1	r	**Reserved**

Register description SCU

XTAL Status Set Register

XTALSTATS
XTAL Status Set Register **(0020_H)** **RESET_TYPE_4 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															XTAL_FAIL_STSSET
r															w

Field	Bits	Type	Description
XTAL_FAIL_STSSET	0	w	**XTAL Watchdog Fail Latched Status Set** This bit is set by software and automatically cleared by hardware. 0_B **NoSET**, Status not Set 1_B **SET**, Status Set
RES	31:1	r	**Reserved**

Interrupt Node 0 Mapping Register

INP0
Interrupt Node 0 Mapping Register **(0024_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										INP_CSC	INP_ARVG	INP_CANTX	INP_BDRV_IRQ1	INP_BDRV_IRQ0	INP_PMU
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
INP_PMU	0	rw	**PMU Interrupt Mapping** 0_B **IRQ0**, PMU INT mapped to IRQ0 1_B **IRQ1**, PMU INT mapped to IRQ1
INP_BDRV_IRQ0	1	rw	**Bridge Driver Interrupt 0 Mapping** 0_B **IRQ0**, BDRV_IRQ0 mapped to IRQ0 1_B **IRQ1**, BDRV_IRQ0 mapped to IRQ1

Field	Bits	Type	Description
INP_BDRV_IRQ1	2	rw	**Bridge Driver Interrupt 1 Mapping** 0_B **IRQ0**, BDRV_IRQ1 mapped to IRQ0 1_B **IRQ1**, BDRV_IRQ1 mapped to IRQ1
INP_CANTX	3	rw	**CANTX Interrupt Mapping** 0_B **IRQ0**, CANTX INT mapped to IRQ0 1_B **IRQ1**, CANTX INT mapped to IRQ1
INP_ARVG	4	rw	**ARVG Interrupt Mapping** 0_B **IRQ0**, ARVG INT mapped to IRQ0 1_B **IRQ1**, ARVG INT mapped to IRQ1
INP_CSC	5	rw	**CSC Interrupt Mapping** 0_B **IRQ0**, CSC INT mapped to IRQ0 1_B **IRQ1**, CSC INT mapped to IRQ1
RES	31:6	r	**Reserved**

Interrupt Node 1 Mapping Register

INP1
Interrupt Node 1 Mapping Register **(0028_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										INP_GPT2CR	INP_GPT2T6	INP_GPT2T5	INP_GPT1T4	INP_GPT1T3	INP_GPT1T2
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
INP_GPT1T2	0	rw	**GPT1T2 Interrupt Mapping** 0_B **IRQ7**, GPT1T2 INT mapped to IRQ7 1_B **IRQ8**, GPT1T2 INT mapped to IRQ8
INP_GPT1T3	1	rw	**GPT1T3 Interrupt Mapping** 0_B **IRQ7**, GPT1T3 INT mapped to IRQ7 1_B **IRQ8**, GPT1T3 INT mapped to IRQ8
INP_GPT1T4	2	rw	**GPT1T4 Interrupt Mapping** 0_B **IRQ7**, GPT1T4 INT mapped to IRQ7 1_B **IRQ8**, GPT1T4 INT mapped to IRQ8
INP_GPT2T5	3	rw	**GPT2T5 Interrupt Mapping** 0_B **IRQ7**, GPT2T5 INT mapped to IRQ7 1_B **IRQ8**, GPT2T5 INT mapped to IRQ8
INP_GPT2T6	4	rw	**GPT2T6 Interrupt Mapping** 0_B **IRQ7**, GPT2T6 INT mapped to IRQ7 1_B **IRQ8**, GPT2T6 INT mapped to IRQ8

Field	Bits	Type	Description
INP_GPT2CR	5	rw	**GPT2CR Interrupt Mapping** 0_B **IRQ7**, GPT2CR INT mapped to IRQ7 1_B **IRQ8**, GPT2CR INT mapped to IRQ8
RES	31:6	r	**Reserved**

Interrupt Node 2 Mapping Register

INP2
Interrupt Node 2 Mapping Register **($002C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES													**INP_MON3**	**INP_MON2**	**INP_MON1**
r													rw	rw	rw

Field	Bits	Type	Description
INP_MON1	0	rw	**MON1 Interrupt Mapping** 0_B **IRQ12**, MON1 INT mapped to IRQ12 1_B **IRQ13**, MON1 INT mapped to IRQ13
INP_MON2	1	rw	**MON2 Interrupt Mapping** 0_B **IRQ12**, MON2 INT mapped to IRQ12 1_B **IRQ13**, MON2 INT mapped to IRQ13
INP_MON3	2	rw	**MON3 Interrupt Mapping** 0_B **IRQ12**, MON3 INT mapped to IRQ12 1_B **IRQ13**, MON3 INT mapped to IRQ13
RES	31:3	r	**Reserved**

Interrupt Node 3 Mapping Register

INP3
Interrupt Node 3 Mapping Register **(0030_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES											**INP_BEMF2**	**INP_BEMF1**	**INP_BEMF0**	**INP_SDADC1**	**INP_SDADC0**
r											rw	rw	rw	rw	rw

Field	Bits	Type	Description
INP_SDADC0	0	rw	**SDADC0 Interrupt Mapping** 0_B **IRQ18**, SDADC0 INT mapped to IRQ18 1_B **IRQ19**, SDADC0 INT mapped to IRQ19
INP_SDADC1	1	rw	**SDADC1 Interrupt Mapping** 0_B **IRQ18**, SDADC1 INT mapped to IRQ18 1_B **IRQ19**, SDADC1 INT mapped to IRQ19
INP_BEMF0	2	rw	**BEMF0 Interrupt Mapping** 0_B **IRQ18**, BEMF0 INT mapped to IRQ18 1_B **IRQ19**, BEMF0 INT mapped to IRQ19
INP_BEMF1	3	rw	**BEMF1 Interrupt Mapping** 0_B **IRQ18**, BEMF1 INT mapped to IRQ18 1_B **IRQ19**, BEMF1 INT mapped to IRQ19
INP_BEMF2	4	rw	**BEMF2 Interrupt Mapping** 0_B **IRQ18**, BEMF2 INT mapped to IRQ18 1_B **IRQ19**, BEMF2 INT mapped to IRQ19
RES	31:5	r	**Reserved**

Interrupt Node 4 Mapping Register

INP4

Interrupt Node 4 Mapping Register **(0034_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

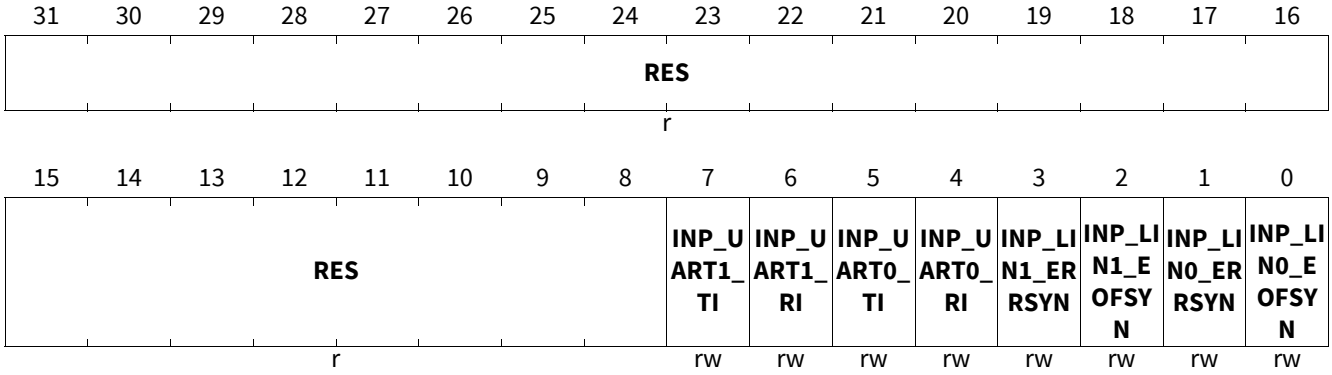
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES												INP_EXINT3	INP_EXINT2	INP_EXINT1	INP_EXINT0
r												rw	rw	rw	rw

Field	Bits	Type	Description
INP_EXINT0	0	rw	**EXINT0 Interrupt Mapping** 0_B **IRQ20**, EXINT0 INT mapped to IRQ20 1_B **IRQ21**, EXINT0 INT mapped to IRQ21
INP_EXINT1	1	rw	**EXINT1 Interrupt Mapping** 0_B **IRQ20**, EXINT1 INT mapped to IRQ20 1_B **IRQ21**, EXINT1 INT mapped to IRQ21
INP_EXINT2	2	rw	**EXINT2 Interrupt Mapping** 0_B **IRQ20**, EXINT2 INT mapped to IRQ20 1_B **IRQ21**, EXINT2 INT mapped to IRQ21
INP_EXINT3	3	rw	**EXINT3 Interrupt Mapping** 0_B **IRQ20**, EXINT3 INT mapped to IRQ20 1_B **IRQ21**, EXINT3 INT mapped to IRQ21

Field	Bits	Type	Description
RES	31:4	r	**Reserved**

Interrupt Node 5 Mapping Register

INP5
Interrupt Node 5 Mapping Register **(0038_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
INP_LIN0_EOFSYN	0	rw	**LIN0_EOFSYN Interrupt Mapping** 0_B **IRQ22**, LIN0_EOFSYN INT mapped to IRQ22 1_B **IRQ23**, LIN0_EOFSYN INT mapped to IRQ23
INP_LIN0_ERRSYN	1	rw	**LIN0_ERRSYN Interrupt Mapping** 0_B **IRQ22**, LIN0_ERRSYN INT mapped to IRQ22 1_B **IRQ23**, LIN0_ERRSYN INT mapped to IRQ23
INP_LIN1_EOFSYN	2	rw	**LIN1_EOFSYN Interrupt Mapping** 0_B **IRQ22**, LIN1_EOFSYN INT mapped to IRQ22 1_B **IRQ23**, LIN1_EOFSYN INT mapped to IRQ23
INP_LIN1_ERRSYN	3	rw	**LIN1_ERRSYN Interrupt Mapping** 0_B **IRQ22**, LIN1_ERRSYN INT mapped to IRQ22 1_B **IRQ23**, LIN1_ERRSYN INT mapped to IRQ23
INP_UART0_RI	4	rw	**UART0_RI Interrupt Mapping** 0_B **IRQ22**, UART0_RI INT mapped to IRQ22 1_B **IRQ23**, UART0_RI INT mapped to IRQ23
INP_UART0_TI	5	rw	**UART0_TI Interrupt Mapping** 0_B **IRQ22**, UART0_TI INT mapped to IRQ22 1_B **IRQ23**, UART0_TI INT mapped to IRQ23
INP_UART1_RI	6	rw	**UART1_RI Interrupt Mapping** 0_B **IRQ22**, UART1_RI INT mapped to IRQ22 1_B **IRQ23**, UART1_RI INT mapped to IRQ23
INP_UART1_TI	7	rw	**UART1_TI Interrupt Mapping** 0_B **IRQ22**, UART1_TI INT mapped to IRQ22 1_B **IRQ23**, UART1_TI INT mapped to IRQ23
RES	31:8	r	**Reserved**

Interrupt Node 6 Mapping Register

INP6
Interrupt Node 6 Mapping Register **($003C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES									INP_SSC1_EIR	INP_SSC1_TIR	INP_SSC1_RIR	RES	INP_SSC0_EIR	INP_SSC0_TIR	INP_SSC0_RIR
r									rw	rw	rw	r	rw	rw	rw

Field	Bits	Type	Description
INP_SSC0_RIR	0	rw	**SSC0_RIR Interrupt Mapping** 0_B **IRQ24**, SSC0_RIR INT mapped to IRQ24 1_B **IRQ25**, SSC0_RIR INT mapped to IRQ25
INP_SSC0_TIR	1	rw	**SSC0_TIR Interrupt Mapping** 0_B **IRQ24**, SSC0_TIR INT mapped to IRQ24 1_B **IRQ25**, SSC0_TIR INT mapped to IRQ25
INP_SSC0_EIR	2	rw	**SSC0_EIR Interrupt Mapping** 0_B **IRQ24**, SSC0_EIR INT mapped to IRQ24 1_B **IRQ25**, SSC0_EIR INT mapped to IRQ25
RES	3, 31:7	r	**Reserved**
INP_SSC1_RIR	4	rw	**SSC1_RIR Interrupt Mapping** 0_B **IRQ24**, SSC1_RIR INT mapped to IRQ24 1_B **IRQ25**, SSC1_RIR INT mapped to IRQ25
INP_SSC1_TIR	5	rw	**SSC1_TIR Interrupt Mapping** 0_B **IRQ24**, SSC1_TIR INT mapped to IRQ24 1_B **IRQ25**, SSC1_TIR INT mapped to IRQ25
INP_SSC1_EIR	6	rw	**SSC1_EIR Interrupt Mapping** 0_B **IRQ24**, SSC1_EIR INT mapped to IRQ24 1_B **IRQ25**, SSC1_EIR INT mapped to IRQ25

Interrupt Node 7 Mapping Register

INP7
Interrupt Node 7 Mapping Register **(0040_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES							INP_DMATRERR	INP_DMACH7	INP_DMACH6	INP_DMACH5	INP_DMACH4	INP_DMACH3	INP_DMACH2	INP_DMACH1	INP_DMACH0
r							rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
INP_DMACH0	0	rw	**DMACH0 Interrupt Mapping** 0_B **IRQ29**, DMACH0 INT mapped to IRQ29 1_B **IRQ30**, DMACH0 INT mapped to IRQ30
INP_DMACH1	1	rw	**DMACH1 Interrupt Mapping** 0_B **IRQ29**, DMACH1 INT mapped to IRQ29 1_B **IRQ30**, DMACH1 INT mapped to IRQ30
INP_DMACH2	2	rw	**DMACH2 Interrupt Mapping** 0_B **IRQ29**, DMACH2 INT mapped to IRQ29 1_B **IRQ30**, DMACH2 INT mapped to IRQ30
INP_DMACH3	3	rw	**DMACH3 Interrupt Mapping** 0_B **IRQ29**, DMACH3 INT mapped to IRQ29 1_B **IRQ30**, DMACH3 INT mapped to IRQ30
INP_DMACH4	4	rw	**DMACH4 Interrupt Mapping** 0_B **IRQ29**, DMACH4 INT mapped to IRQ29 1_B **IRQ30**, DMACH4 INT mapped to IRQ30
INP_DMACH5	5	rw	**DMACH5 Interrupt Mapping** 0_B **IRQ29**, DMACH5 INT mapped to IRQ29 1_B **IRQ30**, DMACH5 INT mapped to IRQ30
INP_DMACH6	6	rw	**DMACH6 Interrupt Mapping** 0_B **IRQ29**, DMACH6 INT mapped to IRQ29 1_B **IRQ30**, DMACH6 INT mapped to IRQ30
INP_DMACH7	7	rw	**DMACH7 Interrupt Mapping** 0_B **IRQ29**, DMACH7 INT mapped to IRQ29 1_B **IRQ30**, DMACH7 INT mapped to IRQ30
INP_DMATRERR	8	rw	**DMATRERR Interrupt Mapping** 0_B **IRQ29**, DMATRERR INT mapped to IRQ29 1_B **IRQ30**, DMATRERR INT mapped to IRQ30
RES	31:9	r	**Reserved**

NMI Control Register

NMICON
NMI Control Register **(0044_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES													NMIPLL1EN	NMIPLL0EN	NMIXTALEN
r													rw	rw	rw

Field	Bits	Type	Description
NMIXTALEN	0	rw	**XTAL Watchdog Fail NMI Enable** 0_B **Disable**, NMI disabled 1_B **Enable**, NMI enabled
NMIPLL0EN	1	rw	**PLL0 Loss of Lock NMI Enable** 0_B **Disable**, NMI disabled 1_B **Enable**, NMI enabled
NMIPLL1EN	2	rw	**PLL1 Loss of Lock NMI Enable** PLL1 Loss of Lock NMI Enable 0_B **Disable**, NMI disabled 1_B **Enable**, NMI enabled
RES	31:3	r	**Reserved**

NMI Status Register

NMISR
NMI Status Register **(0048_H)** **RESET_TYPE_4 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES													NMIPLL1	NMIPLL0	NMIXTAL
r													rhxre	rhxre	rhxre

Field	Bits	Type	Description
NMIXTAL	0	rhxre	**XTAL Watchdog Fail NMI Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoNMI**, NMI not occurred 1_B **NMI**, NMI occurred
NMIPLL0	1	rhxre	**PLL0 Loss of Lock NMI Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoNMI**, NMI not occurred 1_B **NMI**, NMI occurred
NMIPLL1	2	rhxre	**PLL1 Loss of Lock NMI Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoNMI**, NMI not occurred 1_B **NMI**, NMI occurred
RES	31:3	r	**Reserved**

NMI Status Clear Register

NMISRC

NMI Status Clear Register **($004C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES													NMIPLL1CLR	NMIPLL0CLR	NMIXTALCLR
r													w	w	w

Field	Bits	Type	Description
NMIXTALCLR	0	w	**XTAL Watchdog Fail NMI Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, NMI not cleared 1_B **Cleared**, NMI cleared
NMIPLL0CLR	1	w	**PLL0 Loss of Lock NMI Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, NMI not cleared 1_B **Cleared**, NMI cleared
NMIPLL1CLR	2	w	**PLL1 Loss of Lock NMI Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, NMI not cleared 1_B **Cleared**, NMI cleared

Field	Bits	Type	Description
RES	31:3	r	**Reserved**

NMI Status Set Register

NMISRS
NMI Status Set Register **(0050_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES													**NMIPLL1SET**	**NMIPLL0SET**	**NMIXTALSET**
r													w	w	w

Field	Bits	Type	Description
NMIXTALSET	0	w	**XTAL Watchdog Fail NMI Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, NMI not set 1_B **SET**, NMI set
NMIPLL0SET	1	w	**PLL0 Loss of Lock NMI Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, NMI not set 1_B **SET**, NMI set
NMIPLL1SET	2	w	**PLL1 Loss of Lock NMI Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, NMI not set 1_B **SET**, NMI set
RES	31:3	r	**Reserved**

MON Interrupt Enable Register

MONIEN
MON Interrupt Enable Register **(0054_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES													**MON3EN**	**MON2EN**	**MON1EN**
r													rw	rw	rw

Field	Bits	Type	Description
MON1EN	0	rw	**MON1 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
MON2EN	1	rw	**MON2 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
MON3EN	2	rw	**MON3 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
RES	31:3	r	**Reserved**

MON Interrupt Status Register

MONIS

MON Interrupt Status Register **(0058_H)** **RESET_TYPE_4 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										**MON3F**	**MON3R**	**MON2F**	**MON2R**	**MON1F**	**MON1R**
r										rhxre	rhxre	rhxre	rhxre	rhxre	rhxre

Field	Bits	Type	Description
MON1R	0	rhxre	**MON1 Rising Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
MON1F	1	rhxre	**MON1 Falling Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
MON2R	2	rhxre	**MON2 Rising Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred

Field	Bits	Type	Description
MON2F	3	rhxre	**MON2 Falling Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
MON3R	4	rhxre	**MON3 Rising Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
MON3F	5	rhxre	**MON3 Falling Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
RES	31:6	r	**Reserved**

MON Interrupt Status Clear Register

MONISC

MON Interrupt Status Clear Register **($005C_H$)** **RESET_TYPE_3 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										MON3 FCLR	MON3 RCLR	MON2 FCLR	MON2 RCLR	MON1 FCLR	MON1 RCLR
r										w	w	w	w	w	w

Field	Bits	Type	Description
MON1RCLR	0	w	**MON1 Rising Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
MON1FCLR	1	w	**MON1 Falling Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
MON2RCLR	2	w	**MON2 Rising Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared

Field	Bits	Type	Description
MON2FCLR	3	w	**MON2 Falling Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
MON3RCLR	4	w	**MON3 Rising Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
MON3FCLR	5	w	**MON3 Falling Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
RES	31:6	r	**Reserved**

MON Interrupt Status Set Register

MONISS

MON Interrupt Status Set Register **(0060_H)** **RESET_TYPE_3 Value: 0000 0000$_H$**

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

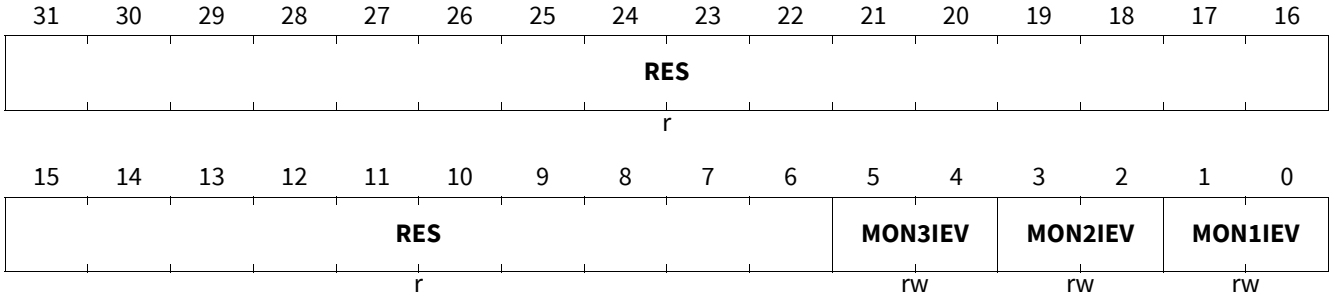
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										MON3FSET	MON3RSET	MON2FSET	MON2RSET	MON1FSET	MON1RSET
r										w	w	w	w	w	w

Field	Bits	Type	Description
MON1RSET	0	w	**MON1 Rising Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
MON1FSET	1	w	**MON1 Falling Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
MON2RSET	2	w	**MON2 Rising Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
MON2FSET	3	w	**MON2 Falling Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET

Field	Bits	Type	Description
MON3RSET	4	w	**MON3 Rising Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
MON3FSET	5	w	**MON3 Falling Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
RES	31:6	r	**Reserved**

MON Interrupt Configuration Register

MONCON
MON Interrupt Configuration Register **(0064_H)** **RESET_TYPE_3 Value: 0000 0000_H**



Field	Bits	Type	Description
MON1IEV	1:0	rw	**MON1 Interrupt Event Select** 00_B **Disable**, MON Int disabled 01_B **Enable**, MON Int on rising edge 10_B **FallingEdge**, MON Int on falling edge 11_B **BothEdges**, MON Int on both edges
MON2IEV	3:2	rw	**MON2 Interrupt Event Select** 00_B **Disable**, MON Int disabled 01_B **Enable**, MON Int on rising edge 10_B **FallingEdge**, MON Int on falling edge 11_B **BothEdges**, MON Int on both edges
MON3IEV	5:4	rw	**MON3 Interrupt Event Select** 00_B **Disable**, MON Int disabled 01_B **Enable**, MON Int on rising edge 10_B **FallingEdge**, MON Int on falling edge 11_B **BothEdges**, MON Int on both edges
RES	31:6	r	**Reserved**

External Interrupt Enable Register

EXTIEN
External Interrupt Enable Register **(0068_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES												EXTINT3EN	EXTINT2EN	EXTINT1EN	EXTINT0EN
r												rw	rw	rw	rw

Field	Bits	Type	Description
EXTINT0EN	0	rw	**External EXTINT0 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
EXTINT1EN	1	rw	**External EXTINT1 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
EXTINT2EN	2	rw	**External EXTINT2 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
EXTINT3EN	3	rw	**External EXTINT3 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
RES	31:4	r	**Reserved**

External Interrupt Status Register

EXTIS
External Interrupt Status Register **($006C_H$)** **RESET_TYPE_4 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								EXTINT3F	EXTINT3R	EXTINT2F	EXTINT2R	EXTINT1F	EXTINT1R	EXTINT0F	EXTINT0R
r								rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre

Field	Bits	Type	Description
EXTINT0R	0	rhxre	**External EXTINT0 Rising Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
EXTINT0F	1	rhxre	**External EXTINT0 Falling Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
EXTINT1R	2	rhxre	**External EXTINT1 Rising Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
EXTINT1F	3	rhxre	**External EXTINT1 Falling Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
EXTINT2R	4	rhxre	**External EXTINT2 Rising Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
EXTINT2F	5	rhxre	**External EXTINT2 Falling Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
EXTINT3R	6	rhxre	**External EXTINT3 Rising Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
EXTINT3F	7	rhxre	**External EXTINT3 Falling Edge Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
RES	31:8	r	**Reserved**

External Interrupt Status Clear Register

EXTISC
External Interrupt Status Clear Register (0070$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								EXTINT3FCLR	EXTINT3RCLR	EXTINT2FCLR	EXTINT2RCLR	EXTINT1FCLR	EXTINT1RCLR	EXTINT0FCLR	EXTINT0RCLR
r								w	w	w	w	w	w	w	w

Field	Bits	Type	Description
EXTINT0RCLR	0	w	**External EXTINT0 Rising Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
EXTINT0FCLR	1	w	**External EXTINT0 Falling Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
EXTINT1RCLR	2	w	**External EXTINT1 Rising Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
EXTINT1FCLR	3	w	**External EXTINT1 Falling Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
EXTINT2RCLR	4	w	**External EXTINT2 Rising Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
EXTINT2FCLR	5	w	**External EXTINT2 Falling Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
EXTINT3RCLR	6	w	**External EXTINT3 Rising Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared

Field	Bits	Type	Description
EXTINT3FCLR	7	w	**External EXTINT3 Falling Edge Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
RES	31:8	r	**Reserved**

External Interrupt Status Set Register

EXTISS

External Interrupt Status Set Register **(0074_H)** **RESET_TYPE_3 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								EXTINT3FSET	EXTINT3RSET	EXTINT2FSET	EXTINT2RSET	EXTINT1FSET	EXTINT1RSET	EXTINT0FSET	EXTINT0RSET
r								w	w	w	w	w	w	w	w

Field	Bits	Type	Description
EXTINT0RSET	0	w	**External EXTINT0 Rising Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
EXTINT0FSET	1	w	**External EXTINT0 Falling Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
EXTINT1RSET	2	w	**External EXTINT1Rising Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
EXTINT1FSET	3	w	**External EXTINT1Falling Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
EXTINT2RSET	4	w	**External EXTINT2 Rising Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET

Field	Bits	Type	Description
EXTINT2FSET	5	w	**External EXTINT2 Falling Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
EXTINT3RSET	6	w	**External EXTINT3 Rising Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
EXTINT3FSET	7	w	**External EXTINT3 Falling Edge Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
RES	31:8	r	**Reserved**

EXT Interrupt Configuration Register

EXTCON
EXT Interrupt Configuration Register **(0078_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTINT3INSEL		**EXTINT2INSEL**		**EXTINT1INSEL**		**EXTINT0INSEL**		**EXTINT3IEV**		**EXTINT2IEV**		**EXTINT1IEV**		**EXTINT0IEV**	
rw		rw		rw		rw		rw		rw		rw		rw	

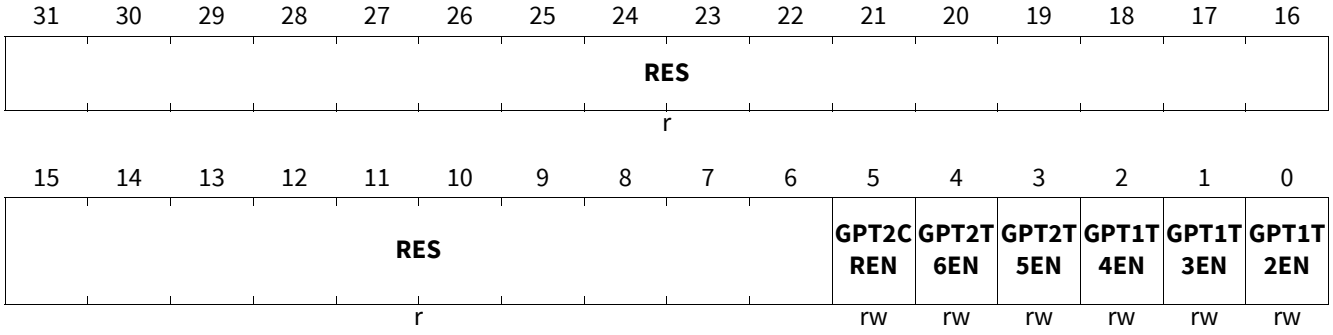
Field	Bits	Type	Description
EXTINT0IEV	1:0	rw	**External EXTINT0 Interrupt Event Select** 00_B **Disable**, EXT Int disabled 01_B **Enable**, EXT Int on rising edge 10_B **FallingEdge**, EXT Int on falling edge 11_B **BothEdges**, EXT Int on both edges
EXTINT1IEV	3:2	rw	**External EXTINT1 Interrupt Event Select** 00_B **Disable**, EXT Int disabled 01_B **Enable**, EXT Int on rising edge 10_B **FallingEdge**, EXT Int on falling edge 11_B **BothEdges**, EXT Int on both edges
EXTINT2IEV	5:4	rw	**External EXTINT2 Interrupt Event Select** 00_B **Disable**, EXT Int disabled 01_B **Enable**, EXT Int on rising edge 10_B **FallingEdge**, EXT Int on falling edge 11_B **BothEdges**, EXT Int on both edges

Field	Bits	Type	Description
EXTINT3IEV	7:6	rw	**External EXTINT3Interrupt Event Select** 00_B **Disable**, EXT Int disabled 01_B **Enable**, EXT Int on rising edge 10_B **FallingEdge**, EXT Int on falling edge 11_B **BothEdges**, EXT Int on both edges
EXTINT0INSEL	9:8	rw	**External EXTINT0 Input Select** 00_B **EXTINT0A**, selected … 11_B **EXTINT0D**, selected
EXTINT1INSEL	11:10	rw	**External EXTINT1 Input Select** 00_B **EXTINT1A**, selected … 11_B **EXTINT1D**, selected
EXTINT2INSEL	13:12	rw	**External EXTINT2 Input Select** 00_B **EXTINT2A**, selected … 11_B **EXTINT2D**, selected
EXTINT3INSEL	15:14	rw	**External EXTINT3 Input Select** 00_B **EXTINT3A**, selected … 11_B **EXTINT3D**, selected
RES	31:16	r	**Reserved**

General Purpose Timer 12 Interrupt Enable Register

GPTIEN

General Purpose Timer 12 Interrupt Enable Register($007C_H$) **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
GPT1T2EN	0	rw	**General Purpose Timer 1 T2 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
GPT1T3EN	1	rw	**General Purpose Timer 1 T3 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled

Field	Bits	Type	Description
GPT1T4EN	2	rw	**General Purpose Timer 1 T4 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
GPT2T5EN	3	rw	**General Purpose Timer 2 T5 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
GPT2T6EN	4	rw	**General Purpose Timer 2 T6 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
GPT2CREN	5	rw	**General Purpose Timer 2 CR Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
RES	31:6	r	**Reserved**

General Purpose Timer 12 Interrupt Status Register

GPTIS

General Purpose Timer 12 Interrupt Status Register(0080_H) **RESET_TYPE_4 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										GPT2CR	GPT2T6	GPT2T5	GPT1T4	GPT1T3	GPT1T2
r										rhxre	rhxre	rhxre	rhxre	rhxre	rhxre

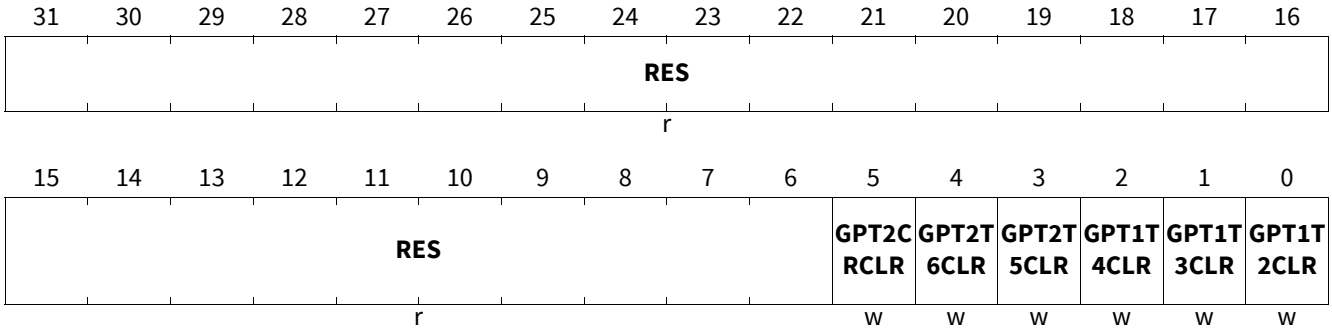
Field	Bits	Type	Description
GPT1T2	0	rhxre	**General Purpose Timer 1 T2 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
GPT1T3	1	rhxre	**General Purpose Timer 1 T3 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
GPT1T4	2	rhxre	**General Purpose Timer 1 T4 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred

Field	Bits	Type	Description
GPT2T5	3	rhxre	**General Purpose Timer 2 T5 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
GPT2T6	4	rhxre	**General Purpose Timer 2 T6 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
GPT2CR	5	rhxre	**General Purpose Timer 2 CR Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
RES	31:6	r	**Reserved**

General Purpose Timer 12 Interrupt Status Clear Register

GPTISC

General Purpose Timer 12 Interrupt Status Clear Register(0084_H) **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
GPT1T2CLR	0	w	**General Purpose Timer 1 T2 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
GPT1T3CLR	1	w	**General Purpose Timer 1 T3 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
GPT1T4CLR	2	w	**General Purpose Timer 1 T4 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared

Field	Bits	Type	Description
GPT2T5CLR	3	w	**General Purpose Timer 2 T5 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
GPT2T6CLR	4	w	**General Purpose Timer 2 T6 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
GPT2CRCLR	5	w	**General Purpose Timer 2 CR Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
RES	31:6	r	**Reserved**

General Purpose Timer 12 Interrupt Status Set Register

GPTISS

General Purpose Timer 12 Interrupt Status Set Register(0088_H) **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										GPT2CRSET	GPT2T6SET	GPT2T5SET	GPT1T4SET	GPT1T3SET	GPT1T2SET
r										w	w	w	w	w	w

Field	Bits	Type	Description
GPT1T2SET	0	w	**General Purpose Timer 1 T2 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
GPT1T3SET	1	w	**General Purpose Timer 1 T3 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
GPT1T4SET	2	w	**General Purpose Timer 1 T4 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
GPT2T5SET	3	w	**General Purpose Timer 2 T5 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET

Field	Bits	Type	Description
GPT2T6SET	4	w	**General Purpose Timer 2 T6 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
GPT2CRSET	5	w	**General Purpose Timer 2 CR Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
RES	31:6	r	**Reserved**

DMA Interrupt Enable Register

DMAIEN
DMA Interrupt Enable Register **($008C_H$)** **RESET_TYPE_3 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES							DMATRERREN	DMACH7EN	DMACH6EN	DMACH5EN	DMACH4EN	DMACH3EN	DMACH2EN	DMACH1EN	DMACH0EN
r							rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
DMACH0EN	0	rw	**DMA Channel 0 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
DMACH1EN	1	rw	**DMA Channel 1 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
DMACH2EN	2	rw	**DMA Channel 2 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
DMACH3EN	3	rw	**DMA Channel 3 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
DMACH4EN	4	rw	**DMA Channel 4 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
DMACH5EN	5	rw	**DMA Channel 5 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled

Field	Bits	Type	Description
DMACH6EN	6	rw	**DMA Channel 6 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
DMACH7EN	7	rw	**DMA Channel 7 Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
DMATRERREN	8	rw	**DMA Error Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
RES	31:9	r	**Reserved**

DMA Interrupt Status Register

DMAIS

DMA Interrupt Status Register **(0090_H)** **RESET_TYPE_4 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
r								rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre

Field	Bits	Type	Description
DMACH0	0	rhxre	**DMA Channel 0 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
DMACH1	1	rhxre	**DMA Channel 1 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
DMACH2	2	rhxre	**DMA Channel 2 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred

Field	Bits	Type	Description
DMACH3	3	rhxre	**DMA Channel 3 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
DMACH4	4	rhxre	**DMA Channel 4 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
DMACH5	5	rhxre	**DMA Channel 5 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
DMACH6	6	rhxre	**DMA Channel 6 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
DMACH7	7	rhxre	**DMA Channel 7 Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
RES	31:8	r	**Reserved**

DAM Interrupt Status Clear Register

DMAISC

DAM Interrupt Status Clear Register **(0094_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								DMACH7CLR	DMACH6CLR	DMACH5CLR	DMACH4CLR	DMACH3CLR	DMACH2CLR	DMACH1CLR	DMACH0CLR
r								w	w	w	w	w	w	w	w

Field	Bits	Type	Description
DMACH0CLR	0	w	**DMA Channel 0 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
DMACH1CLR	1	w	**DMA Channel 1 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
DMACH2CLR	2	w	**DMA Channel 2 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
DMACH3CLR	3	w	**DMA Channel 3 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
DMACH4CLR	4	w	**DMA Channel 4 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
DMACH5CLR	5	w	**DMA Channel 5 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
DMACH6CLR	6	w	**DMA Channel 6 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
DMACH7CLR	7	w	**DMA Channel 7 Interrupt Status Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
RES	31:8	r	**Reserved**

DMA Interrupt Status Set Register

DMAISS

DMA Interrupt Status Set Register **(0098_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								DMACH7SET	DMACH6SET	DMACH5SET	DMACH4SET	DMACH3SET	DMACH2SET	DMACH1SET	DMACH0SET
r								w	w	w	w	w	w	w	w

Field	Bits	Type	Description
DMACH0SET	0	w	**DMA Channel 0 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
DMACH1SET	1	w	**DMA Channel 1 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
DMACH2SET	2	w	**DMA Channel 2 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
DMACH3SET	3	w	**DMA Channel 3 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
DMACH4SET	4	w	**DMA Channel 4 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
DMACH5SET	5	w	**DMA Channel 5 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
DMACH6SET	6	w	**DMA Channel 6 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
DMACH7SET	7	w	**DMA Channel 7 Interrupt Status Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET

Field	Bits	Type	Description
RES	31:8	r	**Reserved**

DMA Channel Select Register CCU7

DMAP_CCU7
DMA Channel Select Register CCU7 ($009C_H$) RESET_TYPE_3 Value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		**CHE**		**T16PM**		**T16CM**		**T15PM**		**T15CM**		**T14PM**		**T14CM**	
r		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T13ZM		**T13PM**		**T13CM**		**T12ZM**		**T12PM**		**T12CM72**		**T12CM71**		**T12CM70**	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
T12CM70	1:0	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH0**, Channel 0 selected 10_B **CH2**, Channel 2 selected 11_B **CH4**, Channel 4 selected
T12CM71	3:2	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH1**, Channel 1 selected 10_B **CH3**, Channel 3 selected 11_B **CH5**, Channel 5 selected
T12CM72	5:4	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH0**, Channel 0 selected 10_B **CH2**, Channel 2 selected 11_B **CH4**, Channel 4 selected
T12PM	7:6	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH1**, Channel 1 selected 10_B **CH3**, Channel 3 selected 11_B **CH5**, Channel 5 selected
T12ZM	9:8	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH0**, Channel 0 selected 10_B **CH2**, Channel 2 selected 11_B **CH4**, Channel 4 selected

Field	Bits	Type	Description
T13CM	11:10	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH1**, Channel 1 selected 10_B **CH3**, Channel 3 selected 11_B **CH5**, Channel 5 selected
T13PM	13:12	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH0**, Channel 0 selected 10_B **CH2**, Channel 2 selected 11_B **CH4**, Channel 4 selected
T13ZM	15:14	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH1**, Channel 1 selected 10_B **CH3**, Channel 3 selected 11_B **CH5**, Channel 5 selected
T14CM	17:16	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH0**, Channel 0 selected 10_B **CH2**, Channel 2 selected 11_B **CH4**, Channel 4 selected
T14PM	19:18	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH1**, Channel 1 selected 10_B **CH3**, Channel 3 selected 11_B **CH5**, Channel 5 selected
T15CM	21:20	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH0**, Channel 0 selected 10_B **CH2**, Channel 2 selected 11_B **CH4**, Channel 4 selected
T15PM	23:22	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH1**, Channel 1 selected 10_B **CH3**, Channel 3 selected 11_B **CH5**, Channel 5 selected
T16CM	25:24	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH0**, Channel 0 selected 10_B **CH2**, Channel 2 selected 11_B **CH4**, Channel 4 selected
T16PM	27:26	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH1**, Channel 1 selected 10_B **CH3**, Channel 3 selected 11_B **CH5**, Channel 5 selected

Field	Bits	Type	Description
CHE	29:28	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH0**, Channel 0 selected 10_B **CH2**, Channel 2 selected 11_B **CH4**, Channel 4 selected
RES	31:30	r	**Reserved**

DMA Channel Select Register ADCs

DMAP_ADC
DMA Channel Select Register ADCs **($00A0_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES				**SDADC_RES1**		**SDADC_RES0**		**ADC1_CMPHI**		**ADC1_CMPLO**		**ADC1_SQ1**		**ADC1_SQ0**	
r				rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADC1_RES7		**ADC1_RES6**		**ADC1_RES5**		**ADC1_RES4**		**ADC1_RES3**		**ADC1_RES2**		**ADC1_RES1**		**ADC1_RES0**	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
ADC1_RES0	1:0	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH2**, Channel 2 selected 10_B **CH4**, Channel 4 selected 11_B **CH6**, Channel 6 selected
ADC1_RES1	3:2	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH3**, Channel 3 selected 10_B **CH5**, Channel 5 selected 11_B **CH7**, Channel 7 selected
ADC1_RES2	5:4	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH2**, Channel 2 selected 10_B **CH4**, Channel 4 selected 11_B **CH6**, Channel 6 selected
ADC1_RES3	7:6	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH3**, Channel 3 selected 10_B **CH5**, Channel 5 selected 11_B **CH7**, Channel 7 selected

Field	Bits	Type	Description
ADC1_RES4	9:8	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH2**, Channel 2 selected 10_B **CH4**, Channel 4 selected 11_B **CH6**, Channel 6 selected
ADC1_RES5	11:10	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH3**, Channel 3 selected 10_B **CH5**, Channel 5 selected 11_B **CH7**, Channel 7 selected
ADC1_RES6	13:12	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH2**, Channel 2 selected 10_B **CH4**, Channel 4 selected 11_B **CH6**, Channel 6 selected
ADC1_RES7	15:14	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH3**, Channel 3 selected 10_B **CH5**, Channel 5 selected 11_B **CH7**, Channel 7 selected
ADC1_SQ0	17:16	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH2**, Channel 2 selected 10_B **CH4**, Channel 4 selected 11_B **CH6**, Channel 6 selected
ADC1_SQ1	19:18	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH3**, Channel 3 selected 10_B **CH5**, Channel 5 selected 11_B **CH7**, Channel 7 selected
ADC1_CMPLO	21:20	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH2**, Channel 2 selected 10_B **CH4**, Channel 4 selected 11_B **CH6**, Channel 6 selected
ADC1_CMPHI	23:22	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH3**, Channel 3 selected 10_B **CH5**, Channel 5 selected 11_B **CH7**, Channel 7 selected
SDADC_RES0	25:24	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH2**, Channel 2 selected 10_B **CH4**, Channel 4 selected 11_B **CH6**, Channel 6 selected

Field	Bits	Type	Description
SDADC_RES1	27:26	rw	**DMA Channel Request Select** 00_B **Disabled**, No Channel selected 01_B **CH3**, Channel 3 selected 10_B **CH5**, Channel 5 selected 11_B **CH7**, Channel 7 selected
RES	31:28	r	**Reserved**

DMA Channel Select Register Timer

DMAP_TIM
DMA Channel Select Register Timer **($00A4_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		GPT12 CR	GPT12 T6	GPT12 T5	GPT12 T4	GPT12 T3	GPT12 T2	RES						T21OV	T2OV
r		rw	rw	rw	rw	rw	rw	r						rw	rw

Field	Bits	Type	Description
T2OV	0	rw	**DMA Channel Request Select** 0_B **Disabled**, No Channel selected 1_B **CH6**, Channel 6 selected
T21OV	1	rw	**DMA Channel Request Select** 0_B **Disabled**, No Channel selected 1_B **CH7**, Channel 7 selected
RES	7:2, 31:14	r	**Reserved**
GPT12T2	8	rw	**DMA Channel Request Select** 0_B **Disabled**, No Channel selected 1_B **CH0**, Channel 0 selected
GPT12T3	9	rw	**DMA Channel Request Select** 0_B **Disabled**, No Channel selected 1_B **CH1**, Channel 1 selected
GPT12T4	10	rw	**DMA Channel Request Select** 0_B **Disabled**, No Channel selected 1_B **CH2**, Channel 2 selected
GPT12T5	11	rw	**DMA Channel Request Select** 0_B **Disabled**, No Channel selected 1_B **CH3**, Channel 3 selected
GPT12T6	12	rw	**DMA Channel Request Select** 0_B **Disabled**, No Channel selected 1_B **CH4**, Channel 4 selected

Register description SCU

Field	Bits	Type	Description
GPT12CR	13	rw	**DMA Channel Request Select** 0_B **Disabled**, No Channel selected 1_B **CH5**, Channel 5 selected

DMA Channel Select Register COM Modules

DMAP_COM

DMA Channel Select Register COM Modules (00A8$_H$) **RESET_TYPE_3 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES													CAN_IR1	RES	CAN_IR0
r													rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UART1_TI		UART1_RI		UART0_TI		UART0_RI		SSC1_TIR		SSC1_RIR		SSC0_TIR		SSC0_RIR	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SSC0_RIR	1:0	rw	**DMA Channel Request Select** 00_B **Disabled0**, No Channel selected 01_B **CH1**, Channel 1 selected 10_B **CH3**, Channel 3 selected 11_B **Disabled3**, No Channel selected
SSC0_TIR	3:2	rw	**DMA Channel Request Select** 00_B **Disabled0**, No Channel selected 01_B **CH0**, Channel 0 selected 10_B **CH2**, Channel 2 selected 11_B **Disabled3**, No Channel selected
SSC1_RIR	5:4	rw	**DMA Channel Request Select** 00_B **Disabled0**, No Channel selected 01_B **CH5**, Channel 5 selected 10_B **CH7**, Channel 7 selected 11_B **Disabled3**, No Channel selected
SSC1_TIR	7:6	rw	**DMA Channel Request Select** 00_B **Disabled0**, No Channel selected 01_B **CH4**, Channel 4 selected 10_B **CH6**, Channel 6 selected 11_B **Disabled3**, No Channel selected
UART0_RI	9:8	rw	**DMA Channel Request Select** 00_B **Disabled0**, No Channel selected 01_B **CH0**, Channel 0 selected 10_B **CH2**, Channel 2 selected 11_B **Disabled3**, No Channel selected

Field	Bits	Type	Description
UART0_TI	11:10	rw	**DMA Channel Request Select** 00_B **Disabled0**, No Channel selected 01_B **CH1**, Channel 1 selected 10_B **CH3**, Channel 3 selected 11_B **Disabled3**, No Channel selected
UART1_RI	13:12	rw	**DMA Channel Request Select** 00_B **Disabled0**, No Channel selected 01_B **CH4**, Channel 4 selected 10_B **CH6**, Channel 6 selected 11_B **Disabled3**, No Channel selected
UART1_TI	15:14	rw	**DMA Channel Request Select** 00_B **Disabled0**, No Channel selected 01_B **CH5**, Channel 5 selected 10_B **CH7**, Channel 7 selected 11_B **Disabled3**, No Channel selected
CAN_IR0	16	rw	**DMA Channel Request Select** 0_B **Disabled**, No Channel selected 1_B **CH0**, Channel 0 selected
RES	17, 31:19	r	**Reserved**
CAN_IR1	18	rw	**DMA Channel Request Select** 0_B **Disabled**, No Channel selected 1_B **CH1**, Channel 1 selected

Power Mode Control Register

PMCON0
Power Mode Control Register **($00AC_H$)** **RESET_TYPE_4 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES													STOP	SLEEP	RES
r													w	w	r

Field	Bits	Type	Description
RES	0, 31:3	r	**Reserved** Always read as 0
SLEEP	1	w	**Sleep Mode Enable** Setting this bit will cause the device to go into Sleep Mode. Reset by HW. 0_B **ACTIVE**, Active Mode 1_B **SLEEP**, SLEEP Mode Entry

Field	Bits	Type	Description
STOP	2	w	**STOP Mode Enable** Setting this bit will cause the device to go into STOP Mode. Reset by HW. 0_B **ACTIVE**, Active Mode 1_B **STOP**, STOP Mode Entry

Peripheral Management Control Register

PMCON
Peripheral Management Control Register (00B0$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

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

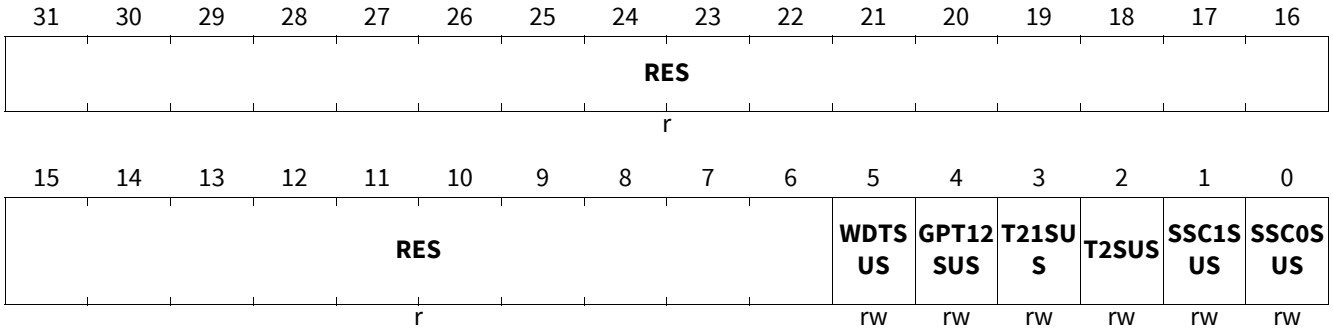
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES											GPT12_DIS	T21_DIS	T2_DIS	SSC1_DIS	SSC0_DIS
r											rw	rw	rw	rw	rw

Field	Bits	Type	Description
SSC0_DIS	0	rw	**SSC0 Module Disable** 0_B **Enable**, SSC0 enabled (normal operation mode) 1_B **Disable**, SSC0 disabled
SSC1_DIS	1	rw	**SSC1 Module Disable** 0_B **Enable**, SSC1 enabled (normal operation mode) 1_B **Disable**, SSC1 disabled
T2_DIS	2	rw	**Timer 20 Module Disable** 0_B **Enable**, T2 enabled (normal operation mode) 1_B **Disable**, T2 disabled
T21_DIS	3	rw	**Timer 21 Module Disable** 0_B **Enable**, T21 enable (normal operation mode) 1_B **Disable**, T21 disabled
GPT12_DIS	4	rw	**General Purpose Timer 12 Module Disable** 0_B **Enable**, GPT12 enabled (normal operation mode) 1_B **Disable**, GPT12 disabled
RES	31:5	r	**Reserved** Always read as 0

Register description SCU

Module Suspend Control Register

SUSCTR
Module Suspend Control Register **($00B4_H$)** **RESET_TYPE_3 Value: $0000\ 0020_H$**



Field	Bits	Type	Description
SSC0SUS	0	rw	**SSC0 Suspend** 0_B **CONT**, module continues on DBG request 1_B **SUS**, module suspended on DBG request
SSC1SUS	1	rw	**SSC1 Suspend** 0_B **CONT**, module continues on DBG request 1_B **SUS**, module suspended on DBG request
T2SUS	2	rw	**Timer 20 Suspend** 0_B **CONT**, module continues on DBG request 1_B **SUS**, module suspended on DBG request
T21SUS	3	rw	**Timer 21 Suspend** 0_B **CONT**, module continues on DBG request 1_B **SUS**, module suspended on DBG request
GPT12SUS	4	rw	**Gerneral Purpose Timer 12 Suspend** 0_B **CONT**, module continues on DBG request 1_B **SUS**, module suspended on DBG request
WDTSUS	5	rw	**SCU Watchdog Timer Suspend** 0_B **CONT**, module continues on DBG request 1_B **SUS**, module suspended on DBG request
RES	31:6	r	**Reserved** Always read as 0

Register description SCU

PCU Control Register

PCU_CTRL
PCU Control Register **($00B8_H$)** **RESET_TYPE_4 Value: $0000\ 8001_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XTALWDG_SD_DIS	OTWARN_SD_DIS	RES						RES							OT_SLEEP_EN
rw	rw	r						r							rw

Field	Bits	Type	Description
OT_SLEEP_EN	0	rw	**System Overtemperature Sleep Mode Enable** If enabled the PMU will enter Sleep Mode if a system overtemperature is detected. 0_B **SleepDisable**, No Sleep Mode in case of SysOT 1_B **SleepEnable**, Sleep Mode in case of SysOT
RES	7:1, 13:8, 23:16, 25, 31:26	r	**Reserved** Always read as 0
OTWARN_SD_DIS	14	rw	**Overtemperature Warning Peripherals Shutdown Disable** 0_B **Enable**, Shutdown of selected peripherals enabled 1_B **Disable**, Shutdown of selected peripherals disabled
XTALWDG_SD_DIS	15	rw	**XTAL Watchdog Peripherals Shutdown Disable** 0_B **Enable**, Shutdown of selected peripherals enabled 1_B **Disable**, Shutdown of selected peripherals disabled
BDRV_SD_EN	24	rw	**BDRV Shutdown Enable** Setting this bit will cause BDRV to be switched off in case a peripheral shutdown event occurs and shutdown source is not disable. 0_B **Disable**, Shutdown disabled 1_B **Enable**, Shutdown enabled

DMA Control Register

DMACTRL
DMA Control Register **($00E0_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

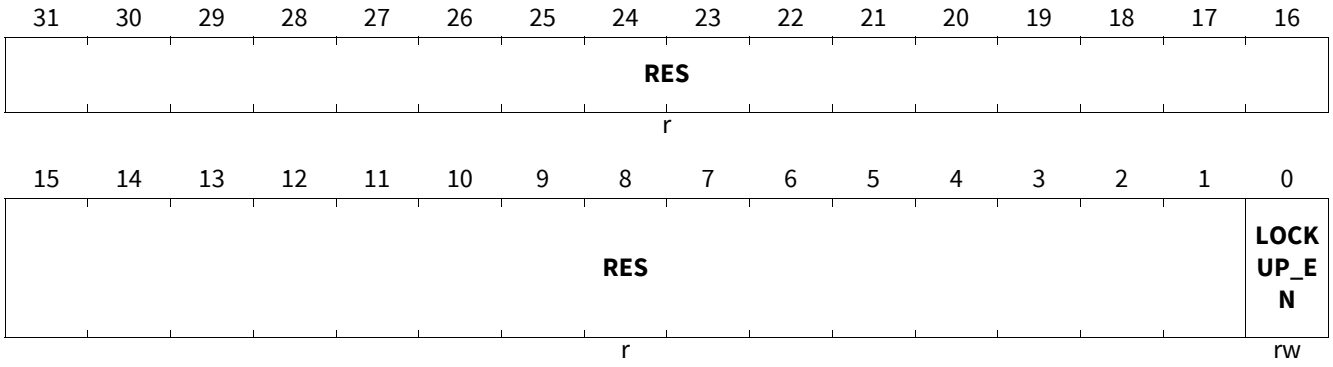
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMAREQINTEN_CH7	DMAREQINTEN_CH6	DMAREQINTEN_CH5	DMAREQINTEN_CH4	DMAREQINTEN_CH3	DMAREQINTEN_CH2	DMAREQINTEN_CH1	DMAREQINTEN_CH0	DEMEN_CH7	DEMEN_CH6	DEMEN_CH5	DEMEN_CH4	DEMEN_CH3	DEMEN_CH2	DEMEN_CH1	DEMEN_CH0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
DEMEN_CH0	0	rw	**DMA Endless Mode Enable Channel 0** 0_B **DISABLED**, Endless mode disabled 1_B **ENABLED**, Endless mode enabled
DEMEN_CH1	1	rw	**DMA Endless Mode Enable Channel 1** 0_B **DISABLED**, Endless mode disabled 1_B **ENABLED**, Endless mode enabled
DEMEN_CH2	2	rw	**DMA Endless Mode Enable Channel 2** 0_B **DISABLED**, Endless mode disabled 1_B **ENABLED**, Endless mode enabled
DEMEN_CH3	3	rw	**DMA Endless Mode Enable Channel 3** 0_B **DISABLED**, Endless mode disabled 1_B **ENABLED**, Endless mode enabled
DEMEN_CH4	4	rw	**DMA Endless Mode Enable Channel 4** 0_B **DISABLED**, Endless mode disabled 1_B **ENABLED**, Endless mode enabled
DEMEN_CH5	5	rw	**DMA Endless Mode Enable Channel 5** 0_B **DISABLED**, Endless mode disabled 1_B **ENABLED**, Endless mode enabled
DEMEN_CH6	6	rw	**DMA Endless Mode Enable Channel 6** 0_B **DISABLED**, Endless mode disabled 1_B **ENABLED**, Endless mode enabled
DEMEN_CH7	7	rw	**DMA Endless Mode Enable Channel 7** 0_B **DISABLED**, Endless mode disabled 1_B **ENABLED**, Endless mode enabled
DMAREQINTEN_CH0	8	rw	**DMA Pending Request Interrupt Enable Channel 0** 0_B **DISABLED**, Pending Request Interrupt disabled 1_B **ENABLED**, Pending Request Interrupt enabled
DMAREQINTEN_CH1	9	rw	**DMA Pending Request Interrupt Enable Channel 1** 0_B **DISABLED**, Pending Request Interrupt disabled 1_B **ENABLED**, Pending Request Interrupt enabled

Field	Bits	Type	Description
DMAREQINTEN_CH2	10	rw	**DMA Pending Request Interrupt Enable Channel 2** 0_B **DISABLED**, Pending Request Interrupt disabled 1_B **ENABLED**, Pending Request Interrupt enabled
DMAREQINTEN_CH3	11	rw	**DMA Pending Request Interrupt Enable Channel 3** 0_B **DISABLED**, Pending Request Interrupt disabled 1_B **ENABLED**, Pending Request Interrupt enabled
DMAREQINTEN_CH4	12	rw	**DMA Pending Request Interrupt Enable Channel 4** 0_B **DISABLED**, Pending Request Interrupt disabled 1_B **ENABLED**, Pending Request Interrupt enabled
DMAREQINTEN_CH5	13	rw	**DMA Pending Request Interrupt Enable Channel 5** 0_B **DISABLED**, Pending Request Interrupt disabled 1_B **ENABLED**, Pending Request Interrupt enabled
DMAREQINTEN_CH6	14	rw	**DMA Pending Request Interrupt Enable Channel 6** 0_B **DISABLED**, Pending Request Interrupt disabled 1_B **ENABLED**, Pending Request Interrupt enabled
DMAREQINTEN_CH7	15	rw	**DMA Pending Request Interrupt Enable Channel 7** 0_B **DISABLED**, Pending Request Interrupt disabled 1_B **ENABLED**, Pending Request Interrupt enabled
RES	31:16	r	**Reserved**

CPU LOCKUP Config Register

LOCKUPCFG
CPU LOCKUP Config Register **($00E8_H$)** **Reset Value: Table 85**



Field	Bits	Type	Description
LOCKUP_EN	0	rw	**CPU LOCKUP Reset Enable** 0_B **Disable**, CPU LOCKUP Reset Disabled 1_B **Enable**, CPU LOCKUP Reset Enabled
RES	31:1	r	**Reserved**

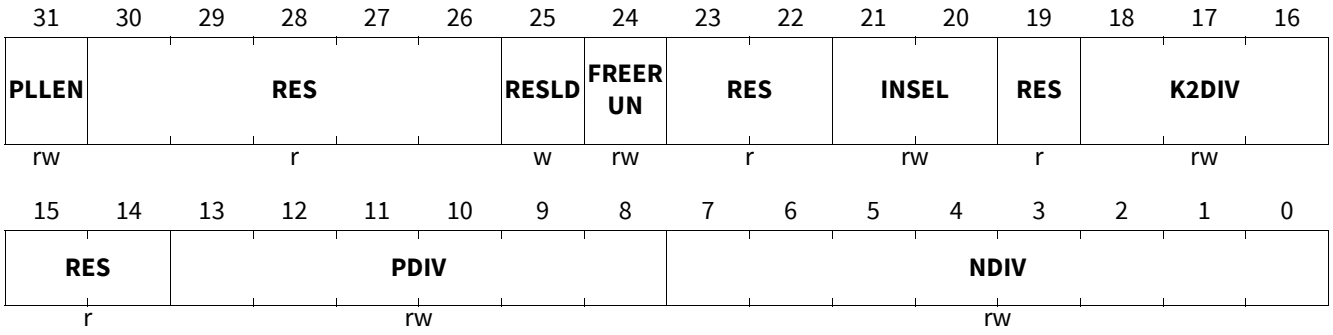
Table 85 Reset Values of LOCKUPCFG

Reset Type	Reset Value	Note
RESET_TYPE_4	$0000\ 0000_H$	RESET_TYPE_4
FW_CTRL	$----\ ----_H$	Set by startup FW

6.9.3 PLLREG Registers

PLL0 Control Register

CON0
PLL0 Control Register **(0000_H)** **RESET_TYPE_6 Value: 0020 0578$_H$**

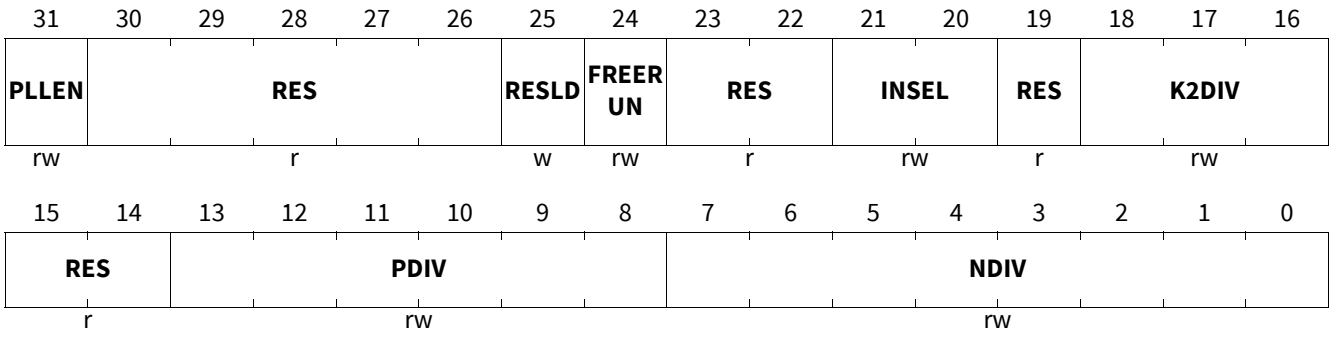


Field	Bits	Type	Description
NDIV	7:0	rw	**N-Divider Setting** Values from 39 to 200 are used, smaller or higher values are saturated. Note: Shall only be written if PLLEN = 0, otherwise new value will not be used.
PDIV	13:8	rw	**P-Divider Setting** Values from 4 to 50 are used, smaller or higher values are saturated.
RES	15:14, 19, 23:22, 30:26	r	**Reserved**
K2DIV	18:16	rw	**K2-Divider Setting** 000_B **div2**, K2 = 2 (default) (decoded value 0x01) 001_B **div3**, K2 = 3 (decoded value 0x02) 010_B **div4**, K2 = 4 (decoded value 0x03) 011_B **div5**, K2 = 5 (decoded value 0x04) 100_B **div6**, K2 = 6 (decoded value 0x05) 101_B **div7**, K2 = 7 (decoded value 0x06) 110_B **div8**, K2 = 8 (decoded value 0x07) 111_B **div9**, K2 = 9 (decoded value 0x08)
INSEL	21:20	rw	**PLL Reference Clock Select** Synchronous switching of reference to hp_clk is not possible if XDP = 1 or no XTAL_CLK is available. Use 1x setting only if no XTAL_CLK is available. 00_B **hp_sync**, HP_CLK selected synchronously as reference clock (use if XTAL_CLK is available) 01_B **xtal_sync**, XTAL_CLK selected as reference clock 10_B **hp_async**, HP_CLK selected asynchronously as reference clock (use if no XTAL_CLK is available) 11_B **hp_async**, HP_CLK selected asynchronously as reference clock (use if no XTAL_CLK is available)

Field	Bits	Type	Description
FREERUN	24	rw	**Free-running Mode Select** 0_B **Connect**, Reference Clock is connected to the PLL 1_B **Disconnect**, Reference Clock is disconnected from PLL (Free Running)
RESLD	25	w	**Lock Detection Reset** 0_B **Noreset**, No effect. 1_B **reset**, Reset lock flag and restart lock detection.
PLLEN	31	rw	**PLL Enable** 0_B **Disable**, PLL disabled 1_B **Enable**, PLL enabled

PLL1 Control Register

CON1
PLL1 Control Register **(0004_H)** **RESET_TYPE_6 Value: $0020\ 0550_H$**



Field	Bits	Type	Description
NDIV	7:0	rw	**N-Divider Setting** Values from 39 to 200 are used, smaller or higher values are saturated. Note: Shall only be written if PLLEN = 0, otherwise new value will not be used.
PDIV	13:8	rw	**P-Divider Setting** Values from 4 to 50 are used, smaller or higher values are saturated.
RES	15:14, 19, 23:22, 30:26	r	**Reserved**
K2DIV	18:16	rw	**K2-Divider Setting** 000_B **div2**, K2 = 2 (default) (decoded value 0x01) 001_B **div3**, K2 = 3 (decoded value 0x02) 010_B **div4**, K2 = 4 (decoded value 0x03) 011_B **div5**, K2 = 5 (decoded value 0x04) 100_B **div6**, K2 = 6 (decoded value 0x05) 101_B **div7**, K2 = 7 (decoded value 0x06) 110_B **div8**, K2 = 8 (decoded value 0x07) 111_B **div9**, K2 = 9 (decoded value 0x08)

Field	Bits	Type	Description
INSEL	21:20	rw	**PLL Reference Clock Select** Synchronous switching of reference to hp_clk is not possible if XDP = 1 or no XTAL_CLK is available. Use 1x setting only if no XTAL_CLK is available. 00_B **hp_sync**, HP_CLK selected synchronously as reference clock (use if XTAL_CLK is available) 01_B **xtal_sync**, XTAL_CLK selected as reference clock 10_B **hp_async**, HP_CLK selected asynchronously as reference clock (use if no XTAL_CLK is available) 11_B **hp_async**, HP_CLK selected asynchronously as reference clock (use if no XTAL_CLK is available)
FREERUN	24	rw	**Freerunning Mode Select** 0_B **Connect**, Reference Clock is connected to the PLL 1_B **Disconnect**, Reference Clock is disconnected from PLL (Free Running)
RESLD	25	w	**Lock Detection Reset** 0_B **Noreset**, No effect. 1_B **reset**, Reset lock flag and restart lock detection.
PLLEN	31	rw	**PLL Enable** 0_B **Disable**, PLL disabled 1_B **Enable**, PLL enabled

PLL NDIV Spread Control Register

SPCTR

PLL NDIV Spread Control Register **(0008_H)** **RESET_TYPE_6 Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SPUPVAL1										**SPRANGE1**		**RES**			**SPEN1**
rwpht										rwpht		r			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPUPVAL0										**SPRANGE0**		**RES**			**SPEN0**
rwpht										rwpht		r			rw

Field	Bits	Type	Description
SPEN0	0	rw	**NDIV PLL0 Spread Enable** 0_B **Disabled**, NDIV spread feature disabled 1_B **Enabled**, NDIV spread feature enabled
RES	3:1, 19:17	r	**Reserved**

Field	Bits	Type	Description
SPRANGE0	5:4	rwpht	**PLL0 NDIV Spread Range Value** This bit field defines the NDIV Value Spread Range for PLL0 NDIV Value will be changed after each update interval in single steps in the defined range Note: NDIV min/max values of 39/200 will not be exceeded. Note: This bit field can only be written if SPEN0 = 0 00_B **SPREAD1**, NDIV Spread in Range of +/- 1 Values 01_B **SPREAD2**, NDIV Spread in Range of +/- 2 Values 10_B **SPREAD3**, NDIV Spread in Range of +/- 3 Values 11_B **SPREAD4**, NDIV Spread in Range of +/- 4 Values
SPUPVAL0	15:6	rwpht	**PLL0 NDIV Spread Update Interval** This bit field defines the NDIV value update interval in PLL0 clock cycles if SPEN0 = 1. Note: This bit field can only be written if SPEN0 = 0 000_H NDIV value updated each PLL cycle ... $3FF_H$ NDIV value updated each 1024 PLL cycles
SPEN1	16	rw	**NDIV PLL1 Spread Enable** 0_B **Disabled**, NDIV spread feature disabled 1_B **Enabled**, NDIV spread feature enabled
SPRANGE1	21:20	rwpht	**PLL1 NDIV Spread Range Value** This bit field defines the NDIV Value Spread Range for PLL1 NDIV Value will be changed after each update interval in single steps in the defined range Note: NDIV min/max values of 39/200 will not be exceeded. Note: This bit field can only be written if SPEN1 = 0 00_B **SPREAD1**, NDIV Spread in Range of +/- 1 Values 01_B **SPREAD2**, NDIV Spread in Range of +/- 2 Values 10_B **SPREAD3**, NDIV Spread in Range of +/- 3 Values 11_B **SPREAD4**, NDIV Spread in Range of +/- 4 Values
SPUPVAL1	31:22	rwpht	**PLL1 NDIV Spread Update Interval** This bit field defines the NDIV value update interval in PLL1 clock cycles if SPEN1 = 1. Note: This bit field can only be written if SPEN1 = 0 000_H NDIV value updated each PLL cycle ... $3FF_H$ NDIV value updated each 1024 PLL cycles

PLL Status Register

STAT
PLL Status Register **($000C_H$)** **RESET_TYPE_6 Value: $0000\ 0030_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES														OSCSEL_STAT1	OSCSEL_STAT0
r														r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										PLL1_LOL_STS	PLL0_LOL_STS	RES		LCK1	LCK0
r										rhxr	rhxr	r		r	r

Field	Bits	Type	Description
LCK0	0	r	**Lock Status PLL 0** This bit shows the current Lock status of PLL0. The Lock status must be locked as pre-condition to clear PLL0_LOL_STS. 0_B **NotLocked**, PLL0 is not locked 1_B **Locked**, PLL0 is locked
LCK1	1	r	**Lock Status PLL1** This bit shows the current Lock status of PLL1. The Lock status must be locked as pre-condition to clear PLL1_LOL_STS. 0_B **NotLocked**, PLL1 is not locked 1_B **Locked**, PLL1 is locked
RES	3:2, 15:6, 31:18	r	**Reserved**
PLL0_LOL_STS	4	rhxr	**PLL0 Loss Of Lock Status** PLL0 loss of lock condition occurred. SELSYSx using PLL0 is switched to hp_clk. To enable PLL0 as SELSYSx output the loss of lock condition has to be cleared. This flag is cleared automatically by hardware when the corresponding clear bit is written and PLL0 is locked. 0_B **noLOL**, no PLL Loss of Lock occurred 1_B **LOL**, a PLL Loss of Lock has occurred

Field	Bits	Type	Description
PLL1_LOL_STS	5	rhxr	**PLL1 Loss Of Lock Status** PLL1 loss of lock condition occurred. SELSYSx using PLL1 is switched to hp_clk. To enable PLL1 as SELSYSx output the loss of lock condition has to be cleared. This flag is cleared automatically by hardware when the corresponding clear bit is written and PLL1 is locked. 0_B **noLOL**, no PLL Loss of Lock occurred 1_B **LOL**, a PLL Loss of Lock has occurred
OSCSEL_STAT0	16	r	**Oscillator Select Status PLL0** 0_B **hp_clk**, hp_clk selected as reference 1_B **xtal_clk**, xtal_clk selected as reference
OSCSEL_STAT1	17	r	**Oscillator Select Status PLL1** 0_B **hp_clk**, hp_clk selected as reference 1_B **xtal_clk**, xtal_clk selected as reference

PLL Status Clear Register

STATC
PLL Status Clear Register **(0010_H)** **RESET_TYPE_5 Value: 0000 0000_H**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														PLL1_LOL_STSCLR	PLL0_LOL_STSCLR
r														w	w

Field	Bits	Type	Description
PLL0_LOL_STSCLR	0	w	**PLL0 Loss Of Lock Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **NotCleared**, Status not cleared 1_B **Cleared**, Status cleared
PLL1_LOL_STSCLR	1	w	**PLL1 Loss Of Lock Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **NotCleared**, Status not cleared 1_B **Cleared**, Status cleared
RES	31:2	r	**Reserved**

Register description SCU

PLL Status Set Register

STATS
PLL Status Set Register **(0014_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														PLL1_LOL_STSSET	PLL0_LOL_STSSET
r														w	w

Field	Bits	Type	Description
PLL0_LOL_STSSET	0	w	**PLL0 Loss Of Lock Status Set** This bit is set by software and automatically cleared by hardware. 0_B **NoSET**, Status not SET 1_B **SET**, Status SET
PLL1_LOL_STSSET	1	w	**PLL1 Loss Of Lock Status Set** This bit is set by software and automatically cleared by hardware. 0_B **NoSET**, Status not SET 1_B **SET**, Status SET
RES	31:2	r	**Reserved**

7 Microcontroller Unit (MCU)

7.1 Features overview

The MCU provides following features:

- Arm® Cortex®-M3 processor (see **Chapter 7.3**)
  - Arm® Cortex®-M3 processor core
  - Arm® System Timer (SYSTICK)
  - Nested Vector Interrupt Controller (NVIC)
  - Arm® CoreSight™ Debug Unit (SW-DP)
- Direct Memory Access (DMA) (see **Chapter 7.4**)
- Memory system (see **Chapter 7.5**)
  - Non-volatile memory uncached (FLASH0)
  - Non-volatile memory cached (FLASH1)
  - Program SRAM memory (PSRAM)
  - Data SRAM memory (DSRAM)
  - Read-only memory (ROM)
- Cache system (see **Chapter 7.6**)
- Multilayer Bus Matrix (see **Chapter 7.7**)
  - Bus Matrix interconnect topology
  - AHB watcher
- Memory Access Control (MAC) (see **Chapter 7.8**)
  - Memory protection
  - Trusted Gate mechanism
  - Firmware
- Peripheral Bridge (PBA0/1) (see **Chapter 7.9**)

7.2 Block diagram

The **Figure 74** illustrates the top-level architecture of the Microcontroller Unit sub-system.

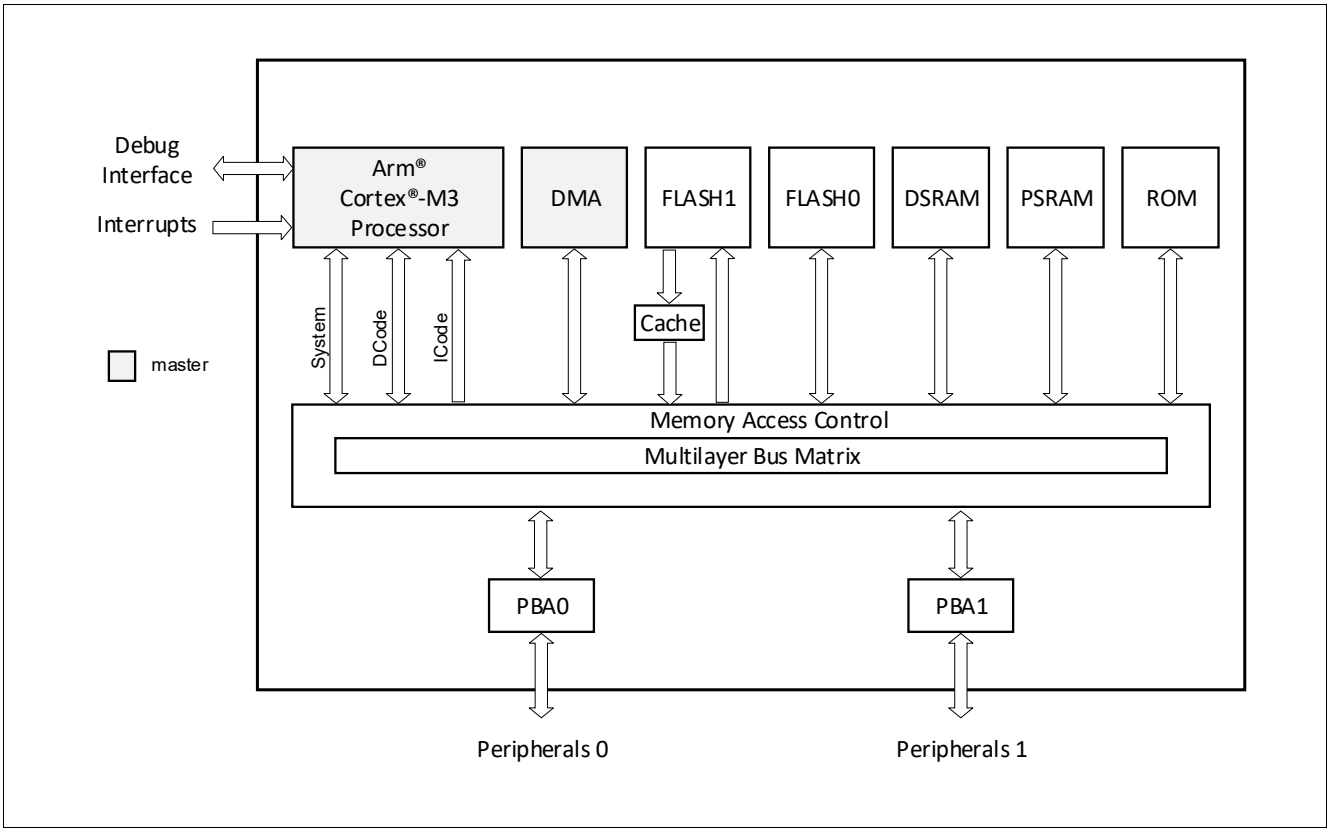


Figure 74 Block diagram MCU

7.3 Arm® Cortex®-M3 processor

The Arm® Cortex®-M3 processor is the top-level of the Arm® Cortex®-M3 CPU sub-system and contains other Arm® building blocks providing all the functional features.

The Arm® Cortex®-M3 processor comes with a 3-stage pipelined Harvard architecture, making it ideal for a wide range of embedded automotive applications. The processor delivers exceptional power efficiency through an efficient instruction set and an optimized design. To address the growing complexity of embedded automotive applications it also includes a range of single-cycle/SIMD multiplication, multiply-and-accumulate, saturating arithmetic and dedicated hardware division instructions. The Arm® Cortex®-M3 processor implements tightly-coupled system components that reduce processor area while significantly improving interrupt handling and system debug capabilities. The instruction set provides the exceptional performance expected of a modern 32-bit architecture with the high code density of 8-bit and 16-bit microcontrollers.

References

[1] Cortex-M3 Technical Reference Manual

[2] CoreSight™ ETM-M3 Technical Reference Manual

[3] CoreSight™ Technology System Design Guide

[4] CoreSight™ Components Technical Reference Manual

[5] Arm® Debug Interface v5 Architecture Specification

[6] Embedded Trace Macrocell Architecture Specification

[7] ARMv7-M Architecture Reference Manual

7.3.1 Block diagram

Figure 75 shows the functional blocks of the Arm® Cortex®-M3.

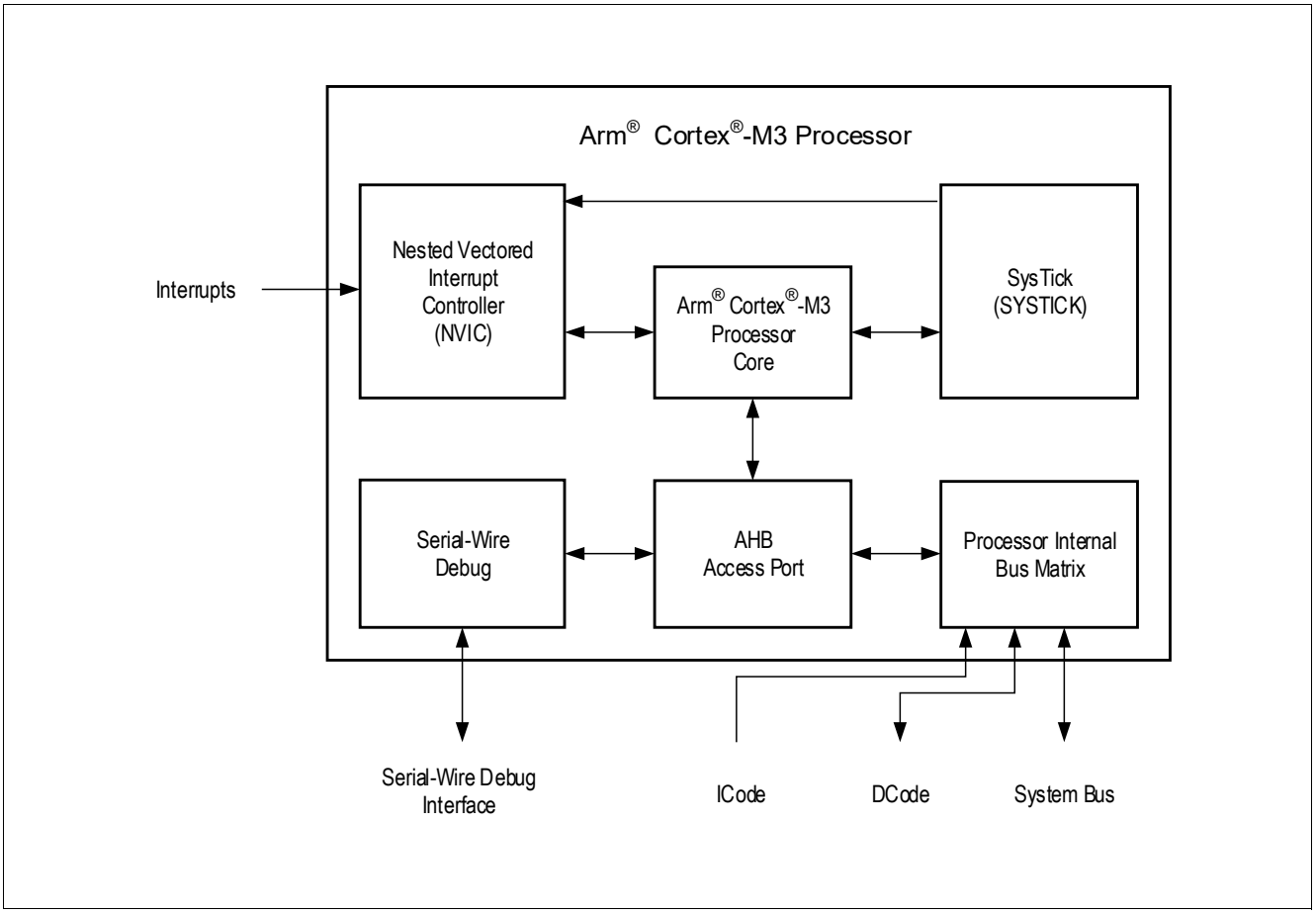


Figure 75 Arm® Cortex®-M3 processor block diagram

7.3.2 Processor core

The Arm® Cortex®-M3 processor core is a central part of the processor, executing code.

The Arm® Cortex®-M3 processor provides the following features:

- Arm® architecture v7-M
- Thumb®-2 instruction set
- 32-bit hardware divide instructions
- 3-stage pipeline
- Banked stack pointers
- Automatic processor state saving and restoration
- Low interrupt latency
- System Timer (SYSTICK)
- Integrated debug interface

7.3.2.1 Processor registers

The processor has the following 32-bit registers:

- 13 general-purpose registers, R0-R12
- Stack pointer (SP), R13 alias of banked registers, SP_process and SP_main
- Link register (LR), R14
- Program counter (PC), R15
- Special-purpose program status registers (xPSR)

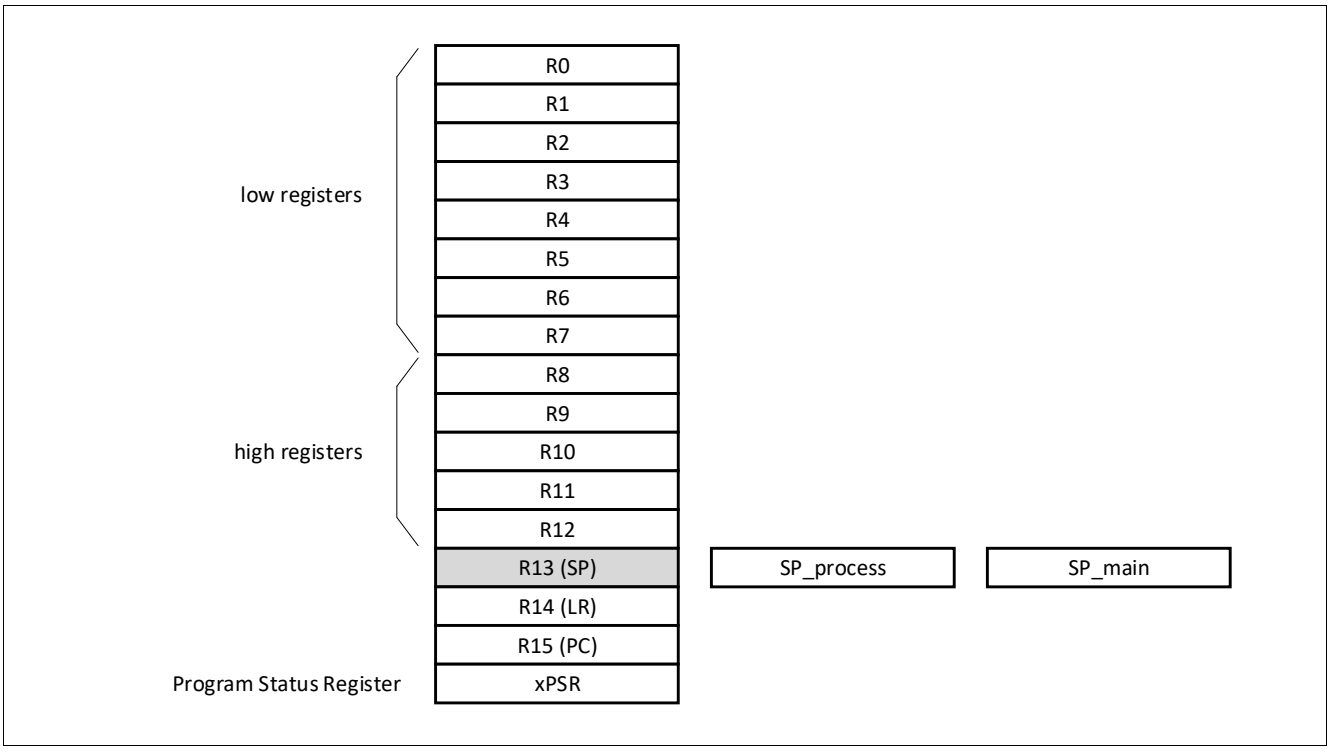


Figure 76 Processor register set

7.3.2.1.1 General-purpose registers

The general-purpose registers R0-R12 have no special architecturally-defined use. Most instructions that can specify a general-purpose register can specify R0-R12.

- **Low registers**
  Registers R0-R7 are accessible by all instructions that specify a general-purpose register.
- **High registers**
  Registers R8-R12 are accessible by all 32-bit instructions that specify a general-purpose register.
  Registers R8-R12 are not accessible by all 16-bit instructions.
- **Stack Pointer**
  Register R13 is used as Stack Pointer (SP).
- **Link register**
  Register R14 is the subroutine Link Register (LR).
- **Program Counter**
  Register R15 is the Program Counter (PC).

7.3.2.1.2 Special-purpose program status registers (xPSR)

Processor status at the system level breaks down into three categories:

- Application PSR
- Interrupt PSR
- Execution PSR

They can be accessed as individual registers, a combination of any two from three, or a combination of all three using the Move to Register from Status (MRS) and MSR instructions.

7.3.2.2 Instruction set summary

This chapter provides the Instruction set. **Table 86** shows the instructions and their cycle counts. The cycle counts are based on a system with zero wait states.

Within the assembler syntax, depending on the operation, the <op2> field can be replaced with one of the following options:

- A simple register
- An immediate shifted register
- A register-shifted register
- An immediate value

For brevity, not all load and store addressing modes are shown.

Table 86 uses the following abbreviations in the cycles column:

- P for the number of cycles required for a pipeline refill
- B for the number of cycles required to perform the barrier operation
- N for the number of registers in the register list to be loaded or stored, including PC or LR
- W for the number of cycles spent waiting for an appropriate event

Table 86 Instruction set summary

Operation	Description	Assembler instruction	Cycles (without wait states)
Move	Register	MOV Rd, <op2>	1
	16-bit immediate	MOVW Rd, #<imm>	1
	Immediate into top	MOVT Rd, #<imm>	1
	To PC	MOV PC, Rm	1 + P
Add	Add	ADD Rd, Rn, <op2>	1
	Add to PC	ADD PC, PC, Rm	1 + P
	Add with carry	ADC Rd, Rn, <op2>	1
	Form address	ADR Rd, <label>	1
Subtract	Subtract	SUB Rd, Rn, <op2>	1
	Subtract with borrow	SBC Rd, Rn, <op2>	1
	Reverse	RSB Rd, Rn, <op2>	1
Multiply	Multiply	MUL Rd, Rn, Rm	1
	Multiply accumulate	MLA Rd, Rn, Rm	2
	Multiply subtract	MLS Rd, Rn, Rm	2
	Long signed	SMULL RdLo, RDHi, Rn, Rm	3 to 5[1)]
	Long unsigned	UMULL RdLo, RdHi, Rn, Rm	3 to 5[1)]
	Long singed accumulate	SMLAL RdLo, RdHi, Rn, Rm	4 to 7[1)]
	Long unsigned accumulate	UMLAL RdLo, RdHi, Rn, Rm	4 to 7[1)]
Divide	Signed	SDIV Rd, Rn, Rm	2 to 12[2)]
	Unsigned	UDIV Rd, Rn, Rm	2 to 12[2)]
Saturate	Signed	SSAT Rd, #i<imm>, <op2>	1
	Unsigned	USAT Rd, #<imm>, <op2>	1
Compare	Compare	CMP Rn, <op2>	1
	Negative	CMN Rn, <op2>	1
Logical	AND	AND Rd, Rn, <op2>	1
	Exclusive OR	EOR Rd, Rn, <op2>	1
	OR	ORR Rd, Rn, <op2>	1
	OR NOT	ORN Rd, Rn, <op2>	1
	Bit clear	BIC Rd, Rn, <op2>	1
	Move NOT	MVN Rd, <op2>	1
	AND test	TST Rn, <op2>	1
	Exclusive OR test	TEQ Rn, <op1>	

Table 86 Instruction set summary (cont'd)

Operation	Description	Assembler instruction	Cycles (without wait states)
Shift	Logical shift left	LSL Rd, Rn, #<imm>	1
	Logical shift left	LSL Rd, Rn, Rs	1
	Logical shift right	LSR Rd, Rn, #<imm>	1
	Logical shift right	LSR Rd, Rn, Rs	1
	Arithmetic shift right	ASR Rd, Rn, #<imm>	1
	Arithmetic shift right	ASR Rd, Rn, Rs	1
Rotate	Rotate right	ROR Rd, Rn, #<imm>	1
	Rotate right	ROR Rd, Rn, Rs	1
	With extension	RRX Rd, Rn	1
Count	Leading zeroes	CLZ Rd, Rn	1
Load	Word	LDR Rd, [Rn, <op2>]	2[3]
	To PC	LDR PC, [Rn, <op2>]	2[3] + P
	Halfword	LDRH Rd, [Rn, <op2>]	2[3]
	Byte	LDRB Rd, [Rn, <op2>]	2[3]
	Signed halfword	LDRSH Rd, [Rn, <op2>]	2[3]
	Signed byte	LDRSB Rd, [Rn, <op2>]	2[3]
	User word	LDRT Rd, [Rn, #<imm>]	2[3]
	User halfword	LDRHT Rd, [Rn, #<imm>]	2[3]
	User byte	LDRBT Rd, [Rn, #<imm>]	2[3]
	User signed halfword	LDRSHT Rd, [Rn, #<imm>]	2[3]
	User signed byte	LDRSBT Rd, [Rn, #<imm>]	2[3]
	PC relative	LDR Rd, [PC, #<imm>]	2[3]
	Doubleword	LDRD Rd, Rd, [Rn, #<imm>]	1 + N
	Multiple	LDM Rn, {<reglist>}	1 + N
	Multiple including PC	LDM Rn, {<reglist>, PC}	1 + N + P
Store	Word	STR Rd, [Rn, <op2>]	2[3]
	Halfword	STRH Rd, [Rn, <op2>]	2[3]
	Byte	STRB Rd, [Rn, <op2>]	2[3]
	Signed halfword	STRSH Rd, [Rn, <op2>]	2[3]
	Signed byte	STRSB Rd, [Rn, <op2>]	2[3]
	User word	STRT Rd, [Rn, #<imm>]	2[3]
	User halfword	STRHT Rd, [Rn, #<imm>]	2[3]
	User byte	STRBT Rd, [Rn, #<imm>]	2[3]
	User signed halfword	STRSHT Rd, [Rn, #<imm>]	2[3]
	User signed byte	STRSBT Rd, [Rn, #<imm>]	2[3]
	Doubleword	STRD RD, Rd, [Rn, #<imm>]	1 + N
	Multiple	STM Rn, {<reglist>}	1 + N

Table 86 Instruction set summary (cont'd)

Operation	Description	Assembler instruction	Cycles (without wait states)
Push	Push	PUSH {<reglist>}	1 + N
	Push with link register	PUSH {<reglist>, LR}	1 + N
Pop	Pop	POP {<reglist>}	1 + N
	Pop and return	POP {<reglist>, PC}	1 + N + P
Semaphore	Load exclusive	LDREX Rd, [Rn, #imm>]	2
	Load exclusive half	LDREXH Rd, [Rn]	2
	Load exclusive byte	LDREXB Rd, [Rn]	2
	Store exclusive	STREX Rd, Rt, [Rn, #<imm>]	2
	Store exclusive half	STREXH Rd, Rt, [Rn]	2
	Store exclusive byte	STREXB Rd, Rt, [Rn]	2
	Clear exclusive monitor	CLREX	1
Branch	Conditional	B <cc> <label>	1 or 1 + P[4)]
	Unconditional	B <label>	1 + P
	With link	BL <label>	1 + P
	With exchange	BX Rm	1 + P
	With link and exchange	BLX Rm	1 + P
	Branch if zero	CBZ Rn, <label>	1 or 1 + P[4)]
	Branch if non-zero	CBZN Rn, <label>	1 or 1 + P[4)]
	Byte table branch	TBB [Rn, Rm]	2 + P
	Halfword table branch	TBH [Rn, Rm, LSL#1]	2 + P
State change	Supervisor call	SVC #<imm>	–
	If-then-else	IT... <cond>	1[5)]
	Disable interrupts	CPSID <flags>	1 or 2
	Enable interrupts	CPSIE <flags>	1 or 2
	Read special register	MRS Rd, <specreg>	1 or 2
	Write special register	MSR <specreg>, Rn	1 or 2
	Breakpoint	BKPT #<imm>	–
Extend	Signed halfword to word	SXTH Rd, <op2>	1
	Signed byte to word	SXTB Rd, <op2>	1
	Unsigned halfword	UXTH Rd, <op2>	1
	Unsigned byte	UXTB Rd, <op2>	1
Bit field	Extract unsigned	UBFX Rd, Rn, #<imm>, #<imm>	1
	Extract signed	SBFX Rd, Rn, #<imm>, #<imm>	1
	Clear	BFC Rd, Rn, #<imm>, #<imm>	1
	Insert	BFI Rd, Rn, #<imm>, #<imm>	1

Table 86 Instruction set summary (cont'd)

Operation	Description	Assembler instruction	Cycles (without wait states)
Reverse	Bytes in word	REV Rd, Rm	1
	Bytes in both halfwords	REV16 Rd, Rm	1
	Signed bottom halfword	REVSH Rd, Rm	1
	Bits in word	RBIT Rd, Rm	1
Hint	Send event	SEV	1
	Wait for event	WFE	1 + W
	Wait for interrupt	WFI	1 + W
	No operation	NOP	1
Barriers	Instruction synchronization	ISB	1 + B
	Data memory	DMB	1 + B
	Data synchronization	DSB <flags>	1 + B

1) UMULL, SMULL, UMLAL, and SMLAL instructions use early termination depending on the size of the source values. These are interruptible, that is abandoned and restarted, with worst case latency of one cycle.
2) Division operations use early termination to minimize the number of cycles required based on the number of leading ones and zeros in the input operands.
3) Neighboring load and store single instructions can pipeline their address and data phases. This enables these instructions to complete in a single execution cycle.
4) Conditional branch completes in a single cycle if the branch is not taken.
5) An IT instruction can be folded onto a preceding 16-bit Thumb instruction, enabling execution in zero cycles.

7.3.3 System Timer (SYSTICK)

The System Timer (SYSTICK) can be used as general-purpose timer for e.g. periodic operating system tick. The SYSTICK event can request an exception at the NVIC.

The System Timer (SYSTICK) can be used as general-purpose timer, as a periodic system tick.

The SYSTICK has following features:

- 24-bit down-counting timer
- Auto-reload from programmable reload value
- Selectable clock source
- CPU exception (system tick)

7.3.3.1 Block diagram SYSTICK

The SYSTICK timer diagram is shown in **Chapter 77**.

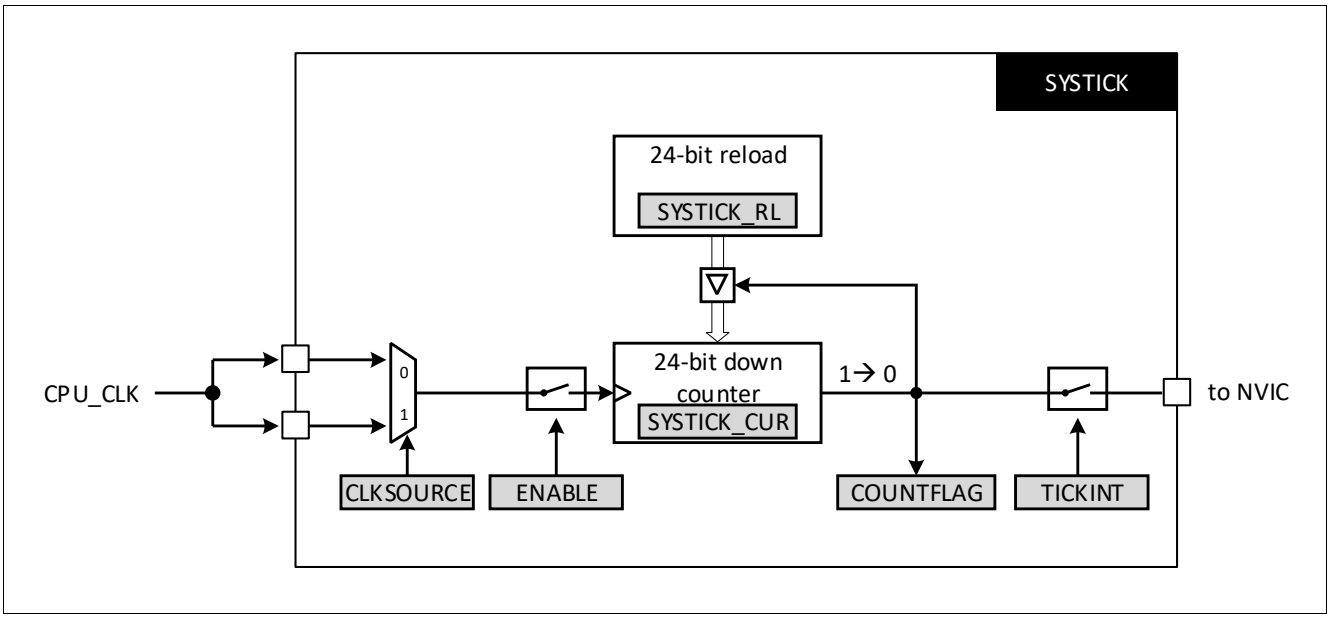


Figure 77 SYSTICK block diagram

7.3.3.2 SYSTICK interrupts

The SYSTICK event can request an exception at the NVIC.

7.3.3.3 SYSTICK control scheme

The SYSTICK has the following features:

- There is a 24-bit down counter (SYSTICK_CUR) which is clocked from a selectable clock source. The clock source selection is done via SYSTICK_CS.CLKSOURCE.
- The down-counter is loaded initially from a programmable reload value (SYSTICK_RL), when the bit SYSTICK_CS.ENABLE is set to 1.
- The down-counter is reloaded automatically when the current count value transitions from 1 to 0 (SYSTICK event).
- The SYSTICK event is displayed in SYSTICK_CS.COUNTFLAG and can request an exception at NVIC when enabled via SYSTICK_CS.TICKINT.
- The count period is the SYSTICK_RL.RELOAD + 1.

- The current count value can be read from SYSTICK_CUR.
- There is a reference value which reflects a SYSTICK period of 10ms for a defined clock (here 60 MHz). This value is hard-coded in SYSTICK_CAL.TENMS (here value 600000 or 0x927C0).

7.3.3.4 SYSTICK suspend features

The SYSTICK is halted when debugging.

7.3.3.5 SYSTICK operation mode behavior

The SYSTICK is part of the MCU sub-system and behaves according to **Table 87**.

Table 87 Operation mode behavior SYSTICK

Reset	The SYSTICK is reset via HOT_RESET. It has the following effects: • All SFRs of the SYSTICK are reset.
Power-up/ Power-down	• The SYSTICK is kept in reset state by HOT_RESET as long as its supply and clock are not in specified operating range. • The SYSTICK is released from reset state by HOT_RESET when its supply or clock are within the specified operating range.
Active mode	• The SYSTICK (and the MCU sub-system) is automatically set to active mode with releasing HOT_RESET.
Stop mode	• The SYSTICK clock is stopped automatically at stop mode entry. The configuration is kept. • The SYSTICK clock is started automatically at stop mode exit. The SYSTICK recovers from stop mode to active mode with the configuration at the stop mode entry.
Sleep mode	• The sleep mode entry transition is like a power-down. The SYSTICK clock and supply are switched off automatically, HOT_RESET is asserted. All configuration settings are lost in sleep mode entry. • The sleep mode exit transition is like a power-up. The SYSTICK recovers from sleep mode to active mode from its reset state.
Fail-sleep	• None

7.3.4 Nested Vector Interrupt Controller (NVIC)

The interrupt system is built of the Nested Vector Interrupt Controller (NVIC) integrated in the Arm® Cortex®-M3 Processor. It provides several interrupt nodes for on-chip peripheral interrupts and exception nodes for internal core exceptions as well as for Non-Maskable Interrupts (NMI).

The NVIC handles the interrupt requests directed to the interrupt nodes. A prioritization scheme allows to organize the processing order in case of simultaneous requests or nested interrupts. Multiple peripheral interrupts can be connected to shared nodes. The peripheral interrupt sources are controlled inside the corresponding peripheral module.

The interrupt system provides the following features:

- 32 interrupt nodes for peripheral events, 8 exception nodes for core and system-related events
- 32 interrupt levels with grouping scheme and dynamic re-prioritization option
- Support for tail-chaining for minimum latency at nested interrupt requests
- Relocatable interrupt vector table
- Automatic processor state saving at interrupt entry and restoring at interrupt exit

7.3.4.1 Block diagram NVIC

The block diagram **Table 78** illustrates the top-level features of the NVIC module in the context of the interrupt system.

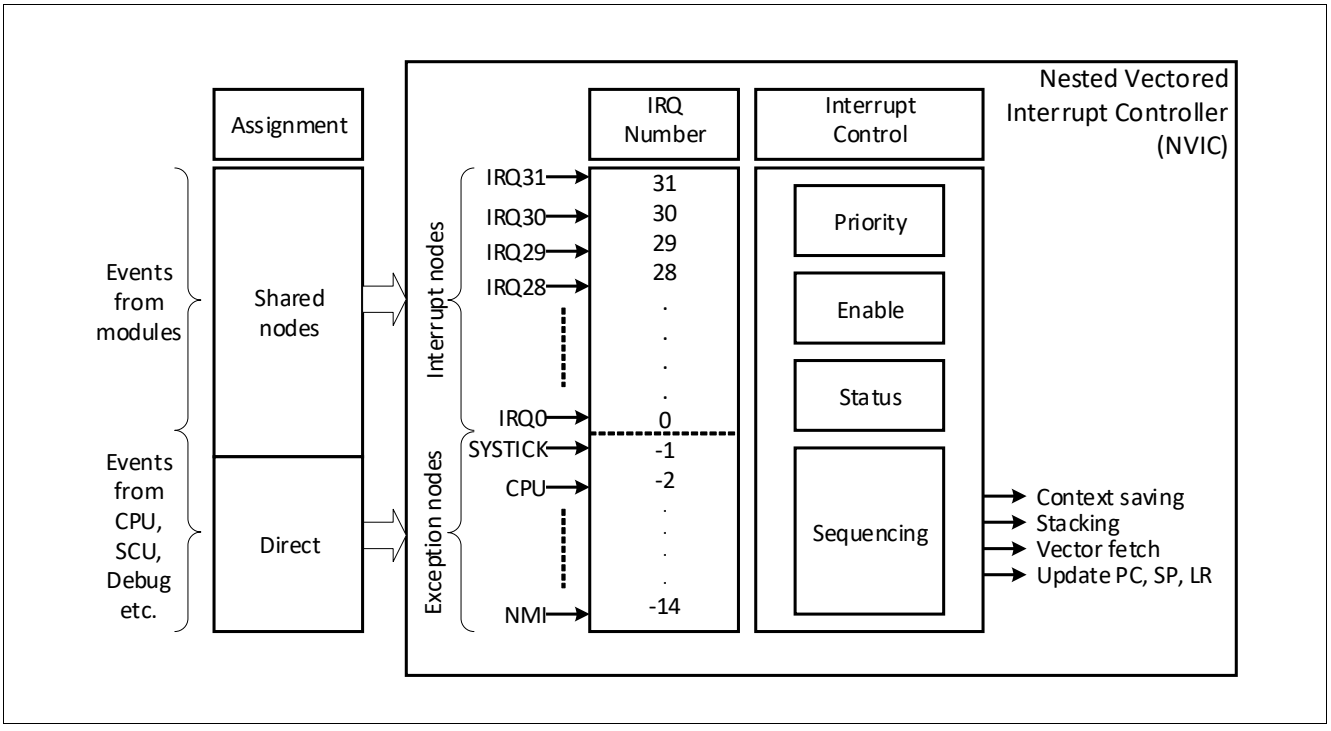


Figure 78 NVIC block diagram

7.3.4.2 Interrupt request sources

- 32 interrupt request lines for on-chip peripheral events
- 1 interrupt request line for NMI events
- 1 interrupt request line for SYSTICK events
- 6 exception sources for events of CPU, AHB and debug
- Software interrupt (SVCall using SVC instruction)

The interrupt nodes have individual enable and status flags

7.3.4.3 Interrupt prioritization

There are 32 interrupt priority levels can be programmed individually per interrupt node (the lower the setting number, the higher the priority of the interrupt). The interrupt priority can be dynamically changed during run-time. There is also a priority grouping scheme available to select of preemptive and not-preemptive interrupt handling.

Prioritization features

- 32 interrupt priority levels which can be programmed individually per interrupt node
- Interrupt priority can be dynamically changed (reprioritized)
- Priority grouping scheme enables the selection of preemptive and not-preemptive interrupt handling and nested interrupts

7.3.4.4 Interrupt sequencing

Sequencing features

- The processor state is automatically saved on interrupt entry, and restored on interrupt exit, with no instruction overhead (interrupt sequencing)
- The interrupt latency is deterministic
- There is a support for tail-chaining and late arrival of interrupts. This enables back-to-back interrupt processing without the overhead of state saving and restoration between interrupts
- The interrupt vector table is relocatable

7.3.4.5 Edge sensitive interrupt requests

Regardless of the event, the interrupt request which is sent to the NVIC is always edge/pulse sensitive.

7.3.4.6 NVIC operation mode behavior

The NVIC is part of the MCU sub-system and behaves according to **Table 88**.

Table 88 Operation mode behavior NVIC

Reset	The NVIC is reset via RESET_TYPE_3. It has the following effects: • All SFRs of the NVIC are reset.
Power-up/ Power-down	• The NVIC is kept in reset state by RESET_TYPE_3 as long as its supply and clock are not in specified operating range. • The NVIC is released from reset state by RESET_TYPE_3 when its supply or clock are within the specified operating range.
Active mode	• The NVIC (and the MCU sub-system) is automatically set to active mode with releasing RESET_TYPE_3.
Stop mode	• The NVIC clock is stopped automatically at stop mode entry. The configuration is kept. • The NVIC clock is started automatically at stop mode exit. The NVIC recovers from stop mode to active mode with the configuration at the stop mode entry.
Sleep mode	• The sleep mode entry transition is like a power-down. The NVIC clock and supply are switched off automatically, RESET_TYPE_3 is asserted. All configuration settings are lost in sleep mode entry. • The sleep mode exit transition is like a power-up. The NVIC recovers from sleep mode to active mode from its reset state.
Fail-sleep	• None

7.3.5 Debug system

The debug system is used to observe or modify the state of the device during application software development.

The TLE989x/TLE988x provides a variety of debug and test features which are implemented with the Arm® CoreSight™ module. They are integrated into the Arm® Cortex®-M3.

The implemented debug system provides the following features:

- Arm® CoreSight™ module connected to a Serial Wire Debug Port (SW-DP)
- Two hardware breakpoints
- Halt after reset (HAR)
- Single-stepping
- Processor core registers access
- Vector catching (use exceptions as breakpoints)
- One data watchpoint
- System memory access

7.3.5.1 Debug access

To use the debug system on the TLE989x/TLE988x, two dedicated pins are used to establish an SWD connection. The dedicated pins are P0.0 and TMS which are utilized as SWD_CLK (P0.0) and SWD_DIO (TMS). The SWD_DIO signal (TMS pin) is gated by the permanent memory protection. The permanent memory protection is enabled/disabled by the BootROM firmware via BSL (see **Figure 79**).

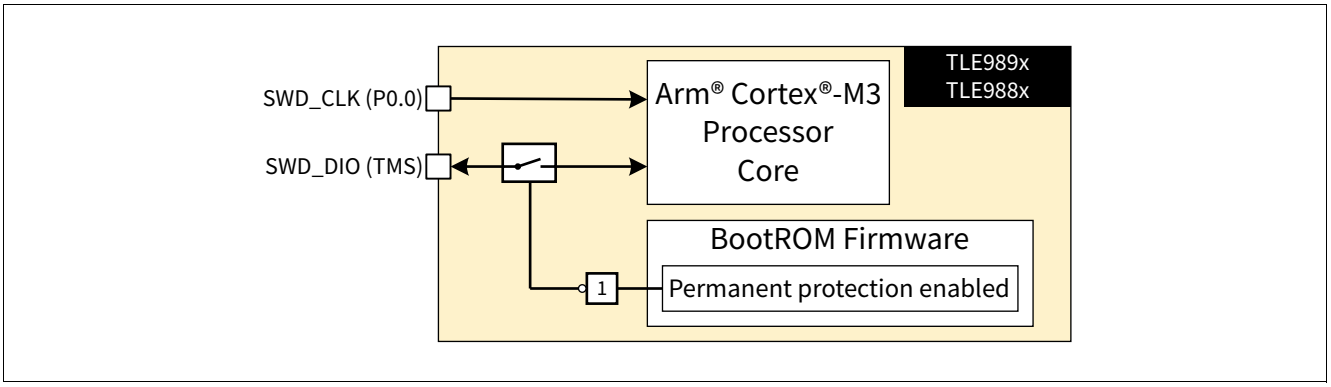


Figure 79 Internal SWD connection

7.3.5.2 Debug mode

The debug mode is an extended user mode where debug is enabled in case debug access is granted (see Debug access). The entry and exit of the debug mode are controlled by the BootROM firmware and external pins. To switch between user and debug mode, a device reset in conjunction with TMS and P0.0 control is needed (see **Figure 80**).

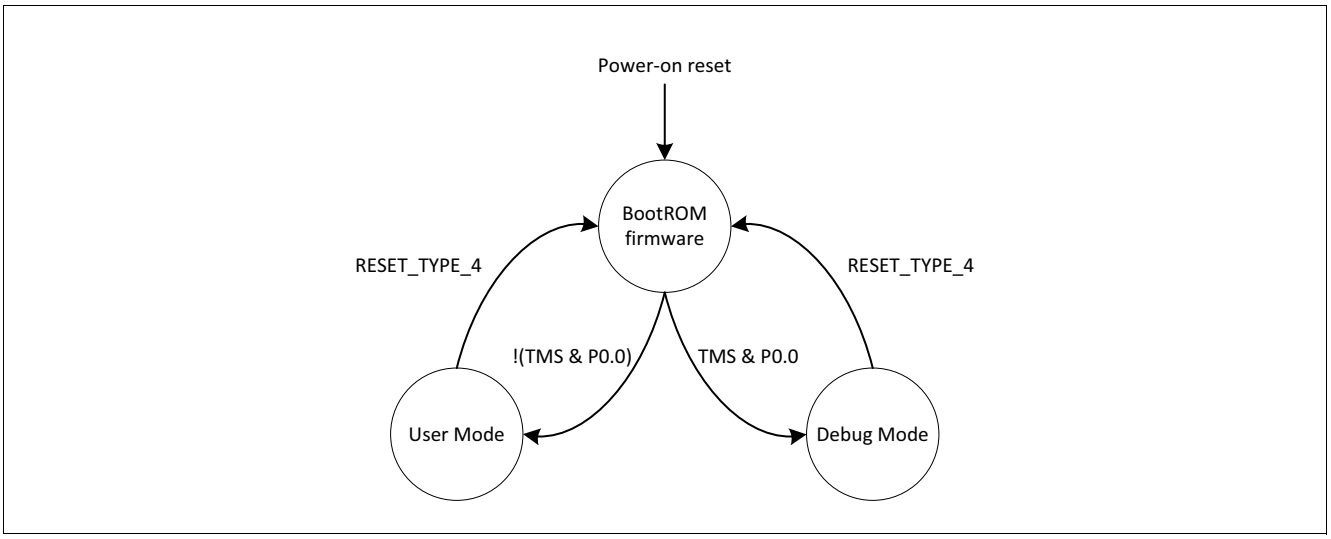


Figure 80 Switching between debug and user mode

TMS device reset

To reset a device which is not already in debug mode, TMS and P0.0 pins can be used. The reset is triggered externally by keeping the TMS and P0.0 pins high for at least 200 clock cycles of the master clock (f_{MCLK}). To successfully reset the device, consider the minimum clock speed of f_{MCLK} and the oscillator settling time in the datasheet.

After triggering the reset externally (TMS_RST), this mechanism is disabled until the next RESET_TYPE_4 occurs, which resets it. This is to prevent any accidental reset of the device during SWD communication.

Debug mode entry

To enter the debug mode, TMS and P0.0 need both to be driven high externally after device reset. If both pins are latched high after reset, the BootROM firmware proceeds with the device startup in debug mode instead of user mode. This ultimately enables the debug capabilities of the device. The device reset, after which TMS and P0.0 are latched, can be triggered by TMS and P0.0 (see “**TMS device reset**”) or by the P0.10 pin if it is configured as reset pin. After entering debug mode, the DHCSR.C_DEBUGEN bit can be set by the debugger.

The complete procedure (using the TMS device reset) is shown in **Figure 81**.

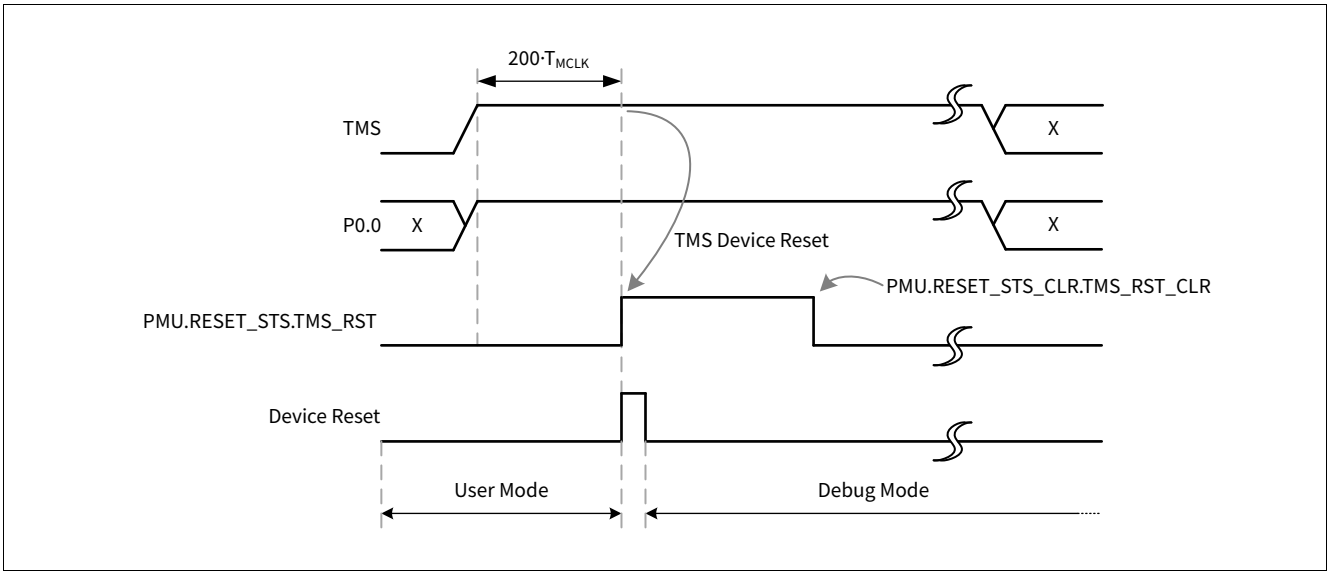


Figure 81 Device reset and debug mode entry

Note: A reset via TMS/P0.0 is signalized by the PMU.RESET_STS.TMS_RST bit. This status bit should be cleared afterwards with the PMU.RESET_STS_CLR.TMS_RST_CLR bit.

Debug mode exit

To exit the debug mode, the device has to be reset with a RESET_TYPE_4. The TMS pin has to be kept low during and after the device reset to ensure that it is not latched as high by the BootROM firmware. Otherwise, the device boots up into debug mode instead of user mode. The device reset can for example be triggered by setting the PMU.MISC_CTRL.TRIG_RST bit. The procedure is depicted in **Figure 82**.

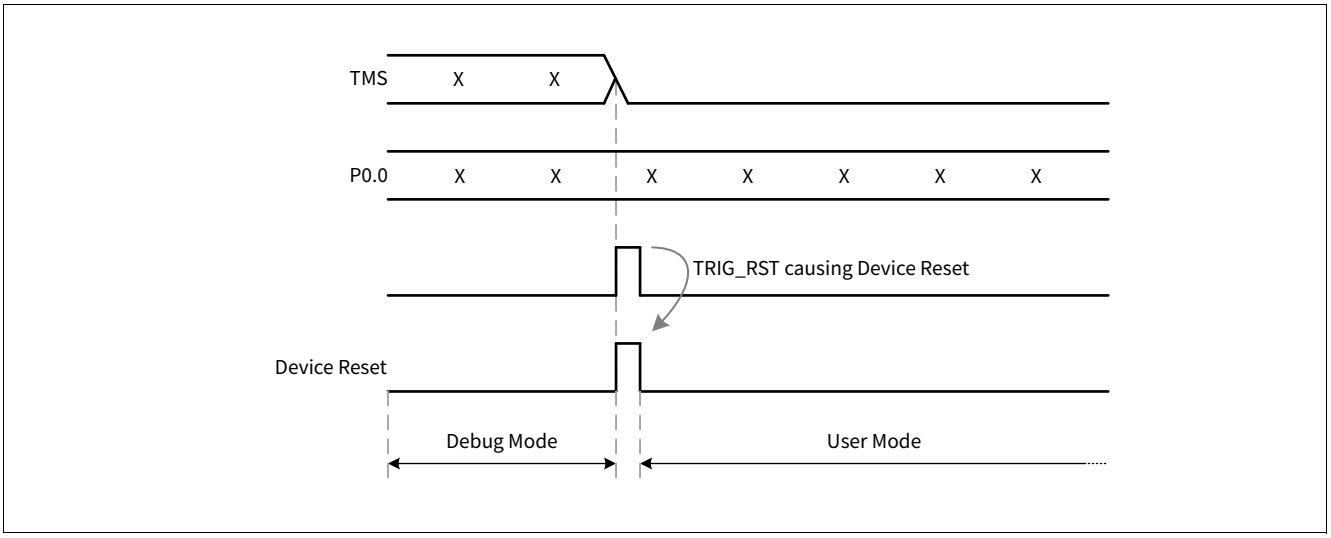


Figure 82 Device reset and debug mode exit

By resetting the device to exit the debug mode, the external reset mechanism through TMS and P0.0 is reset as well. It can then be used to enter the debug mode again at a later point in time.

Note: The status bit indicating which reset type was used should be cleared afterwards.

Device reset in debug mode

If a device reset is needed while in debug mode, it has to be ensured that TMS and P0.0 are kept high during and after the device reset. If both pins are then latched high prior to the BootROM firmware execution, the device proceeds to enter the debug mode again, hence preserving the debug mode during device reset. The procedure is depicted in **Figure 83**.

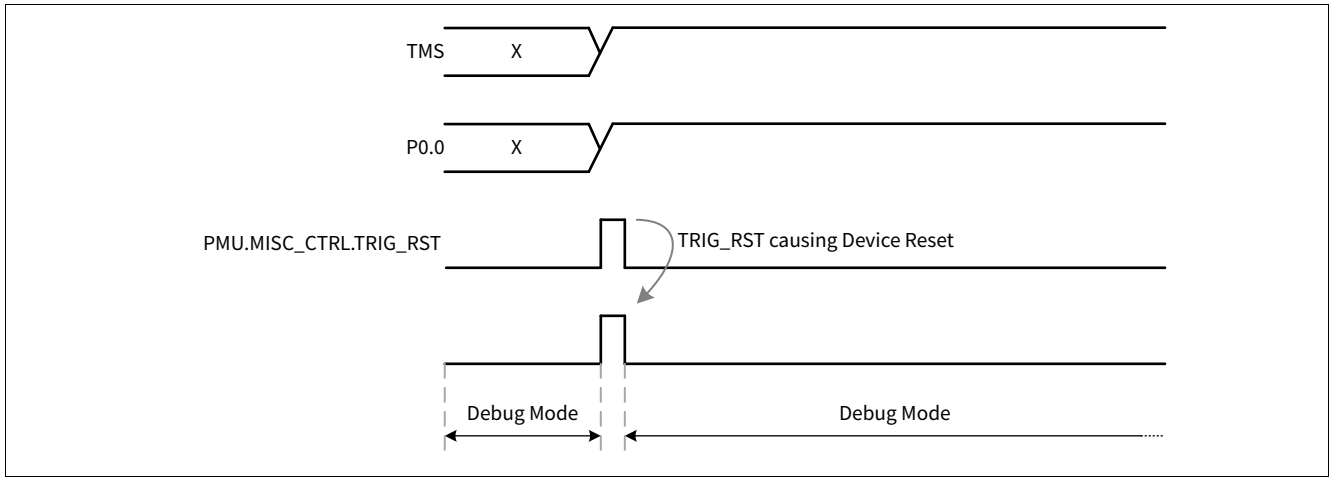


Figure 83 Preserving debug mode during device reset

Since the reset with PMU.MISC_CTRL.TRIG_RST causes a RESET_TYPE_4, the debugger has to re-establish the connection afterwards and also restore all debug settings (for example breakpoints). To overcome this issue, a soft reset like AIRCR.RESETSYSREQ can be used. Another advantage of the soft reset is, that TMS and P0.0 are not latched and hence they do not need to be pulled high (see **Figure 84**).

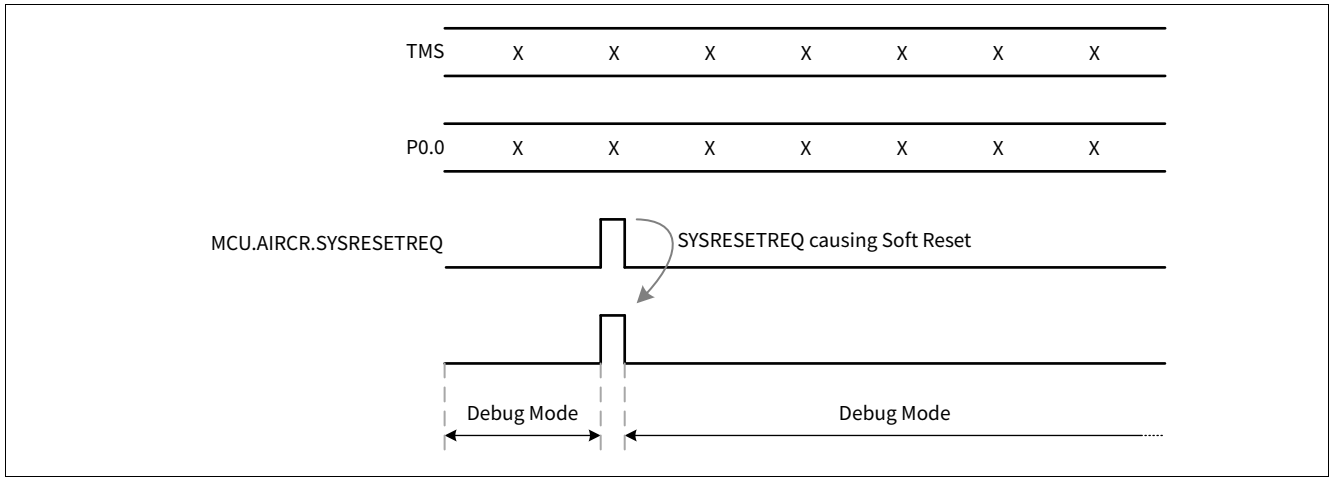


Figure 84 Preserving debug mode by using soft reset

7.3.5.3 Debug monitor

In case the halting debug mode cannot be used due to time critical software, the built-in debug monitor can be enabled. By enabling the debug monitor, debug events can be used as interrupt source. This can be set up with the Debug Exception and Monitor Control register (DEMCR). The priority of the debug monitor interrupt is set within the System Handler Priority register 3 (SHPR3).

Note: This is only possible if debug is disabled (DHCSR.C_DEBUGEN = 0).

7.4 Direct Memory Access (DMA)

The Direct Memory Access (DMA) system allows to transfer data between different registers and/or memory locations in the background, off-loading the CPU. Efficient use of the DMA controller can significantly improve system performance.

The DMA controller PL230 of Arm® is a low-power, high-performance module that is designed for optimal compatibility with Arm® Cortex®.

The DMA controller provides the following features:

- Up to 8 DMA channels
- Support for transfer sources and destinations (see **Multilayer Bus Matrix**):
  - Memory-to-memory transfers
  - Memory-to-peripheral transfers
  - Peripheral-to-peripheral transfers
- Multiple cycle types
- Multiple transfer data widths:
  - Byte
  - Word
  - DWord
- Each DMA channel can access a primary and an alternate channel control data structure
- Each DMA channel has a programmable priority level
- All the channel control data is stored in system memory in little-endian format
- The destination data width is equal to the source data width
- The number of transfers in a single DMA cycle can be programmed from 1 to 1024
- The transfer address increment can be greater than the data width

References

[8] PrimeCell® µDMA Controller (PL230), Revision: r0p0, Technical Reference Manual

7.4.1 Usage

The generic usage of the DMA involves several interactions as illustrated in **Figure 85**:

1. DMA configuration installation.
2. DMA hardware trigger and DMA service request configuration in SCU.
3. DMA service request generation from a peripheral towards SCU.
4. DMA service triggering passed from SCU to DMA.
5. DMA configuration fetching from DSRAM.
6. Data transfer between sources and destinations according to the channel configuration.
7. Service request towards SCU generation on DMA transfer completion.
8. Interrupt request to NVIC after DMA completion.

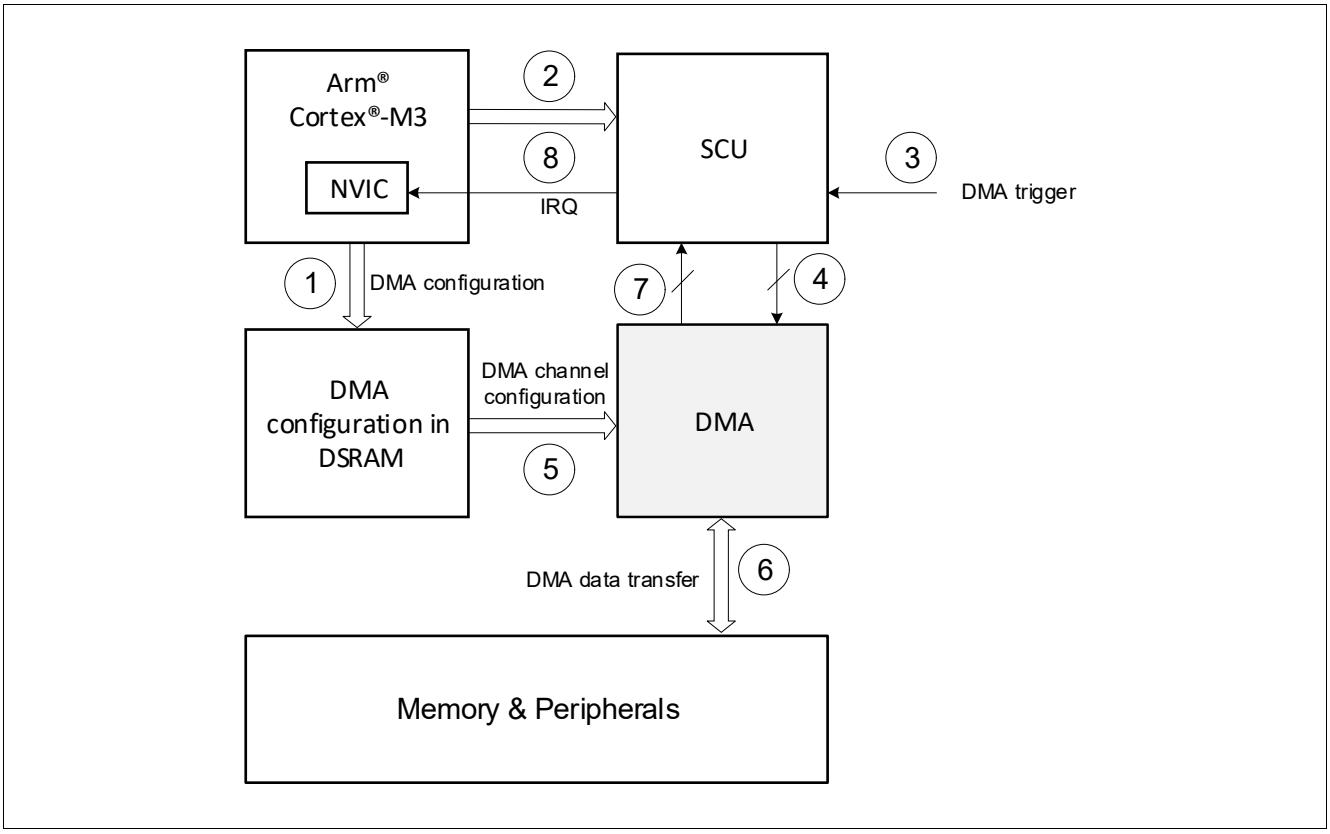


Figure 85 DMA control scheme

Various variations of the sequence are possible depending on the desired DMA operation mode. The hardware DMA trigger may be substituted with a software trigger depending on the application needs. In case of repeatable transfer actions the DMA may be configured to apply the Auto Reload/Rearm feature (see **Chapter 7.4.10**).

7.4.2 Protection control

The controller enables you to configure the AHB-Lite protection control signals HPROT:

- HPROT[0] is tied to high (1) to indicate a data access
- HPROT[1] indicates if the access is privileged (1) or not privileged (0)
- HPROT[2] indicates if the access is bufferable (1) or not bufferable (0)
- HPROT[3] indicates if the access is cacheable (1) or not cacheable (0)

For each DMA cycle, you can configure the source transfer and destination transfer to use different protection control settings.

You can also configure different settings when you access the channel control data structure.

7.4.3 Address increment

The controller enables you to configure the address increments that it uses when it reads the source data or when it writes the destination data. The available increments depend on the size of the data packet being transferred (see **Table 89**).

Table 89 Address increment

Packet data width	Size of address increment		
	Byte	Half-Word	Word
8 bits	X	X	X
16 bits		X	X
32 bits			X

The minimum address increment must always be equal in size to the width of the data packet. The maximum address increment permitted by the controller is one word.

7.4.4 DMA control block

he DMA control block contains the control logic that provides the following features:

- Arbitrates the incoming request
- Indicates which channel is active
- Indicates when a channel is complete
- Indicates when an error has occurred on the bus interface
- Enables slow peripherals to stall the completion of a DMA cycle
- Waits for a request to clear before completing a DMA cycle
- Performs multiple or single DMA transfers for each request
- Performs the following types of DMA transfers

7.4.5 Arbitration rate

You can configure when the controller arbitrates during a DMA transfer, enabling you to reduce the latency to service a higher priority channel.

The arbitration rate which indicates how many AHB bus transfers occur before the controller rearbitrates depends on the R_power bits raised to the power of 2.

- Arbitration rate = 2^R_power if R_power ≤ 9
- Arbitration rate = 1024 if R_power > 9

For example, if R_power = 4, then the controller arbitrates every 2^4 = 16 DMA transfers.

You must take care not to assign a low-priority channel with a large R_power, because this prevents the controller from servicing high-priority requests until it rearbitrates.

7.4.6 Priority

When the controller arbitrates, it determines the next channel to service depending on the channel number and the priority level assigned to that channel. This can be configured in the register CHNL_PRIORITY_SET.

7.4.7 Channel control data structure

You must provide an area of system memory to contain the channel control data structure. This system memory must:

- Provide a contiguous area of system memory that the controller and the host processor can access
- Have a base address that is an integer multiple of the total size of the channel control data structure

7.4.7.1 Base address

You can configure the base address for the primary data structure by writing the appropriate value in the CTRL_BASE_PTR register.

It is not necessary for you to calculate the base address of the alternate data structure, it is provided in the ALT_CTRL_BASE_PTR register.

7.4.7.2 Source end pointer

The src_data_end_ptr memory location contains a pointer to the end address of the source data.

The controller reads this memory location when it starts a 2^R DMA transfer.

7.4.7.3 Destination end pointer

The dst_data_end_ptr memory location contains a pointer to the end address of the destination data.

The controller reads this memory location when it starts a 2^R DMA transfer.

7.4.7.4 Control data configuration

The DMA channel configuration is held in a channel-indexed configuration structure stored in DSRAM. The configuration structure exists in two instances, a primary structure and an alternate structure. These structures hold the desired configuration for each DMA channel. Which of the two configuration structure instances is used or whether both instances are needed depends on the transfer mode. The configuration structures can be reallocated in the DSRAM module.

For each DMA transfer, the channel_cfg memory location provides the control information for the controller.

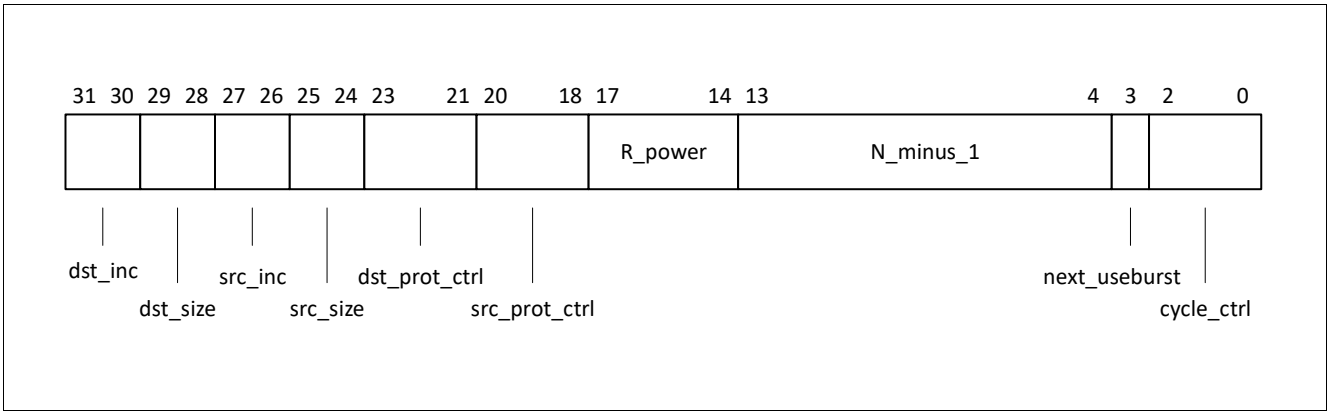


Figure 86 Control data encoding overview

Table 90 Configuration data encoding

Field	Bits	Description			
dst_inc	31:30	Destination address increment, depending on the source data size			
		Destination address increment	Sourced data size		
			Byte	Halfword	Word
		Byte b00	X		
		Halfword b01	X	X	
		Word b10	X	X	X
		No increment b11	X	X	X

Table 90 Configuration data encoding (cont'd)

Field	Bits	Description			
dst_size	29:28	Destination data size, it must contain the same value than src_size b00 Byte b01 Halfword b10 Word b11 Reserved			
src_inc	27:26	Source address increment, depending on the source data size			
		Source address increment	Sourced data size		
			Byte	Halfword	Word
		Byte b00	X		
		Halfword b01	X	X	
		Word b10	X	X	X
		No increment b11	X	X	X
src_size	25:24	Source data size, it must contain the same value than src_size b00 Byte b01 Halfword b10 Word b11 Reserved			
dst_prot_ctrl	23:21	Control of the state of HPROT[3:1] when the controller writes the destination data.			
src_prot_ctrl	25:18	Control of the state of HPROT[3:1] when the controller reads the source data.			
R_power	17:14	Number of DMA transfers which occur before the controller rearbitration. Arbitration rate = 2^R_power			
n_minus_1	13:4	Total number of DMA transfers 0x0, 1 DMA transfer 0x1, 2 DMA transfers … 0x1F, 1024 DMA transfers The controller updates this field immediately prior to it entering the arbitration process			
next_useburst	3	Controls if the CHNL_USEBURST_SET[C] bit is set to 1, when the controller is performing a peripheral scatter-gather and is completing a DMA cycle that uses the alternate data structure			
cycle_ctrl	2:0	Operating mode of the DMA cycle			

At the start of a DMA cycle, the controller fetches the channel_cfg from system memory. After it performs 2^R or N transfers, it stores the updated channel_cfg in the system memory.

After the controller completes the N transfers, it sets the cycle_ctrl field to b000 (Invalid) to indicate that the channel_cfg data is invalid. This prevents it from repeating the same DMA transfer.

7.4.7.5 Address calculation

To calculate the source address of a DMA transfer, the controller performs a left shift operation on the n_minus_1 value by a shift amount specified by src_inc and then subtracts the resulting value from the source data end pointer.

Similarly to calculate the destination address of a DMA transfer, the controller performs a left shift operation on the n_minus_1 value by a shift amount specified by dst_inc and then subtracts the resulting value from the destination end pointer.

Table 91 Address calculation

		src_in			
		b00	b01	b10	b11
dst_inc	b00	rc_add = src_data_end_ptr – n_minus_1 dst_add = dst_data_end_ptr – n_minus_1			
	b01		src_add = src_data_end_ptr – (n_minus_1 << 1) dst_add = dst_data_end_ptr – (n_minus_1 << 1		
	b10			src_add = src_data_end_ptr – (n_minus_1 << 2) dst_add = dst_data_end_ptr – (n_minus_1 << 2)	
	b11				rc_add = src_data_end_ptr dst_add = dst_data_end_ptr

7.4.7.6 Programmer's guide

You can find below the parameter's configuration of the control data structure used in the low-level driver. The fields marked red are fixed for the considered cycle type, the other fields must be configured by the user (see **Table 92**).

Table 92 Configuration

Field	Configuration		
	Basic	Memory Scatter Gather	Peripheral Scatter Gather
dst_inc	User configurable	Memory Scatter Gather	Peripheral Scatter Gather
dst_size	User configurable	2u	2u
src_inc	User configurable	2u	2u
src_size	User configurable	2u	2u
dst_prot_ctrl	7u	7u	7u
src_prot_ctrl	7u	7u	7u
R_power	7u	2u	2u
n_minus_1	User configurable	User configurable (N multiple of 4)	User configurable (N multiple of 4)
next_useburst	0u	0u	0u
cycle_ctrl	1u	4u	6u

7.4.8 DMA cycle types

The cycle_ctrl bits in the channel control data structure controls how the controller performs a DMA cycle (see **Table 93**).

Table 93 DMA transfer types

cycle_ctrl	Description
b000	Invalid
b001	Basic DMA transfer
b010	Auto-request
b011	Ping-pong
b100	Memory scatter-gather using the primary data structure
b101	Memory scatter-gather using the alternate data structure
b110	Peripheral scatter-gather using the primary data structure
b111	Peripheral scatter-gather using the alternate data structure

7.4.8.1 Invalid transfer

After completing a DMA cycle, the controller sets the cycle type to invalid, to prevent it from repeating the same DMA cycle.

7.4.8.2 Basic transfer

A basic transfer performs a predefined number of transfers from a source location (memory or register) to a destination location (memory or register) upon one trigger request. Once the requested transfers are completed the corresponding channel deactivates itself and can trigger a DMA complete interrupt request. The interrupt must be enabled.

In this type, the controller can use either the primary or the alternate data structure (see **Table 87**).

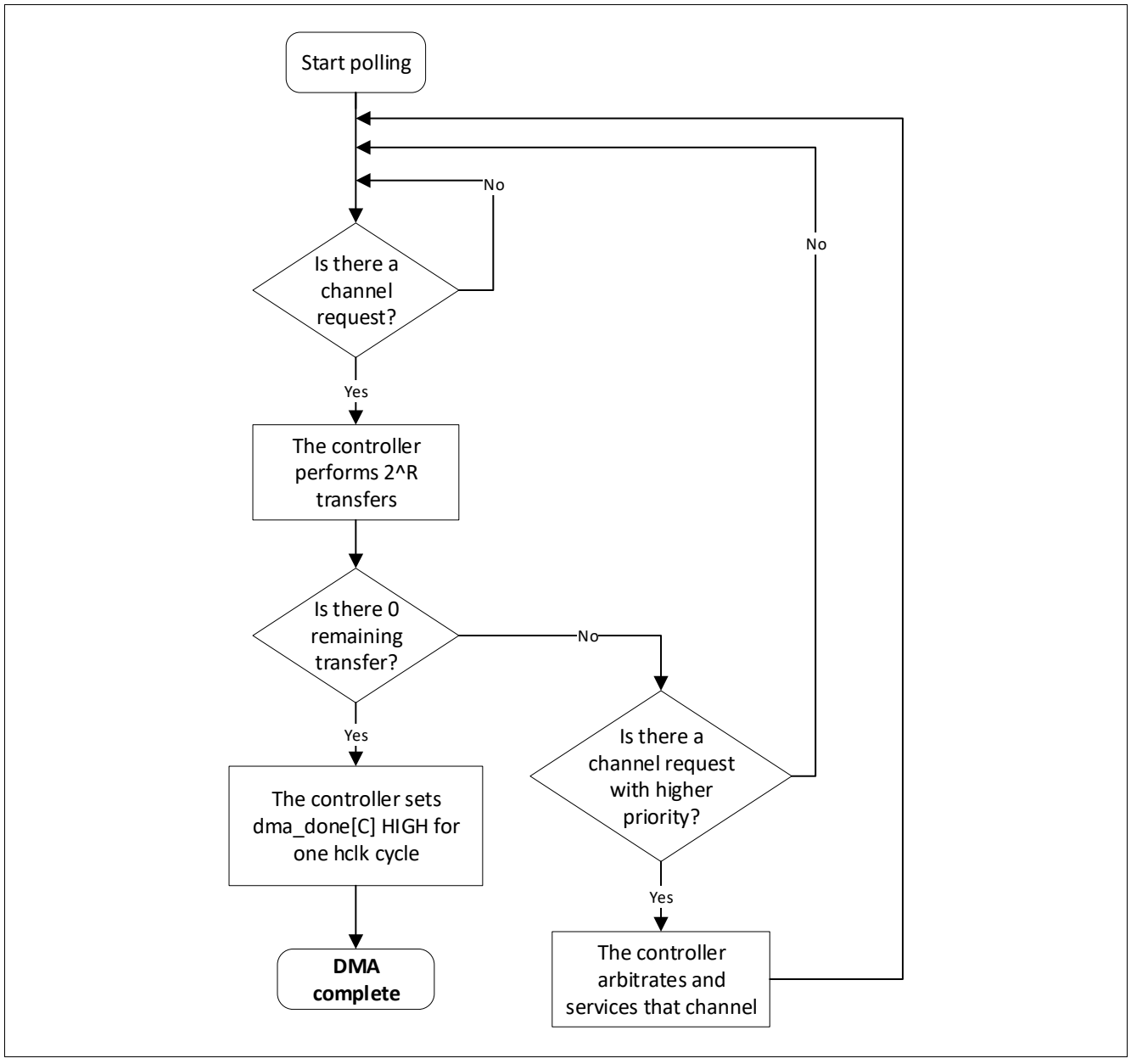


Figure 87 DMA basic transfer flow

7.4.8.3 Auto-request transfer

This mode is similar to the basic mode but the controller only needs to receive one single request to enable it to complete the entire DMA cycle.

This enables a large data transfer to occur, without significantly increasing the latency for servicing higher priority requests or requiring multiple requests from the processor or peripheral.

7.4.8.4 Ping-pong transfer

In this mode, the controller performs a DMA cycle using alternately the primary data structure and the alternate data structure. It switches between these structures until it reads an invalid data structure or until the host disables the channel (see **Table 88**).

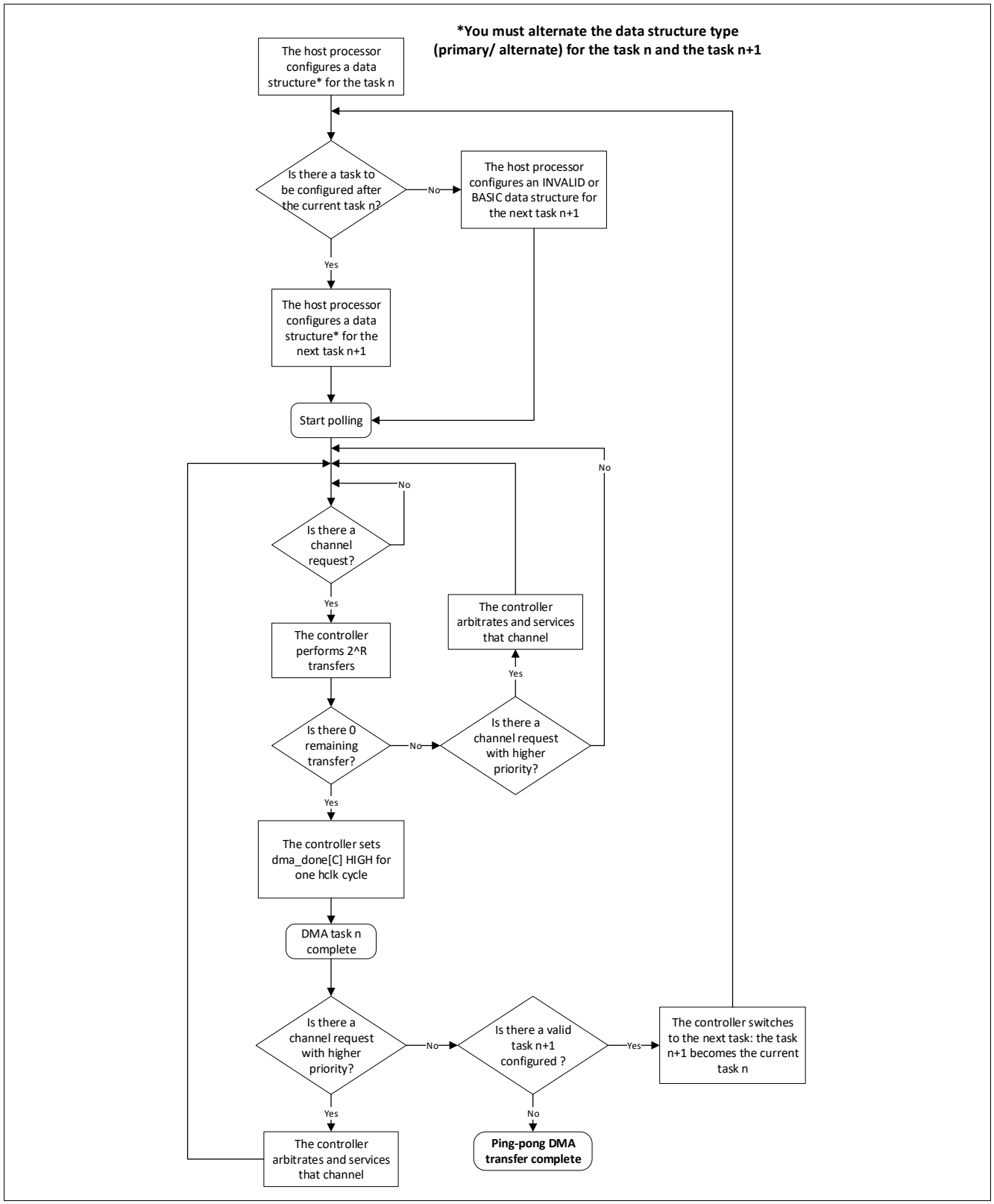


Figure 88 DMA ping-pong transfer flow

7.4.8.5 Memory scatter-gather

In this mode, the controller receives an initial request and then performs 4 DMA transfers using alternately the primary data structure or the alternate data structure.

It continues to switch from primary to alternate and from alternate to primary until the host processor configures the alternate data structure for a basic cycle or until it reads an invalid data structure (see **Figure 89**).

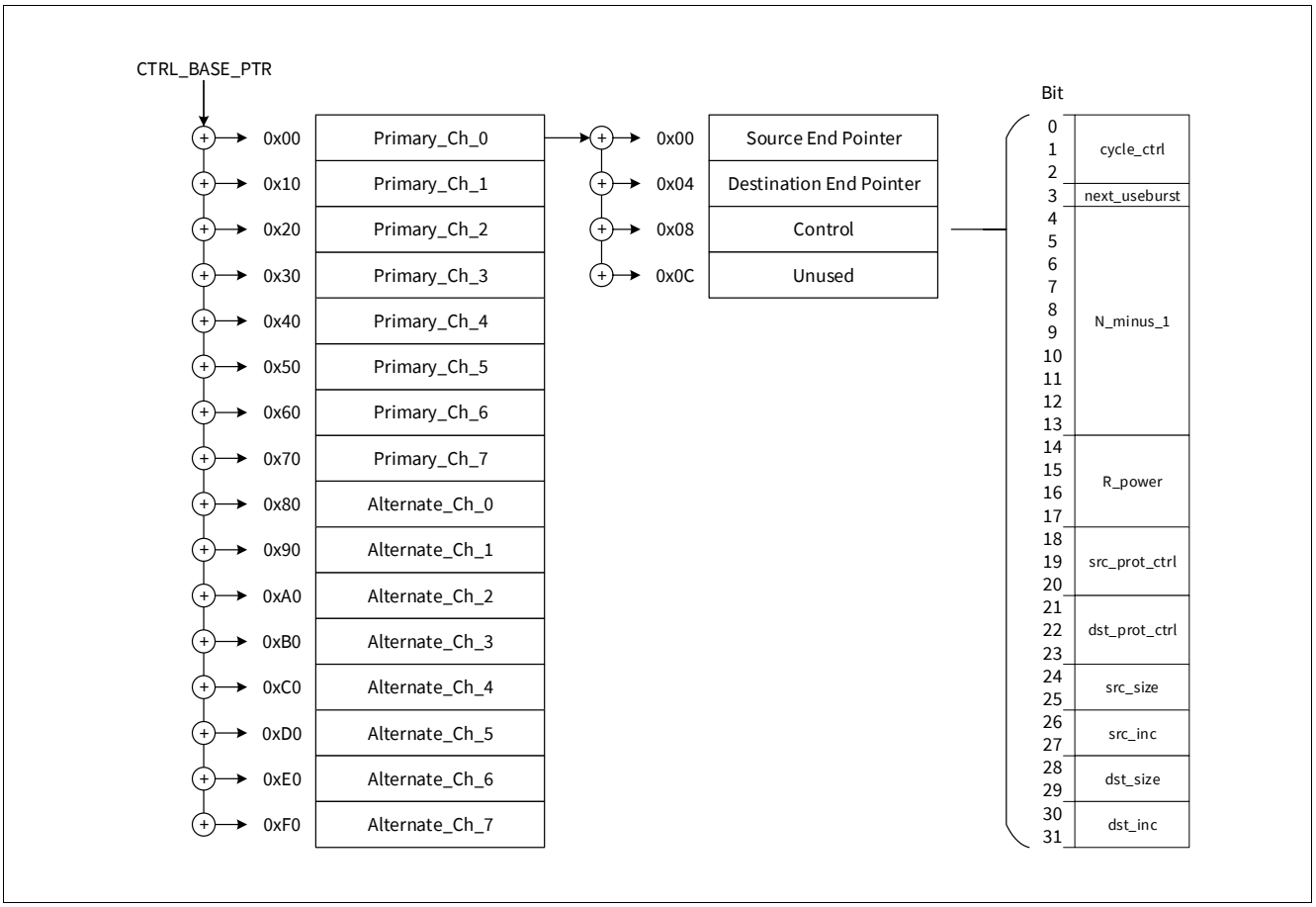


Figure 89 DMA memory scatter-gather transfer flow

7.4.8.6 Peripheral scatter-gather

This mode is similar to the memory scatter-gather mode. The differences are the following:

- there is no arbitration until the controller completes the task
- several requests must be sent in case N > 2^R

7.4.9 Error signaling

If the controller detects an ERROR response on the AHB-Lite master interface:

It disables the channel that corresponds to the ERROR it sets dma_err HIGH

The host processor can detect the faulty channel by reading the register CHNL_ENABLE_SET to create a list of disabled channels and comparing it with the record of the channels that have recently set their dma_done[] outputs. The channel with no record of dma_done[C] being set is the channel on which the ERROR occurred.

7.4.10 Auto-reload/Auto-rearm

The DMA provides the feature to reload the corresponding slot in the control structure in order to be ready to perform the same transfer job again once the DMA transfer is completed.

The auto-rearm feature enables the selected channel once the auto-reload has been performed, to make the channel ready to be triggered.

Both features can be set individually for each channel regardless of what transfer mode is being used by setting the corresponding DMACTRL.DEMEN_CHx bit within the SCU.

7.4.11 Trigger sources

The chapter describes the input sources of the DMA module. The DMA provides an individual trigger input for each channel. The trigger inputs are provided by the SCU module (for reference please see DMA request mapping chapter).

7.4.11.1 Hardware trigger

Each DMA channel trigger input is connected to a dedicated hardware resource of the device. Once an input trigger of a configured and enabled channel is actuated the set-up transfer is performed.

7.4.11.2 Software trigger

Each DMA channel can be triggered individually by software.

7.4.12 DMA Interrupts

The DMA can request an interrupt through the SCU. The registers to configure these interrupts are located within the SCU. The signal being used for the interrupt generation at the SCU/NVIC is DMACH[C] and DMATRERR. The first corresponds to the different DMA channels available. The latter can be used to raise an interrupt in case an error has occurred.

The DMACH[C] is driven by the dma_done[C] signal within the DMA, which is asserted high as the DMA has finished its transfer (one or multiple transfers). In case channel C is disabled, the DMA_SRQ[C] signal can be used as the interrupt source to trigger an interrupt indicating that a transfer request is available but the DMA channel is disabled. This routing is disabled by default and can be enabled by setting the corresponding DMACTRL.DMAINTREQEN_CH[C] bit within the SCU (see **Figure 90**).

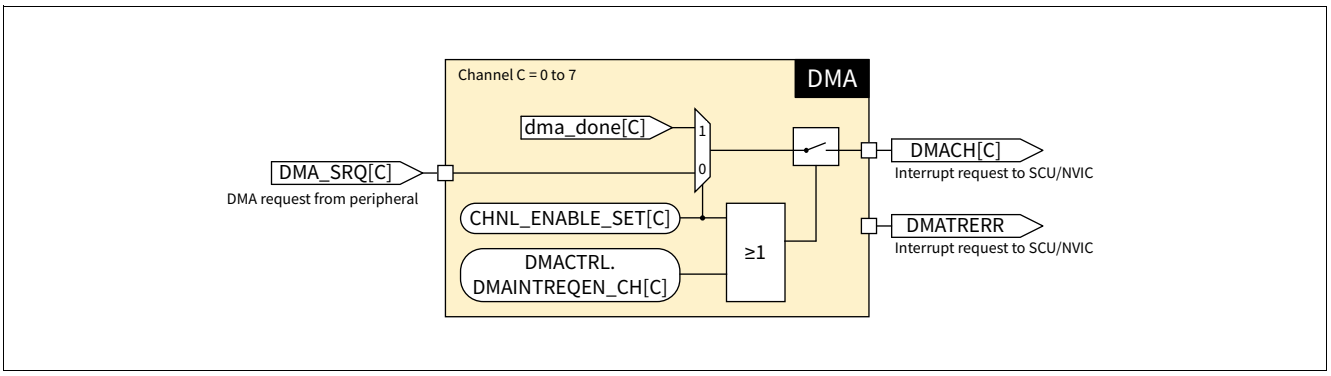


Figure 90 DMA interrupt logic (available for each channel separately)

7.5 Memory system

The memory system of the comprises a set of on-chip memories for program storage and data manipulation enabling application code execution.

This chapter contains feature description of the functional features of the individual memories.
The product specific sizes of the memories of the MCU are specified in the **Overview** (**Chapter 1**) and **Memory map** (**Chapter 3.12**).

The memory system provides the following features:

- Memory modules
  - FLASH0 for non-volatile code and data storage with ECC, and high endurance (N_{ER_high}) non-cached
  - Kbyte FLASH1 for non-volatile code and data storage with ECC, cached
  - Bytes of 100 Times Programmable Memory with ECC (100TP)
  - Byte DSRAM with ECC for data only
  - Byte PSRAM with ECC for program only
- Logical memory blocks in non-volatile memories
  - ROM for startup firmware, Bootstrap Loader and NVM service routines
  - Key Storage
  - Secured Software Container
  - Crypto Library
  - Config sectors
  - User BSL
  - Sector size of 4 Kbytes
  - Page size of 128 Bytes

7.5.1 Memory partitioning

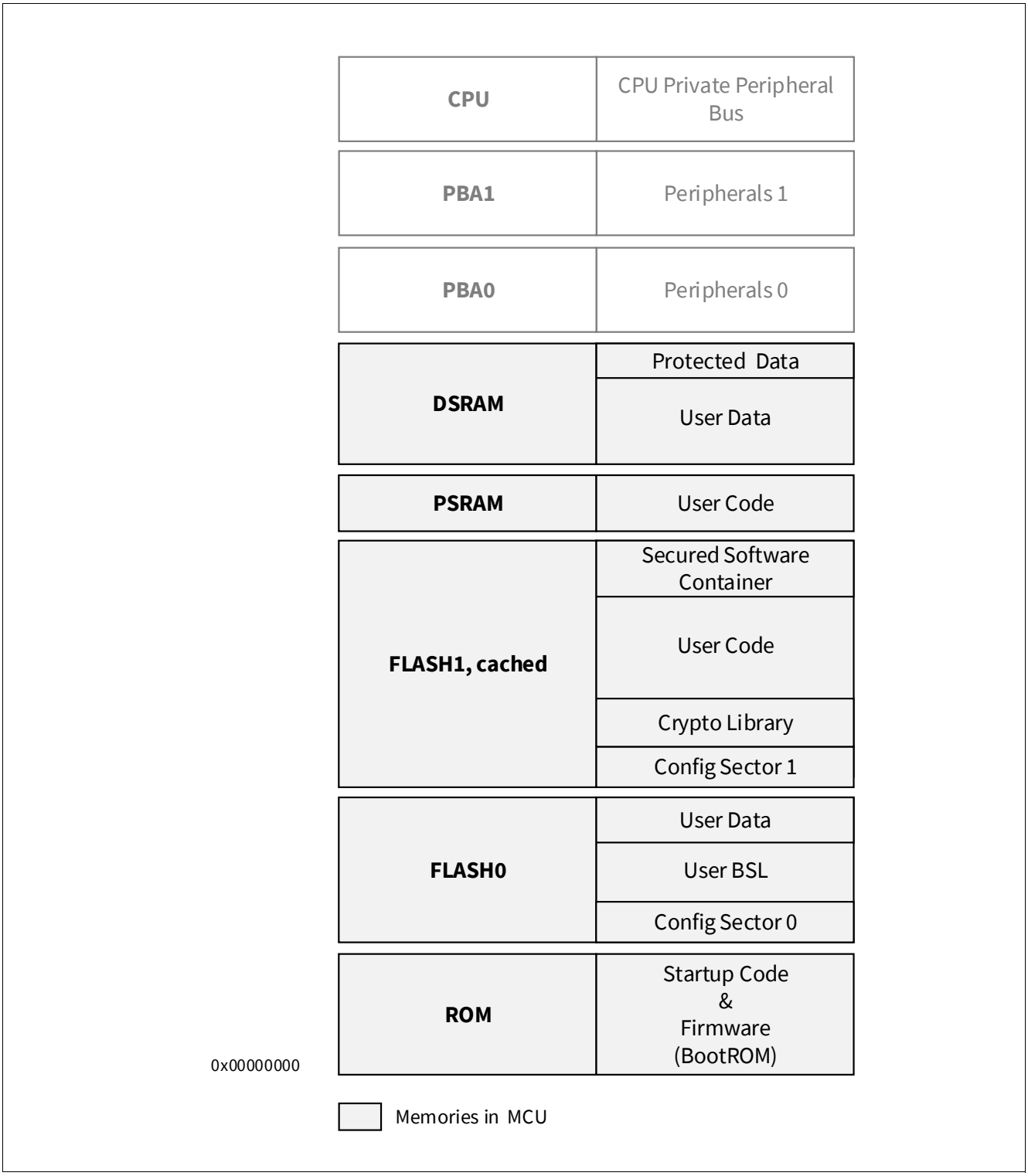


Figure 91 Memory partitioning

7.5.2 Dual-Flash access

The Dual-Flash feature provides the possibility to write/erase physical flash pages on one flash module, while the core continues to execute code from the other flash module. Writing/erasing physical flash pages within the same flash module where the core is currently executing code from is possible, but not simultaneously. Read accesses to the flash module which is under write/erase/verify operation is also not allowed.

The Dual-Flash feature allows the user to execute the following NVM operation in parallel to the code execution:

- Page write
- Page erase
- Sector erase

The BootROM firmware provides user API functions to use this so-called read-while-write (RWW) functionality. It is handled by the BootROM firmware solely. Nonetheless, the status and result of such a RWW operation can be read from the NVM_OP_STS and NVM_OP_RESULT registers. More information on the usage can be found in the firmware User Manual.

7.5.3 FLASH0

The FLASH0 memory is primarily intended for User Bootstrap Loader (UBSL) and user data storage. The FLASH0 is not cached.

7.5.3.1 FLASH0 segmentation

The FLASH0 contains several logical blocks optimized for their functions:

- **Config sector**
  - **100TP pages**
- **User BSL**
  - **Startup page and application entry**
- **User data**

For a detailed memory map please refer to **Figure 91** and **Chapter 3.12.5**.

7.5.3.2 Config sector

The config sector of FLASH1 contains device-specific configuration and trimming data which are needed during the boot up process. It also contains the security key storage.

7.5.3.2.1 100TP pages

The 100TP segment contains eight pages, where each page has a limit of 100 write accesses. The 100TP pages are partly used to hold user-adjustable calibration data for the ADCs.

The 100TP pages are accessible with write/erase and read routines via dedicated Firmware APIs in the ROM and/or device default BSL. The 100TP pages may be used to store application constant data.

For each 100TP page the user can decide whether a page belongs to the User BSL segment (accessible by the User BSL segment only) or User Code segment (accessible by the User BSL segment or User Code segment only). By default none of the pages are access-limited, they can be accessed from User BSL segment and User Code segment.

By default the page write access counter is stored within the page and is incremented by the Firmware routine upon every user write operation. The user may also write a new value to the write access counter in order to reduce the number of remaining write attempts purposely e.g. setting the remaining write attempts to zero in

order to emulate one-time programmable feature. A new write access counter value is only accepted by the Firmware routine if the new value is greater than the already stored value. If the write access counter value provided by the user is lower or equal to the stored write access count value then the default handling of the write access counter applies i.e. the counter is incremented.

The 100TP pages by default are write protected. By calling the provided BootROM user API the write protection is temporary disabled for the update process and protection is reinstalled automatically when write operation is completed.

7.5.3.3 User BSL

The User BSL segment contains the entry point for the user application software after the startup-routine completion upon boot-up. The CPU loads the initial stack pointer from the startup page (SP reset address) into the stack pointer register and is handing over control to the user application software by branching to the application entry point specified within the startup page (PC reset address).

7.5.3.3.1 Startup page and application entry

Table 94 lists the first three pages of the UBSL including the startup page as well as the addresses for user application entry and the interrupt/exception vector table.

Table 94 Memory map of the startup page and application entry

Name	Start address	Size [bytes]	Default value	Description
Page 1 (0x11000000 - 0x1100007F)				
Non-activity counter (NAC)	0x11000000	1	0xFF	Time that the BSL waits for a valid passphrase
Inverted NAC	0x11000001	1	0x00	Inverted value of the NAC
Node address (NAD)	0x11000002	1	0xFF	Node address of the device
Inverted NAD	0x11000003	1	0x00	Inverted value of the NAD
Reserved	0x11000004	4	–	–
Secure boot hash	0x11000008	16	0x00	Hash value to be checked at secure boot
Secure boot code size	0x11000018	4	0x00	Size of the code to be checked at secure boot (size in bytes)
Secure boot key ID	0x1100001C	1	0xFF	Secure key index to be used for secure boot
Reserved	0x1100001D	39	–	–
Debug handler	0x11000044	60	–	This page contains the reset and halt code needed by the debugger

Table 94 Memory map of the startup page and application entry (cont'd)

Name	Start address	Size [bytes]	Default value	Description
Page 2 (0x11000080 - 0x110000FF)				
Reserved	0x11000080	128	–	–
Page 3 (0x11000100 - 0x1100017F)				
SP reset address	0x11000100	4	Application dependent	Starting address of the stack pointer
PC reset address	0x11000104	4	0x1100 0044 (Debug handler)	Application code entry address
Interrupt/exception table	0x11000108	–	–	Start of the interrupt and exceptions table

Note: In case dualboot is used, each alternative image starts with an altered copy of the above table. See dualboot for more details.

Non-activity counter (NAC)

Waiting time is determined by (NAC - 1) × 5 ms if NAC is lower or equal to 0x1C. For a value greater than 0x1C the BSL waits forever.

Inverted NAC

If ~NAC does not match Inverted NAC, the BSL waits forever.

Node address (NAD)

The device node address determining if the BSL corresponds to an incoming BSL passphrase or not. If set to 0xFF, the device ignores the BSL passphrase NAD value.

Inverted NAD

If ~NAD does not match Inverted NAD, the NAD is assumed to be 0xFF and hence the device ignores the BSL passphrase NAD value.

Secure boot hash

This is the stored hash value which is compared to the calculated hash value when using the secure boot mechanism. In case both hash values match, the startup procedure is continued (execution of UBSL image). If the hash values do not match, the device enters the fail sleep mode.

This value is ignored if the secure key ID is set to 0xFF.

Secure boot code size

This is the length of the code to be checked by secure boot. If the specified range is within the UBSL segment, the code size to be checked is valid. In case the specified range is exceeding the UBSL segment, the code size is invalid and the device enters an endless loop. The size is given in bytes.

The User BSL code size is limited to maximum 8 KB.

This value is ignored if the secure key ID is set to 0xFF.

Secure key ID

This is the key ID of the key to be used for the secure boot mechanism. The key is used to compute the hash code of the UBSL image which is then compared to the stored hash value.

If the key ID is set to 0xFF, secure boot is disabled no integrity check is done.

Debug handler

To be able to have a reset and halt available for debugging, this debug handler is needed. In case a debug connection is requested, it keeps the CPU in a busy loop after startup until a debugger has established a connection.

SP reset address

This is the start address of the stack pointer after reset. It depends on the heap size, stack size, and static variables. Hence, it is application dependent and no default value can be given.

PC reset address

After any RESET, the BootROM firmware runs the startup-routine. The startup-routine then prepares the PC by setting it to the address of the reset handler. However, since a debug handler is needed to make a halt after reset available, the PC is set to the address of the debug handler instead of the reset handler. After the execution of the debug handler, a jump to the reset handler is performed by the debug handler.

Note: The debug handler is delivered with the SDK but is not mandatory to be used unless halt after reset is needed. The PC reset address can also be set to the application entry point (reset handler) if no halt after reset is needed.

7.5.3.4 User data

The user data segment can be used to store data that need to be updated during run-time, for example execution error log. The User Data segment provides an EEPROM emulation done by a mapping mechanism, which maps logical accesses to physical pages. The EEPROM emulation allows up to 990k logical write/erase accesses to a single User Data flash sector.

The 990 k logical write accesses can be freely distributed among the 32 logical pages of this sector. If the User Data segment is configured to contain more than one sector then each sector provides 990 k of write/erase cycles individually. The BootROM provides the necessary functionality and APIs to realize the EEPROM emulation.

7.5.3.5 Data sector handling

The NVM provides a special sector for data storage. Through a non-linear mapping of the address space, the FW and the NVM module provides a special feature to increase the maximum number of write-erase cycles a logical page can stand and to reduce the risk of data loss in case of interrupted NVM operations (tearing events).

The handling of this special Data sector requires the usage of an NVM internal look-up table (MapRAM) which is used to store and handle the link between logical and physical addresses of the sector’s pages.

Since the MapRAM is a volatile memory, the firmware takes care to rebuild the MapRAM content at each power-up based on mapping information stored into a specific field of the Data sectors pages (mapblock). This process is called Data sector initialization (MapRAM initialization).

During this initialization phase, mapping errors induced by tearing events might be found. This would then prevent the firmware from properly restoring the link between the logical and physical addresses thus preventing proper usage of this sector. In this case, the firmware provides a specific algorithm (Service

Algorithm) to identify and solve these errors. In particular, the Service Algorithm tries to repair bad pages created unintentionally into the NVM Data region due to, for example, an NVM program or erase operation interrupted by any reset or power loss (tearing events). The Service Algorithm is triggered during the startup by the NVM data sector initialization in case mapping issues are found.

The Service Algorithm provides proper analysis features to try to preserve the integrity of the NVM Data region in case ongoing NVM operation (program or erase) is unintentionally and unexpectedly aborted (e.g. due to power loss). Anyhow, it is not meant to cover all possible scenarios that can be created by an interrupted NVM operation. The user shall put in place proper action to avoid any possible interruption of NVM operation (e.g. using proper capacitor on the power supply).

The NVM Data sector initialization and Service Algorithm flows are described below.

NVM Data sector initialization

After any reset, as part of the startup, the firmware triggers an NVM initialization of the NVM data sector. This initialization is performed by a hardware state machine which takes care to restore the mapping information into the MapRAM reading specific bytes (called mapblock) of the NVM Data sector pages (see **Figure 92**). The state machine accesses these bytes and, page by page, reads out the logical page to which the current physical page has to be linked to, updating accordingly the dedicated MapRAM location.

In case a mapblock is read as erased, the physical page is not mapped. All the logical pages for which no valid mapping is found are marked into the MapRAM as unmapped.

While reading out the info from the mapblock, the hardware state machine might find incorrect mapping info. In particular, following scenarios might appear:

- More physical pages are mapped to the same logical page (double or higher mapping)
- The mapblock information cannot be read correctly due to ECC errors (faulty page)

In this case, the hardware state machine stops the initialization on the first incorrect mapping. In case of power-on reset, brown-out reset, pin reset, WDT1 reset or wake-up from sleep in addition the execution of the Service Algorithm (SA) is triggered.

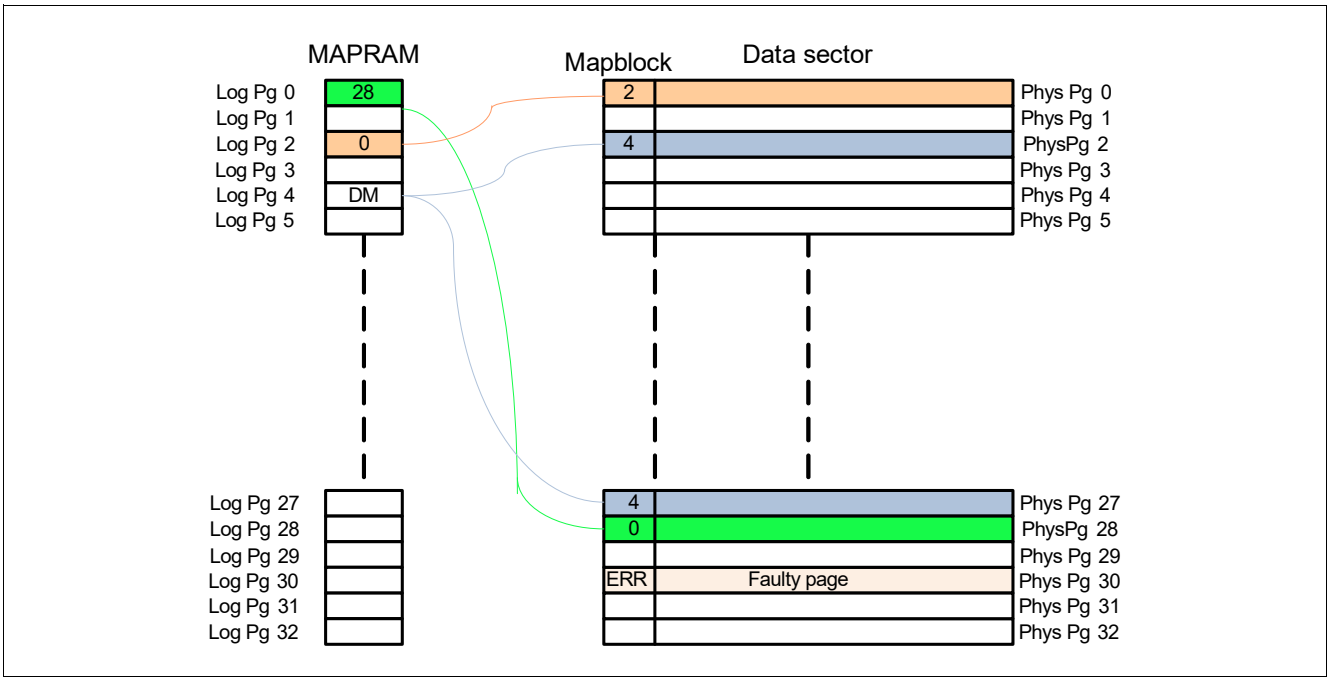


Figure 92 MapRAM and mapblocks

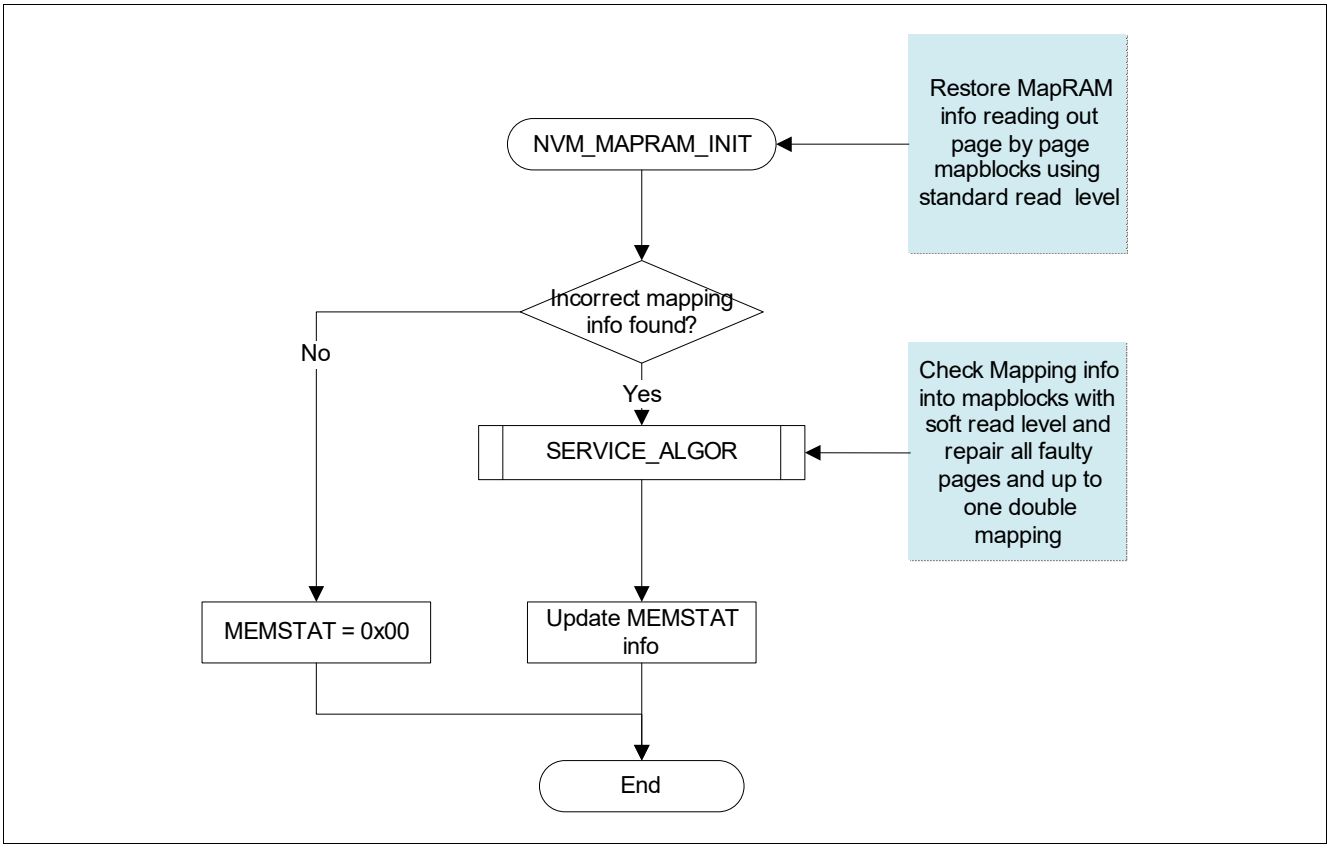


Figure 93 NVM data sector initialization flow

In order to detect pages whose mapblock is marginal towards the standard read level, the NVM finite state machine that performs the mapping initialization is triggered three times with three different read levels: standard read margin, soft read level erased and soft read level programmed. As soon as the first incorrect mapping (faulty or multiple mapping) is detected by any of these three initialization sequences, depending on the reset type, the Service Algorithm is called.

At the end of the Service Algorithm execution, a new initialization of the Data sector is needed to properly initialize the mapping info. This final initialization is again executed by triggering the NVM Finite State Machine and is performed using only the standard read level.

Note: For any reset the result of the last NVM Data sector initialization executed during the startup flow is reported to the user via the bit 1 of the SYS_STRTUP_STS register (MRMINITSTS). If this bit is set to 1 then the last initialization failed and the mapping info might be corrupted. In this case, a reset (power-on reset, brown-out reset, pin reset or wake-up reset) can be issued in order to start the Service Algorithm to try to fix the integrity issue inside the Data NVM. If the MRAMINITSTS is still flagged afterwards, the Data NVM sector has to be re-initialized by performing a sector erase.

Service Algorithm

The Service Algorithm is called by the NVM Data sector initialization in case incorrect mapping issues have been found. The Service Algorithm checks the d-ta sector page by page reading the mapblocks with soft read levels.

At first, the Service Algorithm looks for faulty pages and tries to repair them by erasing these pages. Following, the algorithm proceeds looking for double or higher mappings.

In case two or more double mappings or at least one triple or higher mapping were found the SA stops execution and reports an error on the MEMSTAT register (MEMSTAT set to $B0_H$). In case, instead only one double mapping is found, the algorithm selects which page has to be erased according to the following steps:

1. The SA checks the 2 pages linked to the double mapping with standard, soft and hard read levels to detect which one has better quality (more margin towards the standard read level. The page with smaller margin is then erased.
2. In case both pages have same quality, the algorithm checks some specific bits of the mapblock (called map counter) to check which of the pages has been programmed last. In this case, the older one is erased.
3. In case both pages have same map counter value, the SA cannot decide which page has to be erased and ends the flow reporting an error on the MEMSTAT register (e.g. MEMSTAT set to $B0_H$ for a 256 Kbyte variant).

Whenever the SA is triggered, information regarding the addressed data sector number will be stored in SECTORINFO (this is an indication that the SA was executed during the startup phase). In addition, in case the SA cannot recover all incorrect mapped pages, the SA reports a fail into the SASTATUS field of the MEMSTAT writing the value 10_B. In such a case, the user shall properly handle the reported mapping issue by either triggering a reset (power-on reset, pin reset, WDT1 reset, brown-out reset or wake-up from sleep reset) in order to trigger a new NVM initialization or to erase the whole NVM data sector to reset the mapping info.

Detailed description of the MEMSTAT register can be found in the following **Table 95**.

Table 95 MEMSTAT register status for NVM integrity handling

Field	Bits	Description
SASTATUS	7:6	**Service Algorithm Status** 00 Depending on SECTORINFO, 2 possible outcomes. For SECTORINFO = 00H: NVM initialization successful, no SA is executed. For SECTORINFO = Otherwise: SA execution successful. Only 1 mapping error fixed. 01 SA execution successful. At least 1 mapping error fixed. 10 SA execution failed. Map error in data sector. 11 Reserved
SECTORINFO	5:0	**Sector Information** At the startup, the value of this field is set to 000000_B and it is written anytime the SA is executed. This field is internally divided into two parts: Bits 5:0: NVM Class identifier 30_H: 256 Kbyte variants 20_H: 128 Kbyte variants 10_H: 64 Kbyte variants 09_H: 36 Kbyte variants Others: Reserved Once the SA has been executed, regardless of the execution status, the last access sector information will be stored here.

Note: The MEMSTAT register has a dual function. It is used to store the return value of the SA as well as input value for the NVM operations to indicate the Emergency Operation. For this reason, the user shall reset the MEMSTAT register after every power-on reset, brown-out reset, pin reset or wake-up reset before the execution of any NVM operation.

During the repair phase, pages with incorrect mapping are erased. Each page erase operation takes up to 4.5 ms.

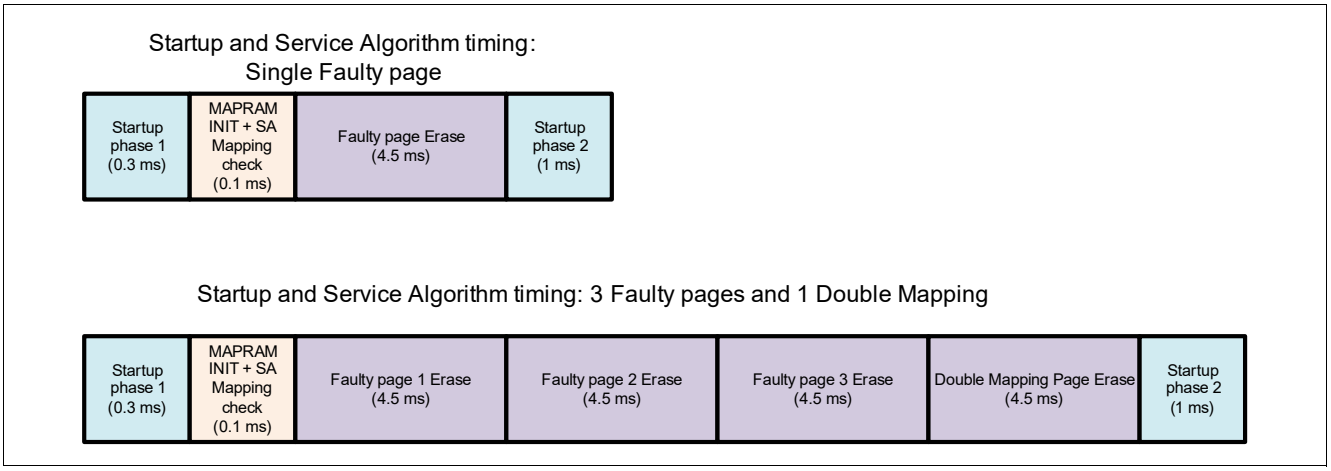


Figure 94 Service Algorithm: Timing examples

Due to the duration of the first WDT1 open window after reset (long open window), the maximum number of pages that can be repaired in one Service Algorithm execution is 13.

The result of the Service Algorithm repair phase is reported in the MEMSTAT register. At the end of the startup procedure, user shall evaluate the content of this register to properly handle fails and clear the register before performing any NVM operation.

The value is only available after reset before any NVM operation (Program, Erase, OpenAB) is started.

Service Algorithm and NVM protection

In case the Service Algorithm detects mapping issues, it tries to repair mapping by erasing the wrong pages (either faulty or double mapped pages). Consequently, the repair step can modify the NVM Data sector content. To avoid data loss, the SA checks the NVM Data sector protection and proceeds towards the repair step only if the protection is not enabled.

In case protection is enabled, instead, the repair actions are not performed and a warning is provided to the user by writing the value FE_H in the MEMSTAT register.

Via a dedicated 100TP sector parameter the user can always allow Service Algorithm to perform the repair step even in case the Data sector is protected. The control byte for this feature, CS_SA_WITH_PROT_EN, is stored into the first 100TP page. When this parameter is set to the value $A5_H$ the repair step is executed even in case protection is set. The repair flow saves the protection setting, removes temporarily the protection on the data sector, performs the needed repair operation and then restores the original protection settings. The temporary protection disabling is performed via NVM protection register setting, no access or changes to the user defined NVM protection password is performed. By default the CS_SA_WITH_PROT_EN parameter is set to 00_H (i.e. protection status is considered).

7.5.4 FLASH1

The user code segment by default is located in the FLASH1 memory. It is also recommended to place constant data in the user code segment. The FLASH1 accesses are cached.

7.5.4.1 FLASH1 segmentation

The FLASH1 memory is partitioned into the following segments:

- **Config sector**
  - **Security Key Storage**
- **User Code**
- **Secured Software Container**

For a detailed memory map please refer to **Figure 91** and **Chapter 3.12.5**.

7.5.4.2 Config sector

The config sector of FLASH1 contains device-specific configuration and trimming data which are needed during the boot up process. It also contains the security key storage.

7.5.4.2.1 Security Key Storage

The Security Key Storage is a special protected block of memory that can be accessed by dedicated Firmware routines performing cryptographic operation. Direct access from User Code is prohibited and will result in access error. For details please refer to **Chapter 7.8.2.2**.

7.5.4.3 User Code

The User Code sector is the memory block intended for user application code. It offers high performance in terms of code fetching due to caching effect and large memory capacity.

7.5.4.4 Secured Software Container

The Secured Software Container is intended for special software that can be called from the user application level but cannot be read out as data. See **Chapter 7.8.1.9** for further details.

7.5.5 PSRAM

The PSRAM memory is intended to store volatile user data, as well as volatile user code. The PSRAM is optimized for code execution but not limited to this purpose. The PSRAM is automatically reserved for the use with Secured Software when the Secured Software Container is activated.

For a detailed memory map please refer to **Figure 91** and **Chapter 3.12.5**.

7.5.6 DSRAM

The DSRAM memory is intended to store volatile user data only like e.g. stack, heap. DSRAM does not support fetching of executable code.

7.5.6.1 DSRAM segmentation

The DSRAM memory is partitioned into the following segments:

- **User Data segment**
- **Security Data segment**

7.5.6.1.1 User Data segment

The User Data segment is intended for user application relevant data i.e. user stack, user heap, user variables, constants etc. This memory is not capable of fetching executable code and attempts to execute code from this segment will result in a bus error response.

7.5.6.1.2 Security Data segment

The Security Data segment is intended for cyber protection routines data i.e. security stack, security heap, security variables etc. This memory is not accessible from the user code and attempts to execute code or access data in this segment will result in a bus error response.

7.5.7 ROM

The ROM contains the device startup code, user utility API routines and **Crypto library** entry API s. The code in the ROM is also referred to as BootROM.

7.5.8 Error Checking and Correction (ECC)

All memory modules provide an ECC mechanism in order to protect data integrity of the memory content upon read accesses. The ECC operates on memory chunks of 64 bits. For each 64 bit an 8 bit ECC word is stored inside the memory module.

- Detection up to two bit failures within each data read from the memory
- Correction a single bit failure
- Autonomous operation in the background without any user interaction
- Cannot be disabled

7.5.8.1 ECC mechanism

For NVM the ECC mechanism is able to request an NMI in case a double-bit error was detected within a memory chunk. In addition a status flag is set in an ECC status register **NMISR**. The last address of the ECC occurrence can be read by calling a Firmware routine in the ROM via dedicated API.

For RAM the ECC mechanism is able to request an NMI in case of double-bit errors and also to flag a single bit error in the **NMISR** register. The last address of the ECC occurrence can be read by calling a Firmware routine in the ROM via dedicated API.

Upon a write operation to a memory the ECC function calculates an 8 bit ECC code based on the data of the memory chunk. The ECC word is stored inside the memory module along with the user data.

Upon a read operation from a memory the ECC mechanism calculates the ECC check word for the memory chunk and compares the result with the stored ECC check word. Single bit failures will be corrected immediately. Double bit failures are detected but not corrected, an interrupt can be requested instead.

7.5.9 Memory system interrupts

The Memory system can generate various NMI exceptions and interrupts as listed in **Table 96** and in **Table 97**.

Table 96 Memory NMI sources

NMI name	NMI description
NMIDS	DSRAM ECC Double Bit Error
NMIPS	PSRAM ECC Double Bit Error
NMICD	Cache DataRAM ECC Double Bit Error
NMINVM0	NVM0 ECC Double Bit Error
NMINVM1	NVM1 ECC Double Bit Error
NMIMAP0	NVM0 Map RAM Error from MPU
NMIMAP1	NVM1 Map RAM Error from MPU
NMISTOF	User Stack Overflow

Table 97 Memory interrupt sources

Interrupt name	Interrupt description
NVM0OPC	Operation complete interrupt raised by the NVM FSM, used for RWW
NVM1OPC	Operation complete interrupt raised by the NVM FSM, used for RWW

The NMI and interrupt connectivity is shown in **Figure 95** and **Figure 96**.

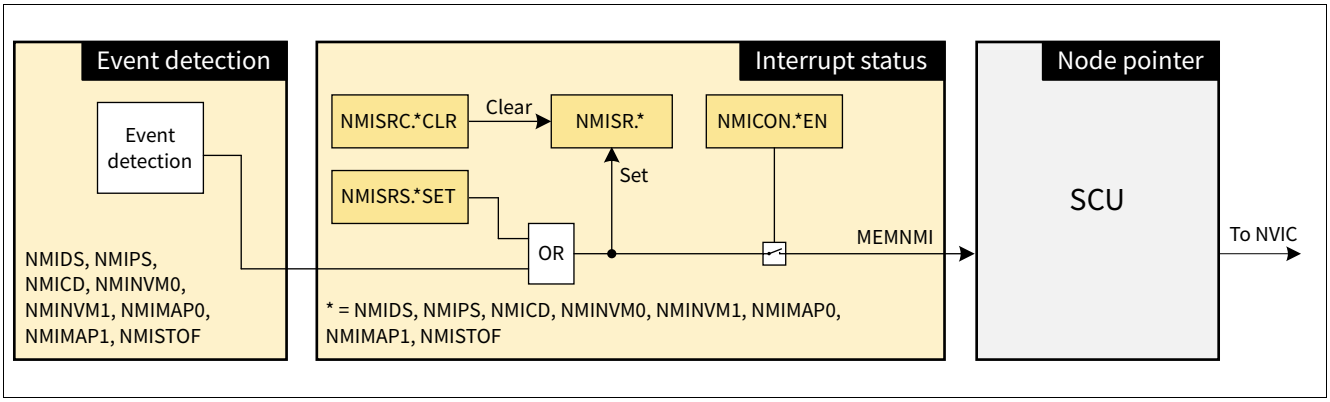


Figure 95 NMI event connectivity

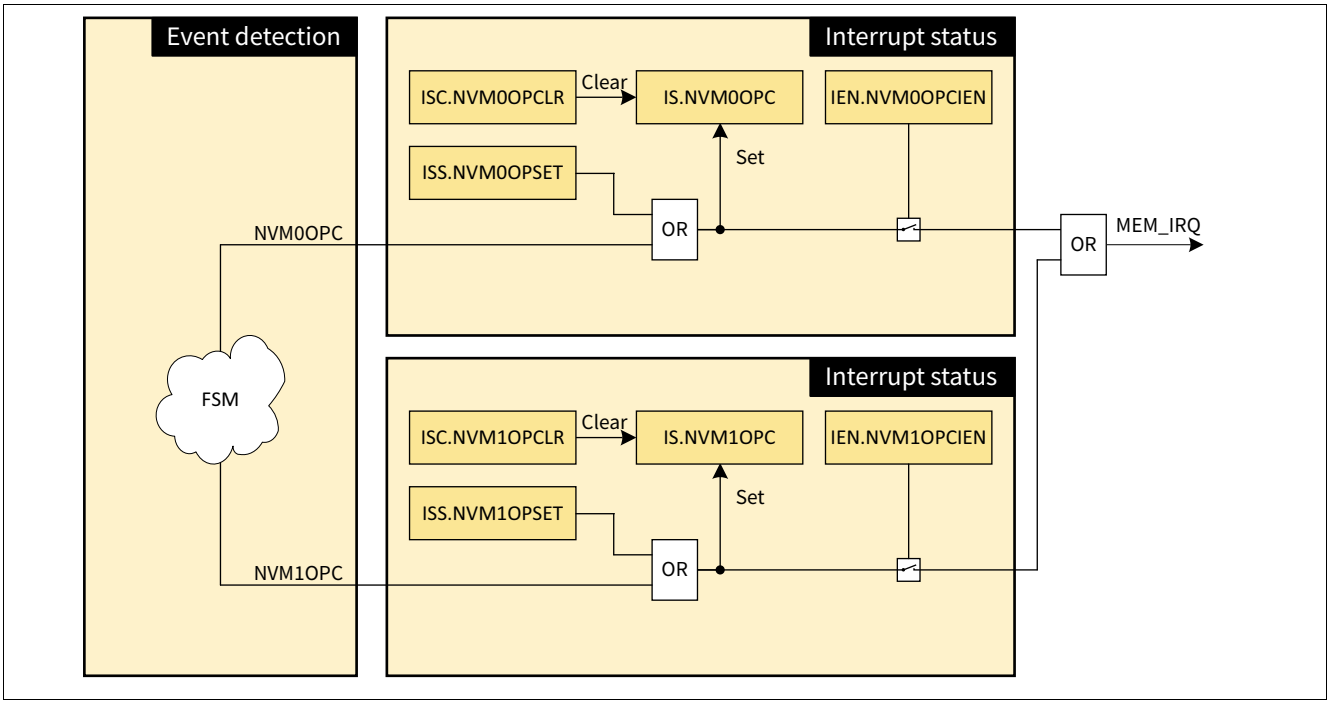


Figure 96 Interrupt event connectivity

Note: In case the RWW functionality is used, the MEM_IRQ interrupt (hard-wired to NVIC IRQ 6) needs to point to the nvm_user_isr_handler located in the BootROM firmware at 0xC1.

7.6 Cache system

The following chapter describes the cache module accelerating access to the FLASH1 module (see **Figure 97**).

The read-only cache is located between the CPU and FLASH1. It holds a temporary copy of a small portion of the FLASH1 memory contents and in average is considerably faster than uncached access to the FLASH1. In addition to its fast access speed the cache also consumes less power upon read access with respect to the power consumed upon FLASH1 memory read accesses.

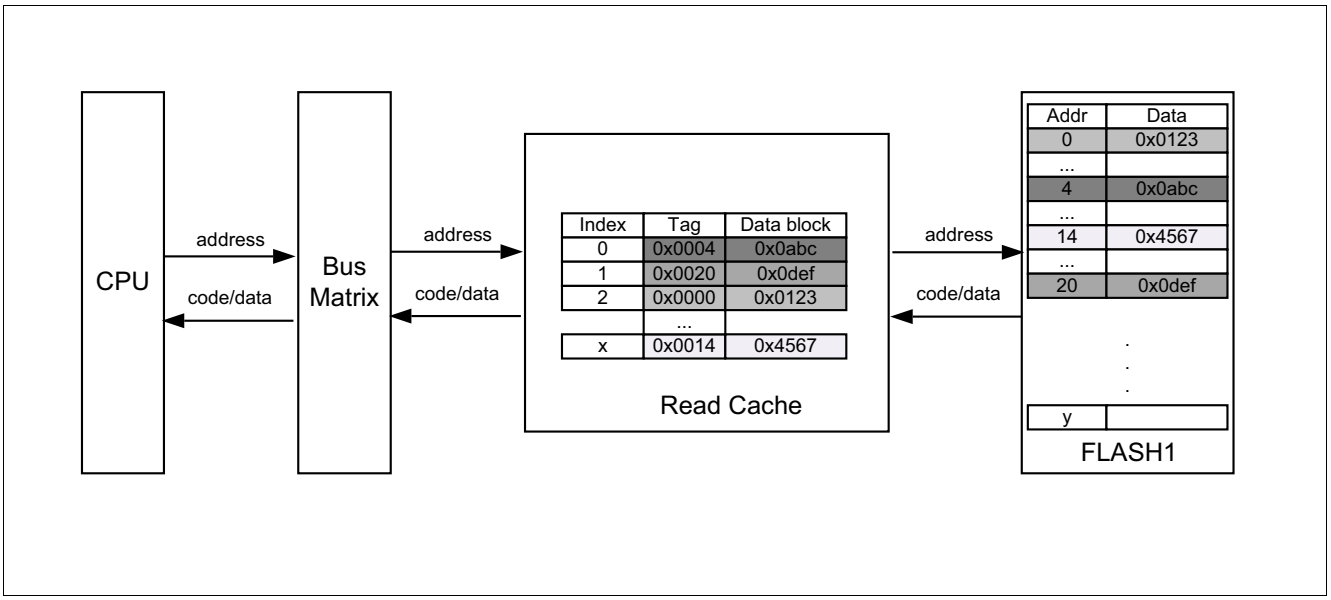


Figure 97 Cache overview

The cache provides the following features:

- Read-only
- 4-way associative
- Least Recently Used (LRU) replacement policy
- Support for *lock* and *unlock* of a part of cache
- Cache block-, set-, and all-clean (including invalidation of tags)
- Cache pre-fetching in the background
- One-cycle access for read and execute hits
- No cache line wrap penalty

7.6.1 Cache system operation

The cache system operation principle inherently depends on locality aspect. That means that programs are inclined to utilize a particular section of the address space for their processing over a period of time. The content of the cache is automatically updated according to the demand of the code or data at a given time (please refer to **Figure 97**). By efficient code and data organization the cache operation can be optimized and the system performance can be enhanced.

The cache is connected to the FLASH1 module, all code fetch and read accesses to FLASH1 are cached. The minimum data portion handled by the cache is 16 bytes, one cache block.

Cache hit

If the processor core fetches code from FLASH1 the cache controller checks whether the requested chunk of data is already present in the cache. If that is the case then a cache hit occurs, the data is read from the cache and provided to the processor with zero wait states.

Cache miss

If the requested chunk of data is not present in the cache then a cache miss occurs, the cache controller will load one cache block (16 bytes) from FLASH1 into the cache. For this operation the cache must have a free cache block available. If all cache blocks are used then the least frequently used (LRU) cache block will be discarded and filled with the new data from FLASH1. For this period of time the processor will be halted. Once the cache block has been transferred to the cache the processor will be released and the requested chunk is provided to the processor.

Cache touch

If a cache block has not been used for a long time it might be discarded by the cache controller in case a free cache block is needed. In order to prevent a cache block from being discarded it can be touched by the user by writing to **CACHE_BT** register in the cache controller which increases the access count value of the addresses cache block. Touching a cache block is a temporary effect, over time the access count value might decrease turning this cache block in a least-frequently-used block.

Cache lock

If a cache block has not been used for a long time it might be discarded by the cache controller in case a free cache block is needed. In order to prevent a cache block from being discarded it can be locked by the user by writing to **CACHE_BL** register in the cache controller. By locking a cache block this cache block will be taken out of the pool of cache blocks. This cache block will never be discarded. If the locking of a cache block is no longer needed it can be unlocked and so returned back to the pool of cache blocks available for caching. Only 50% of the cache blocks can be locked at once.

Cache operation principle

The effectiveness of a cache is measured in terms of its so-called hit ratio. It corresponds with the probability of detecting a requested address in the cache so that the corresponding content can be directly provided. The absence of a matching address entry in the cache is referred to as a miss, with the consequence that the relevant information has to be procured from the memory. The cache utilizes a replacement policy selecting the victim cache way which needs to be evicted before the newly requested line can be placed into the cache memory.

The replacement of the cache ways can be realized using different algorithms referred to as replacement policies selectable with the cache configuration registers:

- Least Recently Used (LRU) replacement policy selects the way that has been unused the longest (in a set) for replacement

Part of a requested address is compared against the tag entries for a match (hit). If found, the corresponding block in the data section returns the corresponding data of the input address. If the access results in a miss, however, a tag and the corresponding data block in the cache are then replaced by the new address and data. The cache also provides an option for locking cache entries in order to avoid replacing the data which may be useful in some situations, e.g. when the use of the data is anticipated with a high probability and/or at high repetition rate.

7.6.2 Cache architecture

Figure 98 illustrates the cache principle for detecting a specified address to return the data for its assigned address in the address space. A hit is shown in this instance.

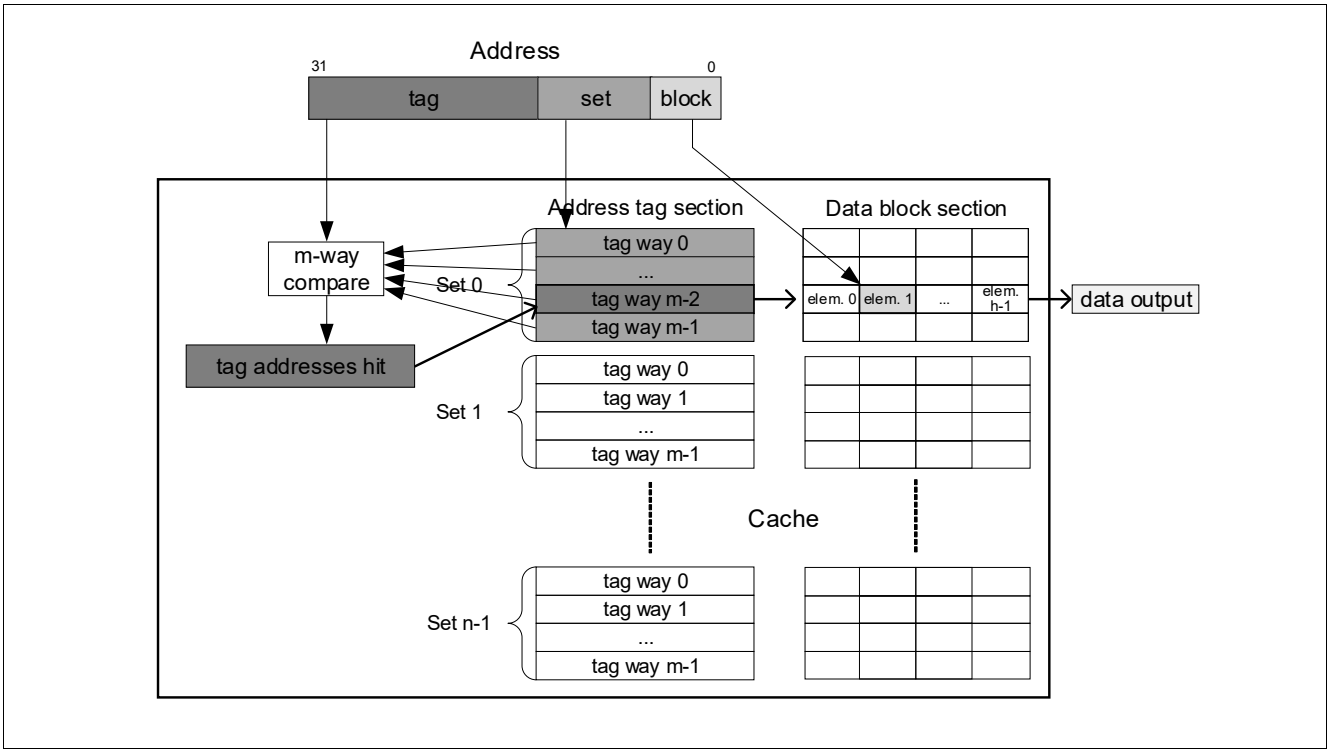


Figure 98 Cache organization

The ways in this set are then searched for a hit with the tag information using the *tag* bits in the address. If a hit is found the cache uses further the *block* bits in the address to locate the appropriate data in the block (see **Figure 98**).

The relevant data element in the data block section contains the data read from the address corresponding to the mapped address in the FLASH1 memory.

7.6.3 Cache configurability

The cache peripheral provides a number of SFR controlled functions for optimized handling of the cache. For example, the user can clean a block specified by an address, or lock a block specified by its address so that those cache entries cannot be replaced. If the lock is no longer needed an unlock function is provided. The cache SFRs are described in **Chapter 7.10.4.2**.

7.7 Multilayer Bus Matrix

The Bus Matrix enables multiple bus masters to be connected to multiple slaves. This allows a high level of concurrency in data transfers and thus a low response time and a high throughput.

- Arm® AHB-Lite protocol
- Little endian
- Support for byte- / half word- / word access
- Support for unaligned access
- Bus ERROR signaling when all bits within an AHB write access are protected (byte, halfword, word)
- Bus ERROR when all bits within an AHB write access are read only (byte, halfword, word)
- Bus ERROR if there is a read/write access to a reserved address

7.7.1 Bus Matrix interconnect topology

The TLE989x/TLE988x implements 4 master interfaces and 7 slave interfaces with 32 bit address and data buses. The details of the internal Bus Matrix topology and accessary shown in **Table 98**.

The Bus Matrix determines the master access priority criteria upon access request to a slave and establishes connection between them in order to fulfil.

Table 98 Multilayer Bus Matrix internal connectivity and access priorities[1)]

Master\Slave	ROM	PSRAM	DSRAM	FLASH0	FLASH1	PBA0	PBA1
CPU D-Code interface	0	0	0	0	0	no access	no access
CPU System interface	no access	no access	no access	no access	no access	0	0
CPU I-Code interface	1	1	no access	1	1	no access	no access
DMA	no access	no access	1	no access	no access	1	1

1) The priority is anti-proportional to its assigned value. Hence, a priority of 0 is higher than a priority of 1.

7.7.2 AHB watcher

As the Arm® Cortex® M3 cannot log the AHB address on an imprecise bus fault, the MEMCTRL logs the fault addresses in order to provide a debug possibility of bus faults. The AHB Watcher is instantiated twice in the system and allows to watch the following interfaces of the core:

- DCode interface
- System interface

The fault addresses are stored in registers **DBFA** and **SBFA**. If a bus fault occurs the corresponding status flag is set in the register **BFSTS** and can be cleared by register bits in **BFSTSC**. The status bits can also be set by the register bits in **BFSTSS**.

7.7.3 Bus Matrix interrupts

The Bus Matrix can generate exception with bus error upon access fault. The AHB Watchers store the address for which bus error occurred in the DCode Interface and System Interface in the **DBFA** and **SBFA** registers respectively.

7.8 Memory Access Control (MAC)

The Memory Access Control has the role of protection against erroneous code execution that could lead to unstable system operation. It also blocks unauthorized access attempts to protected resources on the system being result of a cyber attack. The MAC is comprised of several hardware and firmware components which provide a mechanism against cyber-attacks targeted on the Security Key Storage.

7.8.1 Memory Protection

The purpose of the Memory Protection mechanism is to prevent access to security sensitive resources like e.g. system configuration data or Key Storage from unprivileged software components. The protection mechanism is based on the execution address range and access initiator type. Privileged Firmware routines are exposed to the user software via dedicated defined API s can access the required secrets in order to perform required operation using the Security Keys without exposing the sensitive data to the user application code. For more detail on the Firmware please refer to the **Chapter 7.8.3**.

The memory protection system provides the following features:

- **Permanent protection**
- **Temporary protection**
- **FLASHx write protection**
- **FLASHx read protection**
- **User stack overflow protection (USOP)**
- **Secure Stack Overflow Protection (SSOP)**
- **Key storage protection**
- **Secured Software Container segment protection**
- **Debug access protection**

7.8.1.1 Permanent protection

The permanent protection will be installed during the boot up process by the BootROM before execution of the user code starts and/or a debugger attempts to connect via SWD interface.

The permanent protection cannot be modified by the user application code (no user API).

The permanent protection can only be removed via the device internal BSL mode.

Permanent protection features

- Handles write protection and read protection for the given NVM segment at once.
- Set by executing a device internal BSL mode command with an NVM segment individual pass phrase.
- Allows to set an individual Erase flag for each protected NVM segment. If the Erase Flag is set, in case of protection removal this segment gets erased. If the Erase Flag is not set, in case of protection removal this segment does not get erased.
- Removes the write and read protection by executing a device internal BSL mode command with the NVM segment specific pass phrase. All segments of the same or a lower privilege level are getting erased. If the pass phrase is correct, the segment gets protected and the Erase flag is set.
- Blocks any debugger connection through SWD if protection is set for the given NVM segment.
- Blocks any device internal BSL command which can be used to download code to the device if the protection is set for the NVM segment.
- Blocks any device internal BSL command which can be used to upload code from the device if the protection is set for the NVM segment.

- Prevents protecting the NVM segment if the higher ranked (according to the privilege level) NVM segments is not protected.

7.8.1.2 Temporary protection

A temporary protection is a protection set or removed during run-time. It applies to write protection avoiding accidental overwrite of the content. It only changes the protection status of a given segment for a given session. It does not alter the state of the permanent protection.

Temporary protection features

- Does not alter the Permanent Protection of any NVM segment
- Allows setting the write protection individually for each NVM segment (User BSL, User Data, User Code), obeying the Segment access privileges, see **Chapter 7.8.2.1**
- Allows clearing the Write Protection individually for each NVM segment (User BSL, User Data, User Code), obeying the Segment access privileges, see **Chapter 7.8.2.1**
- Can be set through BootROM user API calls
- Can be cleared through BootROM user API calls
- BootROM user API calls are protected against unintended calls

7.8.1.3 FLASHx write protection

The FLASH0 and FLASH1 write protection can be set individually for each segment. It prevents the intentional or unintentional modification of a protected segment. It can be set or cleared either by a user API or by the device internal BSL mode.

FLASHx write protection features

- Can be controlled by the Permanent Protection, see **Chapter 7.8.1.1**
- Can be controlled by the Temporary Protection, see **Chapter 7.8.1.2**
- Blocks any write access to an NVM segment (User BSL, User Data, User Code) if the NVM segment is write protected

7.8.1.4 FLASHx read protection

The FLASH0 and FLASH1 read protection can be set individually for each segment. It prevents the content of a protected segment from being read by other segments, according to the access rights. Once the read protection is set for any segment the serial wire debug (SWD) port is disabled, the device will no longer establish a connection to a debugger.

FLASHx read protection features

- Controlled by the Permanent Protection, see **Chapter 7.8.1.1**
- Blocks read accesses to any target segment according to the segment access privileges, see **Chapter 7.8.2.1**

7.8.1.5 User stack overflow protection (USOP)

The User Stack Overflow Protection allows the User to configure a Guard Band region which enables detection of the stack overflow within the DSRAM memory.

A stack overflow occurs when the stack pointer exceeds the stack assigned address space. When a stack overflow occurs the stack access might point to unused address space or might corrupt values stored in other reserved region. In both case, it typically results in an unpredictable behavior.

In the register **STACK_OVF_ADDR** user can define a DSRAM address range to be used as Guard Band. This Guard Band region should be at the end of the DSRAM region reserved for stack (see **Figure 99**). The detection logic monitors each DSRAM write access. In case a CPU driven DSRAM write access targets an address inside the guard-band region a stack overflow detection flag is set (see **NMISR**.NMISTOF description) and a NMISTOF flag is triggered. The stack overflow detection flag can be cleared via the register **NMISRC**.

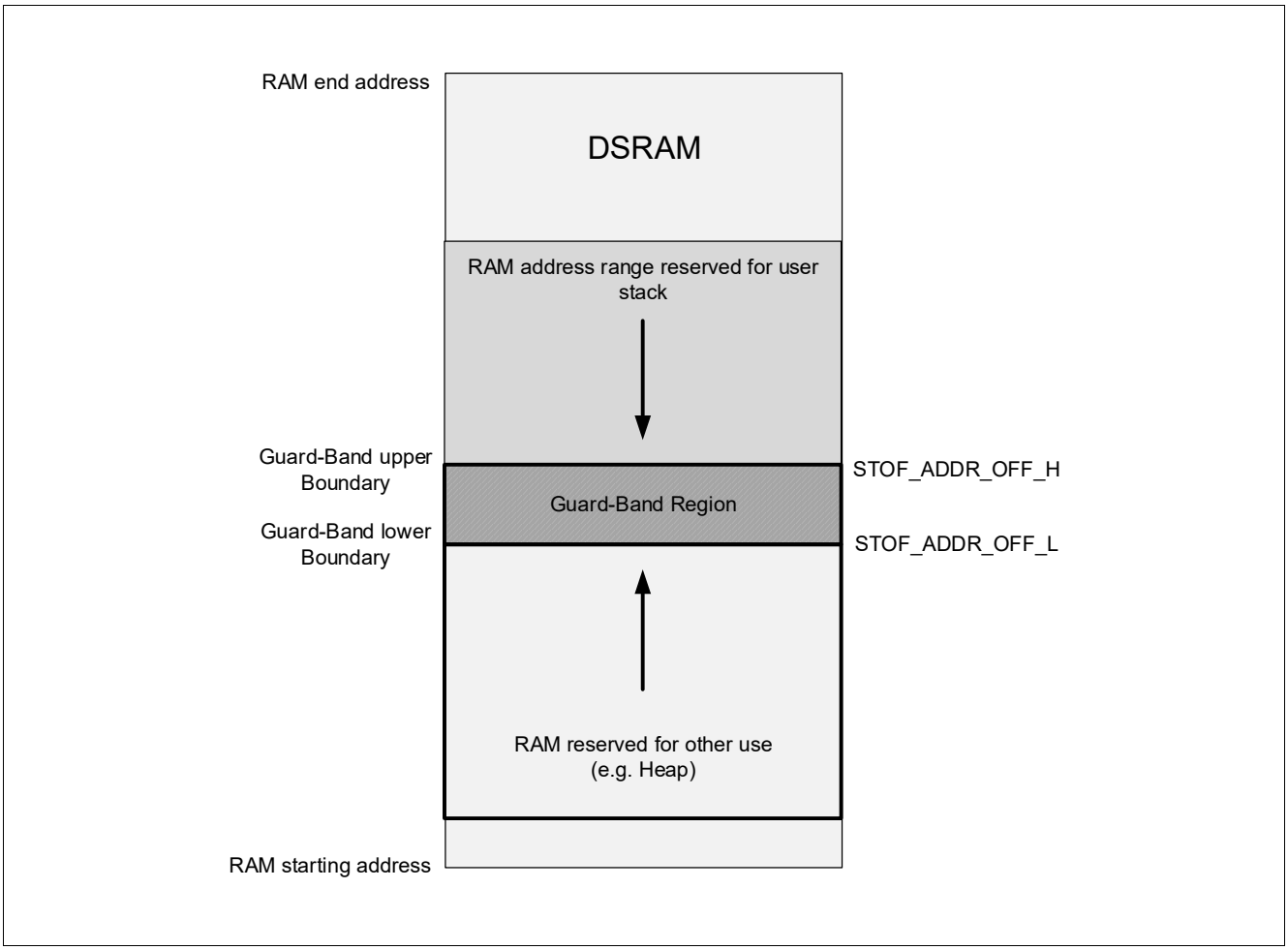


Figure 99 Stack overflow address boundary

The stack overflow detection logic does not flag any status bit or NMI in case the write access is triggered by the debugger.

7.8.1.5.1 USOP interrupts

The USOP can generate NMI interrupt to NVIC upon guard-band region access attempt.

7.8.1.6 Secure Stack Overflow Protection (SSOP)

The Secure Stack Overflow Protection allows the Firmware to configure a Guard Band region which enables detection of the stack overflow within the secure DSRAM memory segment holding dedicated stack accessible by the Firmware only.

A stack overflow occurs when the stack pointer exceeds the stack assigned address space. When a stack overflow occurs the stack access might point to unused address space or might corrupt values stored in other reserved region. In both case, it typically results in an unpredictable behavior.

7.8.1.6.1 SSOP Interrupts

The SSOP does not generate NMI interrupt to NVIC upon guard-band region access attempt. Instead reset of the microcontroller is activated. The source of the reset must be examined for of debugging.

7.8.1.7 Security library segment protection

The security library segment is intended to store the crypto-library, see **Chapter 7.8.3.3**. The security segment has the same privilege level as the BootROM user functions, including the same protection. Execution is permitted via dedicated API calls.

Security library segment protection features

- Blocks read accesses to Security segment from segments with insufficient privilege level, excl. self-reading. A bus fault is triggered in case of illegal access attempt
- Only updateable by test program
- Utilizes protected RAM that can only be used for security sensitive data which is protected against exposure to the user or any user routines
- The Security Segment container for the Crypto Library is located in NVM1

7.8.1.8 Key storage protection

The key storage is protected against uncontrolled access from user address space. All accesses to the key storage are controlled by dedicated BootROM routines. Any access including execution to the key storage from user mode will cause a bus fault.

Key storage protection features

- Rewrite protection
- Read protection
- Boot key protection
- Failure Analysis Request (FAR) management protecting user keys integrity

Key write protection

The key storage by default is write protected.

An empty slot is not protected and it can be written by using dedicated BootROM API. When writing a key the corresponding rewrite protection is set. It effectively defines whether a key slot can be rewritten.

7.8.1.9 Secured Software Container segment protection

The Secured Software Container (SSC) segment may contain libraries of special algorithms, complete or in parts, or, any 3rd party software library. Though the SSC segment has the same privilege level as the user code

segment, none of the user codes (user BSL, user CODE, PSRAM) is able to read the SSC segment and any read attempt will cause a bus fault. Write attempts to the SSC segment originated from any user segments are blocked. Execution of functions inside the SSC segment from any user segment is permitted. The MCTRL library in encrypted form can only be updated using a dedicated API of the firmware.

Secured Software Container segment protection features

- The Secured Software Container is protected against data accesses and unintended exposure to untrusted code. Data accesses initiated by program parts executed from other regions (i.e. not from the Secured Software Container itself) are blocked. The only exception are BootROM API routines required for secure update and maintenance of the of the Secured Software Container. A bus fault is triggered on illegal access attempt.
- The Flash Protection enables a permanent write protection of the Secured Software Container segment. The BootROM can temporary remove the protection for update purpose.
- The Secured Software Container segment can only be updated by user API provided in the BootROM. Use of a key is required for this operation.
- The device implements a protected RAM for execution of the Secured Software Container code. The Secured Software Container code can be executed in the protected mode from PSRAM.

7.8.1.10 Debug access protection

The debug interface operation is restricted to user application code and data. The protected parts of Firmware and data cannot be accessed from the debugger level. Any attempt to perform a forbidden action will be ignored. That includes but does not limit to:

- Breakpoints during execution of the Firmware
- Debug stepping during execution of the Firmware
- Exposure of the CPU general-purpose registers content during execution of the Firmware
- Debug watchpoints set in the protected memories
- Memory dumps of the protected memories

7.8.2 Trusted Gate mechanism

The TLE989x/TLE988x device provides mechanisms supporting protection against electromagnetic interferences as well as cyber security threats. The Protected Context (see **Figure 100**) is intended for utilization of cyber protection related features during run-time while safeguarding direct access initiated from the User Context.

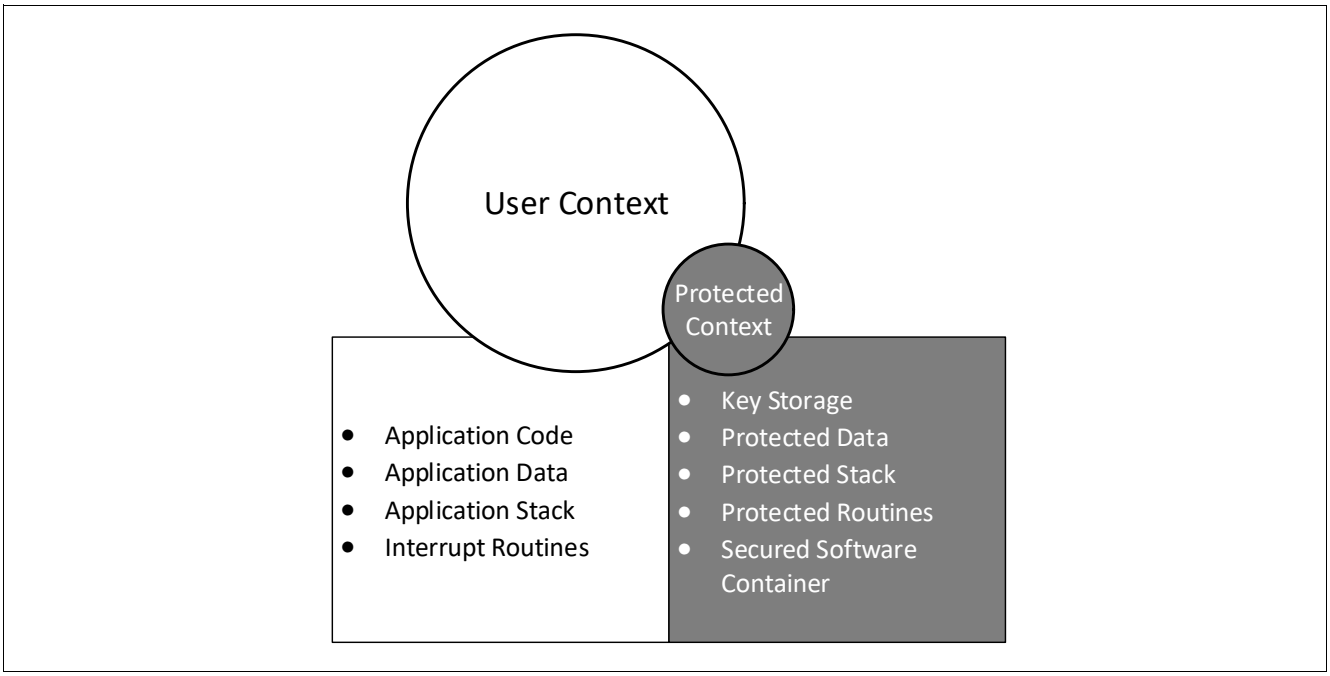


Figure 100 Protected assets

The switching between User Context and Protected Context strives to shield the critical resources from direct alteration caused by cyber attack attempts or by electromagnetic interference. The Trusted Gate mechanism utilizes privileged firmware routines, which offer various services via APIs. In combination with a hardware mechanism, it provides a logical separation between different resources (see **Figure 101**). Execution of the privileged routines can be initiated via dedicated entry vectors and any attempt to bypass the entry points will result in an error exception. This rule applies also to the User Interrupt Service Routines (ISR) triggered during execution of the Protected Code.

Note: Any interrupt triggered while execution in Protected Content has added total latency of $t_{cntxlat}$ due to the procedures of context switching.

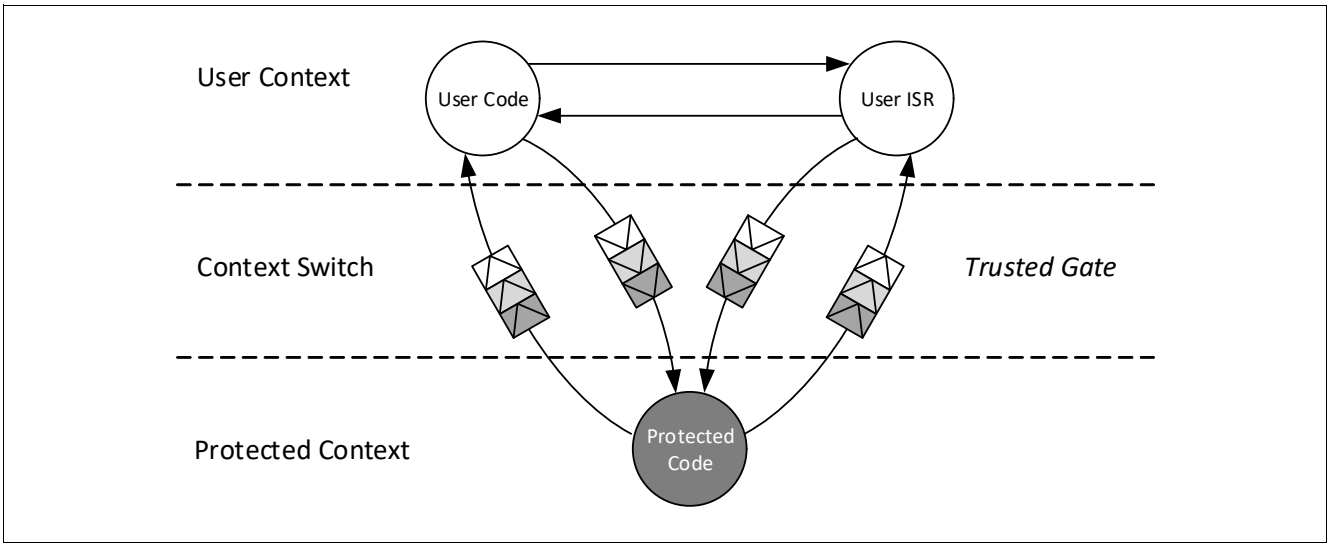


Figure 101 Context switching with Trusted Gate

7.8.2.1 Segment access privileges

The various segments are ranked with a given privilege level determining direct access right of the requestors to their content. The higher the ranking the higher the privilege level (see **Table 99**). According to the privilege level of the given segment the code execution and/or data access is allowed when an access request is initiated to a segment of equal or lower Privilege Level. The memory accesses according to the assigned Privilege Levels of the requestor and the target segment are controlled with the **Memory Protection**.

Accesses from segments with a lower privilege level to segments with a higher privilege level can also be performed via special firmware APIs, as illustrated in **Figure 102**. For the references to the Firmware API info please refer to **Chapter 7.8.3**.

Table 99 Privilege levels of the segments

Privilege level	Segment	Notes
7	BootROM Startup Code	read-protected, execute-protected
6	BootROM User Functions	read-protected
5	Key Storage	read-protected, write-protected
3 or 4	User BSL	privilege level can be set to 3 by BSL, default is 4
3	User Code, PSRAM	user-accessible
2	User Data	user-accessible
1	DSRAM	user-accessible
0	BootROM User API Entry Vectors	read-protected, the User API Entry vectors are accessible for execution

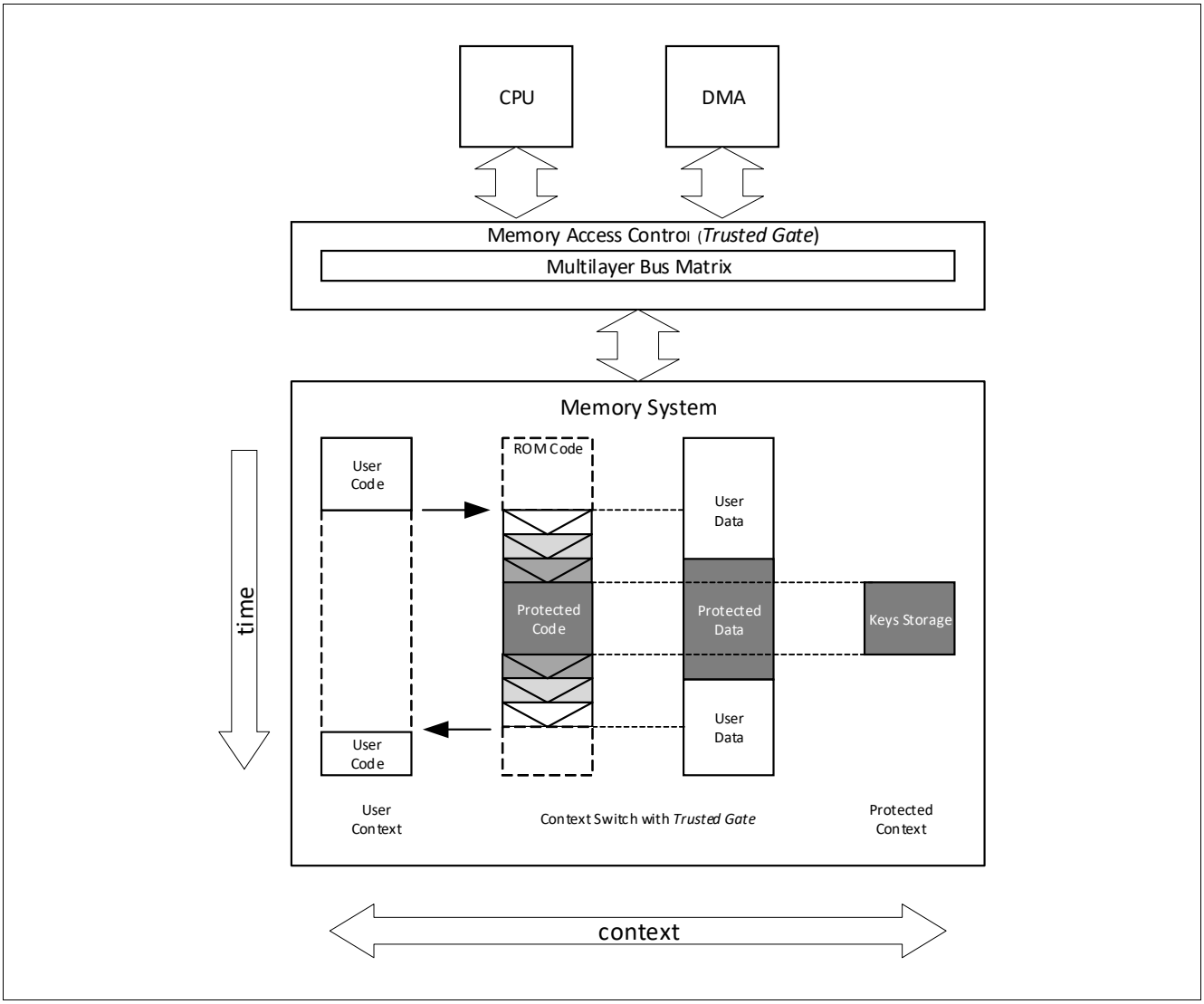


Figure 102 Memory Access Control with Trusted Gate

Cyber-protection support features

- Allows execution (call) of segments with the same or lower privilege level as/than the caller
- Denies execution (call) of segments with a higher privilege level than the caller, if the target segment is read protected
- Allows reading of data/code from segments with the same or lower privilege level as/than the caller
- Denies reading of data/code from segments with a higher privilege level than the caller, if the target segment is read protected
- Allows altering the temporary write protection state of segments with the same or lower privilege level as/than the caller
- Denies altering the temporary write protection state of segments with a higher privilege level than the caller
- Grants access (call) to the BootROM User API Entry Vectors from any segment, regardless of its privilege level
- Grants access (read/write) to the Key Storage (privilege level 5) from privilege level 6 and 7
- Denies any access to the Key Storage (privilege level 5) from any segment of privilege level 5 or lower

- Allows the User BSL segments privilege level to be changed to level 3, same as User Code
- Hard-fault exception is requested upon access denial

7.8.2.2 Key storage management

A dedicated sector of the FLASH1 module can be used to configure the key storage. If the key storage is activated it is not addressable by user code memory operations anymore. All accesses to the key storage are controlled by the BootROM via an user API. The BootROM provides an user API to access the key storage. The key storage is organized in key slots, in which symmetrical keys can be stored.

Key storage features

- Up to 13 key slots
- Key slot length of 256 bits each
- Located in the config sector of the FLASH1 module
- Write protected by hardware
- A regular write/update access removes the write protection temporarily
- An irregular write/update attempt results in a bus fault

7.8.2.2.1 Key slot content

Each key slot contains the following information:

- Security Key
- Key version
- Protection configuration

7.8.3 Firmware

The Firmware located in different non-volatile parts of the Memory System serves several purposes:

- Startup sequence including hardware configuration and trimming
- User utility routines exposed via dedicated APIs
- BSL support
- Test & Diagnostics routines

For details of the Firmware features please refer to the Firmware User Manual.

7.8.3.1 Secure boot

If enabled, the secure boot mechanism provides an automatic integrity check of the designated UBSL image at device startup. The integrity check can be applied to the default BSL image or to another image making use of the dualboot feature.

To enable the secure boot, the startup page of the designated UBSL image needs to contain a valid Secure Boot Key ID other than 0xFF. Writing 0xFF into the Secure Boot Key ID disables the secure boot mechanism. If secure boot is enabled, the BootROM firmware expects the Secure Boot Code Size and the Secure Boot Hash to be stored in the startup page as well.

In case secure boot is enabled, the BootROM firmware starts the secure check at device startup. The secure boot is done as follows (depicted in **Figure 103**):

1. BootROM firmware checks if the code size is valid, meaning that the code to be checked resides within the UBSL:

- (Image start) + 0x100 + (secure code size) < (UBSL end)
- If the code size is invalid, the device transitions in an endless loop (waiting for the watchdog expiration in case of User Boot Mode)

2. The key specified by the secure boot key ID is read out of the key storage
3. Firmware calculates the hash value of the specified UBSL address range:
   - Address range start at: (image start) + 0x100
   - Address range end at: (image start) + 0x100 + (secure code size)
   - The hash is calculated with the CMAC algorithm using the loaded secure boot key
4. The calculated hash value is compared to the Secure Boot Hash stored in the startup page:
   - If the hash values match, the firmware proceeds to execute the UBSL image (branch to application)
   - Otherwise, the device transitions in an endless loop (waiting for the watchdog expiration in case of User Boot Mode)

*Note: The time needed for the secure boot consumes time of the FS_WDT Long Open Window. This has to be considered when using secure boot and limits the maximum code size to be secured to 8 KB. Refer to the **Chapter 3** "**Product definitions**" for more information.*

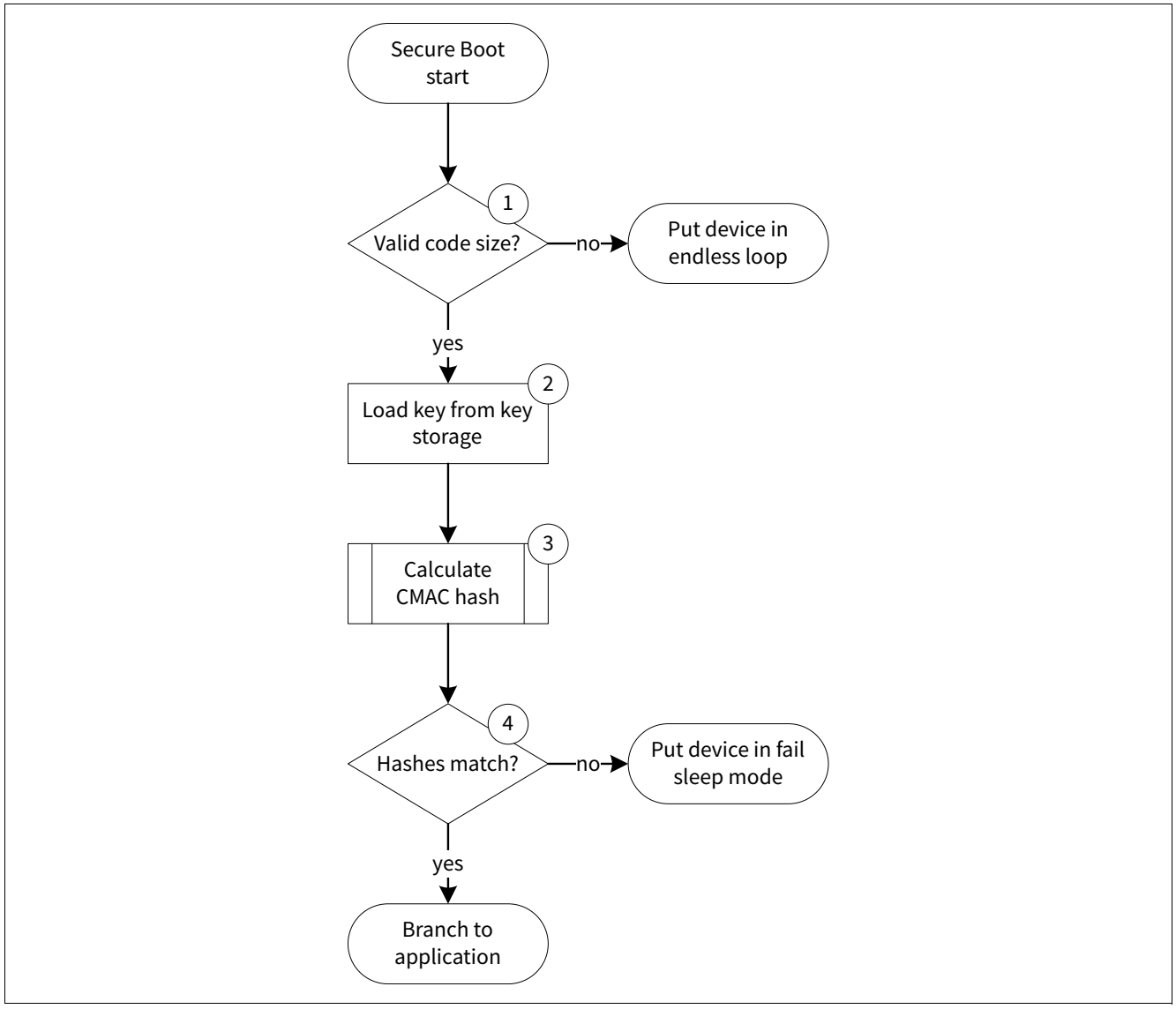


Figure 103 The secure boot flow (the numbers refer to the description above)

The image start for the default UBSL image is always 0x11000000. For other Dualboot images, the start address is specified within config sector of FLASH0. It can be set by using the related BootROM firmware API (see **Dualboot** for more information). The offset of 0x100, which is added to the image start address, is used to exclude the startup page from the hash calculation. Hence, the hash calculation starts at the user vector table which stores the different addresses for resets, interrupts, and exceptions, followed by the application code. The complete data flow can be seen in **Figure 104**.

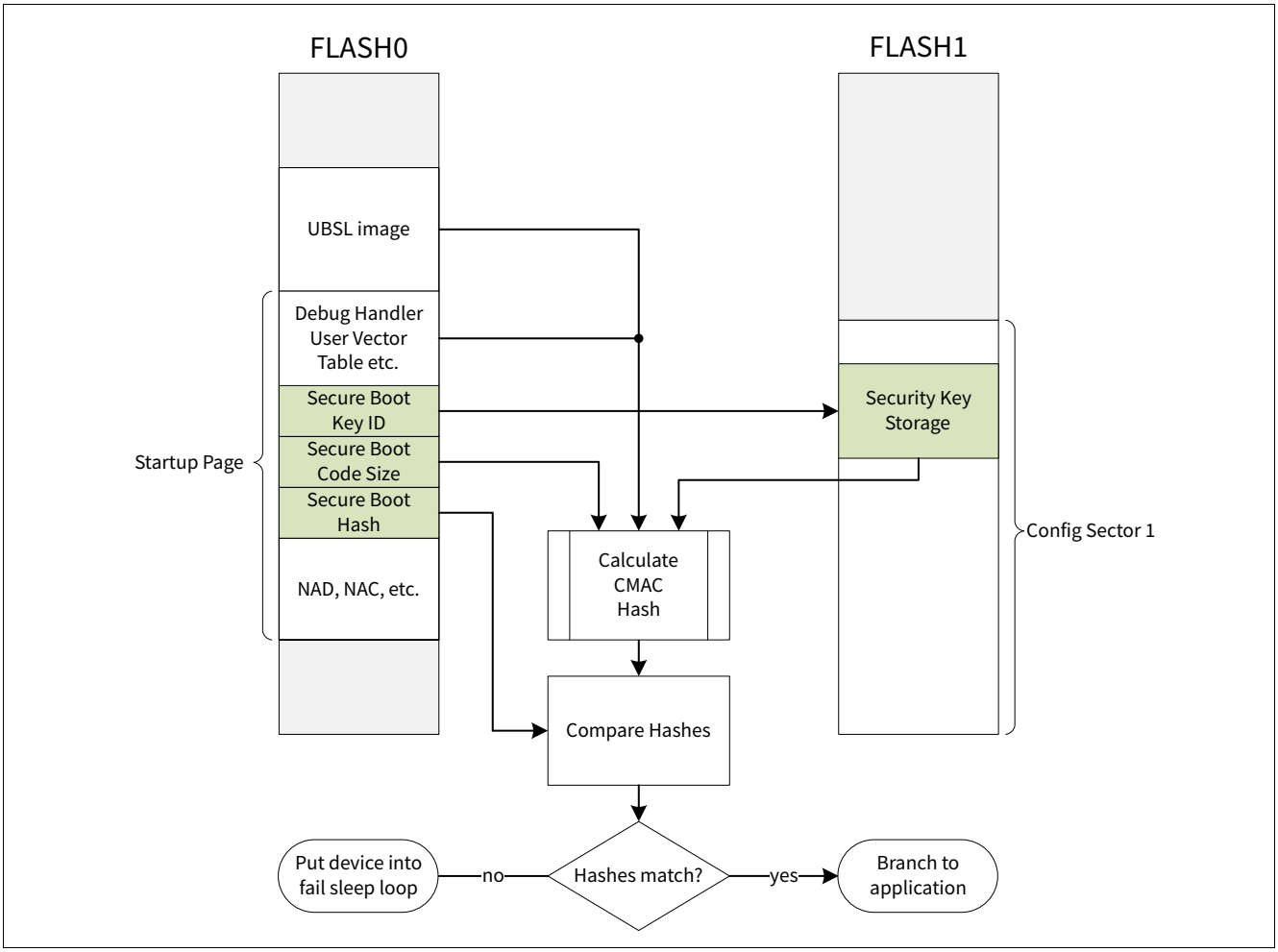


Figure 104 Secure boot data flow

*Note: The depicted memory section sizes in **Figure 104** do not reflect the real memory sizes in bytes.*

7.8.3.2 Dualboot

The TLE989x/TLE988x comes with a dualboot feature. The dualboot can be used to enable a second bootable image to be executed after device startup. By this, it is possible to change between different images to switch for example between a user bootloader and the user application.

To make use of the dualboot functionality, an additional image has to be stored in the UBSL. This additional image will also need its own startup page and user vector table (see “Startup page and application entry” for more information). Once everything is set up, a dedicated BootROM firmware API function can be used to prepare the device to execute the additional image after the next device reset. The API function to perform this preparation is called user_secure_dualboot(uint32_t image_offset). See the firmware user manual for more information.

The address offset provided as input parameter to the API function is added to the UBSL base address (0x1100000) to determine the address of the dualboot image, which is then stored in the config sector of FLASH0. The actual dualboot is then performed by a device reset. After reset, the firmware detects the dualboot image address and starts with the typical startup-routine but is booting into the specified image afterwards. The startup-routine also covers the execution of the secure boot for that additional image if enabled by the startup page of the respective image. An overview of the memory map used for dualboot is depicted in **Figure 105**.

If everything is set up properly, to change from one image to the other, the API function is called to set up the image address followed by a device reset (generated by the application software).

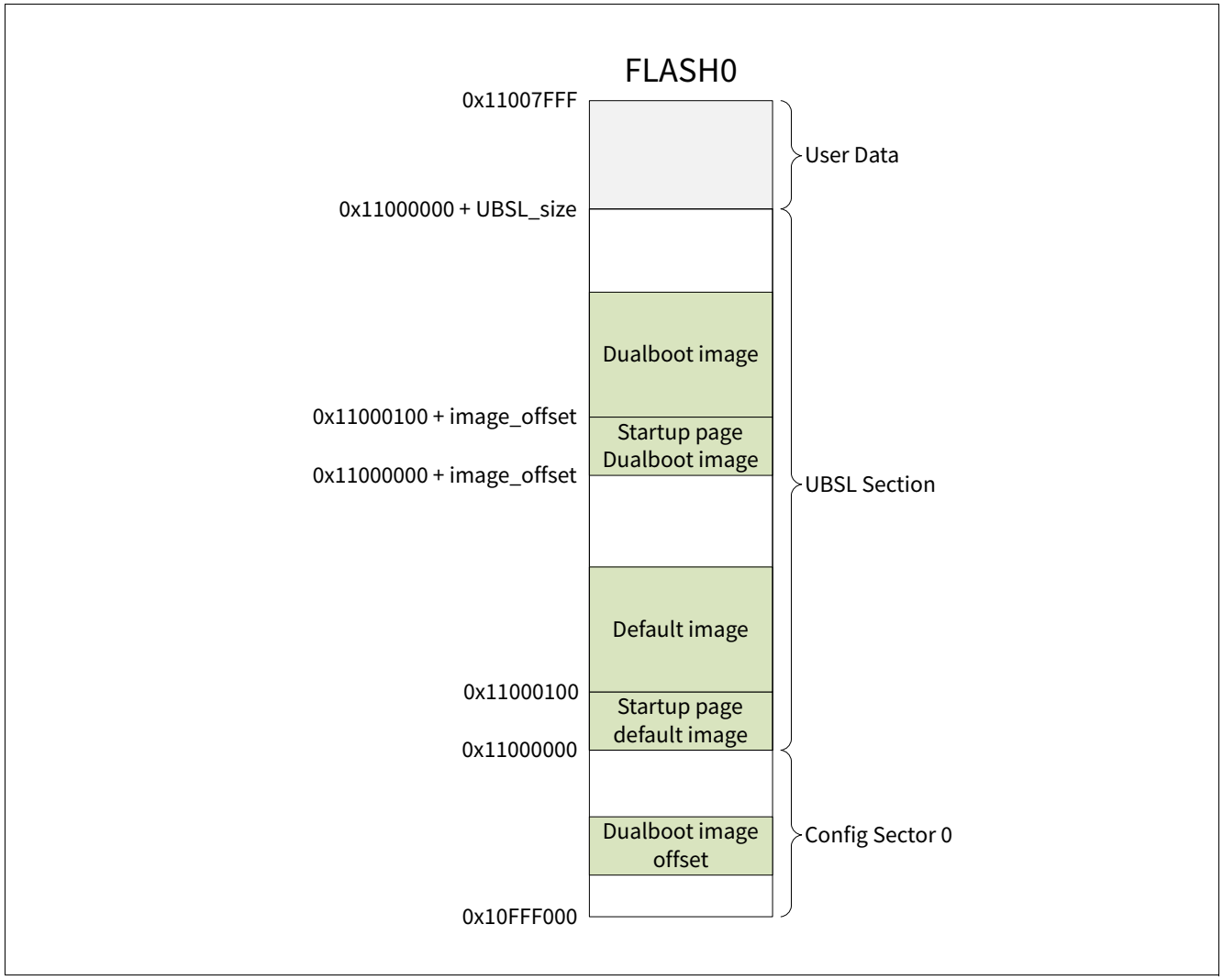


Figure 105 Memory map of FLASH0 for dualboot

7.8.3.3 Crypto library

The BootROM provides several cryptographic routines for calculating signatures, encrypting data chunks or generating hashes. The routines are exposed to the User Code via dedicated firmware APIs.

Supported cryptographic algorithms

- AES
- CMAC

The Crypto library is always executed always in the Protected Context.

7.8.4 MAC Interrupts

The Memory Access Protection unit can generate exception with the bus error upon illegal access attempt.

7.9 Peripheral Bridge (PBA0/1)

The Peripheral Bridge provides the following functions:

- Address decoding for the individual slaves within the Peripheral Bridge domain
- Bus error generation (Default Slave)

7.9.1 PBA0/1 interrupts

The PBA0 and PBA1 bridges can generate exceptions upon bus errors from the connected peripherals.

7.10 Register description MCU

7.10.1 MCU Address Space

Table 100 Register Address Space - MCU

Module	Base Address	End Address	Note
DMA	48034000_H	$48037FFF_H$	DMA Registers, see **Chapter 7.10.2**
MEMCTRL	48044000_H	$48047FFF_H$	MEMCTRL Registers, see **Chapter 7.10.3**
Cache	48048000_H	$4804BFFF_H$	CACHE Registers, see **Chapter 7.10.4**
CPU	$E000E000_H$	$E000EFFF_H$	Arm Core CPU Registers, see **Chapter 7.10.5**

7.10.2 Register description DMA

7.10.2.1 DMA Register Overview

Table 101 Register Overview - DMA (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
DMA_STATUS	DMA Status Register	0000_H	**330**
DMA_CFG	DMA Configuration Register	0004_H	**331**
CTRL_BASE_PTR	Channel Control Data Base Pointer Register	0008_H	**332**
ALT_CTRL_BASE_PTR	Channel Alternate Control Data Base Pointer Register	$000C_H$	**333**
DMA_WAITONREQ_STATUS	Channel Wait on Request Status Register	0010_H	**333**
CHNL_SW_REQUEST	Channel Software Request Register	0014_H	**334**
CHNL_USEBURST_SET	Channel Useburst Set Register	0018_H	**334**
CHNL_USEBURST_CLR	Channel Useburst Clear Register	$001C_H$	**335**
CHNL_REQ_MASK_SET	Channel Request Mask Set Register	0020_H	**336**
CHNL_REQ_MASK_CLR	Channel Request Mask Clear Register	0024_H	**336**
CHNL_ENABLE_SET	Channel Enable Set Register	0028_H	**337**
CHNL_ENABLE_CLR	Channel Enable Clear Register	$002C_H$	**338**
CHNL_PRI_ALT_SET	Channel Primary-Alternate Set Register	0030_H	**338**
CHNL_PRI_ALT_CLR	Channel Primary-Alternate Clear Register	0034_H	**339**
CHNL_PRIORITY_SET	Channel Priority Set Register	0038_H	**340**

Table 101 Register Overview - DMA (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Page Number
CHNL_PRIORITY_CLR	Channel Priority Clear Register	$003C_H$	340
ERR_CLR	Bus Error Status and Clear Register	$004C_H$	341

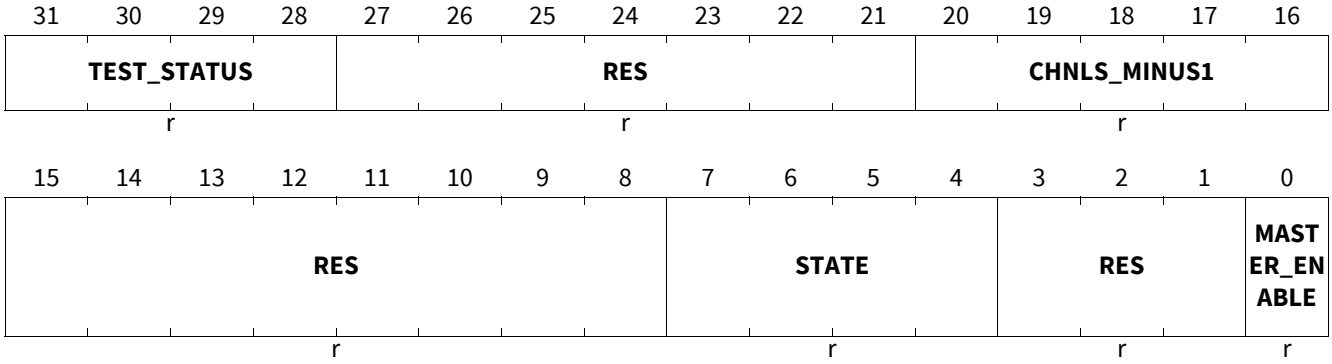
7.10.2.2 DMA Registers

7.10.2.2.1 DMA Registers

DMA Status Register

The read-only DMA_STATUS Register returns the status of the controller.

DMA_STATUS
DMA Status Register **(0000_H)** **RESET_TYPE_3 Value: $0007\ 0000_H$**



Field	Bits	Type	Description
MASTER_ENABLE	0	r	**Controller Status Enable** 0_B **Disabled**, controller is disabled 1_B **Enabled**, controller is enabled
RES	3:1, 15:8, 27:21	r	**Reserved**
STATE	7:4	r	**Current State of the Control State Machine** State can be one of the following bit combinations (other bit combinations are undefined) 0_H **IDLE**, idle 1_H **RCCD**, reading channel controller date 2_H **RSDEP**, reading source data end pointer 3_H **RDDEP**, reading destination data end pointer 4_H **RSD**, reading source data 5_H **WDD**, writing destination data 6_H **WDC**, waiting for DMA request to clear 7_H **WCCD**, writing channel controller data 8_H **STALLED**, stalled 9_H **DONE**, done A_H **PSGT**, peripheral scatter-gather transition B_H **UNDEF_11**, … F_H **UNDEF_15**,

Field	Bits	Type	Description
CHNLS_MINUS1	20:16	r	**Number of DMA Channels** This bitfield value is calculated as the number of available DMA channels minus one. 07_H Controller configured to use 8 DMA Channels
TEST_STATUS	31:28	r	**Test Status** To reduce the gate count you can configure the controller to execute the integration test logic. Bit field combination 2_H to F_H are undefined. 0_H **NoITL**, controller does not include the integration test logic 1_H **ITL**, controller includes the integration test logic 2_H **UNDEF_2**, … F_H **UNDEF_15**,

DMA Configuration Register

DMA_CFG
DMA Configuration Register **(0004_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES1															
w															

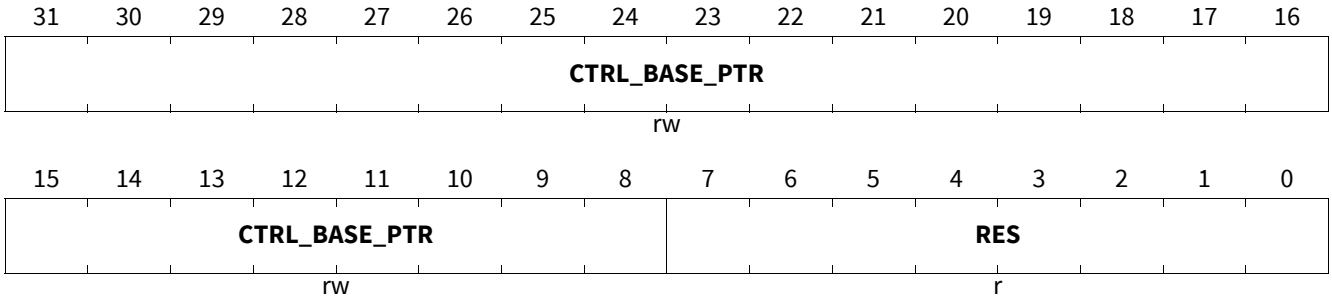
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES1								CHN1_PROT_CTRL			RES0				MASTER_ENABLE
w								w			w				w

Field	Bits	Type	Description
MASTER_ENABLE	0	w	**DMA Controller Enable** 0_B **Disabled**, disables the controller 1_B **Enabled**, enables the controller
RES0	4:1	w	**Reserved** Write as 0.

Field	Bits	Type	Description
CHN1_PROT_CTRL	7:5	w	**AHB Protection** Sets the AHB-Lite protection by controlling the HPROT[3:1] signal levels as follows: bit 7 controls HPROT[3] to indicate if a cacheable access is occurring. Bit 6 controls HPROT[2] to indicate if a bufferable access is occurring. Bit 5 controls HPROT[1] to indicate if a privileged access is occurring. Note:When bit[n]=1, then the corresponding HPROT is HIGH.Note:When bit[n]=1, then the corresponding HPROT is HIGH. The CHN1_PROT_CTRL bits must not be changed when the MASTER_ENABLE bit is set because this may cause a protocol error on the AHB master interface. As the DMA_CFG register is write-only the user must read the status of the master enable bit from the DMA_STATUS register.
RES1	31:8	w	**Reserved** Write as 0.

Channel Control Data Base Pointer Register

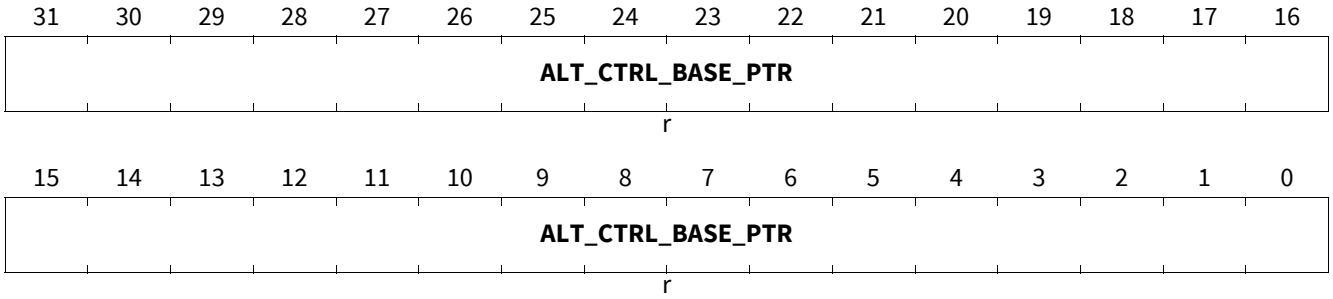
CTRL_BASE_PTR
Channel Control Data Base Pointer Register (0008$_H$) RESET_TYPE_3 Value: 0000 0000$_H$



Field	Bits	Type	Description
RES	7:0	r	**Reserved**
CTRL_BASE_PTR	31:8	rw	**Pointer to the base address of the primary data structure**

Channel Alternate Control Data Base Pointer Register

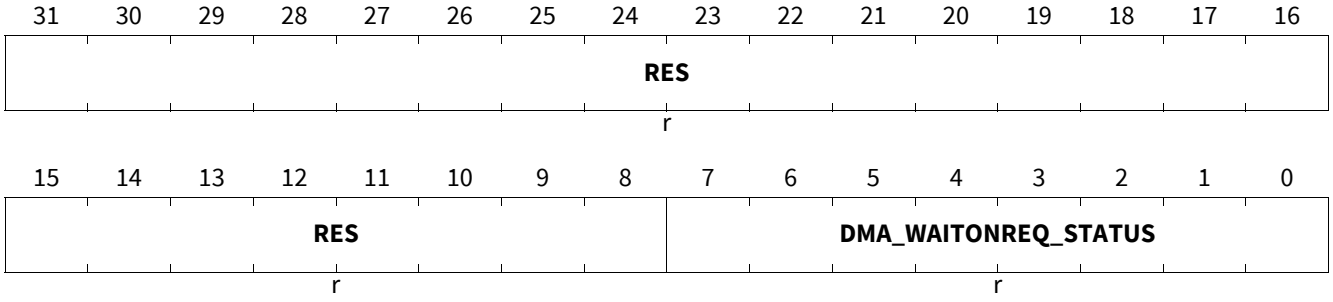
ALT_CTRL_BASE_PTR
Channel Alternate Control Data Base Pointer Register($000C_H$) **RESET_TYPE_3 Value: $0000\ 0080_H$**



Field	Bits	Type	Description
ALT_CTRL_BASE_PTR	31:0	r	**Base Address of the Alternate Data Structure**

Channel Wait on Request Status Register

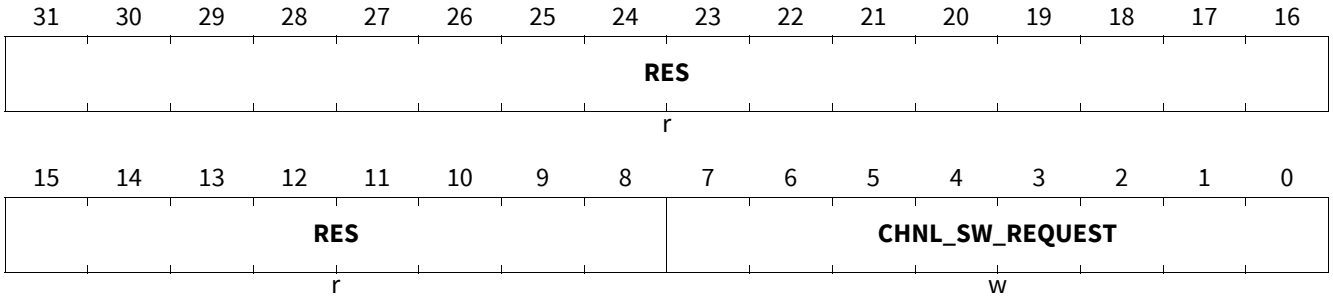
DMA_WAITONREQ_STATUS
Channel Wait on Request Status Register (0010_H) **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
DMA_WAITONREQ_STATUS	7:0	r	**Channel Wait on Request Status** Each bit from 0 to 7 corresponds to a channel C and can have the following values: 0_B dma_waitonreq[C] is LOW. 1_B dma_waitonreq[C] is HIGH.
RES	31:8	r	**Reserved**

Channel Software Request Register

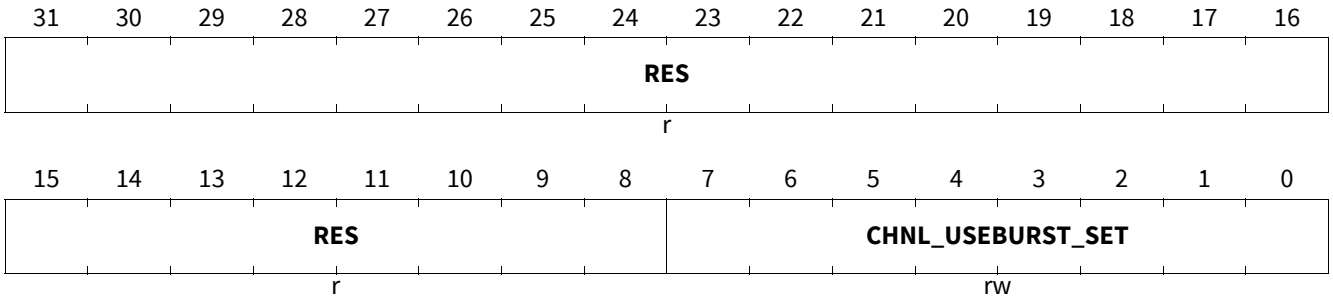
CHNL_SW_REQUEST
Channel Software Request Register **(0014_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHNL_SW_REQUEST	7:0	w	**Software DMA Request** Each bit from 0 to 7 corresponds to a channel C and can have the following values: 0_B does not create a DMA request for channel C. 1_B creates a DMA request for channel C.
RES	31:8	r	**Reserved**

Channel Useburst Set Register

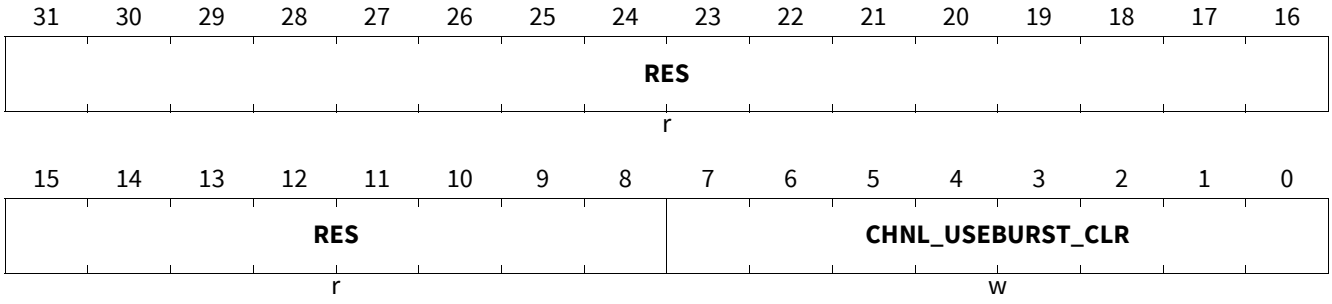
CHNL_USEBURST_SET
Channel Useburst Set Register **(0018_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHNL_USEBURST_SET	7:0	rw	**Channel Useburst Set** Each bit from 0 to 7 corresponds to a channel C and can have the following values: 0_B Set. On a read operation, the DMA channel C responds to received requests on dma_req[C] or dma_sreq[C]. The controller performs 2^R transfers or a single bus transfer. On a write operation, there is no effect. Use the CHNL_USEBURST_CLR register to set the corresponding bit C to 0. 1_B Not_Set. On a read operation, the DMA channel C does not respond to received requests on dma_req[C] or dma_sreq[C]. The controller only responds to dma_req[C] requests and performs 2 transfers. On a write operation, dma_sreq[C] is disabled for the DMA requests generation.
RES	31:8	r	**Reserved**

Channel Useburst Clear Register

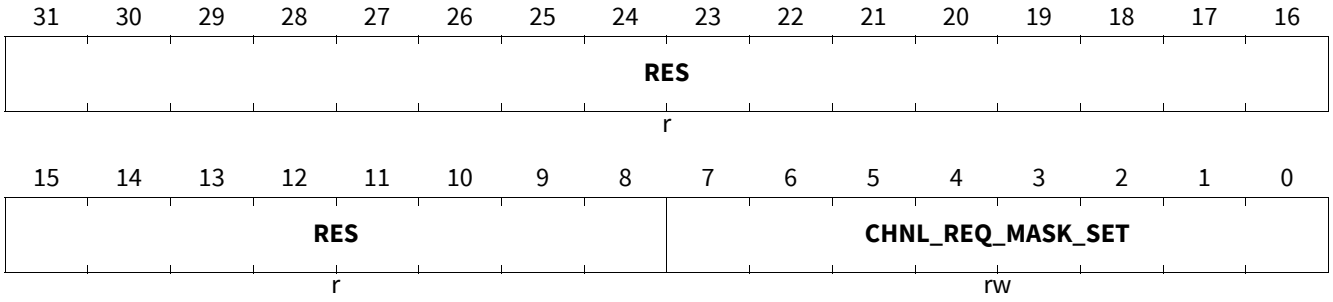
CHNL_USEBURST_CLR
Channel Useburst Clear Register **($001C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHNL_USEBURST_CLR	7:0	w	**Channel Useburst Clear** Each bit from 0 to 7 corresponds to a channel C and can have the following values: 0_B Not_Cleared, set the corresponding bit in the CHNL_USEBURST_SET register to 1 to disable dma_sreq[C] from generating requests 1_B Cleared, enables dma_sreq[C] to generate DMA requests
RES	31:8	r	**Reserved**

Channel Request Mask Set Register

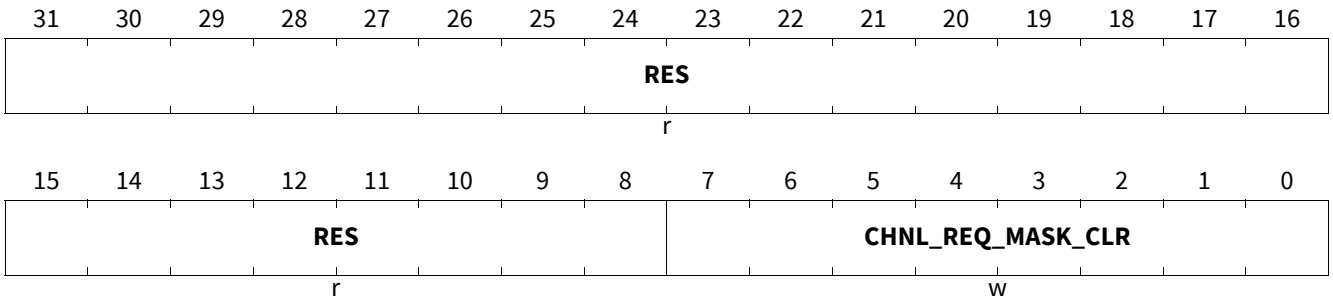
CHNL_REQ_MASK_SET
Channel Request Mask Set Register **(0020_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHNL_REQ_MASK_SET	7:0	rw	**Channel Request Mask Set** Each bit from 0 to 7 corresponds to a channel C and can have the following values: 0_B Set. On a read operation, external requests are enabled for the channel C. On a write operation, there is no effect. Use the register CHNL_REQ_MASK_CLR to enable DMA requests. 1_B Not_Set. On a read operation, external requests are disabled for the channel C. On a write operation, dma_req[C] and dma_sreq[C] are disabled and cannot generate DMA requests.
RES	31:8	r	**Reserved**

Channel Request Mask Clear Register

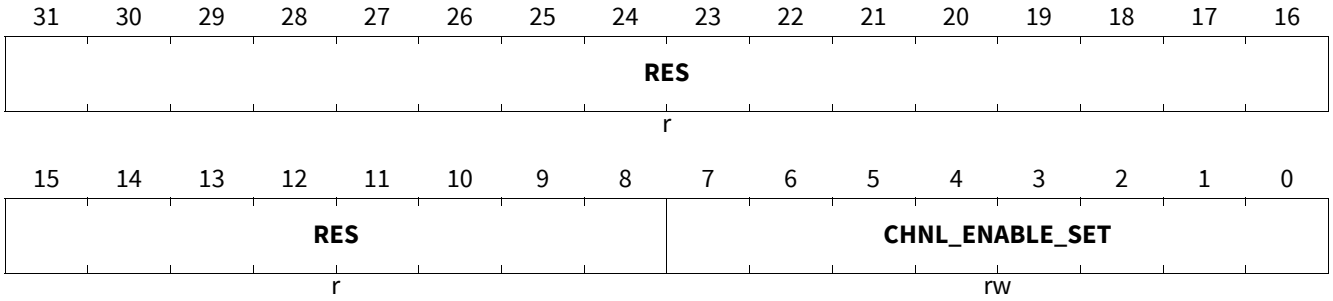
CHNL_REQ_MASK_CLR
Channel Request Mask Clear Register **(0024_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHNL_REQ_MASK_CLR	7:0	w	**Channel Request Mask Clear** Each bit from 0 to 7 corresponds to a channel C and can have the following values: 0_B Not_Cleared, set the corresponding bit in the CHNL_REQ_MASK_SET register to 1 to disable dma_req[C] and dma_sreq[C] from generating requests 1_B Cleared, enables dma_req[C] or dma_sreq[C] to generate DMA requests
RES	31:8	r	**Reserved**

Channel Enable Set Register

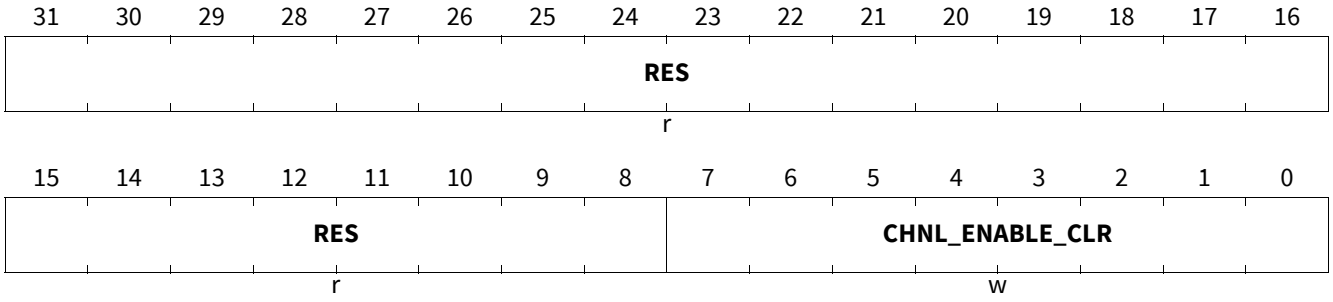
CHNL_ENABLE_SET
Channel Enable Set Register **(0028_H)** **RESET_TYPE_3 Value: 0000 0000_H**



Field	Bits	Type	Description
CHNL_ENABLE_SET	7:0	rw	**Channel Enable Set** Each bit from 0 to 7 corresponds to a channel C and can have the following values: 0_B Not_Set. On a read operation, the DMA channel C is disabled. On a write operation, there is no effect. Use the CHNL_ENABLE_CLR register to disable a channel. 1_B Set. On a read operation, the DMA channel C is enabled. On a write operation, it enables the DMA channel C.
RES	31:8	r	**Reserved**

Channel Enable Clear Register

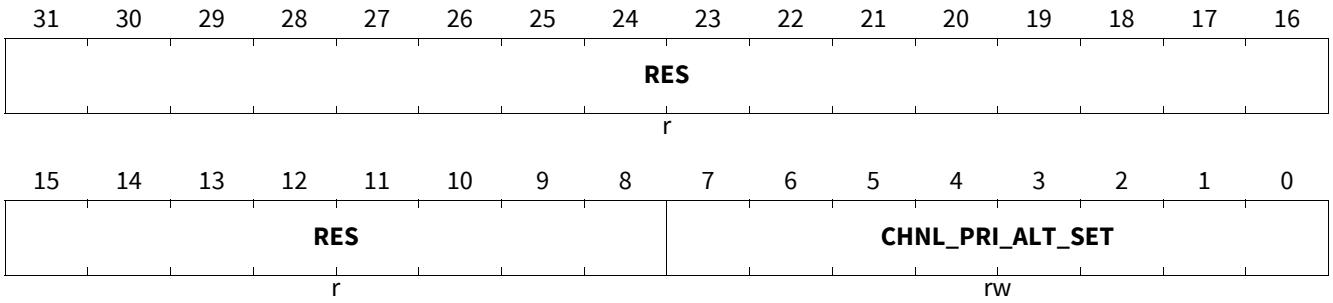
CHNL_ENABLE_CLR
Channel Enable Clear Register **($002C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHNL_ENABLE_CLR	7:0	w	**Channel Enable Clear** Each bit from 0 to 7 corresponds to a channel C and can have the following values: 0_B Not_Cleared, set the corresponding bit in the CHNL_ENABLE_SET register to 1 to enable the corresponding DMA channel C 1_B Cleared, disables the DMA channel C
RES	31:8	r	**Reserved**

Channel Primary-Alternate Set Register

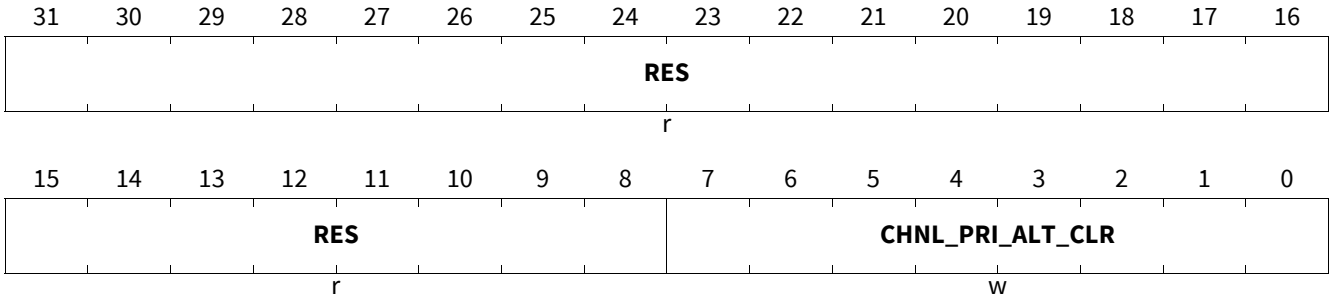
CHNL_PRI_ALT_SET
Channel Primary-Alternate Set Register **(0030_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHNL_PRI_ALT_SET	7:0	rw	**Channel Primary Alternate Set** Each bit from 0 to 7 corresponds to a channel C and can have the following values: 0_B Not_Set. On a read operation, the DMA channel C is using the primary data structure. Setting the corresponding bit in the write-only register CHNL_PRI_ALT_CLR register allows you to configure a DMA channel C to use the primary data structure. 1_B Set. On a read operation, the DMA channel C is using the alternate data structure. On a write operation, it selects the alternate data structure for the DMA channel C
RES	31:8	r	**Reserved**

Channel Primary-Alternate Clear Register

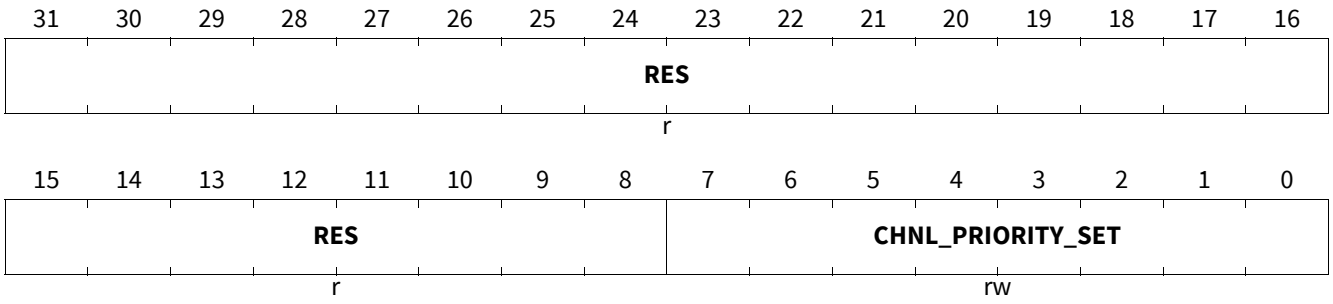
CHNL_PRI_ALT_CLR
Channel Primary-Alternate Clear Register **(0034_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHNL_PRI_ALT_CLR	7:0	w	**Channel Primary Alternate Clear** Each bit from 0 to 7 corresponds to a channel C and can have the following values: 0_B Not_Cleared, set the corresponding bit in the CHNL_PRI_ALT_SET register to 1 to select the alternate data structure 1_B Cleared, selects the primary data structure for the DMA channel C
RES	31:8	r	**Reserved**

Channel Priority Set Register

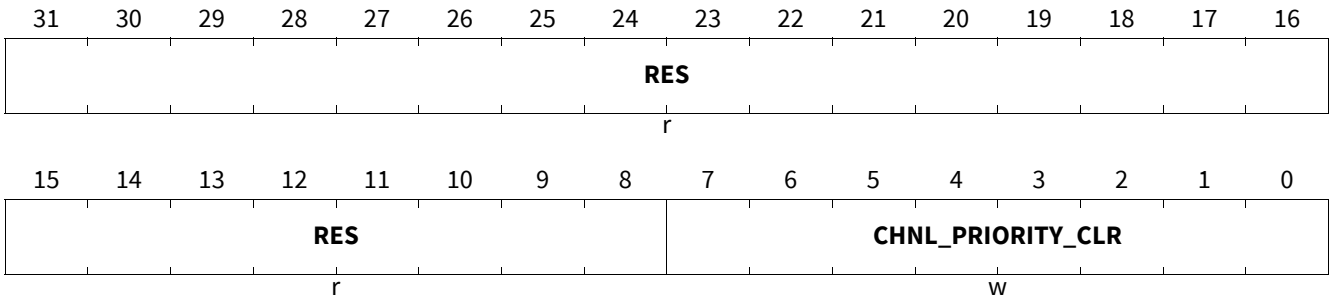
CHNL_PRIORITY_SET
Channel Priority Set Register **(0038_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHNL_PRIORITY_SET	7:0	rw	**Channel Priority Set** Each bit from 0 to 7 corresponds to a channel C and can have the following values: 0_B Not_Set. On a read operation, the DMA channel C is using the default priority level. On a write operation, there is no effect. Set the corresponding bit in the CHNL_ENABLE_CLR register to 1 to set the channel C with the default priority level. 1_B Set. On a read operation, the DMA channel is using a high priority level. On a write operation, it sets a high priority level to the DMA channel C."
RES	31:8	r	**Reserved**

Channel Priority Clear Register

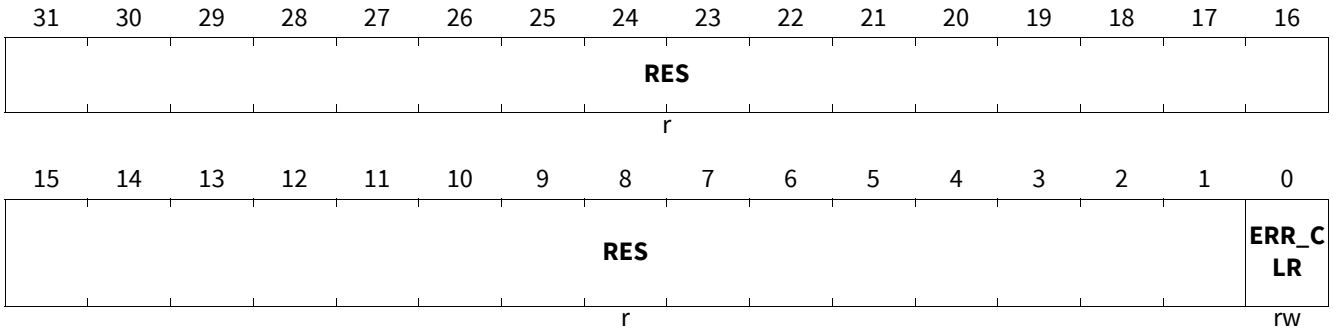
CHNL_PRIORITY_CLR
Channel Priority Clear Register **($003C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHNL_PRIORITY_CLR	7:0	w	**Channel Priority Clear** Each bit from 0 to 7 corresponds to a channel C and can have the following values: 0_B Not_Cleared, set the corresponding bit in the CHNL_ENABLE_SET register to 1 to set the DMA channel C to the high priority level 1_B Cleared, the DMA channel C uses the default priority level.
RES	31:8	r	**Reserved**

Bus Error Status and Clear Register

ERR_CLR
Bus Error Status and Clear Register **($004C_H$)** **RESET_TYPE_3 Value: 0000 0000_H**



Field	Bits	Type	Description
ERR_CLR	0	rw	**Error Clear** 0_B RDL_WRNE. On a read operation DMA_ERR is low. On a write operation there is no effect. 1_B, RDH_WRL. On a read operation DMA_ERR is high. On a write operation, it sets DMA_ERR to low. 0_B **RDL_WRNE**, on read: dma_err is LOW. on write: No effect, status of dma_err is unchanged. 1_B **RDH_WRL**, on read: dma_err is HIGH. on write: Sets dma_err LOW.
RES	31:1	r	**Reserved**

7.10.3 Register description MEMCTRL

7.10.3.1 MEMCTRL Register Overview

Table 102 Register Overview - MEMCTRL (ascending Offset Address)

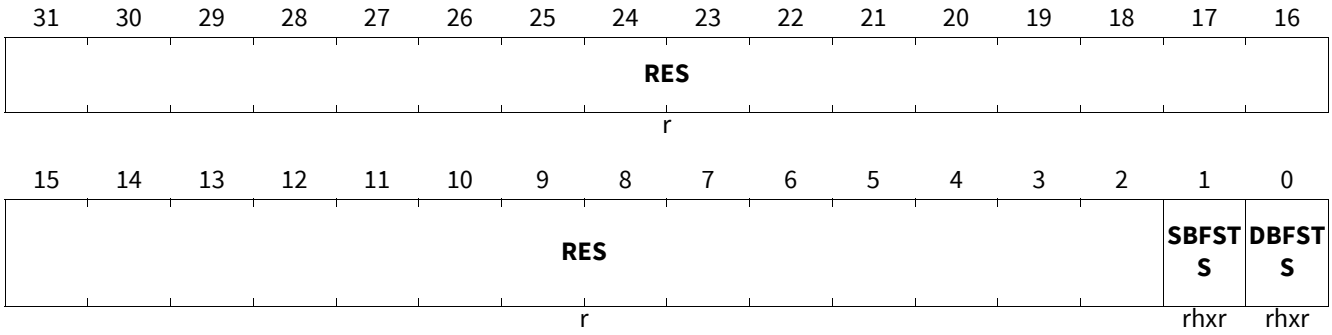
Short Name	Long Name	Offset Address	Page Number
BFSTS	Bus Fault Status Register	0000_H	343
BFSTSC	Bus Fault Status Clear Register	0004_H	343
BFSTSS	Bus Fault Status Set Register	0008_H	344
DBFA	Data Bus Fault Address Register	$000C_H$	345
SBFA	System Bus Fault Address Register	0010_H	345
NMICON	NMI Control Register	0014_H	345
NMISR	NMI Status Register	0018_H	346
NMISRC	NMI Status Clear Register	$001C_H$	348
NMISRS	NMI Status Set Register	0020_H	349
IEN	MEMCTRL Interrupt Enable Register	0024_H	350
IS	MEMCTRL Interrupt Status Register	0028_H	351
ISC	MEMCTRL Interrupt Status Clear Register	$002C_H$	352
ISS	MEMCTRL Interrupt Status Set Register	0030_H	352
MEMSTS	Memory Protection and Error Status Register	0034_H	353
MEMSTSC	Memory Protection and Error Status Register Clear	0038_H	355
MEMSTSS	Memory Protection and Error Status Register Set	$003C_H$	357
NVM_OP_STS	NVM Operation Status	$009C_H$	359
NVM_OP_RESULT	NVM operation result	$00A0_H$	360
MEMSTAT	Memory Status Register	$00A4_H$	360
STACK_OVF_CTRL	Stack Overflow Control Register	$00A8_H$	361
STACK_OVF_ADDR	Stack Overflow Address Register	$00AC_H$	361
STCALIB	System Tick Calibration Register	$00B4_H$	362
SYSWDTCON	System Watchdog Timer Control Register	$00B8_H$	363
SYSWDTREL	System Watchdog Timer Reload Register	$00BC_H$	364
SYSWDT	System Watchdog Timer Value	$00C0_H$	364
SYSWDTWINB	System Watchdog Window-Boundary Count	$00C4_H$	365

7.10.3.2 MEMCTRL Registers

7.10.3.2.1 MEMCTRL Registers

Bus Fault Status Register

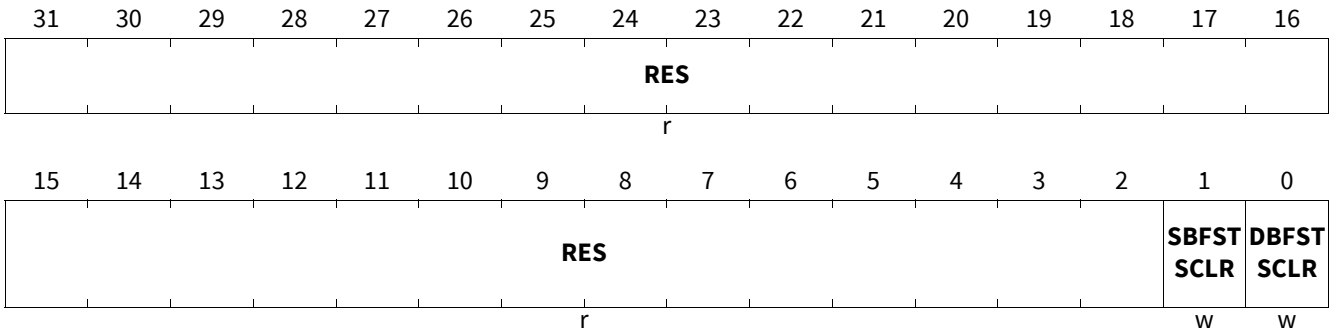
BFSTS
Bus Fault Status Register **(0000_H)** **RESET_TYPE_3 Value: 0000 0000_H**



Field	Bits	Type	Description
DBFSTS	0	rhxr	**Data Bus Fault Status Valid Flag** Address of an occured Data Bus Fault is valid and can be read out of DBFA Register. 0_B **Invalid**, Fault Address is not valid 1_B **Valid**, the address of an occurred Data Bus Fault is valid and can be read out of the DBFA register
SBFSTS	1	rhxr	**System Bus Fault Status Valid Flag** Address of an occured System Bus Fault is valid and can be read out of BFA Register. 0_B **Invalid**, Fault Address is not valid 1_B **Valid**, the address of an occurred System Bus Fault is valid and can be read out of the BFA register
RES	31:2	r	**Reserved** Always read as 0

Bus Fault Status Clear Register

BFSTSC
Bus Fault Status Clear Register **(0004_H)** **RESET_TYPE_3 Value: 0000 0000_H**



Field	Bits	Type	Description
DBFSTSCLR	0	w	**Data Bus Fault Status Valid Flag Clear** This bit is set by software and automatically cleared by hardware 0_B **not_cleared**, Fault Address is not cleared 1_B **Clear**, Fault Address is cleared
SBFSTSCLR	1	w	**System Bus Fault Status Valid Flag Clear** This bit is set by software and automatically cleared by hardware 0_B **not_cleared**, Fault Address is not cleared 1_B **Clear**, Fault Address is cleared
RES	31:2	r	**Reserved** Always read as 0

Bus Fault Status Set Register

BFSTSS

Bus Fault Status Set Register **(0008_H)** **RESET_TYPE_3 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														SBFSTSSET	DBFSTSSET
r														w	w

Field	Bits	Type	Description
DBFSTSSET	0	w	**Data Bus Fault Status Valid Flag Set** This bit is set by software and automatically cleared by hardware 0_B **Not_Set**, Fault Address is not Set 1_B **Set**, Fault Address is Set
SBFSTSSET	1	w	**System Bus Fault Status Valid Flag Set** This bit is set by software and automatically cleared by hardware 0_B **Not_Set**, Fault Address is not Set 1_B **Set**, Fault Address is Set
RES	31:2	r	**Reserved** Always read as 0

Register description MCU

Data Bus Fault Address Register

DBFA
Data Bus Fault Address Register **($000C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
DBFA	31:0	r	**Data Bus Fault Address**

System Bus Fault Address Register

SBFA
System Bus Fault Address Register **(0010_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
SBFA	31:0	r	**System Bus Fault Address**

NMI Control Register

NMICON
NMI Control Register **(0014_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES							NMISTOFEN	NMIWDTEN	NMIMAP1EN	NMIMAP0EN	NMINVM1EN	NMINVM0EN	NMICDEN	NMIPSEN	NMIDSEN
r							rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
NMIDSEN	0	rw	**DSRAM Double Bit ECC Error NMI Enable** 0_B **Disabled**, NMI disabled 1_B **Enabled**, NMI enabled
NMIPSEN	1	rw	**PSRAM Double Bit ECC Error NMI Enable** 0_B **Disabled**, NMI disabled 1_B **Enabled**, NMI enabled
NMICDEN	2	rw	**Cache Data RAM Double Bit ECC Error NMI Enable** 0_B **Disabled**, NMI disabled 1_B **Enabled**, NMI enabled
NMINVM0EN	3	rw	**NVM0 Double Bit ECC Error NMI Enable** 0_B **Disabled**, NMI disabled 1_B **Enabled**, NMI enabled
NMINVM1EN	4	rw	**NVM1 Double Bit ECC Error NMI Enable** 0_B **Disabled**, NMI disabled 1_B **Enabled**, NMI enabled
NMIMAP0EN	5	rw	**NVM0 MAP Error NMI Enable** 0_B **Disabled**, NMI disabled 1_B **Enabled**, NMI enabled
NMIMAP1EN	6	rw	**NVM1 MAP Error NMI Enable** 0_B **Disabled**, NMI disabled 1_B **Enabled**, NMI enabled
NMIWDTEN	7	rw	**Watchdog Timer NMI Enable** 0_B **Disabled**, NMI disabled 1_B **Enabled**, NMI enabled
NMISTOFEN	8	rw	**Stack Overflow NMI Enable** 0_B **Disabled**, NMI disabled 1_B **Enabled**, NMI enabled
RES	31:9	r	**Reserved** Always read as 0

NMI Status Register

NMISR
NMI Status Register **(0018_H)** **RESET_TYPE_4 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES							**NMISTOF**	**NMIWDT**	**NMIMAP1**	**NMIMAP0**	**NMINVM1**	**NMINVM0**	**NMICD**	**NMIPS**	**NMIDS**
r							rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre

Field	Bits	Type	Description
NMIDS	0	rhxre	**DSRAM Double Bit ECC Error NMI Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoNMI**, NMI not occurred 1_B **NMI**, NMI occurred
NMIPS	1	rhxre	**PSRAM Double Bit ECC Error NMI Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoNMI**, NMI not occurred 1_B **NMI**, NMI occurred
NMICD	2	rhxre	**Cache Data RAM Double Bit ECC Error NMI Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoNMI**, NMI not occurred 1_B **NMI**, NMI occurred
NMINVM0	3	rhxre	**NVM0 Double Bit ECC Error NMI Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoNMI**, NMI not occurred 1_B **NMI**, NMI occurred
NMINVM1	4	rhxre	**NVM1 Double Bit ECC Error NMI Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoNMI**, NMI not occurred 1_B **NMI**, NMI occurred
NMIMAP0	5	rhxre	**NVM0 MAP Error NMI Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoNMI**, NMI not occurred 1_B **NMI**, NMI occurred
NMIMAP1	6	rhxre	**NVM1 MAP Error NMI Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoNMI**, NMI not occurred 1_B **NMI**, NMI occurred
NMIWDT	7	rhxre	**Watchdog Timer NMI Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoNMI**, NMI not occurred 1_B **NMI**, NMI occurred
NMISTOF	8	rhxre	**Stack Overflow NMI Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoNMI**, NMI not occurred 1_B **NMI**, NMI occurred

Field	Bits	Type	Description
RES	31:9	r	**Reserved** Always read as 0

NMI Status Clear Register

NMISRC
NMI Status Clear Register **($001C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES							NMISTOFCLR	NMIWDTCLR	NMIMAP1CLR	NMIMAP0CLR	NMINVM1CLR	NMINVM0CLR	NMICDCLR	NMIPSCLR	NMIDSCLR
r							w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
NMIDSCLR	0	w	**DSRAM Double Bit ECC Error NMI Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, NMI not cleared 1_B **Cleared**, NMI cleared
NMIPSCLR	1	w	**PSRAM Double Bit ECC Error NMI Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, NMI not cleared 1_B **Cleared**, NMI cleared
NMICDCLR	2	w	**Cache Data RAM Double Bit ECC Error NMI Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, NMI not cleared 1_B **Cleared**, NMI cleared
NMINVM0CLR	3	w	**NVM 0 Double Bit ECC Error NMI Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, NMI not cleared 1_B **Cleared**, NMI cleared
NMINVM1CLR	4	w	**NVM1 Double Bit ECC Error NMI Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, NMI not cleared 1_B **Cleared**, NMI cleared
NMIMAP0CLR	5	w	**NVM0 MAP Error NMI Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, NMI not cleared 1_B **Cleared**, NMI cleared

Field	Bits	Type	Description
NMIMAP1CLR	6	w	**NVM1 MAP Error NMI Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, NMI not cleared 1_B **Cleared**, NMI cleared
NMIWDTCLR	7	w	**Watchdog Timer NMI Status Clear** This bit is set by software and automatically cleared by hardware 0_B **Not_Cleared**, NMI not cleared 1_B **Cleared**, NMI cleared
NMISTOFCLR	8	w	**Stack Overflow NMI Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, NMI not cleared 1_B **Cleared**, NMI cleared
RES	31:9	r	**Reserved** Always read as 0

NMI Status Set Register

NMISRS
NMI Status Set Register **(0020_H)** **RESET_TYPE_3 Value: 0000 0000_H**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES							NMISTOFSET	NMIWDTSET	NMIMAP1SET	NMIMAP0SET	NMINVM1SET	NMINVM0SET	NMICDSET	NMIPSSET	NMIDSSET
r							w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
NMIDSSET	0	w	**DSRAM Double Bit ECC Error NMI Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, NMI not Set 1_B **Set**, NMI Set
NMIPSSET	1	w	**PSRAM Double Bit ECC Error NMI Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, NMI not Set 1_B **Set**, NMI Set
NMICDSET	2	w	**Cache Data RAM Double Bit ECC Error NMI Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, NMI not Set 1_B **Set**, NMI Set

Field	Bits	Type	Description
NMINVM0SET	3	w	**NVM 0 Double Bit ECC Error NMI Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, NMI not Set 1_B **Set**, NMI Set
NMINVM1SET	4	w	**NVM1 Double Bit ECC Error NMI Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, NMI not Set 1_B **Set**, NMI Set
NMIMAP0SET	5	w	**NVM0 MAP Error NMI Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, NMI not Set 1_B **Set**, NMI Set
NMIMAP1SET	6	w	**NVM1 MAP Error NMI Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, NMI not Set 1_B **Set**, NMI Set
NMIWDTSET	7	w	**Watchdog Timer NMI Status Set** This bit is set by software and automatically cleared by hardware 0_B **Not_Set**, NMI not Set 1_B **Set**, NMI Set
NMISTOFSET	8	w	**Stack Overflow NMI Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, NMI not Set 1_B **Set**, NMI Set
RES	31:9	r	**Reserved** Always read as 0

MEMCTRL Interrupt Enable Register

IEN
MEMCTRL Interrupt Enable Register **(0024_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														NVM1OPCIEN	NVM0OPCIEN
r														rw	rw

Field	Bits	Type	Description
NVM0OPCIEN	0	rw	**NVM0 Operation Complete Interrupt Enable** 0_B **Disabled**, Interrupt disabled 1_B **Enabled**, Interrupt enabled
NVM1OPCIEN	1	rw	**NVM1 Operation Complete Interrupt Enable** 0_B **Disabled**, Interrupt disabled 1_B **Enabled**, Interrupt enabled
RES	31:2	r	**Reserved** Always read as 0

MEMCTRL Interrupt Status Register

IS

MEMCTRL Interrupt Status Register **(0028_H)** **RESET_TYPE_4 Value: 0000 0000_H**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														**NVM1 OPC**	**NVM0 OPC**
r														rhxre	rhxre

Field	Bits	Type	Description
NVM0OPC	0	rhxre	**NVM0 Operation Complete Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
NVM1OPC	1	rhxre	**NVM1 Operation Complete Interrupt Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
RES	31:2	r	**Reserved** Always read as 0

MEMCTRL Interrupt Status Clear Register

ISC

MEMCTRL Interrupt Status Clear Register (002C$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														NVM1OPCLR	NVM0OPCLR
r														w	w

Field	Bits	Type	Description
NVM0OPCLR	0	w	**NVM0 Operation Complete Interrupt Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
NVM1OPCLR	1	w	**NVM1 Operation Complete Interrupt Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
RES	31:2	r	**Reserved** Always read as 0

MEMCTRL Interrupt Status Set Register

ISS

MEMCTRL Interrupt Status Set Register (0030$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														NVM1OPSET	NVM0OPSET
r														w	w

Field	Bits	Type	Description
NVM0OPSET	0	w	**NVM0 Operation Complete Interrupt Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Interrupt not Set 1_B **Set**, Interrupt Set

Field	Bits	Type	Description
NVM1OPSET	1	w	**NVM1 Operation Complete Interrupt Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Interrupt not Set 1_B **Set**, Interrupt Set
RES	31:2	r	**Reserved** Always read as 0

Memory Protection and Error Status Register

MEMSTS

Memory Protection and Error Status Register (0034_H) RESET_TYPE_4 Value: $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES													CDSBE	PSSBE	DSSBE
r													rhxre	rhxre	rhxre

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					PSRAM_PROT_ERR	DSRAM_PROT_ERR	ROM_PROT_ERR	NVM1_SFR_ADDR_ERR	NVM1_SFR_PROT_ERR	NVM1_ADDR_ERR	NVM1_PROT_ERR	NVM0_SFR_ADDR_ERR	NVM0_SFR_PROT_ERR	NVM0_ADDR_ERR	NVM0_PROT_ERR
r					rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre

Field	Bits	Type	Description
NVM0_PROT_ERR	0	rhxre	**NVM0 Access Protection Error** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **No_error**, No Protection error 1_B **Error**, Protection error
NVM0_ADDR_ERR	1	rhxre	**NVM0 Address Protection Error** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **No_error**, No Protection error 1_B **Error**, Protection error
NVM0_SFR_PROT_ERR	2	rhxre	**NVM0 SFR Access Protection Error** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **No_error**, No Protection error 1_B **Error**, Protection error
NVM0_SFR_ADDR_ERR	3	rhxre	**NVM0 SFR Address Protection Error** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **No_error**, No Protection error 1_B **Error**, Protection error

Field	Bits	Type	Description
NVM1_PROT_ERR	4	rhxre	**NVM1 Access Protection Error** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **No_error**, No Protection error 1_B **Error**, Protection error
NVM1_ADDR_ERR	5	rhxre	**NVM1 Address Protection Error** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **No_error**, No Protection error 1_B **Error**, Protection error
NVM1_SFR_PROT_ERR	6	rhxre	**NVM1 SFR Access Protection Error** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **No_error**, No Protection error 1_B **Error**, Protection error
NVM1_SFR_ADDR_ERR	7	rhxre	**NVM1 SFR Address Protection Error** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **No_error**, No Protection error 1_B **Error**, Protection error
ROM_PROT_ERR	8	rhxre	**ROM Access Protection Error** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **No_error**, No Protection error 1_B **Error**, Protection error
DSRAM_PROT_ERR	9	rhxre	**DSRAM Access Protection Error** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **No_error**, No Protection error 1_B **Error**, Protection error
PSRAM_PROT_ERR	10	rhxre	**PSRAM Access Protection Error** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **No_error**, No Protection error 1_B **Error**, Protection error
RES	15:11, 31:19	r	**Reserved** Always read as 0
DSSBE	16	rhxre	**DSRAM Single Bit Error Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoSBE**, Single Bit Error not occurred 1_B **SBE**, Single Bit Error occurred

Field	Bits	Type	Description
PSSBE	17	rhxre	**PSRAM Single Bit Error Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoSBE**, Single Bit Error not occured 1_B **SBE**, Single Bit Error occured
CDSBE	18	rhxre	**Cache Data RAM Single Bit Error Status** This flag is cleared automatically by hardware when the corresponding clear bit is written. 0_B **NoSBE**, Single Bit Error not occured 1_B **SBE**, Single Bit Error occured

Memory Protection and Error Status Register Clear

MEMSTSC

Memory Protection and Error Status Register Clear (0038_H) **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES													CDSBECLR	PSSBECLR	DSSBECLR
						r							w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					PSRAM_PROT_ERRCLR	DSRAM_PROT_ERRCLR	ROM_PROT_ERRCLR	NVM1_SFR_ADDR_ERRC	NVM1_SFR_PROT_ERRC	NVM1_ADDR_ERRCLR	NVM1_PROT_ERRCLR	NVM0_SFR_ADDR_ERRC	NVM0_SFR_PROT_ERRC	NVM0_ADDR_ERRCLR	NVM0_PROT_ERRCLR
		r			w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
NVM0_PROT_ERRCLR	0	w	**NVM0Access Protection Error Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared
NVM0_ADDR_ERRCLR	1	w	**NVM0 Address Protection Error Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared
NVM0_SFR_PROT_ERRCLR	2	w	**NVM0 SFR Access Protection Error Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared

Field	Bits	Type	Description
NVM0_SFR_ADDR_ERRCLR	3	w	**NVM0 SFR Address Protection Error Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared
NVM1_PROT_ERRCLR	4	w	**NVM1 Access Protection Error Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared
NVM1_ADDR_ERRCLR	5	w	**NVM1 Address Protection Error Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared
NVM1_SFR_PROT_ERRCLR	6	w	**NVM1 SFR Access Protection Error Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared
NVM1_SFR_ADDR_ERRCLR	7	w	**NVM1 SFR Address Protection Error Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared
ROM_PROT_ERRCLR	8	w	**ROM Access Protection Error Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared
DSRAM_PROT_ERRCLR	9	w	**DSRAM Access Protection Error Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared
PSRAM_PROT_ERRCLR	10	w	**PSRAM Access Protection Error Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared
RES	15:11, 31:19	r	**Reserved** Always read as 0

Field	Bits	Type	Description
DSSBECLR	16	w	**DSRAM Single Bit Error Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared
PSSBECLR	17	w	**PSRAM Single Bit Error Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared
CDSBECLR	18	w	**Cache Data RAM Single Bit Error Status Clear** This bit is set by software and automatically cleared by hardware. 0_B **Not_Cleared**, Error not cleared 1_B **Cleared**, Error cleared

Memory Protection and Error Status Register Set

MEMSTSS

Memory Protection and Error Status Register Set ($003C_H$) **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES													CDSBESET	PSSBESET	DSSBESET
						r							w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					PSRAM_PROT_ERRSET	DSRAM_PROT_ERRSET	ROM_PROT_ERRSET	NVM1_SFR_ADDR_ERRS	NVM1_SFR_PROT_ERRS	NVM1_ADDR_ERRSET	NVM1_PROT_ERRSET	NVM0_SFR_ADDR_ERRS	NVM0_SFR_PROT_ERRS	NVM0_ADDR_ERRSET	NVM0_PROT_ERRSET
		r			w	w	w	w	w	w	w	w	w	w	w

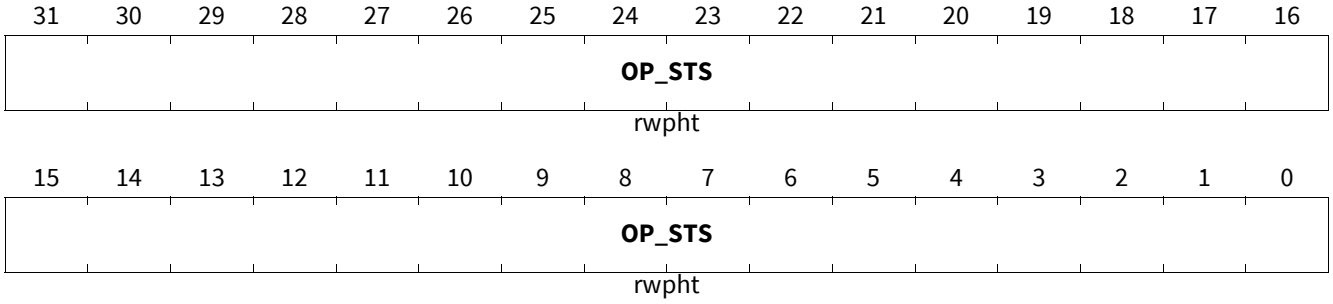
Field	Bits	Type	Description
NVM0_PROT_ERRSET	0	w	**NVM0 Access Protection Error Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set
NVM0_ADDR_ERRSET	1	w	**NVM0 Address Protection Error Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set

Field	Bits	Type	Description
NVM0_SFR_PROT_ERRSET	2	w	**NVM0 SFR Access Protection Error Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set
NVM0_SFR_ADDR_ERRSET	3	w	**NVM0 SFR Address Protection Error Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set
NVM1_PROT_ERRSET	4	w	**NVM1 Access Protection Error Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set
NVM1_ADDR_ERRSET	5	w	**NVM1 Address Protection Error Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set
NVM1_SFR_PROT_ERRSET	6	w	**NVM1 SFR Access Protection Error Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set
NVM1_SFR_ADDR_ERRSET	7	w	**NVM1 SFR Address Protection Error Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set
ROM_PROT_ERRSET	8	w	**ROM Access Protection Error Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set
DSRAM_PROT_ERRSET	9	w	**DSRAM Access Protection Error Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set
PSRAM_PROT_ERRSET	10	w	**PSRAM Access Protection Error Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set

Field	Bits	Type	Description
RES	15:11, 31:19	r	**Reserved** Always read as 0
DSSBESET	16	w	**DSRAM Single Bit Error Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set
PSSBESET	17	w	**PSRAM Single Bit Error Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set
CDSBESET	18	w	**Cache Data RAM Single Bit Error Status Set** This bit is set by software and automatically cleared by hardware. 0_B **Not_Set**, Error not Set 1_B **Set**, Error Set

NVM Operation Status

NVM_OP_STS
NVM Operation Status **($009C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
OP_STS	31:0	rwpht	**Operation Status** 0_H: flash0/1 is not busy 1_H: flash0 is busy 2_H: flash1 is busy 3_H: reserved ... $FFFFFFFF_H$: reserved

Register description MCU

NVM operation result

NVM_OP_RESULT
NVM operation result **($00A0_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OP_RESULT															
rwpht															

Field	Bits	Type	Description
OP_RESULT	31:0	rwpht	**NVM operation result in case of a write/erase operation in the background** The result is only meaningful when NVM0/1 is not busy. Error log codes: 00_H: ERR_LOG_SUCCESS $FFFFFFD9_H$: ERR_LOG_CODE_ACCESS_AB_MODE_ERROR $FFFFFFD8_H$: ERR_LOG_CODE_NVM_ECC2_DATA_ERROR $FFFFFFD7_H$: ERR_LOG_CODE_NVM_VER_ERROR $FFFFFFD6_H$: ERR_LOG_CODE_MAPRAM_INIT_FAIL $FFFFFFD5_H$: ERR_LOG_CODE_VERIFY_AND_MAPRAM_INIT_FAIL

Memory Status Register

MEMSTAT
Memory Status Register **($00A4_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
SECTORINFO	5:0	rw	**Sector number where the Service Algorithm is running**

Field	Bits	Type	Description
SASTATUS	7:6	rw	**Service Algorithm Status** 00_B **Success_1**, If MEMSTAT.SECTORINFO = 0, the NVM is initialized successfully and no Service Algorithm is executed. Otherwise, the Service Algorithm is executed and only one map error is fixed. 01_B **Success_2**, The Service Algorithm is successful and more than one map error are fixed. 10_B **Error_1**, The Service Algorithm is not successful and map error exists. 11_B **Error_2**, The NVM initialization has failed. The Service Algorithm is called but cannot find any page to fix. A soft error is present.
RES	31:8	r	**Reserved** Always read as 0

Stack Overflow Control Register

STACK_OVF_CTRL
Stack Overflow Control Register **($00A8_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
STOF_EN	0	rw	**Stack Overflow Enable** 0_B **Disabled**, stack overflow disabled 1_B **Enabled**, stack overflow enabled
RES	31:1	r	**Reserved** Always read as 0

Stack Overflow Address Register

STACK_OVF_ADDR
Stack Overflow Address Register **($00AC_H$)** **RESET_TYPE_4 Value: $0000\ 0000_H$**

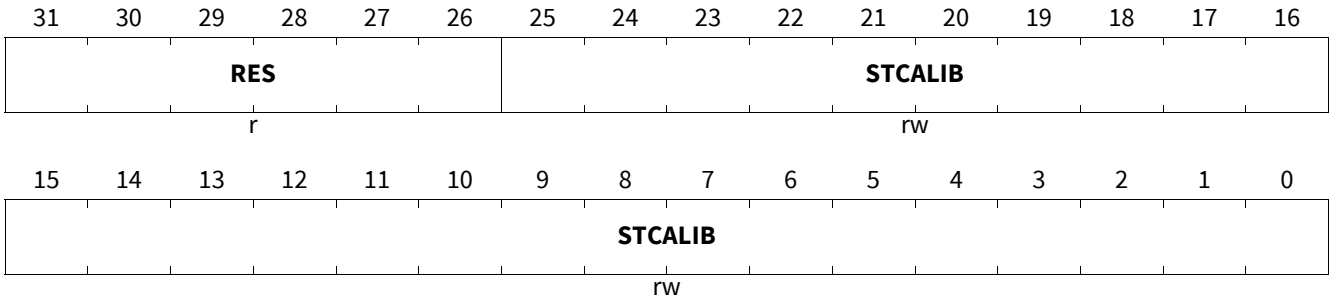
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES							STOF_ADDR_OFF_H								RES
r							rw								r

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

Field	Bits	Type	Description
RES	1:0, 17:15, 31	r	**Reserved** Always read as 0
STOF_ADDR_OFF_L	14:2	rw	**Lower DSRAM address offset boundary for stack overflow protection** The offset is defined from the DSRAM base address (0x18002000). Bits [1:0] of the offset cannot be written and are set to 0.
STOF_ADDR_OFF_H	30:18	rw	**Higher DSRAM address offset boundary for stack overflow protection** The offset is defined from the DSRAM base address (0x18002000). Bits [1:0] of the offset cannot be written and are set to 0.

System Tick Calibration Register

STCALIB
System Tick Calibration Register **($00B4_H$)** **RESET_TYPE_4 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
STCALIB	25:0	rw	**System Tick Calibration** [25]: Noref [24]: Skew [23:0]: Reload Value to use for system tick calibration NOTE: systick clock enable is set each forth cpu_clk cycle
RES	31:26	r	**Reserved** Always read as 0

System Watchdog Timer Control Register

SYSWDTCON
System Watchdog Timer Control Register (00B8$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

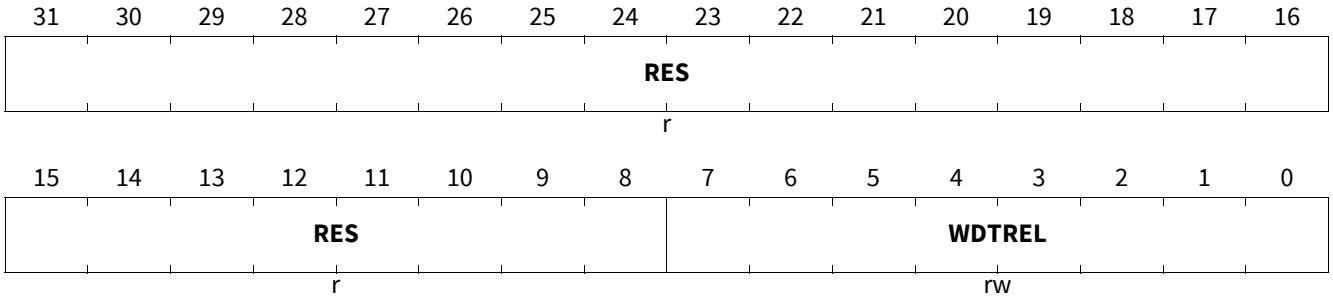
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										WDTBEN	WDTPR	RES	WDTEN	WDTRS	WDTIN
r										rw	r	r	rwv	rwv	rw

Field	Bits	Type	Description
WDTIN	0	rw	**Watchdog Timer Input Frequency Selection** 0_B **DIV2**, Input Frequency is fcpu_clk/2 1_B **DIV128**, Input Frequency is fcpu_clk/128
WDTRS	1	rwv	**WDT Refresh Start** Active high. Set to start refresh operation on the watchdog timer. Cleared automatically by hardware after it is set by software.
WDTEN	2	rwv	**WDT Enable** Note: Clearing WDTEN bit during Prewarning Mode (WDTPR = 1) has no effect. 0_B **Disabled**, WDT is disabled 1_B **Enabled**, WDT is enabled
RES	3, 31:6	r	**Reserved** Always read as 0
WDTPR	4	r	**Watchdog Prewarning Mode Flag** This bit is set to 1 when a Watchdog error is detected. The Watchdog Timer has issued an NMI trap and is in Prewarning Mode. A reset of the chip occurs after the prewarning period has expired. 0_B **Normal**, Normal Mode (default after reset) 1_B **Prewarn**, A watchdog error is detected. The watchdog timer has issued an NMI trap and is in Prewarning mode. A reset of the chip occurs after the prewarning has expired.
WDTBEN	5	rw	**Watchdog Window-Boundary Enable** 0_B **Disable**, Watchdog Window-Boundary feature is disabled. (default) 1_B **Enable**, Watchdog Window-Boundary feature is enabled.

System Watchdog Timer Reload Register

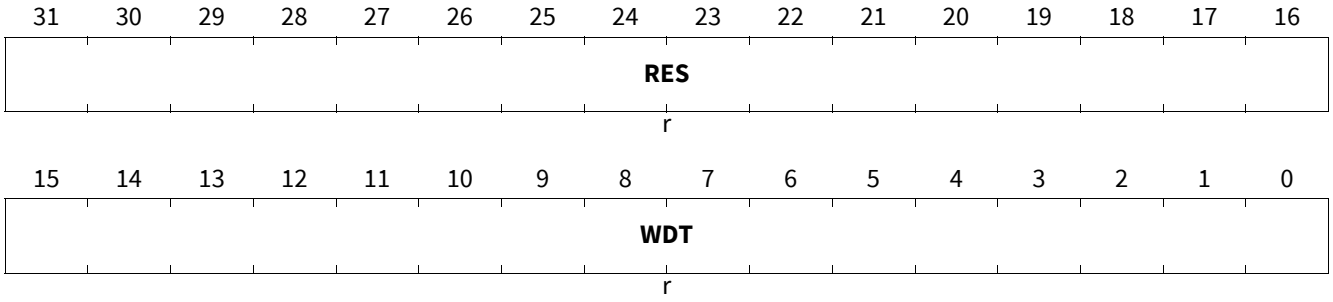
SYSWDTREL
System Watchdog Timer Reload Register (00BC$_H$) RESET_TYPE_3 Value: 0000 0000$_H$



Field	Bits	Type	Description
WDTREL	7:0	rw	**Watchdog Timer Reload Value - Upper Watchdog Timer Byte**
RES	31:8	r	**Reserved** Always read as 0

System Watchdog Timer Value

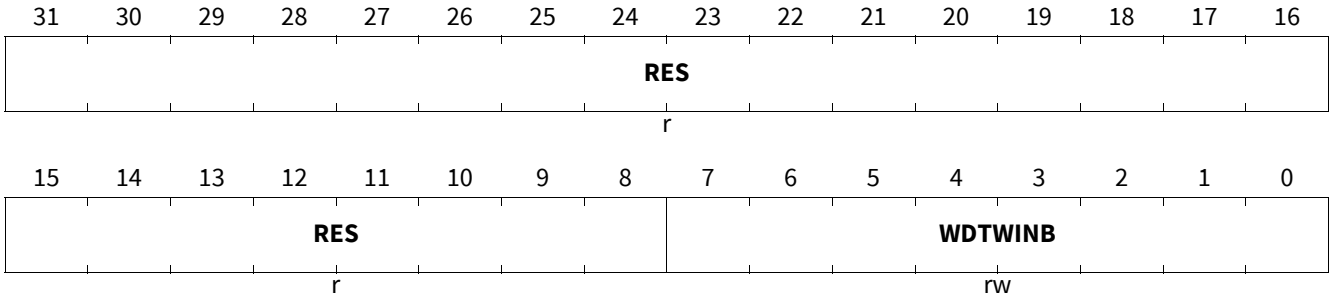
SYSWDT
System Watchdog Timer Value (00C0$_H$) RESET_TYPE_3 Value: 0000 0000$_H$



Field	Bits	Type	Description
WDT	15:0	r	**Watchdog Timer Current Value**
RES	31:16	r	**Reserved** Always read as 0

System Watchdog Window-Boundary Count

SYSWDTWINB
System Watchdog Window-Boundary Count (00C4$_H$) RESET_TYPE_3 Value: 0000 0000$_H$



Field	Bits	Type	Description
WDTWINB	7:0	rw	**Watchdog Window-Boundary Count Value** This value is programmable. Within this Window-Boundary range from 0x0000 to (WDTWINB), the WDT cannot do a Refresh, else it will cause a WDTRST to be asserted. WDTWINB is matched to WDTH.
RES	31:8	r	**Reserved** Always read as 0

7.10.4 Register description CACHE

7.10.4.1 CACHE Register Overview

Table 103 Register Overview - CACHE (ascending Offset Address)

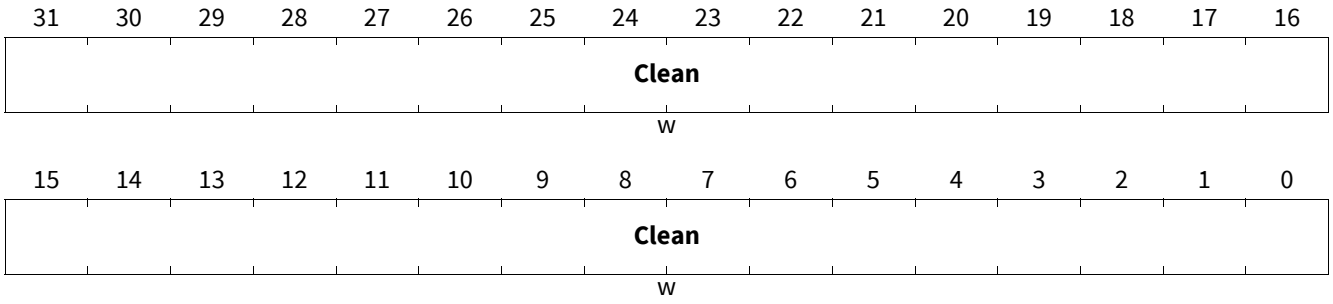
Short Name	Long Name	Offset Address	Page Number
CACHE_AC	Cache All Clean Register	0100_H	367
CACHE_SC	Cache Set Clean Register	0104_H	367
CACHE_BT	Cache Block Touch Register	0114_H	368
CACHE_BL	Cache Block Lock Register	0118_H	368
CACHE_BU	Cache Block Unlock Register	$011C_H$	369

7.10.4.2 CACHE Registers

7.10.4.2.1 CACHE Registers

Cache All Clean Register

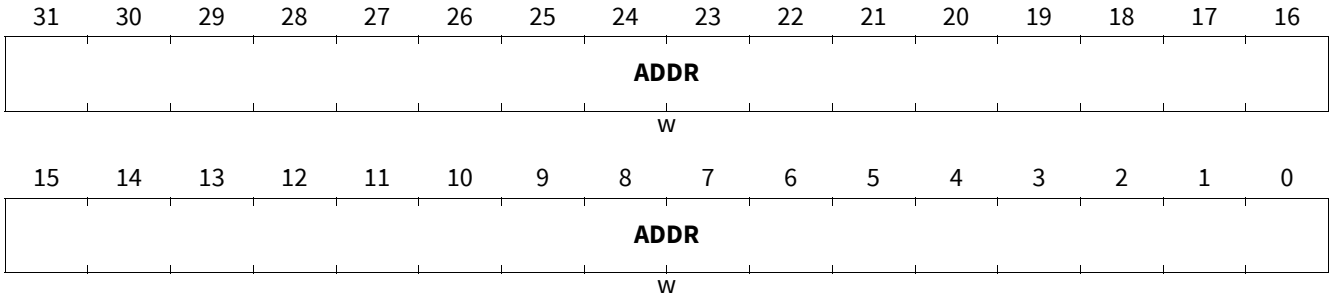
CACHE_AC
Cache All Clean Register **(0100_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
Clean	31:0	w	**Clean all valid blocks from the cache** Any value can be used to trigger a cache cleaning. All valid blocks are then unlocked and invalidated. A read returns 0.

Cache Set Clean Register

CACHE_SC
Cache Set Clean Register **(0104_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

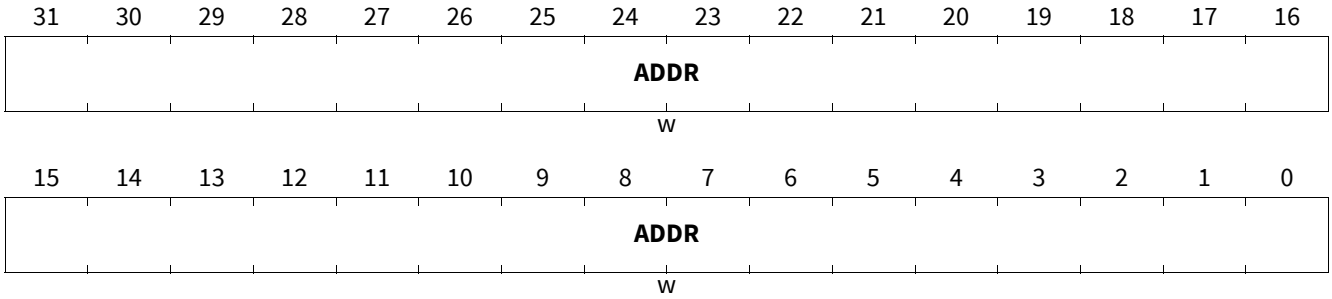


Field	Bits	Type	Description
ADDR	31:0	w	**Clean all valid blocks of one set at a specified address** The address bits depend on the cache size configuration. All valid blocks in this set are then unlocked and invalidated. A read returns 0.

Register description MCU

Cache Block Touch Register

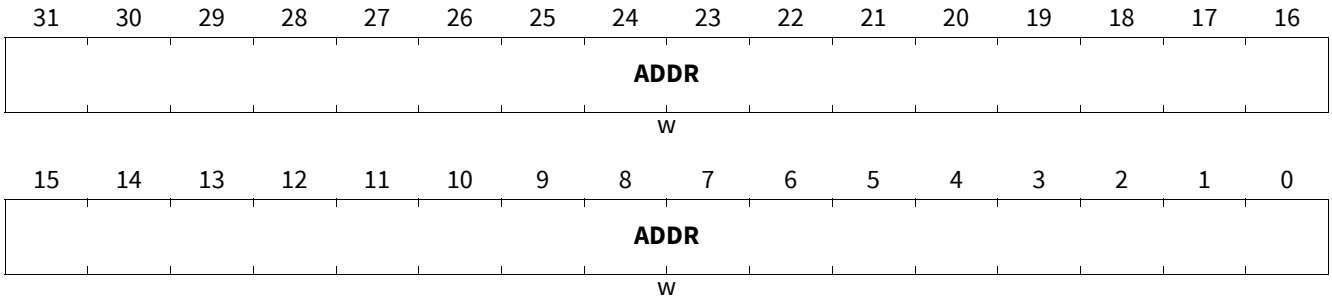
CACHE_BT
Cache Block Touch Register **(0114_H)** **RESET_TYPE_3 Value: 0000 0000$_H$**



Field	Bits	Type	Description
ADDR	31:0	w	**Load a memory block from a specified address into the cache** The address bits depend on the block size configuration. ADDR[3:0] are ignored. The block is then marked as most recently used. A BusFault during the load is not propagated and no block is loaded. If the address has a hit in the cache, no load is performed. A miss will be executed in the background. This command is always a single cycle instruction. A read returns 0.

Cache Block Lock Register

CACHE_BL
Cache Block Lock Register **(0118_H)** **RESET_TYPE_3 Value: 0000 0000$_H$**

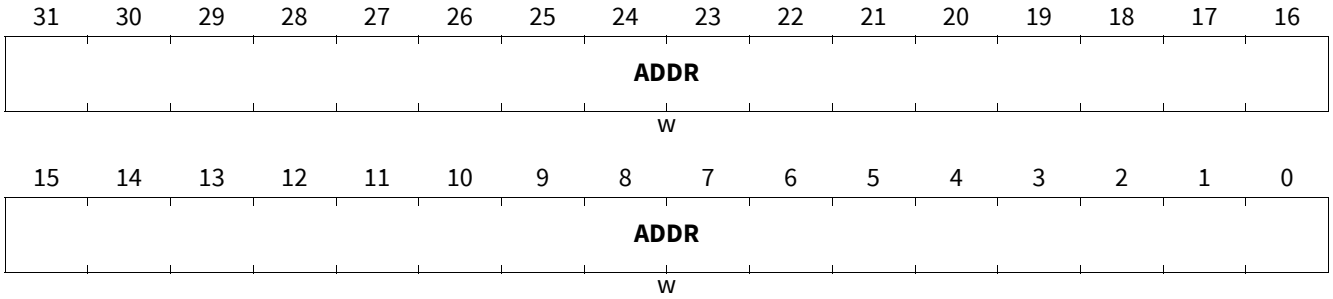


Field	Bits	Type	Description
ADDR	31:0	w	**Lock a memory block from a specified address in the cache** The address bits depend on the block size configuration. ADDR[3:0] are ignored. If the address has a hit in the cache, no fetch and lock is performed. Two ways per set will always remain unlocked. It means that if the maximum number of lockable ways are already locked (the maximum is given by CACHE_CONFIG.WAYS - 2), no fetch and lock is performed. A BusFault is then thrown. A read returns 0.

Register description MCU

Cache Block Unlock Register

CACHE_BU
Cache Block Unlock Register **($011C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
ADDR	31:0	w	**Unlock a memory block from a specified address** The address bits depend on the block size configuration. ADDR[3:0] are ignored. If the address has no hit in the cache, no unlock is performed. Specifically no fill is performed. A read returns 0.

7.10.5 Register description CPU

7.10.5.1 CPU Register Overview

Table 104 Register Overview - CPU (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
ICT	Interrupt Controller Type	0004_H	372
SYSTICK_CS	SysTick Control and Status	0010_H	372
SYSTICK_RL	SysTick Reload Value	0014_H	373
SYSTICK_CUR	SysTick Current Value	0018_H	373
SYSTICK_CAL	SysTick Calibration Value	$001C_H$	374
NVIC_ISER	Interrupt Set-Enable	0100_H	375
NVIC_ICER	Interrupt Clear-Enable	0180_H	377
NVIC_ISPR	Interrupt Set-Pending	0200_H	381
NVIC_ICPR	Interrupt Clear-Pending	0280_H	385
NVIC_IABR	Active Bit Register	0300_H	390
NVIC_IPR0	Interrupt Priority	0400_H	394
NVIC_IPR1	Interrupt Priority	0404_H	394
NVIC_IPR2	Interrupt Priority	0408_H	395
NVIC_IPR3	Interrupt Priority	$040C_H$	396
NVIC_IPR4	Interrupt Priority	0410_H	397
NVIC_IPR5	Interrupt Priority	0414_H	398
NVIC_IPR6	Interrupt Priority	0418_H	398
NVIC_IPR7	Interrupt Priority	$041C_H$	399
CPUID	CPU ID Base Register	$0D00_H$	400
ICSR	Interrupt Control State Register	$0D04_H$	401
VTOR	Vector Table Offset Register	$0D08_H$	402
AIRCR	Application Interrupt/Reset Control Register	$0D0C_H$	403
SCR	System Control Register	$0D10_H$	404
CCR	Configuration Control Register	$0D14_H$	405
SHPR1	System Handler Priority Register 1	$0D18_H$	406
SHPR2	System Handler Priority Register 2	$0D1C_H$	407
SHPR3	System Handler Priority Register 3	$0D20_H$	407
SHCSR	System Handler Control and State Register	$0D24_H$	408
CFSR	Configurable Fault Status Register	$0D28_H$	409
HFSR	Hard Fault Status Register	$0D2C_H$	411
MMFAR	MemManage Fault Address Register	$0D34_H$	412
BFAR	Bus Fault Address Register	$0D38_H$	413

Table 104 Register Overview - CPU (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Page Number
AFSR	Auxiliary Fault Status Register	$0D3C_H$	413
DHCSR	Debug Halting Control and Status Register	$0DF0_H$	414
DCRSR	Debug Core Register Selector Register	$0DF4_H$	416
DCRDR	Debug Core Register Data Register	$0DF8_H$	416
DEMCR	Debug Exception and Monitor Control Register	$0DFC_H$	417
STIR	Software Triggered Interrupt	$0F00_H$	420

7.10.5.2 CPU Registers

7.10.5.2.1 Arm Core CPU Registers

Interrupt Controller Type

ICT

Interrupt Controller Type **(0004_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
INTLINESNUM	4:0	r	**Total number of interrupt lines**
RES	31:5	r	**Reserved**

SysTick Control and Status

SYSTICK_CS

SysTick Control and Status **(0010_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															COUNTFLAG
r															r

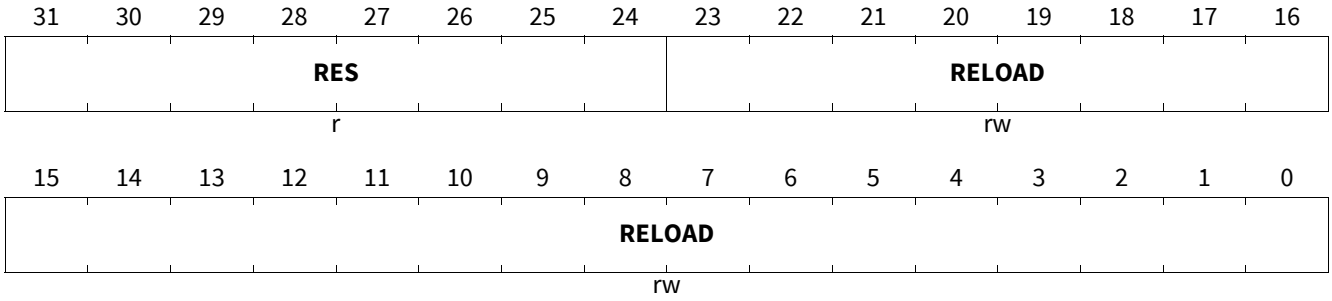
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES													CLKSOURCE	TICKINT	ENABLE
r													rw	rw	rw

Field	Bits	Type	Description
ENABLE	0	rw	**SysTick counter enable** 0_B **Disabled**, counter disabled. 1_B **Enabled**, the counter operates in a multi-shot way. The counter is loaded with the Reload value and begins to count down. When reaching 0, the bit COUNTFLAG is set to 1. Depending on the bit TICKINT, the SysTick handler may be pended. Then the Reload value is loaded again and the countdown starts again.

Field	Bits	Type	Description
TICKINT	1	rw	**TICKINT** 0_B **NoException**, counting down to 0 does not assert the SysTick exeption request 1_B **Exception**, counting down to 0 asserts the SysTick exeption request
CLKSOURCE	2	rw	**CLK SysTick timer clock source selection** 0_B **CPU_CLK**, external reference clock 1_B **CPU_CLK**, CPU clock
RES	15:3, 31:17	r	**Reserved**
COUNTFLAG	16	r	**Count Flag** 0_B **NoCNT**, No count to 0 since last read of the register 1_B **CNT**, The SysTick timer counted to 0 since the last read of this register

SysTick Reload Value

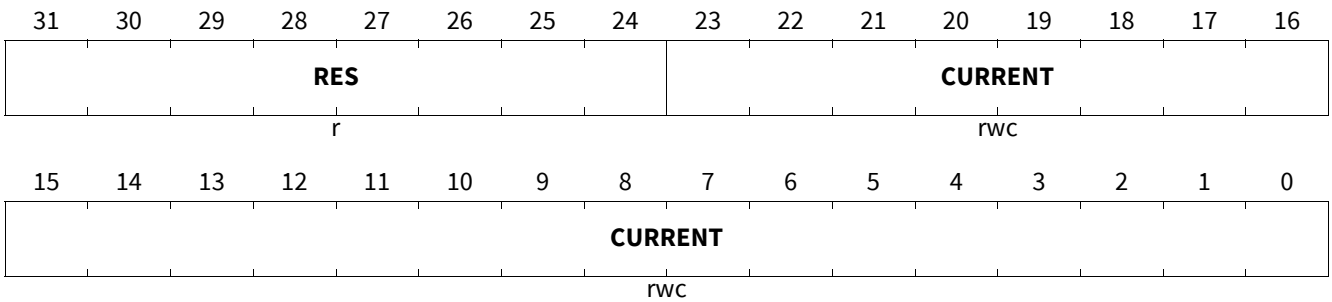
SYSTICK_RL
SysTick Reload Value **(0014_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
RELOAD	23:0	rw	**Reload value for SysTick** This value is loaded into the register SYSTICK_CUR when the counter is enabled and reaches 0."
RES	31:24	r	**Reserved**

SysTick Current Value

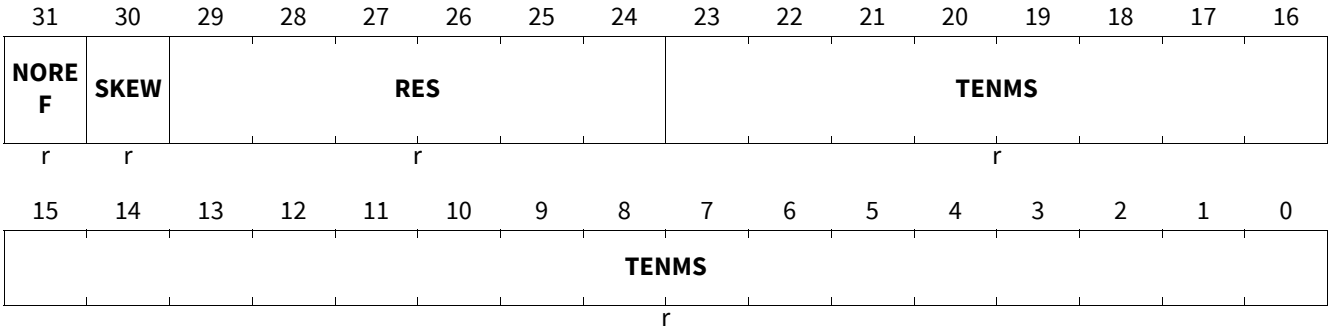
SYSTICK_CUR
SysTick Current Value **(0018_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CURRENT	23:0	rwc	**Current value of SysTick** No read/modify/write protection is provided, so change this value with care. This register is write-clear: writing any value will clear the register to 0. The bit SYSTICK_CS.COUNTFLAG will then be set to 0
RES	31:24	r	**Reserved**

SysTick Calibration Value

SYSTICK_CAL
SysTick Calibration Value **($001C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
TENMS	23:0	r	**Reload value used for 10 ms timing** This value is the Reload value to use for 10ms timing. Depending on the value of SKEW, this might be exactly 10ms or might be the closest value. If set to 0, it means that the calibration value is not known. This is probably because the reference clock is an unknown input from the system or scalable dynamically.
RES	29:24	r	**Reserved**
SKEW	30	r	**Skew** 0_B **Exact**, 10ms calibration value is exact 1_B **Inexact**, 10ms calibration value is inexact, because of the clock frequency
NOREF	31	r	**Presence of a reference clock** 0_B **Ref**, a reference clock is provided 1_B **NoRef**, no reference clock is provided

Interrupt Set-Enable

NVIC_ISER
Interrupt Set-Enable **(0100_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
IRQEN0	0	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN1	1	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN2	2	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN3	3	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN4	4	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN5	5	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN6	6	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN7	7	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN8	8	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN9	9	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt

Field	Bits	Type	Description
IRQEN10	10	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN11	11	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN12	12	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN13	13	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN14	14	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN15	15	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN16	16	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN17	17	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN18	18	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN19	19	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN20	20	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN21	21	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN22	22	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN23	23	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN24	24	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt

Field	Bits	Type	Description
IRQEN25	25	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN26	26	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN27	27	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN28	28	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN29	29	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN30	30	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt
IRQEN31	31	rw	**IRQx interrupt set enable** 0_B **Disabled**, no effect on write 1_B **Enabled**, enables the associated interrupt

Interrupt Clear-Enable

NVIC_ICER
Interrupt Clear-Enable **(0180_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IRQCLREN31	**IRQCLREN30**	**IRQCLREN29**	**IRQCLREN28**	**IRQCLREN27**	**IRQCLREN26**	**IRQCLREN25**	**IRQCLREN24**	**IRQCLREN23**	**IRQCLREN22**	**IRQCLREN21**	**IRQCLREN20**	**IRQCLREN19**	**IRQCLREN18**	**IRQCLREN17**	**IRQCLREN16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQCLREN15	**IRQCLREN14**	**IRQCLREN13**	**IRQCLREN12**	**IRQCLREN11**	**IRQCLREN10**	**IRQCLREN9**	**IRQCLREN8**	**IRQCLREN7**	**IRQCLREN6**	**IRQCLREN5**	**IRQCLREN4**	**IRQCLREN3**	**IRQCLREN2**	**IRQCLREN1**	**IRQCLREN0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
IRQCLREN0	0	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation

Field	Bits	Type	Description
IRQCLREN1	1	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN2	2	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN3	3	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN4	4	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN5	5	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN6	6	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN7	7	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN8	8	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN9	9	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation

Field	Bits	Type	Description
IRQCLREN10	10	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN11	11	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN12	12	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN13	13	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN14	14	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN15	15	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN16	16	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN17	17	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN18	18	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation

Field	Bits	Type	Description
IRQCLREN19	19	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN20	20	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN21	21	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN22	22	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN23	23	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN24	24	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN25	25	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN26	26	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN27	27	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation

Field	Bits	Type	Description
IRQCLREN28	28	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN29	29	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN30	30	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation
IRQCLREN31	31	rw	**IRQx interrupt clear enable** 0_B **Disabled**, the associated interrupt is disabled on a read operation, there is no effect on a write operation 1_B **Enabled**, the associated interrupt is enabled on a read operation and disabled on a write operation

Interrupt Set-Pending

NVIC_ISPR
Interrupt Set-Pending **(0200_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IRQSETPEND31	**IRQSETPEND30**	**IRQSETPEND29**	**IRQSETPEND28**	**IRQSETPEND27**	**IRQSETPEND26**	**IRQSETPEND25**	**IRQSETPEND24**	**IRQSETPEND23**	**IRQSETPEND22**	**IRQSETPEND21**	**IRQSETPEND20**	**IRQSETPEND19**	**IRQSETPEND18**	**IRQSETPEND17**	**IRQSETPEND16**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQSETPEND15	**IRQSETPEND14**	**IRQSETPEND13**	**IRQSETPEND12**	**IRQSETPEND11**	**IRQSETPEND10**	**IRQSETPEND9**	**IRQSETPEND8**	**IRQSETPEND7**	**IRQSETPEND6**	**IRQSETPEND5**	**IRQSETPEND4**	**IRQSETPEND3**	**IRQSETPEND2**	**IRQSETPEND1**	**IRQSETPEND0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
IRQSETPEND0	0	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending

Field	Bits	Type	Description
IRQSETPEND1	1	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND2	2	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND3	3	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND4	4	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND5	5	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND6	6	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND7	7	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND8	8	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND9	9	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND10	10	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND11	11	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending

Field	Bits	Type	Description
IRQSETPEND12	12	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND13	13	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND14	14	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND15	15	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND16	16	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND17	17	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND18	18	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND19	19	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND20	20	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND21	21	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND22	22	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending

Field	Bits	Type	Description
IRQSETPEND23	23	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND24	24	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND25	25	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND26	26	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND27	27	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND28	28	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND29	29	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND30	30	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending
IRQSETPEND31	31	rw	**IRQx interrupt set pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending

Interrupt Clear-Pending

NVIC_ICPR
Interrupt Clear-Pending **(0280_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IRQCLRPEND31	IRQCLRPEND30	IRQCLRPEND29	IRQCLRPEND28	IRQCLRPEND27	IRQCLRPEND26	IRQCLRPEND25	IRQCLRPEND24	IRQCLRPEND23	IRQCLRPEND22	IRQCLRPEND21	IRQCLRPEND20	IRQCLRPEND19	IRQCLRPEND18	IRQCLRPEND17	IRQCLRPEND16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQCLRPEND15	IRQCLRPEND14	IRQCLRPEND13	IRQCLRPEND12	IRQCLRPEND11	IRQCLRPEND10	IRQCLRPEND9	IRQCLRPEND8	IRQCLRPEND7	IRQCLRPEND6	IRQCLRPEND5	IRQCLRPEND4	IRQCLRPEND3	IRQCLRPEND2	IRQCLRPEND1	IRQCLRPEND0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
IRQCLRPEND0	0	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND1	1	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND2	2	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND3	3	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND4	4	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation

Field	Bits	Type	Description
IRQCLRPEND5	5	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND6	6	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND7	7	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND8	8	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND9	9	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND10	10	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND11	11	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation

Field	Bits	Type	Description
IRQCLRPEND12	12	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND13	13	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND14	14	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND15	15	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND16	16	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND17	17	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND18	18	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation

Field	Bits	Type	Description
IRQCLRPEND19	19	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND20	20	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND21	21	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND22	22	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND23	23	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND24	24	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND25	25	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation

Field	Bits	Type	Description
IRQCLRPEND26	26	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND27	27	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND28	28	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND29	29	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND30	30	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation
IRQCLRPEND31	31	rw	**IRQx interrupt clear pending** 0_B **Not_Pending**, the associated interrupt is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the associated interrupt is pending on a read operation, its status is changed to Not_Pending on a write operation

Active Bit Register

NVIC_IABR
Active Bit Register **(0300_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IRQACTIVE31	IRQACTIVE30	IRQACTIVE29	IRQACTIVE28	IRQACTIVE27	IRQACTIVE26	IRQACTIVE25	IRQACTIVE24	IRQACTIVE23	IRQACTIVE22	IRQACTIVE21	IRQACTIVE20	IRQACTIVE19	IRQACTIVE18	IRQACTIVE17	IRQACTIVE16
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQACTIVE15	IRQACTIVE14	IRQACTIVE13	IRQACTIVE12	IRQACTIVE11	IRQACTIVE10	IRQACTIVE9	IRQACTIVE8	IRQACTIVE7	IRQACTIVE6	IRQACTIVE5	IRQACTIVE4	IRQACTIVE3	IRQACTIVE2	IRQACTIVE1	IRQACTIVE0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
IRQACTIVE0	0	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE1	1	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE2	2	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE3	3	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE4	4	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE5	5	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation

Field	Bits	Type	Description
IRQACTIVE6	6	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE7	7	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE8	8	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE9	9	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE10	10	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE11	11	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE12	12	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE13	13	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE14	14	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation

Field	Bits	Type	Description
IRQACTIVE15	15	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE16	16	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE17	17	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE18	18	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE19	19	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE20	20	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE21	21	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE22	22	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE23	23	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation

Field	Bits	Type	Description
IRQACTIVE24	24	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE25	25	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE26	26	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE27	27	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE28	28	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE29	29	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE30	30	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation
IRQACTIVE31	31	r	**IRQx interrupt active flag** 0_B **Disabled**, the associated interrupt is disabled for a read operation, there is no effect for a write operation 1_B **Enabled**, the associated interrupt is enabled for a read and a write operation

Interrupt Priority

NVIC_IPR0

Interrupt Priority **(0400_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_N3					**RES**			**PRI_N2**					**RES**		
rw					r			rw					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_N1					**RES**			**PRI_N0**					**RES**		
rw					r			rw					r		

Field	Bits	Type	Description
RES	2:0, 10:8, 18:16, 26:24	r	**Reserved**
PRI_N0	7:3	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N1	15:11	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N2	23:19	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N3	31:27	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31

Interrupt Priority

NVIC_IPR1

Interrupt Priority **(0404_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_N7					**RES**			**PRI_N6**					**RES**		
rw					r			rw					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_N5					**RES**			**PRI_N4**					**RES**		
rw					r			rw					r		

Field	Bits	Type	Description
RES	2:0, 10:8, 18:16, 26:24	r	**Reserved**
PRI_N4	7:3	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N5	15:11	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N6	23:19	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N7	31:27	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31

Interrupt Priority

NVIC_IPR2

Interrupt Priority **(0408_H)** **RESET_TYPE_3 Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_N11					**RES**			**PRI_N10**					**RES**		
rw					r			rw					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_N9					**RES**			**PRI_N8**					**RES**		
rw					r			rw					r		

Field	Bits	Type	Description
RES	2:0, 10:8, 18:16, 26:24	r	**Reserved**
PRI_N8	7:3	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31

Field	Bits	Type	Description
PRI_N9	15:11	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N10	23:19	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N11	31:27	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31

Interrupt Priority

NVIC_IPR3
Interrupt Priority **($040C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_N15					**RES**			**PRI_N14**					**RES**		
rw					r			rw					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_N13					**RES**			**PRI_N12**					**RES**		
rw					r			rw					r		

Field	Bits	Type	Description
RES	2:0, 10:8, 18:16, 26:24	r	**Reserved**
PRI_N12	7:3	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N13	15:11	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N14	23:19	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31

Field	Bits	Type	Description
PRI_N15	31:27	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31

Interrupt Priority

NVIC_IPR4

Interrupt Priority **(0410_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_N19					**RES**			**PRI_N18**					**RES**		
rw					r			rw					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_N17					**RES**			**PRI_N16**					**RES**		
rw					r			rw					r		

Field	Bits	Type	Description
RES	2:0, 10:8, 18:16, 26:24	r	**Reserved**
PRI_N16	7:3	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N17	15:11	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N18	23:19	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N19	31:27	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31

Interrupt Priority

NVIC_IPR5

Interrupt Priority **(0414_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_N23					**RES**			**PRI_N22**					**RES**		
rw					r			rw					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_N21					**RES**			**PRI_N20**					**RES**		
rw					r			rw					r		

Field	Bits	Type	Description
RES	2:0, 10:8, 18:16, 26:24	r	**Reserved**
PRI_N20	7:3	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N21	15:11	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N22	23:19	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N23	31:27	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31

Interrupt Priority

NVIC_IPR6

Interrupt Priority **(0418_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_N27					**RES**			**PRI_N26**					**RES**		
rw					r			rw					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_N25					**RES**			**PRI_N24**					**RES**		
rw					r			rw					r		

Field	Bits	Type	Description
RES	2:0, 10:8, 18:16, 26:24	r	**Reserved**
PRI_N24	7:3	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N25	15:11	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N26	23:19	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N27	31:27	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31

Interrupt Priority

NVIC_IPR7

Interrupt Priority **($041C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_N31					**RES**			**PRI_N30**					**RES**		
rw					r			rw					r		

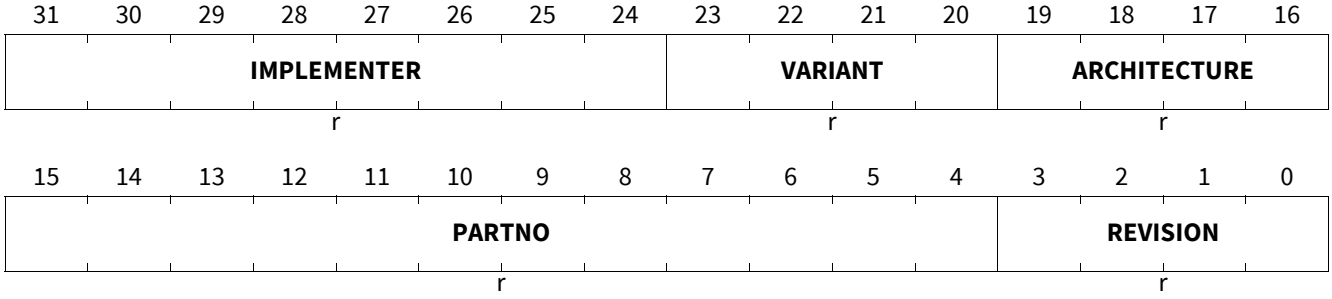
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_N29					**RES**			**PRI_N28**					**RES**		
rw					r			rw					r		

Field	Bits	Type	Description
RES	2:0, 10:8, 18:16, 26:24	r	**Reserved**
PRI_N28	7:3	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31

Field	Bits	Type	Description
PRI_N29	15:11	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N30	23:19	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31
PRI_N31	31:27	rw	**IRQx interrupt priority** 00_H **PRIO_0**, Priority 0 … $1F_H$ **PRIO_31**, Priority 31

CPU ID Base Register

CPUID
CPU ID Base Register **($0D00_H$)** **RESET_TYPE_3 Value: $412F\ C231_H$**



Field	Bits	Type	Description
REVISION	3:0	r	**Revision Number**
PARTNO	15:4	r	**Part Number**
ARCHITECTURE	19:16	r	**Architecture** Read as F_H
VARIANT	23:20	r	**Variant Number**
IMPLEMENTER	31:24	r	**Implementer Code** Assigned by ARM. Read as 41_H for a processor implemented by ARM

Interrupt Control State Register

ICSR
Interrupt Control State Register **($0D04_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NMIPENDSET	RES		PENDSVSET	PENDSVCLR	PENDSTSET	PENDSTCLR	RES	ISRPREEMPT	ISRPending	RES	VECTPending				
rw	r		rw	w	rw	w	r	r	r	r	r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VECTPending				RETTOBASE	RES		VECTACTIVE								
r				r	r		r								

Field	Bits	Type	Description
VECTACTIVE	8:0	r	**Active exception number**
RES	10:9, 21, 24, 30:29	r	**Reserved**
RETTOBASE	11	r	**Presence of preempted active exceptions** Note: In Thread Mode the value of this bit is unknown. 0_B **EXCEPTION**, There is an active exception other than the exception shown by IPSR. 1_B **NoEXCEPTION**, There is no active exception other than any exception shown by IPSR.
VECTPending	20:12	r	**Exception number with the highest priority pending enabled exception** A value of 0 indicates that there is no pending exception.
ISRPending	22	r	**External interrupt generated by the NVIC pending flag** Indicates whether an external interrupt, generated by the NVIC, is pending. 0_B **Not_Pending**, no external interrupt is pending 1_B **Pending**, external interrupt is pending
ISRPREEMPT	23	r	**Service of a pending exception on exit from debug halt state** 0_B **NotSERVICED**, will not service 1_B **SERVICED**, will service a pending exception
PENDSTCLR	25	w	**SysTick exception clear pending** On a read operation, the value is unknown. 0_B **Not_Cleared**, no effect 1_B **Cleared**, the pending status is removed

Field	Bits	Type	Description
PENDSTSET	26	rw	**SysTick exception set pending** On writes, sets the SysTick exception as pending. On reads, indicates the current state of the exception. 0_B **Not_Pending**, the SysTick is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, the SysTick is pending on a read operation, the status of a SysTick exception is set to pending on a write operation
PENDSVCLR	27	w	**PendSV clear pending** Removes the pending status of the PendSV exception 0_B **NoCLEAR**, no effect 1_B **CLEAR**, remove pending status
PENDSVSET	28	rw	**PendSV set pending** Note: Writing 1 to this bit is the only way to set the PendSV exception state to pending. 0_B **Not_Pending**, PendSV is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, PendSV is pending on a read operation, the status of a PendSV exception is set to pending on a write operation
NMIPENDSET	31	rw	**NMI set pending** Note: Given NMI is the exception with highest priority, the processor enters the NMI exception as soon as NMIPENDSET = 1. This bit is then cleared to 0. This means this bit returns a 1 on a read operation only if the NMI signal is reasserted during the handler execution. 0_B **Not_Pending**, an NMI exception is not pending on a read operation, there is no effect on a write operation 1_B **Pending**, an NMI exception is pending on a read operation, the status of an NMI exception is set to pending on write operation

Vector Table Offset Register

VTOR
Vector Table Offset Register **($0D08_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
RES	6:0	r	**Reserved**

Register description MCU

Field	Bits	Type	Description
TBLOFF	31:7	rw	**Vector Table Offset** Bits 31:7 of Vector Table Offset Address.

Application Interrupt/Reset Control Register

AIRCR
Application Interrupt/Reset Control Register (0D0C$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENDIANNESS	RES				PRIGROUP			RES					SYSRESETREQ	VECTCLRACTIVE	VECTRESET
r	r				rw			r					rw	w	w

Field	Bits	Type	Description
VECTRESET	0	w	**Local system reset** Note: The behavior is unpredictable if: the processor is not halted in Debug state and AIRCR.VECTRESET is set to 1 OR the processor is halted in Debug state and both AIRCR.VECTRESET and AIRCR.SYSRESETREQ are set to 1 0_B **NoRESET**, No local system reset 1_B **RESET**, local system reset
VECTCLRACTIVE	1	w	**Active status information clear for fixed and configurable exception** Note: The behavior is unpredictable if the processor is not halted in Debug state and AIRCR.VECTRESET is set to 1 0_B **Not_Cleared**, no information cleared 1_B **Cleared**, all active status information are cleared for fixed and configurable exceptions, the IPSR is also cleared to 0
SYSRESETREQ	2	rw	**System Reset Request** Always read as 0 0_B **NoRESET**, no effect 1_B **RESET**, request system reset
RES	7:3, 14:11	r	**Reserved**
PRIGROUP	10:8	rw	**Priority Grouping** Indicates the binary point position.

Field	Bits	Type	Description
ENDIANNESS	15	r	**Data endianness** This bit is static or configured by hardware input on reset. 0_B **LittleEndian**, little endian 1_B **BigEndian**, big endian
VECTKEY	31:16	rw	**Vector Key** Register writes must write $05FA_H$ to this field, otherwise the write is ignored.

System Control Register

SCR
System Control Register **($0D10_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES											SEVONPEND	RES	SLEEPDEEP	SLEEPONEXIT	RES
r											rw	r	rw	rw	r

Field	Bits	Type	Description
RES	0, 3, 31:5	r	**Reserved**
SLEEPONEXIT	1	rw	**Sleep on exit when returning from Handler mode to Thread mode** Indicates sleep-on-exit when returning from Handler mode to Thread mode. 0_B **NoSLEEP**, do not enter sleep mode 1_B **SLEEP**, enter sleep mode. In this case, an interrupt driven application do not return to an empty main application
SLEEPDEEP	2	rw	**Selection of sleep mode or deep sleep mode as low power mode** 0_B **SLEEP**, selected sleep mode is sleep 1_B **DEEPSLEEP**, selected sleep mode is deep sleep

Field	Bits	Type	Description
SEVONPEND	4	rw	**Send event on pending** When an event or interrupt enters pending state, the event signal wakes up the processor from WFE. If the processor is not waiting for an event, the event is registered and affects the next WFE. The processor also wakes up on execution of a SEV instruction or an external event. 0_B **Enabled**, only enabled interrupts or events can wake-up the processor, disabled interrupts are excluded 1_B **All**, enabled events and all interrupts, including disabled interrupts, can wake-up the processor

Configuration Control Register

CCR
Configuration Control Register **($0D14_H$)** **RESET_TYPE_3 Value: $0000\ 0200_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						STKALIGN	BFHFMIGN	RES			DIV_0_TRP	UNALIGN_TRP	RES	USERSETMPEND	NONBASETHRDENA
r						rw	rw	r			rw	rw	r	rw	rw

Field	Bits	Type	Description
NONBASETHRDENA	0	rw	**Control of the processor entry point into the Thread mode** 0_B **EXCEPTION**, any attempt to enter Thread mode at an execution priority level of other than base level faults. 1_B **THREADMODE**, the processor can enter Thread mode at any execution priority level because of a controlled return value.
USERSETMPEND	1	rw	**Unprivileged software access to the STIR enable** 0_B **NoACCESS**, unprivileged software cannot access register STIR. 1_B **ACCESS**, unprivileged software can access register STIR.
RES	2, 7:5, 31:10	r	**Reserved**
UNALIGN_TRP	3	rw	**Unaligned access traps enable** 0_B **Disabled**, trapping disabled 1_B **Enabled**, trapping enabled
DIV_0_TRP	4	rw	**Trap divide by zero enable** 0_B **Disabled**, do not trap divide by 0 1_B **Enabled**, trap divide by 0

Field	Bits	Type	Description
BFHFMIGN	8	rw	**Effect of precise data access faults on handlers running at a priority -1 or -2** 0_B **LOCKUP**, precise data access fault causes a lockup 1_B **IGNORE**, handler ignores the fault
STKALIGN	9	rw	**Stack frame alignment on exception entry** 0_B **4BYTE**, the exception entry sequence guarantees an 4-byte stack frame alignment, no adjustment is performed 1_B **8BYTE**, the exception entry sequence guarantees an 8-byte stack frame alignment, an adjustment is performed if necessary

System Handler Priority Register 1

SHPR1
System Handler Priority Register 1 **($0D18_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_7					**RES**			**PRI_6**					**RES**		
r					r			rw					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_5					**RES**			**PRI_4**					**RES**		
rw					r			rw					r		

Field	Bits	Type	Description
RES	2:0, 10:8, 18:16, 26:24	r	**Reserved**
PRI_4	7:3	rw	**Priority of System Handler 4, MemManage**
PRI_5	15:11	rw	**Priority of System Handler 5, BusFault**
PRI_6	23:19	rw	**Priority of System Handler 6, UsageFault**
PRI_7	31:27	r	**Reserved for Priority of System Handler 7**

System Handler Priority Register 2

SHPR2

System Handler Priority Register 2 **($0D1C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_11					RES			PRI_10					RES		
rw					r			r					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_9					RES			PRI_8					RES		
r					r			r					r		

Field	Bits	Type	Description
RES	2:0, 10:8, 18:16, 26:24	r	**Reserved**
PRI_8	7:3	r	**Reserved for Priority of System Handler 8**
PRI_9	15:11	r	**Reserved for Priority of System Handler 9**
PRI_10	23:19	r	**Reserved for Priority of System Handler 10**
PRI_11	31:27	rw	**Priority of System Handler 11, SVCall**

System Handler Priority Register 3

SHPR3

System Handler Priority Register 3 **($0D20_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_15					RES			PRI_14					RES		
rw					r			rw					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_13					RES			PRI_12					RES		
r					r			rw					r		

Field	Bits	Type	Description
RES	2:0, 10:8, 18:16, 26:24	r	**Reserved**
PRI_12	7:3	rw	**Priority of System Handler 12, DebugMonitor**
PRI_13	15:11	r	**Reserved for Priority of System Handler 13**
PRI_14	23:19	rw	**Priority of System Handler 14, PendSV**
PRI_15	31:27	rw	**Priority of System Handler 15, SysTick**

System Handler Control and State Register

SHCSR

System Handler Control and State Register (0D24$_H$) RESET_TYPE_3 Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES													USGFAULTENA	BUSFAULTENA	MEMFAULTENA
						r							rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SVCALLPENDED	BUSFAULTPENDED	MEMFAULTPENDED	USGFAULTPENDED	SYSTICKACT	PENDSVACT	RES	MONITORACT	SVCALLACT	RES			USGFAULTACT	RES	BUSFAULTACT	MEMFAULTACT
rw	rw	rw	rw	rw	rw	r	rw	rw	r			rw	r	rw	rw

Field	Bits	Type	Description
MEMFAULTACT	0	rw	**MemManage exception status** 0_B **Inactive**, MemManage is not active 1_B **Active**, MemManage is active
BUSFAULTACT	1	rw	**BusFault exception status** 0_B **NotACTIVE**, BusFault is not active 1_B **ACTIVE**, BusFault is active
RES	2, 6:4, 9, 31:19	r	**Reserved**
USGFAULTACT	3	rw	**UsageFault exception status** 0_B **NotACTIVE**, UsageFault is not active 1_B **ACTIVE**, UsageFault is active
SVCALLACT	7	rw	**SVCall status** 0_B **NotACTIVE**, SVCall is not active 1_B **ACTIVE**, SVCall is active
MONITORACT	8	rw	**Debug monitor status** 0_B **NotActive**, Monitor is not active 1_B **ACTIVE**, Monitor is active
PENDSVACT	10	rw	**PendSV exception status** 0_B **NotACTIVE**, PendSV is not active 1_B **ACTIVE**, PendSV is active
SYSTICKACT	11	rw	**SysTick exception status** 0_B **NotACTIVE**, SysTick is not active 1_B **ACTIVE**, SysTick is active
USGFAULTPENDED	12	rw	**UsageFault exception pending status** 0_B **Not_Pending**, UsageFault is not pending 1_B **Pending**, UsageFault is pending

Register description MCU

Field	Bits	Type	Description
MEMFAULTPENDED	13	rw	**MemManage exception pending status** 0_B **Not_Pending**, MemManage is not pending 1_B **Pending**, MemManage is pending
BUSFAULTPENDED	14	rw	**BusFault exception pending status** 0_B **Not_Pending**, BusFault is not pending 1_B **Pending**, BusFault is pending
SVCALLPENDED	15	rw	**SVCall exception pending status** 0_B **Not_Pending**, SVCall is not pending 1_B **Pending**, SVCall is pending
MEMFAULTENA	16	rw	**MemManage enable** 0_B **Disabled**, Disable MemManage fault 1_B **Enabled**, Enable MemManage fault
BUSFAULTENA	17	rw	**BusFault enable** 0_B **Disabled**, Disable BusFault 1_B **Enabled**, Enable BusFault
USGFAULTENA	18	rw	**UsageFault enable** 0_B **Disabled**, Disable UsageFault 1_B **Enabled**, Enable UsageFault

Configurable Fault Status Register

CFSR
Configurable Fault Status Register **($0D28_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						DIVBYZERO	UNALIGNED	RES				NOCP	INVPC	INVSTATE	UNDEFINSTR
			r			rw	rw			r		rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BFARVALID	RES	LSPERR	STKERR	UNSTKERR	IMPRECISERR	PRECISERR	IBUSERR	MMARVALID	RES	MLSPERR	MSTERR	MUNSTKERR	RES	DACCVIOL	IACCVIOL
rw	r	rw	rw	rw	rw	rw	rw	rw	r	rw	rw	rw	r	rw	rw

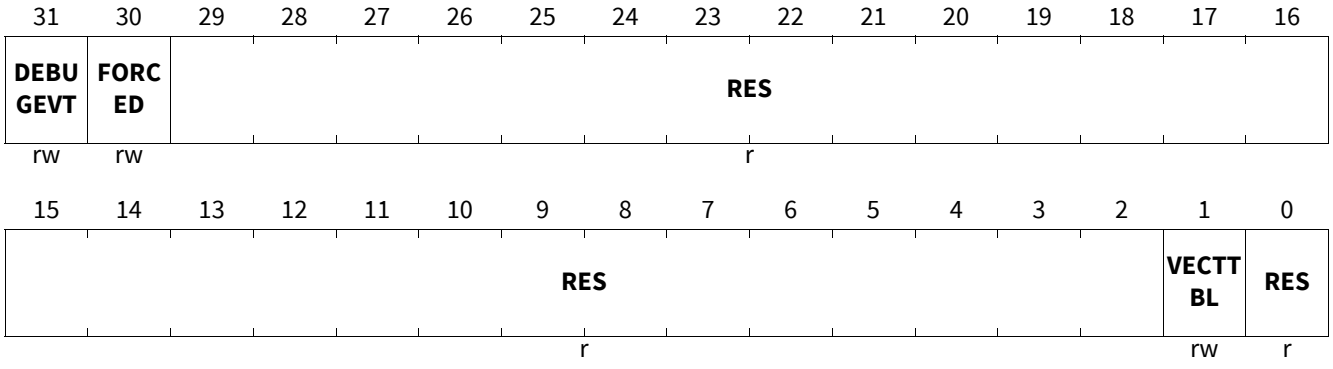
Field	Bits	Type	Description
IACCVIOL	0	rw	**Instruction access violation flag** 0_B **NoVIOL**, No MPU or Execute Never (XN) default memory map access violation has occurred. 1_B **VIOL**, An MPU or Execute Never (XN) default memory map access violation on an instruction fetch has occurred. The fault is signalled only if the instruction is issued.

Field	Bits	Type	Description
DACCVIOL	1	rw	**Data access violation flag** 0_B **NoVIOL**, No data access violation has occurred. 1_B **VIOL**, Data access violation. The MMAR shows the data address that the load or store tried to access.
RES	2, 6, 14, 23:20, 31:26	r	**Reserved**
MUNSTKERR	3	rw	**MemManage fault on unstacking for a return from exception** 0_B **NoERR**, No derived MemManage fault occurred 1_B **ERR**, A derived MemManage fault occurred on exception return
MSTERR	4	rw	**MemManage fault on stacking for exception entry** 0_B **NoERR**, No derived MemManage fault occurred 1_B **ERR**, A derived MemManage fault occurred on exception entry
MLSPERR	5	rw	**MemManage fault during FP lazy state preservation** 0_B **NoERR**, No MemManage fault occurred during FP lazy state preservation 1_B **ERR**, A MemManage fault occurred during FP lazy state preservation
MMARVALID	7	rw	**MemManage Fault Address Register (MMFAR) valid flag** 0_B **Invalid**, MMFAR does not have valid contents. 1_B **Valid**, MMFAR has valid contents.
IBUSERR	8	rw	**Instruction bus error** 0_B **NoERR**, No bus fault on an instruction prefetch has occurred. 1_B **ERR**, A bus fault on an instruction prefetch has occurred. The fault is signalled only if the instruction is issued.
PRECISERR	9	rw	**Precise data bus error** 0_B **NoERR**, No precise data access error has occurred 1_B **ERR**, An imprecise data access error has occurred, and the processor has written the faulting address to the BFAR.
IMPRECISERR	10	rw	**Imprecise data bus error** 0_B **NoERR**, No precise data access error has occurred 1_B **ERR**, An imprecise data access error has occurred.
UNSTKERR	11	rw	**BusFault on unstacking for a return from exception** 0_B **NoERR**, No derived bus fault occurred 1_B **ERR**, A derived bus fault occurred on exception return
STKERR	12	rw	**BusFault on stacking for exception entry** 0_B **NoERR**, No derived bus fault occurred 1_B **ERR**, A derived bus fault occurred on exception entry
LSPERR	13	rw	**BusFault during FP lazy state preservation** 0_B **NoERR**, No bus fault occurred during FP lazy state preservation 1_B **ERR**, A bus fault occurred during FP lazy state preservation
BFARVALID	15	rw	**BusFault Address Register (BFAR) valid flag** 0_B **INVALID**, BFAR does not have valid contents. 1_B **VALID**, BFAR has valid contents.

Field	Bits	Type	Description
UNDEFINSTR	16	rw	**Undefined instruction UsageFault** 0_B **NoUNDEFISTR**, No Undefined Instruction Usage fault has occurred. 1_B **UNDEFISTR**, The processor hat attempted to execute an undefined instruction. This might be an undefined instruction associated with an enabled coprocessor.
INVSTATE	17	rw	**Invalid state UsageFault** 0_B **VALID**, EPSR.T bit and EPSR.IT bits are valid for instruction execution. 1_B **INVALID**, Instruction executed with invalid EPSR.T or EPSR.IT field.
INVPC	18	rw	**Invalid PC load UsageFault** 0_B **NoICERR**, No integrity check error has occurred. 1_B **ICERR**, A integrity check error has occurred.
NOCP	19	rw	**No coprocessor UsageFault** 0_B **NoCPERR**, No coprocessor access error has occurred. 1_B **CPERR**, A coprocessor access error has occurred.
UNALIGNED	24	rw	**Unaligned access UsageFault** Note: Multi-word accesses always fault if not word aligned. Software can configure unaligned word and halfword accesses to fault. 0_B **NoUAERR**, No unaligned access error has occurred. 1_B **UAERR**, A unaligned access error has occurred.
DIVBYZERO	25	rw	**Divide by zero UsageFault** Note: When a SDIV or a UDIV instruction is used with a divisor equal to 0, this fault occurs if CCR.DIV_0_TRP = 1 0_B **NoDIV0**, No Divide by zero error has occurred. 1_B **DIV0**, A divide by zero error has occurred.

Hard Fault Status Register

HFSR
Hard Fault Status Register **($0D2C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
RES	0, 29:2	r	**Reserved**

Field	Bits	Type	Description
VECTTBL	1	rw	**BusFault flag on a vector table read during exception processing** 0_B **NoERR**, No vector table read fault has occurred 1_B **ERR**, Vector table read fault has occurred
FORCED	30	rw	**Forced HardFault flag** 0_B **NoESCAL**, No priority escalation has occurred 1_B **ESCAL**, Processor has escalated a configurable priority exception to HardFault, because it could not be made active due to the priority or because it was disabled.
DEBUGEVT	31	rw	**Debug Event** Note: The processor sets this bit to 1 only when halting debug is disabled and a Debug event occurs. 0_B **NoDBGEV**, No Debug event has occurred 1_B **DBGEV**, Debug event has occurred. The Debug Fault Status Register has been updated.

MemManage Fault Address Register

MMFAR

MemManage Fault Address Register **($0D34_H$)** **RESET_TYPE_3 Value: XXXX $XXXX_H$**

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

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

Field	Bits	Type	Description
ADDRESS	31:0	rw	**Address of the location that generated a MemManage fault** When MMFSR.MMARVALID = 1, this field holds the address of the location that generated the MemManage fault. When an unaligned access faults, the address is the actual address that faulted. Because a single read or write instruction can be split into multiple aligned accesses, the fault address can be any address in the range of the requested access size.

Register description MCU

Bus Fault Address Register

BFAR
Bus Fault Address Register **($0D38_H$)** **RESET_TYPE_3 Value: XXXX XXXX$_H$**

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

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

Field	Bits	Type	Description
ADDRESS	31:0	rw	**Address of the location that generated a BusFault** When BFSR.BFARVALID = 1, this field holds the address of the location that generated the BusFault. When an unaligned access faults the address in the BFAR is the one requested by the instruction, even if it is not the address of the fault.

Auxiliary Fault Status Register

AFSR
Auxiliary Fault Status Register **($0D3C_H$)** **RESET_TYPE_3 Value: 0000 0000$_H$**

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

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

Field	Bits	Type	Description
AUXFAULT	31:0	r	**Additional system fault information** not used - always read 0

Debug Halting Control and Status Register

DHCSR

Debug Halting Control and Status Register (0DF0$_H$) RESET_TYPE_4 Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						S_RESET_ST	S_RETIRE_ST	RES				S_LOCKUP	S_SLEEP	S_HALT	S_REGRDY
r						r	r	r				r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										C_SNAPSTALL	RES	C_MASKINTS	C_STEP	C_HALT	C_DEBUGEN
r										rw	r	rw	rw	rw	rw

Field	Bits	Type	Description
C_DEBUGEN	0	rw	**Debug enable** Setting this bit to 1 via the debugger requires to have DHCSR.C_MASKINTS set to 0, otherwise the behavior is unpredictable. This bit can only be written from the Debug Access Port. Any write operation from the software will be ignored." Pay attention to the layout, no empty lines needed! 0_B **Disabled**, Halting debug disabled 1_B **Enabled**, Halting debug enabled
C_HALT	1	rw	**Processor Halt** 0_B **Run**, Request a halted processor to run 1_B **Halt**, Request a running processor to halt
C_STEP	2	rw	**Processor Step** 0_B **Disabled**, Single-stepping disabled 1_B **Enabled**, Single-stepping enabled
C_MASKINTS	3	rw	**Mask external interrupts, SysTick and PendSV** 0_B **NoIMP**, no action 1_B **AllowIMP**, Allow imprecise entry to Debug state, for example by forcing any stalled load or store instruction to complete.
RES	4, 15:6, 23:20, 31:26	r	**Reserved**

Field	Bits	Type	Description
C_SNAPSTALL	5	rw	**Allow imprecise entry to Debug state** Setting this bit to 1 via the debugger requires to have both DHCSR.C_DEBUGEN and DHCSR.C_HALT set to 1, otherwise the behavior is unpredictable. 0_B **NoIMP**, no action 1_B **AllowIMP**, Allow imprecise entry to Debug state, for example by forcing any stalled load or store instruction to complete.
S_REGRDY	16	r	**Handshake flag for transfers through the DCRDR** 0_B **NotComplete**, There has been a write to the DCRDR, but the transfer is not complete 1_B **Complete**, The transfer to or from the DCRDR is complete
S_HALT	17	r	**Processor halt in debug state** 0_B **NotinDebugstate**, 1_B **InDebugstate**,
S_SLEEP	18	r	**Processor sleep state** 0_B **Notsleeping**, 1_B **Sleeping**,
S_LOCKUP	19	r	**Processor lockup state** 0_B **Notlockedup**, 1_B **Lockedup**,
S_RETIRE_ST	24	r	**Complete instruction flag since last read** 0_B **NoInstructionCompleted**, No instruction has completed since last DHCSR read 1_B **InstructionsCompleted**, At least one instructions has completed since last DHCSR read
S_RESET_ST	25	r	**Processor reset flag since last read** 0_B **NoReset**, No reset since last DHCSR read 1_B **Reset**, At least one reset since last DHCSR read

Register description MCU

Debug Core Register Selector Register

DCRSR

Debug Core Register Selector Register **(0DF4$_H$)** **RESET_TYPE_4 Value: 0000 0000$_H$**

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

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

Field	Bits	Type	Description
REGSEL	6:0	rw	**Selection of the ARM core register or special-purpose register to transfer**
RES	15:7, 31:17	r	**Reserved**
REGWnR	16	rw	**Access type for the transfer** 0_B **Read**, Read Access 1_B **Write**, Write Access

Debug Core Register Data Register

DCRDR

Debug Core Register Data Register **(0DF8$_H$)** **RESET_TYPE_4 Value: 0000 0000$_H$**

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

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

Field	Bits	Type	Description
DBGTMP	31:0	rw	**Data written to the register selected by DCRSR**

Debug Exception and Monitor Control Register

DEMCR

Debug Exception and Monitor Control Register (0DFC$_H$) **Reset Value: Table 105**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES							DWTENA	RES				MON_REQ	MON_STEP	MON_PEND	MON_EN
r							rw	r				rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					VC_HARDERR	VC_INTERR	VC_BUSERR	VC_STATERR	VC_CHKERR	VC_NOCPERR	VC_MMERR	RES			VC_CORERESET
r					rw	rw	rw	rw	rw	rw	rw	r			rw

Field	Bits	Type	Description
VC_CORERESET	0	rw	**Reset vector catch enable** DHCSR.C_DEBUGEN is set to 0, the processor ignores the value of this bit. 0_B **Disabled**, Reset Vector Catch disabled 1_B **Enabled**, Reset Vector Catch enabled, this causes a local reset to halt a running system.
RES	3:1, 15:11, 23:20, 31:25	r	**Reserved**
VC_MMERR	4	rw	**Halting debug trap enable on a MemManage exception** If DHCSR.C_DEBUGEN is set to 0, the processor ignores the value of this bit. 0_B **Disabled**, Halting debug trap disabled 1_B **Enabled**, Halting debug trap enabled
VC_NOCPERR	5	rw	**Halting debug trap enable on a UsageFault caused by an access to the coprocessor** If DHCSR.C_DEBUGEN is set to 0, the processor ignores the value of this bit. 0_B **Disabled**, Halting debug trap disabled 1_B **Enabled**, Halting debug trap enabled

Field	Bits	Type	Description
VC_CHKERR	6	rw	**Halting debug trap enable on a UsageFault caused by a checking error** An alignment check error is an example for a checking error. If DHCSR.C_DEBUGEN is set to 0, the processor ignores the value of this bit. 0_B **Disabled**, Halting debug trap disabled 1_B **Enabled**, Halting debug trap enabled
VC_STATERR	7	rw	**Halting debug trap enable on a UsageFault caused by a state information error** An Undefined Instruction exception is an example for a state information error. If DHCSR.C_DEBUGEN is set to 0, the processor ignores the value of this bit. 0_B **Disabled**, Halting debug trap disabled 1_B **Enabled**, Halting debug trap enabled
VC_BUSERR	8	rw	**Halting debug trap enable on a BusFault exception** If DHCSR.C_DEBUGEN is set to 0, the processor ignores the value of this bit. 0_B **Disabled**, Halting debug trap disabled 1_B **Enabled**, Halting debug trap enabled
VC_INTERR	9	rw	**Halting debug trap enable on a fault during an exception entry or return** If DHCSR.C_DEBUGEN is set to 0, the processor ignores the value of this bit. 0_B **Disabled**, Halting debug trap disabled 1_B **Enabled**, Halting debug trap enabled
VC_HARDERR	10	rw	**Halting debug trap enable on a HardFault exception** If DHCSR.C_DEBUGEN is set to 0, the processor ignores the value of this bit. 0_B **Disabled**, Halting debug trap disabled 1_B **Enabled**, Halting debug trap enabled

Field	Bits	Type	Description
MON_EN	16	rw	**Debug Monitor Exception Enable** If DHCSR.C_DEBUGEN is set to 0, the processor ignores the value of this bit. 0_B **Disabled**, Debug Monitor Exception disabled 1_B **Enabled**, Debug Monitor Exception enabled
MON_PEND	17	rw	**Debug Monitor Pending Exception Set/Clear** When the DebugMonitor exception is pending it becomes active subject to the exception priority rules. A debugger can use this bit to wake up the monitor using the DAP. The effect of setting this bit to 1 is not affected by the value of the MON_EN bit. A debugger can set MON_PEND to 1 and force the processor to take a DebugMonitor exception even when MON_EN is set to 0. 0_B **Not_Pending**, Clear the status of the DebugMonitor exception to not pending 1_B **Pending**, Set the status of the DebugMonitor exception to pending
MON_STEP	18	rw	**Step Request** When MON_EN is set to 0, this feature is disabled and the processor ignores MON_STEP. Setting this bit to 1 makes the step request pending. If this bit is set at an execution priority that is lower than the priority of the DebugMonitor exception, the behavior is unpredictable. 0_B **NoSTEP**, Do not step the processor 1_B **STEP**, Step the processor
MON_REQ	19	rw	**Debug Monitor Semaphore Bit** The processor does not use this bit. The monitor software defines the meaning and use of this bit.
DWTENA	24	rw	**Global enable for the DW unit** 0_B **Disable**, DWT disabled. DWT registers return unknown values on a read operation and it depends on the implementation whether the processor ignores the write operations to DWT. 1_B **Enable**, DWT enabled

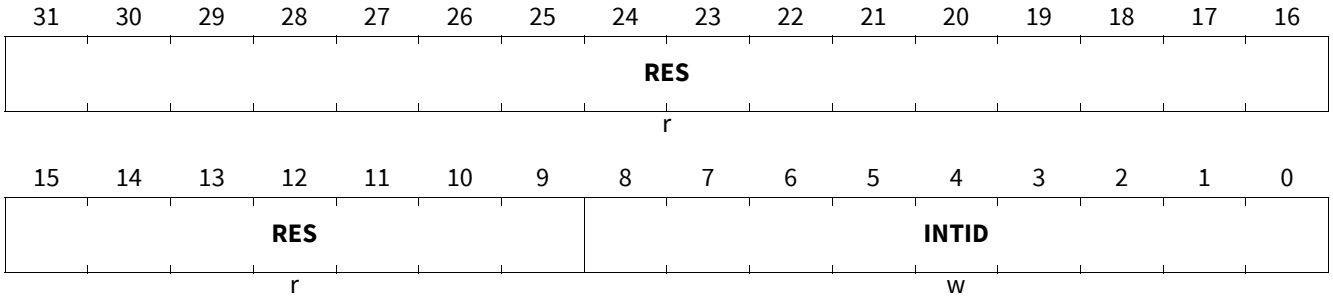
Table 105 Reset Values of DEMCR

Reset Type	Reset Value	Note
RESET_TYPE_3	$XX00\ XXXX_H$	RESET_TYPE_3
RESET_TYPE_4	$00--\ 0000_H$	RESET_TYPE_4

Register description MCU

Software Triggered Interrupt

STIR
Software Triggered Interrupt **(0F00$_H$)** **RESET_TYPE_3 Value: 0000 0000$_H$**



Field	Bits	Type	Description
INTID	8:0	w	**Interrupt ID to be triggered. The value written is (ExceptionNumber - 16)**
RES	31:9	r	**Reserved**

8 System Watchdog Timer (SYSWDT)

8.1 Features overview

The System Watchdog Timer (SYSWDT) belongs to the MCU subsystem. The SYSWDT resets the MCU subsystem in case it is not serviced within a defined time. Therefore it can bring the system into a defined state if the software is not executing according to its normal timing scheme due to a malfunction.

The SYSWDT provides following features:

- 16-bit window watchdog timer
- Programmable watchdog period and window
- Selectable input frequency
- Prewarning interrupt for debug purpose

8.2 Block diagram

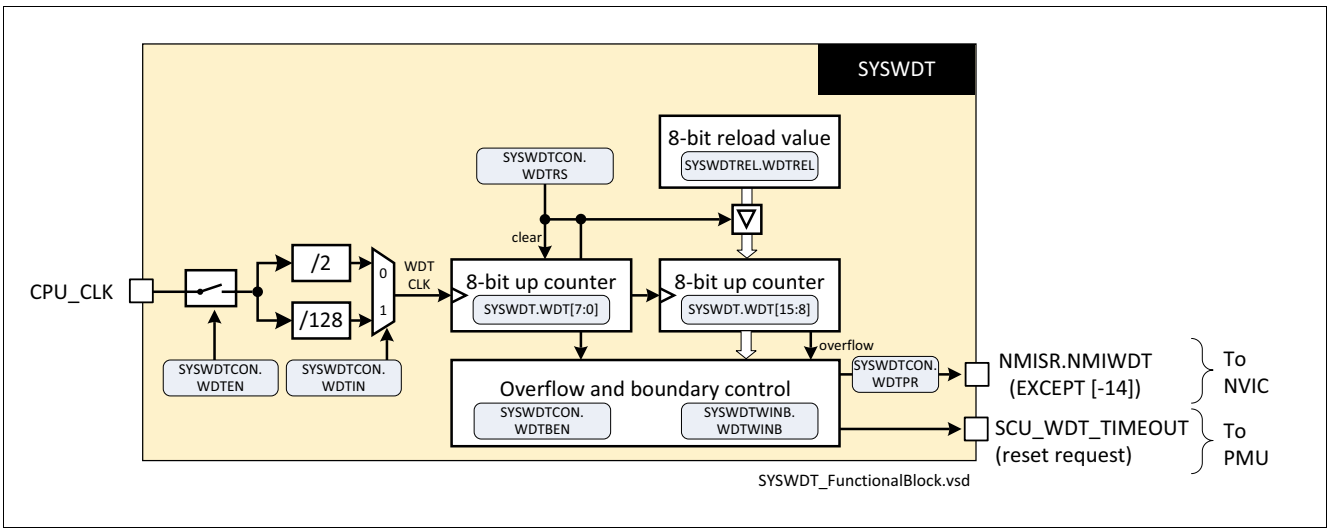


Figure 106 Block diagram SYSWDT

8.3 Toplevel signals

Table 106 SYSWDT power domain

Signal	Direction	Description	From/To
VDDC	input	Switchable supply for digital logic	PMU
VSS		Supply ground	PMU

Table 107 SYSWDT clock domain

Signal	Direction	Description	From/To
CPU_CLK	input	CPU clock	SCU

Table 108 SYSWDT reset

Signal	Direction	Description	From/To
WARM_RESET	input	Resets all digital logic	PMU

Table 109 SYSWDT toplevel block

Signal	Direction	Description	From/To
AHB	input/output	Bus system	CPU
NMIWDT.IRQ	output	SYSWDT interrupt request	SCU
Reset request	output	Request HOT_RESET	PMU

8.4 Interrupts

Events

- The SYSWDT has a prewarning event (WDTPR) which is propagated to SCU’s interrupt node pointer for requesting an NMI on NVIC

8.5 Operation mode behavior

The SYSWDT is part of the MCU subsystem and behaves according to **Table 110**.

Table 110 Operation mode behavior SYSWDT

Reset	The SYSWDT is reset via WARM_RESET. It has following effects: • All SFRs besides bit WDTPR of the SYSWDT are reset
Power-up/ Power-down	• The SYSWDT is kept in reset state as long as its supply and clock is not in the specified operating range • The SYSWDT is released from reset state when its supply or clock is within the specified operating range
Active mode	• The SYSWDT is disabled by default
Stop mode	• The SYSWDT clock is stopped automatically at stop mode entry. The configuration is kept • The SYSWDT clock is started automatically at stop mode exit. The SYSWDT recovers from stop mode to active mode with the configuration at the stop mode entry Note: Before entering stop mode is recommended to disable the SYSWDT or to refresh it.
Sleep mode	• The sleep mode entry transition is like a power-down. The SYSWDT clock and supply are switched off automatically. All configuration settings are lost in sleep mode entry • The sleep mode exit transition is like a power-up. The SYSWDT recovers from sleep mode to active mode from its reset state
Fail-safe state	• In case the SYSWDT is not serviced within the allowed window, a HOT_RESET is requested

8.6 SYSWDT features

The SYSWDT has following features:

- The SYSWDT is a 16-bit up-counter which is clocked by CPU_CLK with two prescaler options. The selection is done via SYSWDTCON.WDTIN.
- The SYSWDT is disabled by default and can be enabled (gated) via SYSWDTCON.WDTEN.
- On overflow of SYSWDT a HOT_RESET is requested and issued by PMU's reset management.
- The SYSWDT displays a prewarning event 48 clocks before overflow in SYSWDTCON.WDTPR. The WDTPR can request an NMI interrupt via SCU. It can be used for software development or diagnostic purpose.
- The SYSWDT can be refreshed via setting SYSWDTCON.WDTRS ("watchdog service") within the allowed time window. Refreshing clears WDTL and reloads WDTH with WDTREL.
- The allowed window for the successful watchdog refresh is by default the full 16 bit period minus the 48 clocks, i.e. count values from 0x0000 to 0xFFCF.
- A watchdog service in the last 48 clocks ("too late") will be ignored, i.e. the SYSWDT will overflow, which leads to a HOT_RESET.
- The allowed service window can be shortened by programming the WDTWINB with an 8 bit granularity. A watchdog service before the counter has reached the window boundary ("too early") leads to a HOT_RESET.
- The occurrence of a SYSWDT reset can be seen in SYSWDTCON.WDTPR or within PMU in RESET_STS.WDT_MCU_RST.

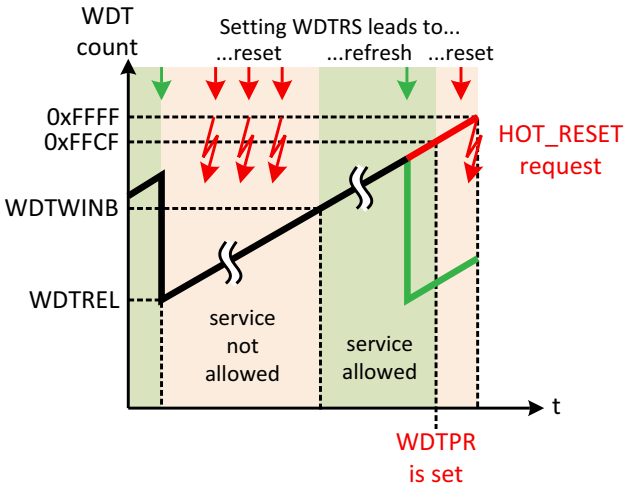


Figure 107 SYSWDT window boundary

The SYSWDT period can be calculated according to **Figure 108**.

$$Period_{SYSWDT} = CPU\_CLK * 2^{(1+\ WDTIN*6)} * (2^{16} - WDTREL * 2^{8})$$

Figure 108 SYSWDT period formula

8.6.1 Suspend features

The SYSWDT is halted when debugging. All SFRs can be accessed by debugger.

8.7 Register descriptions SYSWDT

For a detailed description of the SYSWDT registers click on the page number link in the following table:

Table 111 Register Overview - MEMCTRLREG (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
SYSWDTCON	System Watchdog Timer Control Register	$00B8_H$	363
SYSWDTREL	System Watchdog Timer Reload Register	$00BC_H$	364
SYSWDT	System Watchdog Timer Value	$00C0_H$	364
SYSWDTWINB	System Watchdog Window-Boundary Count	$00C4_H$	365

9 Universal Asynchronous Receiver Transmitter (UART0/1)

9.1 Features overview

The UART0/1 provide a full-duplex asynchronous receiver/transmitter, i.e., it can transmit and receive simultaneously. They are also receive-buffered (1 byte), i.e., they can commence reception of further bytes before a previously received byte has been read from the receive register. However, if the first byte still has not been read by the time the reception of the second byte is complete, the previous byte will be lost. The serial port receive and transmit registers are accessed at Special Function Register (SFR) TXBUF and RXBUF. Writing to TXBUF loads the transmit register, and reading RXBUF accesses a physically separate receive register.

The UART0/1 provides following features:

- Full-duplex asynchronous modes
  - 8-bit or 9-bit data frames, LSB first
  - fixed or variable baud-rate
- Receive buffered (1 Byte)
- Transmit buffered (1 Byte)
- Multiprocessor communication
- Interrupt generation on the completion of a data transmission or reception
- Baud-rate generator with fractional divider for generating a wide range of baud-rates, e.g. 9.6 kBaud, 19.2 kBaud, 115.2 kBaud, 125 kBaud, 250 kBaud, 500 kBaud, 2 MBaud
- Hardware logic for break and synch Byte detection
- Tx inverter logic
- LIN support: connected to timer channel for synchronization to LIN baud-rate

In all modes, transmission is initiated by any instruction that uses TXBUF as a destination register or by writing to the start bit or by an external event. The start selection is programmable. Reception is initiated in the modes by the incoming start bit if REN = 1.

The serial interface also provides interrupt requests when transmission or reception of the frames has been completed. The corresponding interrupt request flags are TI or RI, respectively. If the serial interrupt is not used (i.e., serial interrupt not enabled), TI and RI can also be used for polling the serial interface.

9.2 Block diagram

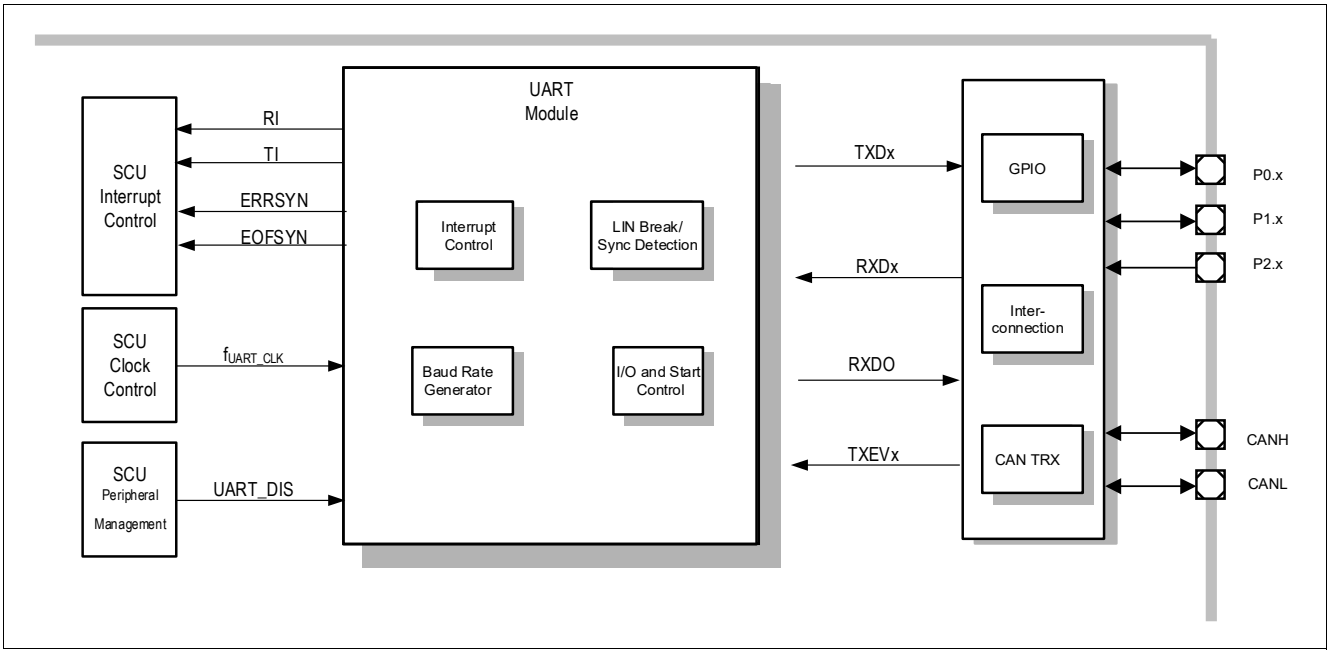


Figure 109 Block diagram UART

9.3 Toplevel signals

Table 112 Toplevel connection

Signal	Direction	Description	From/To
TXD	Output	UART transmit signal	See **Product definitions**, **UART0 interconnections** and **UART1 interconnections**
RXD	Input	UART receive signal	
RXDO	Output	Shifted out in mode 0	
TXEV	Input	External event input	
f_{UART_CLK}	Input	Clock for UART	
RI/TI/ERRSYN/EOFSYN	Output	Interrupt	
UART_DIS	Input	UART disable	

9.4 Interrupts

Events

The UART provides multiple interrupt events:

- TI - Tx Transfer complete interrupt - set when last bit is transmitted
- RI - Rx Receive complete interrupt - set when last receive bit is received and data copied to RXBUF, RB8
- ERRSYS - LIN sync detection error interrupt - errors detected during sync detection
- EOFSYN - LIN sync detection end of sync detected - STOP bit detected at end of 55_H sync byte

Interrupts

An overview of the UART interrupt register handling is shown in **Table 113**.

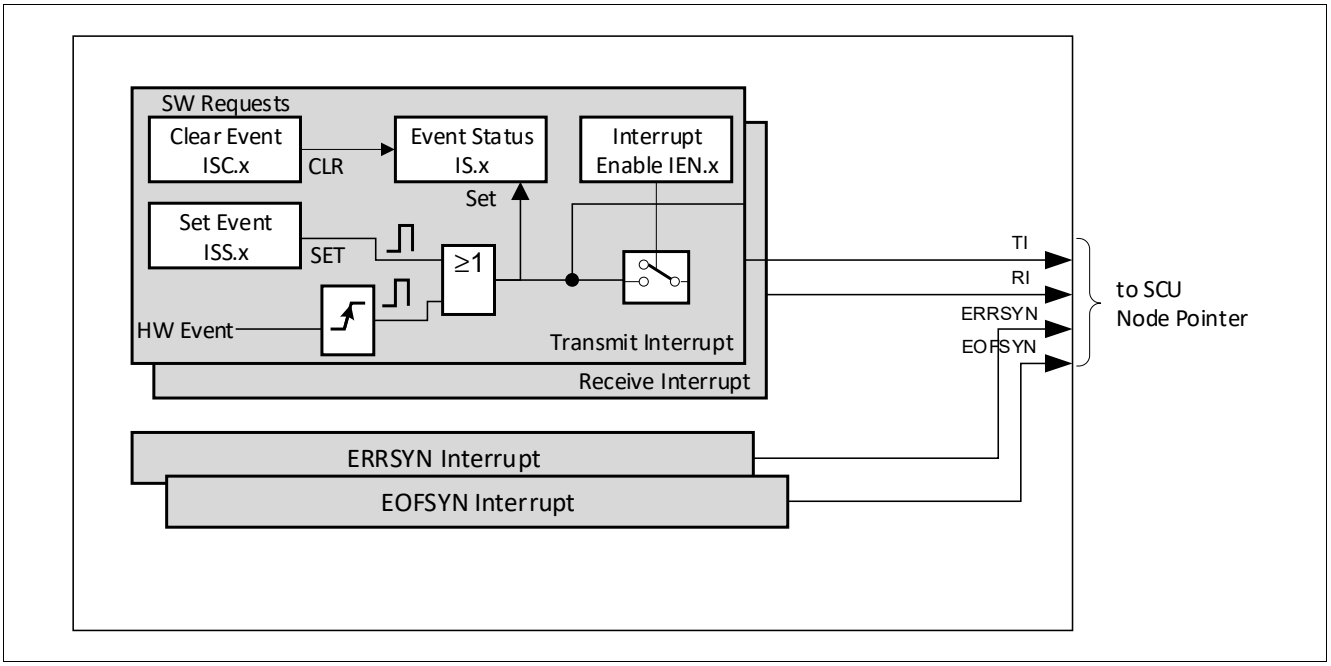


Figure 110 Interrupt handling

Table 113 UART interrupt sources

Interrupt	Interrupt enable bit	Interrupt flag	Interrupt status set bit	Interrupt status clear bit
Reception completed	IEN.RIEN	IS.RI	ISS.RISET	ISC.RICLR
Transmission completed	IEN.TIEN	IS.TI	ISS.TISET	ISC.TICLR
SYN Error	IEN.ERRSYNEN	IS.ERRSYNEN	ISS.ERRSYNSET	ISC.ERRSYNCLR
End of SYN	IEN.EOFSYNEN	IS.EOFSYN	ISS.EOFSYNSET	ISC.EOFSYNCLR

9.5 UART modes

The UART0/1 can be used in four different modes. In mode 0, it operates as an 8-bit shift register. In mode 1, it operates as an 8-Bit serial port. In modes 2 and 3, it operates as a 9-bit serial port. The only difference between mode 2 and mode 3 is the baud-rate, which is fixed in mode 2 but variable in mode 3. The variable baud-rate is set by the underflow rate on the dedicated baud-rate generator.

The different modes are selected by setting bits SM0 and SM1 to their corresponding values, as shown in **Table 114**.

Mode 1 example: 8 data bits, 1 start bit, 1 stop bit, no parity selection, 16 times oversampled, receive & transmit register buffered, Tx/Rx IRQ(s).

Table 114 UART modes

SM0	SM1	Operating mode	Baud-rate
0	0	Mode 0: 8-Bit shift register	$f_{UART\_CLK}/2$
0	1	Mode 1: 8-Bit shift UART	Variable
1	0	Mode 2: 9-Bit shift UART	$f_{UART\_CLK}/64$
1	1	Mode 3: 9-Bit shift UART	Variable

9.5.1 Mode 0, 8-bit shift register, fixed baud-rate

In mode 0, the serial port behaves as an 8-bit shift register. Data is shifted in through RXD, and out through RXDO, while the TXD line is used to provide a shift clock which can be used by external devices to clock data in and out.

The transmission cycle is activated by a write to TXBUF. The data will be written to the transmit shift register with a 1 at the 9th bit position. For the next seven machine cycles, the contents of the transmit shift register are shifted right one position and a zero shifted in from the left so that when the MSB of the data byte is at the output position, it has a 1 and a sequence of zeros to its left. The control block then executes one last shift before setting the TI bit.

Reception is started by the condition REN = 1 and RI = 0. At the start of the reception cycle, $1111\ 1110_B$ is written to the receive shift register. In each machine cycle that follows, the contents of the shift register are shifted left one position and the value sampled on the RXD line in the same machine cycle is shifted in from the right. When the 0 of the initial byte reaches the leftmost position, the control block executes one last shift, loads RXBUF and sets the RI bit.

The baud-rate for the transfer is fixed at $f_{UART\_CLK}/2$ where f_{UART_CLK} is the input clock frequency.

9.5.2 Mode 1, 8-bit UART, variable baud-rate

In mode 1, the UART behaves as an 8-bit serial port. A start bit (0), 8 data bits, and a stop bit (1) are transmitted on TXD or received on RXD at a variable baud-rate.

The transmission cycle is activated by a write to TXBUF. The data are transferred to the transmit shift register and a 1 is loaded to the 9th bit position (as in mode 0). At phase 1 of the machine cycle after the next rollover in the divide-by-16 counter, the start bit is copied to TXD, and data is activated one bit time later. One bit time after the data is activated, the data starts getting shifted right with zeros shifted in from the left. When the MSB gets to the output position, the control block executes one last shift and sets the TI bit.

Reception is started by a high to low transition on RXD (sampled at 16 times the baud-rate). The divide-by-16 counter is then reset and $1111\ 1111_B$ is written to the receive register. If a valid start bit (0) is then detected (based on two out of three samples), it is shifted into the register followed by 8 data bits. If the transition is not followed by a valid start bit, the controller goes back to looking for a high to low transition on RXD. When the

start bit reaches the leftmost position, the control block executes one last shift, then loads RXBUF with the 8 data bits, loads RB8 (SCON.2) with the stop bit, and sets the RI bit, provided RI = 0 (SCON.0), and either SM2 = 0 (SCON.5) (see **Chapter 9.6**) or the received stop bit = 1. If none of these conditions is met, the received byte is lost.

The associated timings for transmit/receive in mode 1 are illustrated in **Figure 111**.

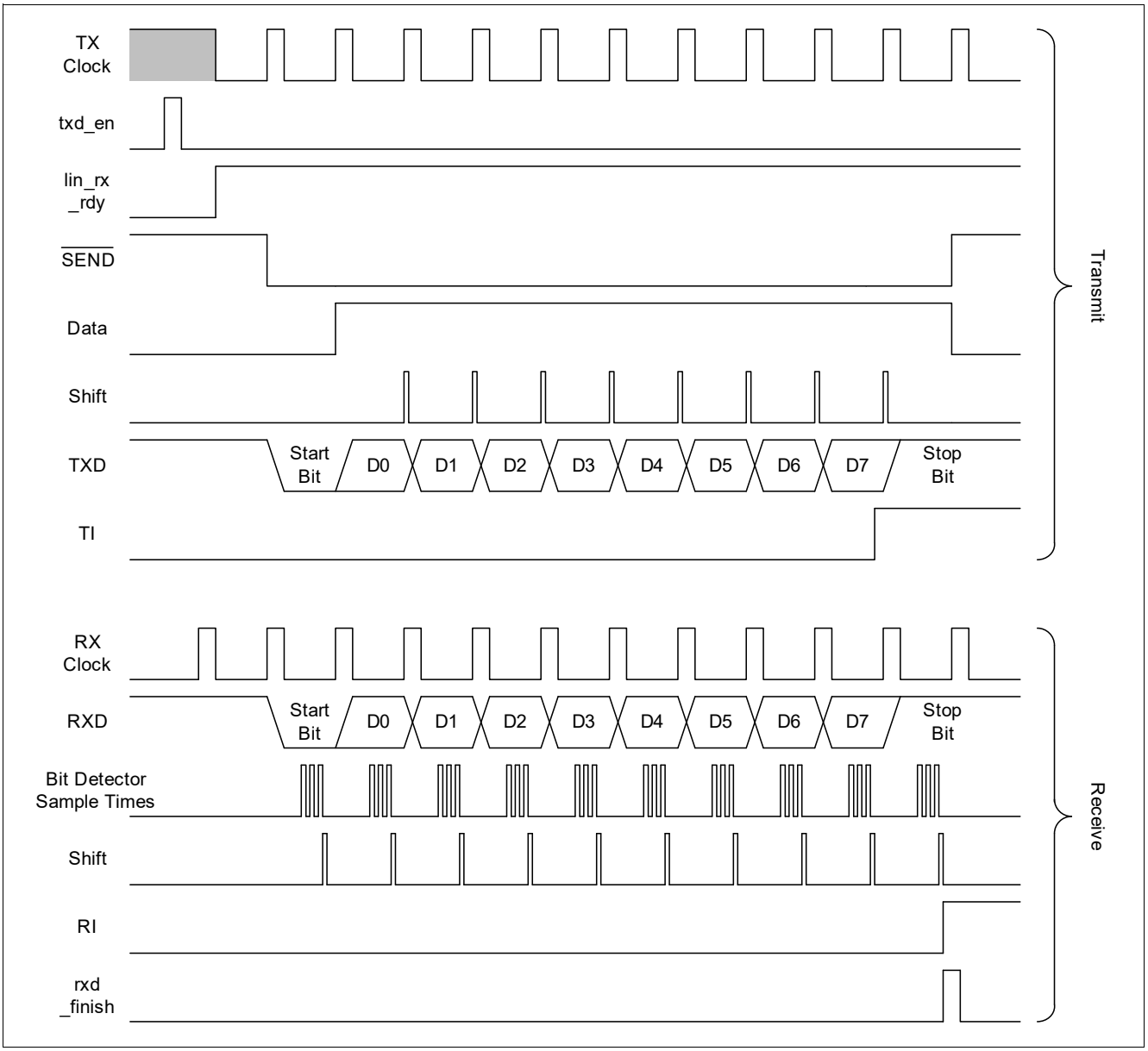


Figure 111 Serial interface, mode 1, timing diagram

9.5.3 Mode 2, 9-bit UART, fixed baud-rate

In mode 2, the UART behaves as a 9-bit serial port. A start bit (0), 8 data bits plus a programmable 9th bit and a stop bit (1) are transmitted on TXD or received on RXD. The 9th bit for transmission is taken from TB8 (SCON.3) while for reception, the 9th bit received is placed in RB8 (SCON.2).

The transmission cycle is activated by a write to TXBUF. The data is transferred to the transmit shift register and TB8 is copied into the 9th bit position. At phase 1 of the machine cycle following the next rollover in the divide-by-16 counter, the start bit is copied to TXD and data is activated one bit time later. One bit time after the data is activated, the data starts shifting right. For the first shift, a stop bit (1) is shifted in from the left and for subsequent shifts, zeros are shifted in. When the TB8 bit gets to the output position, the control block executes one last shift and sets the TI bit.

Reception is started by a high to low transition on RXD (sampled at 16 times of the baud-rate). The divide-by-16 counter is then reset and $1111\ 1111_B$ is written to the receive register. If a valid start bit (0) is then detected (based on two out of three samples), it is shifted into the register followed by 8 data bits. If the transition is not followed by a valid start bit, the controller goes back to looking for a high to low transition on RXD. When the start bit reaches the leftmost position, the control block executes one last shift, then loads RXBUF with the 8 data bits, loads RB8 (SCON.2) with the 9th data bit, and sets the RI bit, provided RI = 0 (SCON.0), and either SM2 = 0 (SCON.5) (see **Chapter 9.6**) or the 9th bit = 1. If none of these conditions is met, the received byte is lost.

The baud-rate for the transfer is fixed at $f_{UART\_CLK}/64$.

9.5.4 Mode 3, 9-bit UART, variable baud-rate

Mode 3 is the same as mode 2 in all respects except that the baud-rate is variable.

The associated timings for transmit/receive in modes 2 and 3 are illustrated in **Figure 112**.

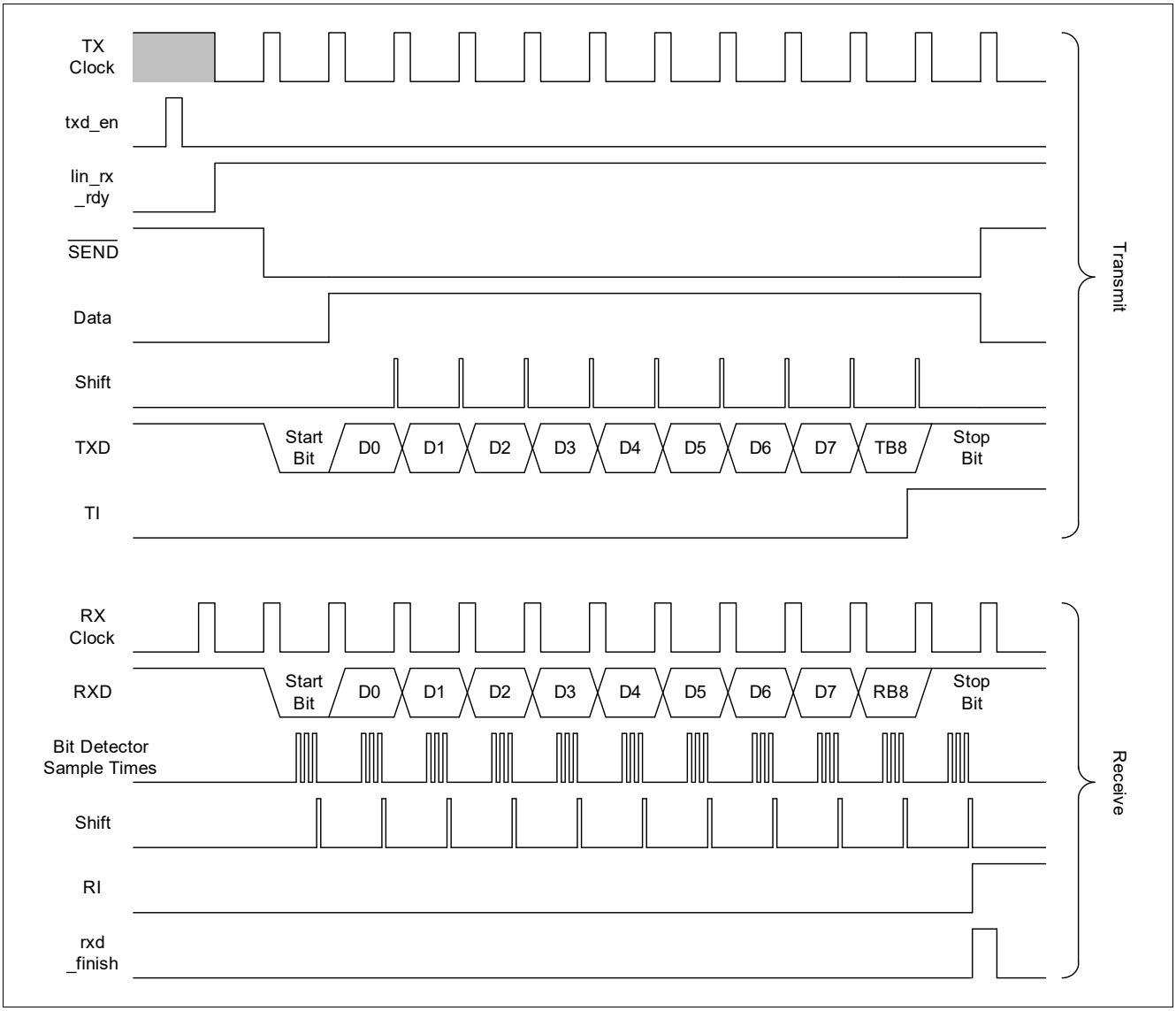


Figure 112 Serial interface, modes 2 and 3, timing diagram

9.6 Multiprocessor communication

Modes 2 and 3 have a special provision for multiprocessor communication using a system of address bytes with bit 9 = 1 and data bytes with bit 9 = 0. In these modes, 9 data bits are received. The 9th data bit goes into RB8 (SCON.2). The communication always ends with one stop bit. The port can be programmed such that when the stop bit is received, the serial port interrupt will be activated only if RB8 = 1.

This feature is enabled by setting bit SM2 in register SCON. One of the ways to use this feature in multiprocessor systems is described in the following paragraph.

When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address byte that identifies the target slave. An address byte differs from a data byte in the 9th bit. The 9th bit in an address byte is 1 and in a data byte the 9th bit is 0. With SM2 = 1, no slave will be interrupted by a data byte. An address byte, however, will interrupt all slaves, so that each slave can examine the received byte and see if it is being addressed. The addressed slave will clear its SM2 bit and prepare to receive the data bytes that will be coming. The slaves that were not being addressed retain their SM2 bits as set and ignore the incoming data bytes.

Note: Bit SM2 has no effect in mode 0. SM2 can be used in mode 1 to check the validity of the stop bit. In a mode 1 reception, if SM2 = 1, the receive interrupt will not be activated unless a valid stop bit is received.

9.7 I/O and start control module (UARTCTRL)

The UARTCTRL module contains logic to select the input signals according to the Alternate Functions used for UART connections. The UARTCTRL also provides the possibility to invert the data output and is responsible for transmission start control.

Input select

For flexible use of the UART modules the RXD and TX Start Event inputs can be selected from each 4 input signals provided by the GPIO Alternate Functions. The UARTCTRL module maps the GPIO Inputs to the UART module according to the setting in INSEL.xSEL.

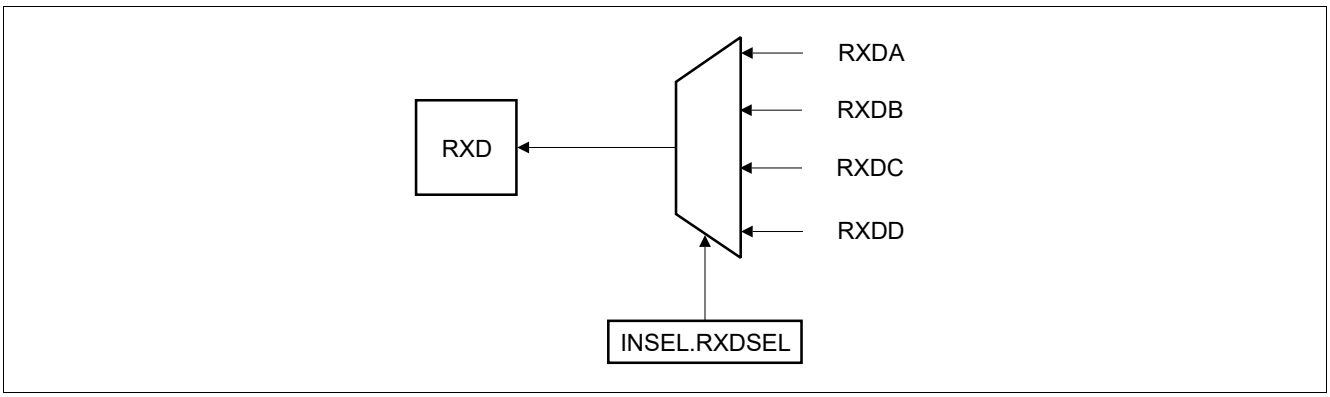


Figure 113 RXD input selection

TDX invert function

The UART provides the possibility to invert the TX Data - txd or rxdo (mode 0) is inverted if SCON.TXINV is set.

Start transmission trigger

An UART transfer can be started by the following events selectable by SCON.TXENSEL

- Write to tx data buffer register TXBUF.TXDATA
- Write to start bit SCON.TXSTART
- A rising edge of the external event input txev selected by INSEL.TXEVSEL

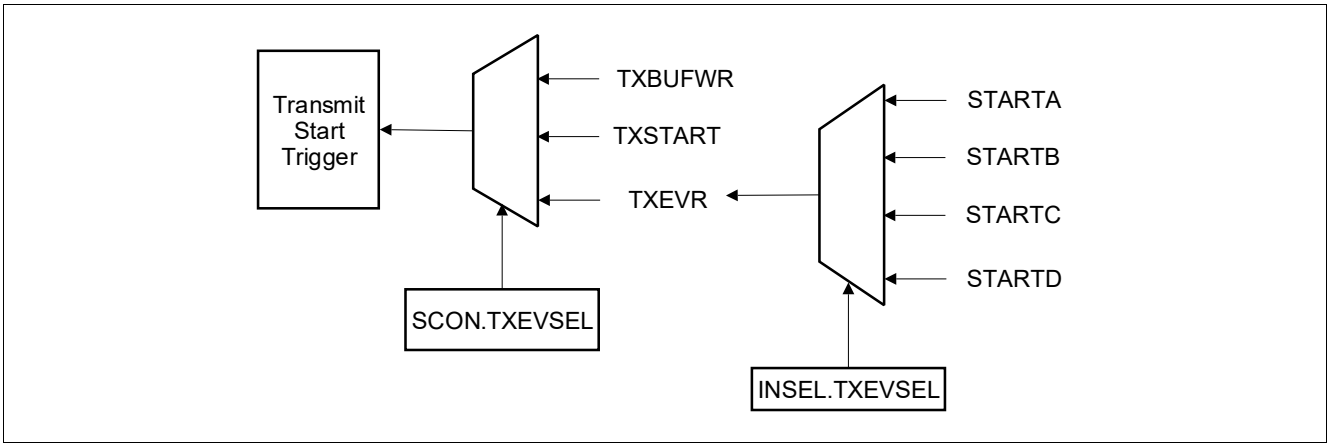


Figure 114 Transmit start trigger

9.8 Baud-rate generation

There are several ways to generate the baud-rate clock for the serial port, depending on the mode in which they are operating.

The baud-rates in modes 0 and 2 are fixed to $f_{UART\_CLK}/2$ and $f_{UART\_CLK}/64$ respectively, while the variable baud-rate in modes 1 and 3 is generated based on the setting of the baud-rate generator.

“Baud-rate clock” and “baud-rate” must be distinguished from each other. The serial interface requires a clock rate that is 16 times the baud-rate for internal synchronization. Therefore, the UART baud-rate generator must provide a “baud-rate clock” to the serial interface where it is divided by 16 to obtain the actual “baud-rate”. The abbreviation f_{UART_CLK} refers to the input clock frequency.

9.8.1 Baud-rate generator

The baud-rate generator is used to generate the variable baud-rate for the UART in modes 1 and 3. It has programmable 11-bit reload value, 3-bit prescaler and 5-bit fractional divider.

The baud-rate generator clock is derived via a prescaler (f_{DIV}) from the input clock f_{UART_CLK}. The baud-rate timer counts downwards and can be started or stopped through the baud-rate control run bit BCON.BR_R. Each underflow of the timer provides one clock pulse to the serial channel. The timer is reloaded with the 11-bit BCON.BG_BR_VALUE each time it underflows. The duration between underflows depends on the ‘n’ value in the fractional divider, which can be selected by the bits BCON.BG_FD_SEL. ‘n’ times out of 32, the timer counts one cycle more than specified by BCON.BG_BR_VALUE. The prescaler is selected by the bits BCON.BR_PRE.

Baud-rate generator support updating the baud-rate during transmission (e.g. after LIN sync detection). To ensure a correct behavior of the transmission the baud-rate registers BCON.BR_PRE, BCON.BG_FD_SEL and BCON.BG_BR_VALUE have shadow registers inside the baud generator module to be updated at an ‘allowed’ point in time. The shadow registers will copy the register values after a new bit-time has started.

The baud-rate of the baud-rate generator depends on the following bits and register values:

- Input clock f_{UART_CLK}
- Value of bit field BCON.BR_PRE
- Value of bit field BCON.BG_FD_SEL
- Value of the 11-bit reload value BCON.BG_BR_VALUE

Figure 115 shows a simplified block diagram of the baud-rate generator.

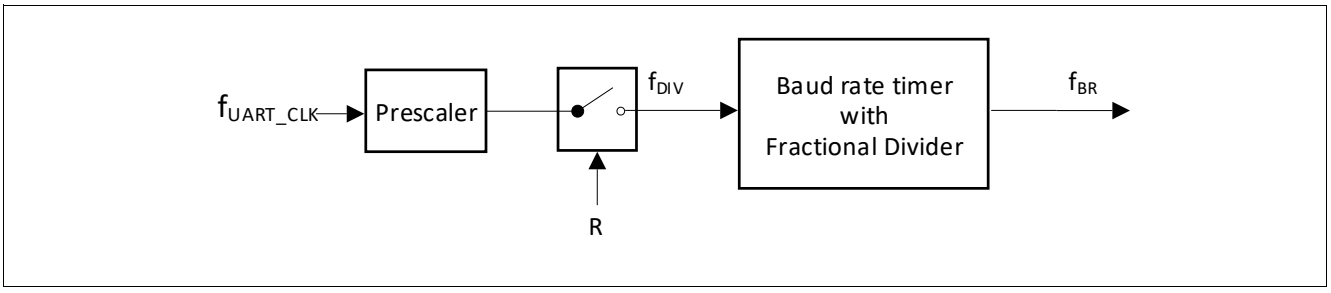


Figure 115 Simplified baud-rate generator block diagram

The following formula calculate the final baud-rate:

$$\text{Baud rate} = \frac{f_{UART\_CLK}}{16 \bullet \text{PRE} \bullet (\text{BR\_VALUE} + \frac{n}{32})} \quad (9.1)$$

The value of PRE (prescaler) is chosen by the bit field BCON.BR_PRE. BR_VALUE represents the contents of the reload value, taken as unsigned 11-bit integer from the bit field BCON.BG_BR_VALUE. n/32 is defined by the fractional divider selection in bit field BCON.BG_FD_SEL.

The maximum baud-rate that can be generated is limited to $f_{UART\_CLK}/32$. Hence, for module clocks of 40 MHz and 80 MHz, the maximum achievable baud-rate is 1.25 MBaud and 2.50 MBaud respectively.

Table 115 lists various commonly used baud-rates together with their corresponding parameter settings and the deviation errors compared to the intended baud-rate.

Table 115 Typical baud-rates of UART (f_{UART_CLK} = 40 MHz)

Baud-rate (f_{UART_CLK} = 40 MHz)	PRE	Reload value (BR_VALUE)	Numerator of fractional value (FD_SEL)	BG register[1)]	Deviation error
250.4 kBaud	1 (BRPRE = 000)	9 (9_H)	31 ($1F_H$)	$0xxx_H$	+0.12%
115.2 kBaud	1 (BRPRE = 000)	21 (15_H)	22 (16_H)	$02B6_H$	+0.06%
20 kBaud	1 (BRPRE = 000)	125 ($7D_H$)	0 (0_H)	$0FA0_H$	0.00%
19.2 kBaud	1 (BRPRE = 000)	130 (82_H)	7 (7_H)	1047_H	-0.01%
9600 Baud	2 (BRPRE = 001)	130 (82_H)	7 (7_H)	1047_H	-0.01%
4800 Baud	4 (BRPRE = 010)	130 (82_H)	7 (7_H)	1047_H	-0.01%
2400 Baud	8 (BRPRE = 011)	130 (82_H)	7 (7_H)	1047_H	-0.01%

1) The value of the 16-bit BG register is obtained by concatenation the 11-bit BRVALUE and 5-bit FD_SEL into a 16-bit value.

9.9 LIN support in UART

The UART module can be used to support the Local Interconnect Network (LIN) protocol for both master and slave operations. The LIN baud-rate detection feature, which consists of the hardware logic for break and synch byte detection, provides the capability to detect the baud-rate within LIN protocol using Timer2. This allows the UART module to be synchronized to the LIN baud-rate for data transmission and reception.

9.9.1 LIN protocol

LIN is a holistic communication concept for local interconnected networks in vehicles. The communication is based on the SCI (UART) data format, a single-master/multiple-slave concept, a clock synchronization for nodes without stabilized time base. An attractive feature of LIN is the self-synchronization of the slave nodes without a crystal or ceramic resonator, which significantly reduces the cost of hardware platform. Hence, the baud-rate must be calculated and returned with every message frame.

The structure of a LIN frame is shown in **Figure 116**. The frame consists of the:

- Header, which comprises a sync break (13-bit time low), synch byte (55_H), and ID field
- Response time
- Data bytes (according to UART protocol)
- Checksum

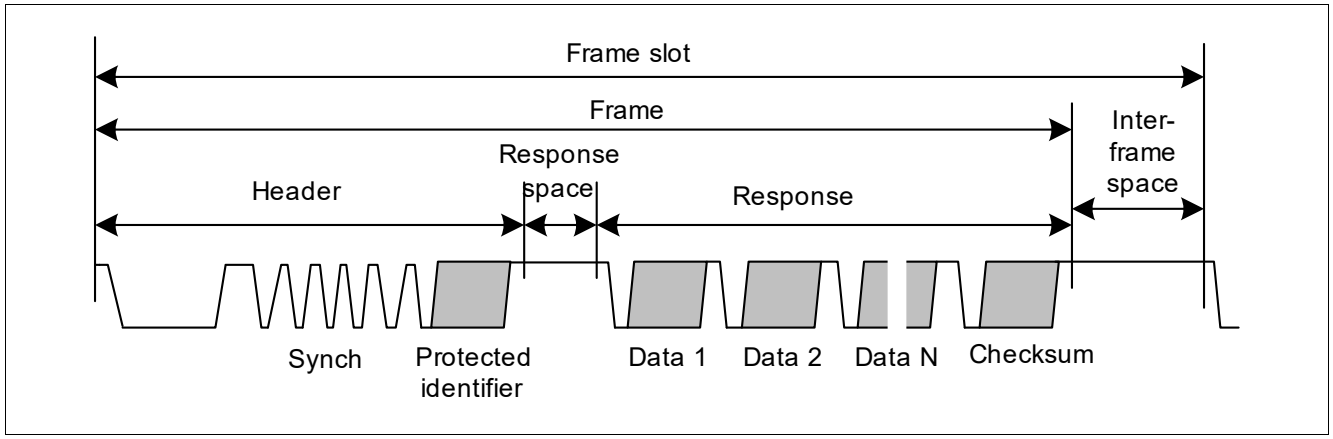


Figure 116 Structure of LIN frame

Each byte field is transmitted as a serial byte, as shown in **Figure 117**. The LSB of the data is sent first and the MSB is sent last. The start bit is encoded as a bit with value zero (dominant) and the stop bit is encoded as a bit with value one (recessive).

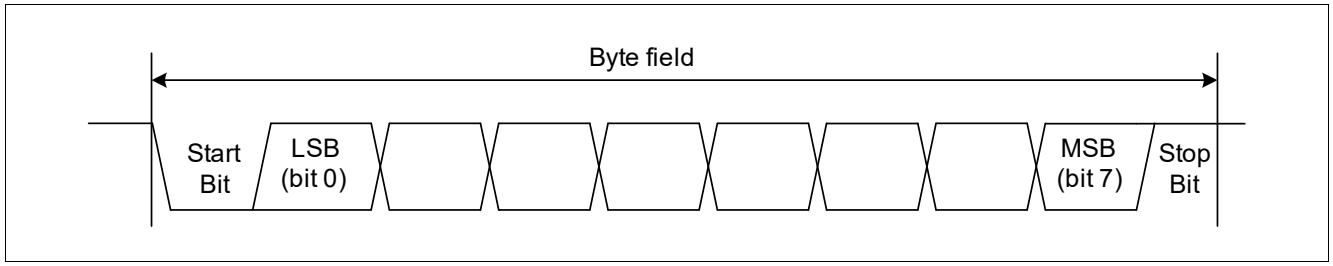


Figure 117 Structure of byte field

The sync break is used to signal the beginning of a new frame. It is the only field that does not comply with **Figure 117**. A sync break is always generated by the master task (in the Master mode) and it must be at least

13 bits of dominant value, including the start bit, followed by a sync break delimiter, as shown in **Figure 118**. The sync break delimiter will be at least one nominal bit time long.

A slave node will use a sync break detection threshold of 11 nominal bit times.

Figure 118 Sync break field

The synch byte is a specific pattern for the determination of the time base. The sync byte field consists of the data value 55_H, as shown in **Figure 119**.

A slave task is always able to detect the sync break/synch sequence, even if it expects a byte field (assuming the byte fields are separated from each other). If this happens, detection of the sync break/synch sequence will abort the transfer in progress and processing of the new frame will commence.

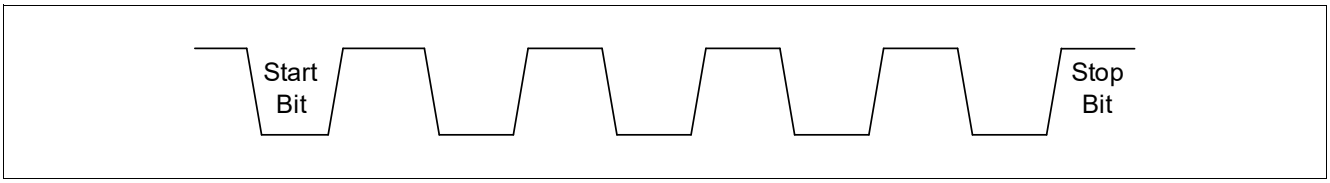


Figure 119 Synch byte field

The slave task will receive and transmit data when an appropriate ID is sent by the master:

1. The slave waits for the synch break
2. The slave synchronizes on the synch byte
3. The slave snoops for the ID
4. According to the ID, the slave determines whether to receive or transmit data, or do nothing
5. When transmitting, the slave sends 2, 4 or 8 data bytes, followed by a check byte

9.9.2 LIN header transmission

LIN header transmission is only applicable in Master mode. In the LIN communication, a master task decides when and which frame is to be transferred on the bus. It also identifies a slave task to provide the data transported by each frame. The information needed for the handshaking between the master and slave, tasks is provided by the master task through the header part of the frame.

The header consists of a sync break and sync byte pattern followed by an identifier. Among these three fields, only the sync break pattern cannot be transmitted as a normal 8-bit UART data. The sync break must contain a dominant value of 13 bits or more to ensure proper synchronization of slave nodes.

In the LIN communication, a slave task is required to be synchronized at the beginning of the protected identifier field of the frame. For this purpose, every frame starts with a sequence consisting of a sync break followed by a synch byte field. This sequence is unique and provides enough information for any slave task to detect the beginning of a new frame and to be synchronized at the start of the identifier field.

9.9.3 Automatic synchronization to the host

Upon entering LIN communication, a connection is established and the transfer speed (baud-rate) of the serial communication partner (host) is automatically synchronized in the following steps that are to be included in the user software:

STEP 1: Initialize interface for reception and timer for baud-rate measurement.

STEP 2: Wait for an incoming LIN frame from host.

STEP 3: Synchronize the baud-rate to the host.

STEP 4: Enter for master request frame or for slave response frame.

The next sections, **Chapter 9.9.4**, **Chapter 9.9.5** and **Chapter 9.9.6** provide some hints on setting up the microcontroller for baud-rate detection of LIN.

Note: Re-synchronization and setup of the baud-rate has always to be done for ***every*** *master request header or slave response header LIN frame by user software.*

9.9.4 Initialization of break/synch field detection logic

The LIN baud-rate detection feature provides the capability to detect the baud-rate within the LIN protocol using Timer2. Initialization consists of:

- Setting of the serial port of the microcontroller to mode 1 (8-bit UART, variable baud-rate) for communication
- Providing the baud-rate range via bit field LINCON.BGSEL
- Toggling of the LINCON.BREN bit (set the bit to 0 before set it back to 1) to initialize the sync break/synch detection logic
- Clearing all status flags LINST.BRK, IS.EOFSYN and IS.ERRSYN to 0
- Setting of Timer2 to capture mode with falling edge trigger at pin T2EX. Setting of the bits T2_MOD.EDGESEL to 0 by default and T2_CON.CP_RL2 to 1
- Enabling Timer2 external events. T2_CON. EXEN2 is set to 1. (EXF2 flag is set when a negative transition occurs at pin T2EX)
- Configuring of f_{T2} by bit field T2_MOD.T2PRE

9.9.5 Baud-rate range selection

The sync break/synch field detection logic supports a maximum number of bits in the sync break field as defined by **Equation (9.2)**.

$$\text{Maximum number of bits} = \text{Baud rate} \bullet \frac{4095}{\text{Sample Frequency}} \tag{9.2}$$

The sample frequency is given by **Equation (9.3)**.

$$\text{Sample Frequency} = \frac{f_{UART\_CLK}}{8 \bullet 2^{BGSEL}} \tag{9.3}$$

If the maximum number of bits in the break field is exceeded, the internal counter will overflow, which results in a baud-rate detection error. Therefore, an appropriate LINST.BGSEL value has to be selected for the required baud-rate detection range.

The baud-rate range defined by different LINST.BGSEL settings is shown in **Table 116**.

Table 116 BGSEL bit field definition for different input frequencies

f_{UART_CLK}	BGSEL	Baud-rate select for detection $f_{UART\_CLK}/(2184*2^{BGSEL})$ to $f_{UART\_CLK}/(72*2^{BGSEL})$
40 MHz	00_B	18.3 kHz to 555.6 kHz
	01_B	9.2 kHz to 277.8 kHz
	10_B	4.6 kHz to 138.9 kHz
	11_B	2.3 kHz to 69.4 kHz

Each BGSEL setting supports a range of baud-rate for detection. If the baud-rate used is outside the defined range, the baud-rate may not be detected correctly.

When f_{UART_CLK} = 40 MHz, the baud-rate range between 2.3 kHz to 555.6 kHz can be detected. The following examples serve as a guide to select the BGSEL value:

- If the baud-rate falls in the range of 2.3 kHz to 4.6 kHz, selected BGSEL value is "11_B"
- If the baud-rate falls in the range of 4.6 kHz to 9.2 kHz, selected BGSEL value is "10_B"
- If the baud-rate falls in the range of 9.2 kHz to 18.3 kHz, selected BGSEL value is "01_B"
- If the baud-rate falls in the range of 18.3 kHz to 555.6 kHz, selected BGSEL value is "00_B"
- If the baud-rate is 20 kHz, the possible values of BGSEL that can be selected are "00_B", "01_B", "10_B", and "11_B". However, it is advisable to select "00_B" for better detection accuracy

9.9.6 LIN baud-rate detection

The baud-rate detection for LIN is shown in **Figure 120**, the header LIN frame consists of the:

- Sync break (13 bit times low)
- Sync byte (55_H)
- Protected ID field

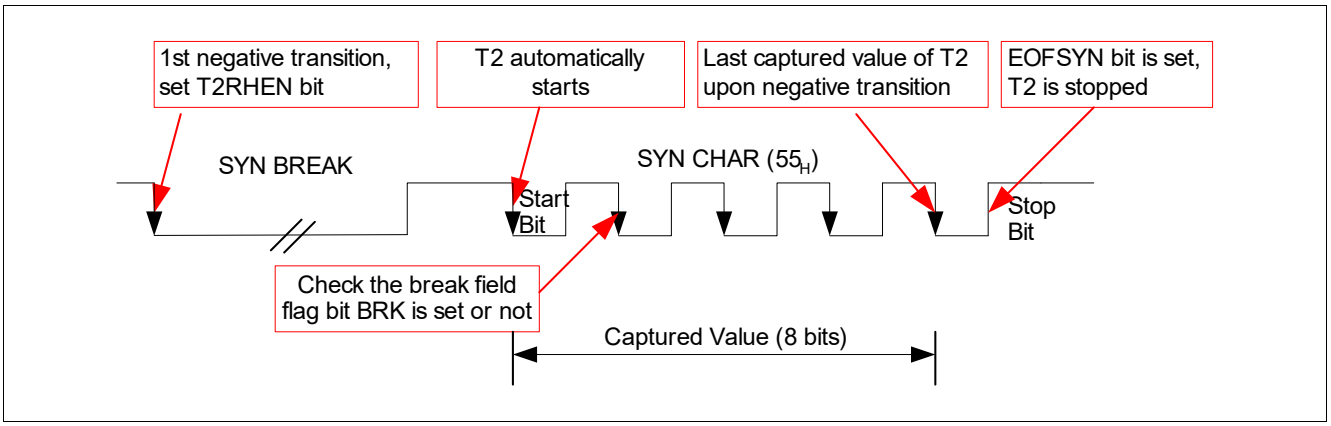


Figure 120 LIN auto baud-rate detection

With the first falling edge:

- The Timer2 external start enable bit (T2_MOD.T2RHEN) is set. The falling edge at pin T2EX is selected by default for Timer2 external start (bit T2_MOD.T2REGS is 0)

With the second falling edge:

- Start Timer2 by the hardware

With the third falling edge:

- Timer2 captures the timing of 2 bits of SYN byte
- Check the break field flag bit LINST.BRK

If the sync break field flag LINST.BRK is set, software may continue to capture 4/6/8 bits of sync byte. Finally, the end of sync byte flag (IS.EOFSYN) is set, Timer2 is stopped. T2 reload/capture register (RC2H/L) is the time taken for 2/4/6/8 bits according to the implementation. Then the LIN routine calculates the actual baud-rate, sets the BR_PRE and BG_BR_VALUE values if the UART module uses the baud-rate generator for baud-rate generation.

After the third falling edge, the software may discard the current operation and continue to detect the next header LIN frame if the following conditions were detected:

- The sync break field flag LINST.BRK is not set, or
- The sync byte error flag IS.ERRSYN is set

9.10 Register description UART0/1

9.10.1 UART0/1 Address Maps

Table 117 Register Address Space

Module	Base Address	End Address	Note
UART0	48018000_H	$4801BFFF_H$	
UART1	$4801C000_H$	$4801FFFF_H$	

Table 118 Register Overview - UART0REG (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
INSEL	Port Input Select Register	0000_H	443
SCON	Serial Channel Control Register	0004_H	443
TSTART	Transmit Start Register	0008_H	444
TXBUF	Serial TX Data Buffer	$000C_H$	445
RXBUF	Serial RX Data Buffer	0010_H	445
IEN	Interrupt Enable Register	0014_H	446
IS	Interrupt Status Register	0018_H	447
ISS	Interrupt Status Set Register	$001C_H$	448
ISC	Interrupt Status Clear Register	0020_H	448
BCON	Baud Rate Control Register	0024_H	449
LINCON	LIN Control Register	0028_H	450
LINST	LIN Status Register	$002C_H$	451
LINSTC	LIN Status Clear Register	0030_H	451
LINSTS	LIN Status Set Register	0034_H	452

9.10.2 UART0REG Registers

Port Input Select Register

INSEL
Port Input Select Register **(0000_H)** **RESET_TYPE_5 Value: 0000 0000_H**

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

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

Field	Bits	Type	Description
RXDSEL	1:0	rw	**RXD Input Select** 00_B **INP0**, RXD inp0 01_B **INP1**, RXD inp1 10_B **INP2**, RXD inp2 11_B **INP3**, RXD inp3
TXEVSEL	3:2	rw	**TX Start Event Input Select** 00_B **INP0**, TXEV inp0 01_B **INP1**, TXEV inp1 10_B **INP2**, TXEV inp2 11_B **INP3**, TXEV inp3
RES	31:4	r	**Reserved** Always read as 0

Serial Channel Control Register

SCON
Serial Channel Control Register **(0004_H)** **RESET_TYPE_5 Value: 0000 0000_H**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						TXENSEL		RES		TXINV	RMOD	REN	SM2	SM1	SM0
r						rw		r		rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SM0	0	rw	**Serial Port Operating Mode Selection** see Table UART Modes

Register description UART0/1

Field	Bits	Type	Description
SM1	1	rw	**Serial Port Operating Mode Selection** see Table UART Modes
SM2	2	rw	**Enable Serial Port Multiprocessor Communication in Mode 2 and 3** Mode 2 or 3: - if SM2 = 1: RI will not be activated if the received 9th data bit (RB8) is 0. Mode 1: - if SM2 = 1: RI will not be activated if no valid stop bit (RB8) was received. Mode 0: - SM2 should be 0.
REN	3	rw	**Enable Receiver of Serial Port** 0_B **Disable**, Serial reception is disabled 1_B **Enable**, Serial reception is enabled
RMOD	4	rw	**Receiver Mode** This bit defines if at the end of Reception the RXBUF Register is updated if IS.RI is still set or RXDATA is lost. 0_B **WaitForInt**, In Mode1-3 RXBUF is not updated if IS.RI is still set 1_B **NoWait**, In Mode1-3 RXBUF is updated if IS.RI is still set
TXINV	5	rw	**TX Data Inverter Enable** 0_B **Disable**, Serial TXD is not inverted 1_B **Enable**, Serial TXD is inverted
RES	7:6, 31:10	r	**Reserved** Always read as 0
TXENSEL	9:8	rw	**Transmit Start Trigger Select** 00_B **TXBUFWR**, Write to TXBUF starts transmission 01_B **TXSTART**, Write to TXSTART starts transmission 10_B **TXEVR**, TXEV rising edge starts transmission 11_B **RES**, Reserved - TXEV rising edge starts transmission

Transmit Start Register

TSTART

Transmit Start Register **(0008_H)** **RESET_TYPE_5 Value: 0000 0000_H**

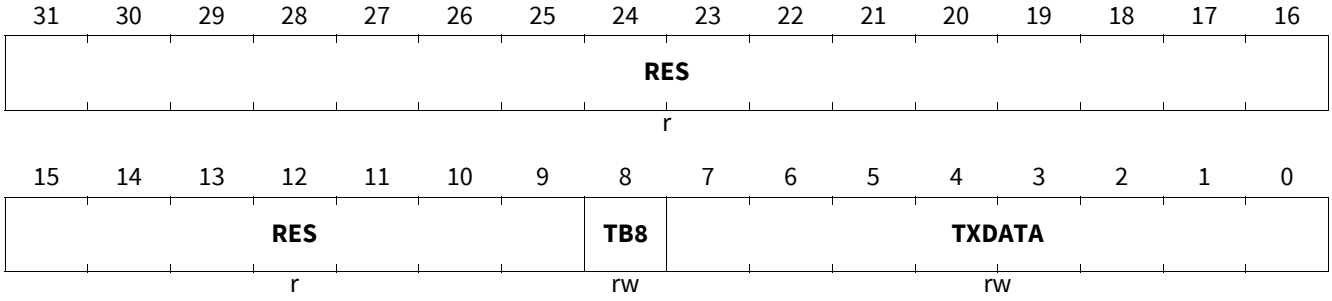
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															TXSTART
r															w

Field	Bits	Type	Description
TXSTART	0	w	**Transmit Start Bit** 0_B **IDLE**, No Data Transmission 1_B **START**, Start Data Transmission
RES	31:1	r	**Reserved** Always read as 0

Serial TX Data Buffer

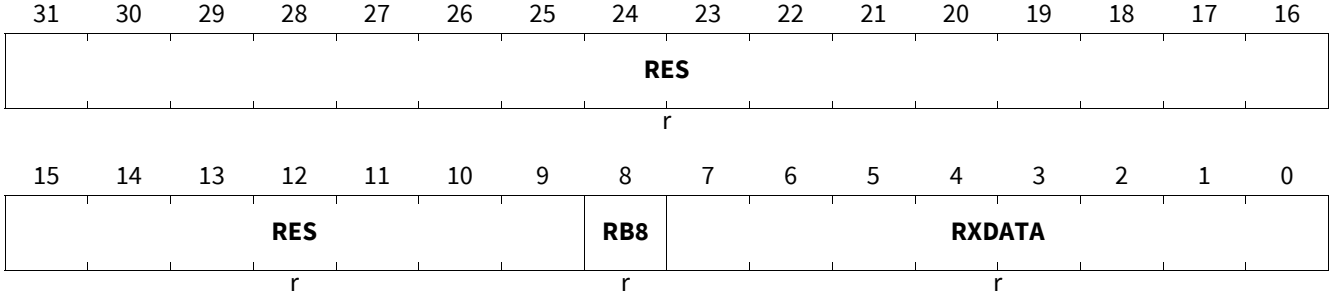
TXBUF
Serial TX Data Buffer **($000C_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**



Field	Bits	Type	Description
TXDATA	7:0	rw	**Serial Interface TxBuffer Register**
TB8	8	rw	**Serial Port Transmitter Bit 9** In modes 2 and 3, this is the 9th data bit sent. In mode 1, this bit is set to 1 In mode 0, this bit is set to 1
RES	31:9	r	**Reserved** Always read as 0

Serial RX Data Buffer

RXBUF
Serial RX Data Buffer **(0010_H)** **RESET_TYPE_5 Value: 0000 0000$_H$**



Field	Bits	Type	Description
RXDATA	7:0	r	**Serial Interface RxBuffer Register**

Register description UART0/1

Field	Bits	Type	Description
RB8	8	r	**Serial Port Receiver Bit 9** In modes 2 and 3, this is the 9th data bit received. In mode 1, this is the stop bit received. In mode 0, this bit is not used.
RES	31:9	r	**Reserved** Always read as 0

Interrupt Enable Register

IEN
Interrupt Enable Register **(0014_H)** **RESET_TYPE_5 Value: 0000 0000_H**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						EOFSYNEN	ERRSYNEN	RES						RIEN	TIEN
r						rw	rw	r						rw	rw

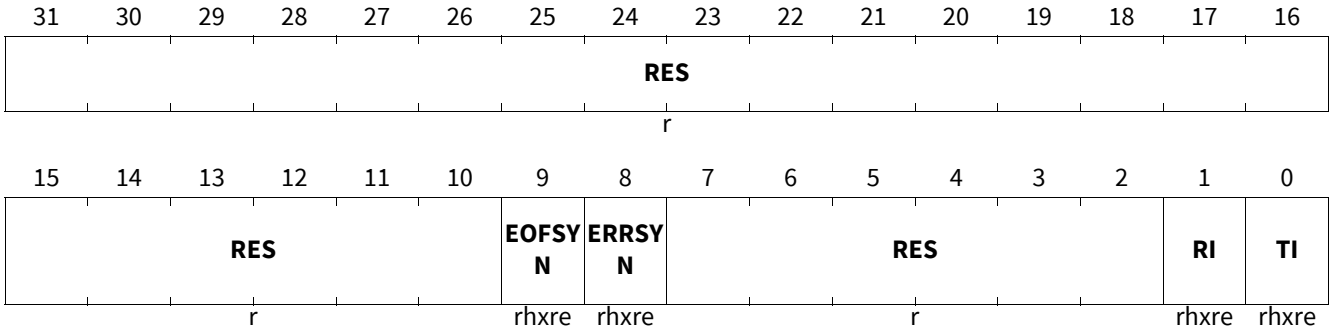
Field	Bits	Type	Description
TIEN	0	rw	**Transmit Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
RIEN	1	rw	**Receive Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
RES	7:2, 31:10	r	**Reserved** Always read as 0
ERRSYNEN	8	rw	**SYN Error Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
EOFSYNEN	9	rw	**End of SYN Byte Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled

Register description UART0/1

Interrupt Status Register

IS

Interrupt Status Register **(0018_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
TI	0	rhxre	**Transmit Buffer Empty Interrupt Flag** This is set by hardware at the end of the 8th bit in mode 0, or at the beginning of the stop bit in modes 1, 2, and 3. This bit is set by hardware and can only be cleared by software. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
RI	1	rhxre	**Receive Interrupt Flag** This is set by hardware at the end of the 8th bit on mode 0, or at the half point of the stop bit in modes 1, 2, and 3. This bit is set by hardware and can only be cleared by software. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
RES	7:2, 31:10	r	**Reserved** Always read as 0
ERRSYN	8	rhxre	**SYN Error Interrupt Flag** This bit is set by hardware and can only be cleared by software 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
EOFSYN	9	rhxre	**End of SYN Byte Interrupt Flag** This bit is set by hardware and can only be cleared by software 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred

Register description UART0/1

Interrupt Status Set Register

ISS

Interrupt Status Set Register **($001C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						EOFSYNSET	ERRSYNSET	RES						RISET	TISET
r						w	w	r						w	w

Field	Bits	Type	Description
TISET	0	w	**Transmit Interrupt Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
RISET	1	w	**Receive Interrupt Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not set 1_B **SET**, Interrupt set
RES	7:2, 31:10	r	**Reserved** Always read as 0
ERRSYNSET	8	w	**SYN Error Interrupt Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not set 1_B **SET**, Interrupt set
EOFSYNSET	9	w	**End of SYN Byte Interrupt Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not set 1_B **SET**, Interrupt set

Interrupt Status Clear Register

ISC

Interrupt Status Clear Register **(0020_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

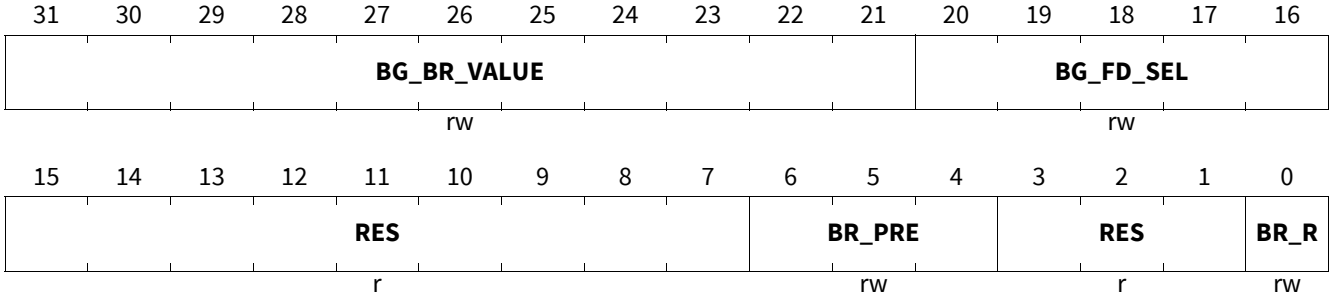
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						EOFSYNCLR	ERRSYNCLR	RES						RICLR	TICLR
r						w	w	r						w	w

Field	Bits	Type	Description
TICLR	0	w	**Transmit Interrupt Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
RICLR	1	w	**Receive Interrupt Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
RES	7:2, 31:10	r	**Reserved** Always read as 0
ERRSYNCLR	8	w	**SYN Error Interrupt Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
EOFSYNCLR	9	w	**End of SYN Byte Interrupt Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared

Baud Rate Control Register

BCON
Baud Rate Control Register **(0024_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
BR_R	0	rw	**Baud Rate Generator Enable Bit** 0_B **Disable**, Baud Rate Generator disabled 1_B **Enable**, Baud Rate Generator enabled
RES	3:1, 15:7	r	**Reserved** Always read as 0

Register description UART0/1

Field	Bits	Type	Description
BR_PRE	6:4	rw	**Prescaler Bit** Selects the baud-rate generator input clock divider f_{DIV} which is derived from the uart_clk clock. 000_B **DIV_1**, fdiv = uart_clk 001_B **DIV_2**, fdiv = uart_clk/2 010_B **DIV_4**, fdiv = uart_clk/4 011_B **DIV_8**, fdiv = uart_clk/8 100_B **DIV_16**, fdiv = uart_clk/16 101_B **DIV_32**, fdiv = uart_clk/32 110_B **NOTUSED6**, fdiv = uart_clk 111_B **NOTUSED7**, fdiv = uart_clk
BG_FD_SEL	20:16	rw	**Fractional Divider Selection** Selects the fractional divider to n/32, where n is the value of BG_FD_SEL and is in the range of 0 to 31. For example, writing 0001_b to BG_FD_SEL selects the fractional divider to 1/32 Note: Fractional divider has no effect if BG_BR_VALUE = 000_H
BG_BR_VALUE	31:21	rw	**Baud Rate Reload Value** 11-bit Baud Rate Timer Reload Value. Note: A value of 0_H sets the Timer into Bypass Mode

LIN Control Register

LINCON
LIN Control Register **(0028_H)** **RESET_TYPE_5 Value: 0000 0000_H**

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

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

Field	Bits	Type	Description
BREN	0	rw	**Baud Rate Detection Enable** 0_B **Disable**, Break/Sync detection is disabled 1_B **Enable**, Break/Sync detection is enabled
BGSEL	2:1	rw	**Baud Rate Select for Detection** For different values of BGSEL the baud rate range for detection is defined by the following formula: $f_{uart\_clk}/(2182*2\text{^}BGSEL) < \text{baud rate range} < f_{uart\_clk}/(72*2\text{^}BGSEL)$ where BGSEL = 00_b, 01_b, 10_b, 11_b.
RES	31:3	r	**Reserved** Always read as 0

Register description UART0/1

LIN Status Register

LINST

LIN Status Register **($002C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															BRK
r															rhxre

Field	Bits	Type	Description
BRK	0	rhxre	**Break Field Detection Flag** This bit is set by hardware and can only be cleared by software. 0_B **NoBRK**, Break Field is not detected 1_B **BRK**, Break Field is detected
RES	31:1	r	**Reserved** Always read as 0

LIN Status Clear Register

LINSTC

LIN Status Clear Register **(0030_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															BRKCLR
r															w

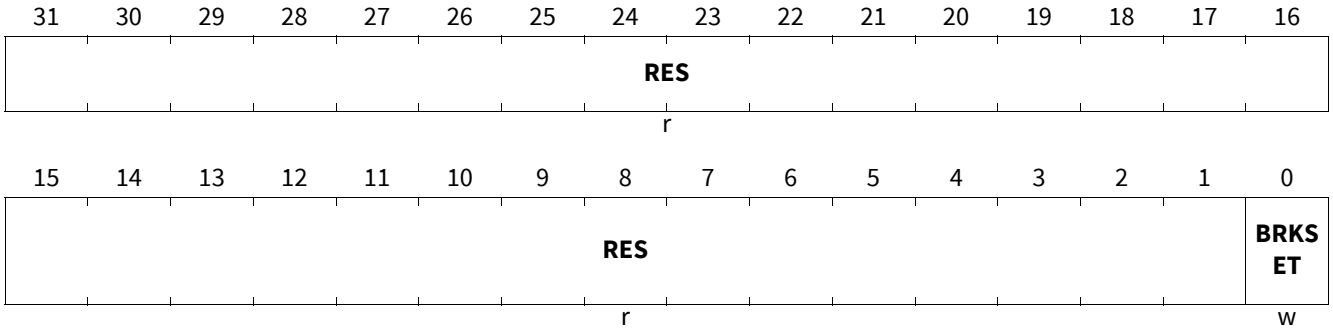
Field	Bits	Type	Description
BRKCLR	0	w	**Break Field Flag Clear** This bit is set by software and automatically cleared by hardware. 0_B **NotCleared**, Break Field is not cleared 1_B **Cleared**, Break Field is cleared
RES	31:1	r	**Reserved** Always read as 0

Register description UART0/1

LIN Status Set Register

LINSTS

LIN Status Set Register **(0034_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
BRKSET	0	w	**Break Field Flag Set** This bit is set by software and automatically cleared by hardware. 0_B **NoSET**, Break Field is not set 1_B **SET**, Break Field is set
RES	31:1	r	**Reserved** Always read as 0

10 High-Speed Synchronous Serial Interface (SSC0/1)

10.1 Features overview

The two high-speed synchronous serial interfaces SSC0/1 support both full-duplex and half-duplex serial synchronous communication.

The SSC0/1 provides following features:

- Master and Slave Mode operation
  - Full-duplex or half-duplex operation
- Transmit and receive buffered
- Flexible data format
  - Programmable number of data bits: 2 to 64-bits
  - Programmable shift direction: Least Significant Bit (LSB) or Most Significant Bit (MSB) shift first
  - Programmable clock polarity: idle low or high state for the shift clock
  - Programmable clock/data phase: data shift with leading or trailing edge of the shift clock
- Variable baud-rate, up to 15 MBaud (Slave mode), 30 MBaud (Master Mode)
- Chip Select (Master), for 1 … 4 slaves
- Chip Select (Slave)
- Compatible with Serial Peripheral Interface (SPI)
- Interrupt generation
  - Interrupt on a transmitter empty condition
  - Interrupt on a receiver full condition
  - Interrupt on an error condition (receive, phase, baud-rate, transmit error)

10.2 Block diagram

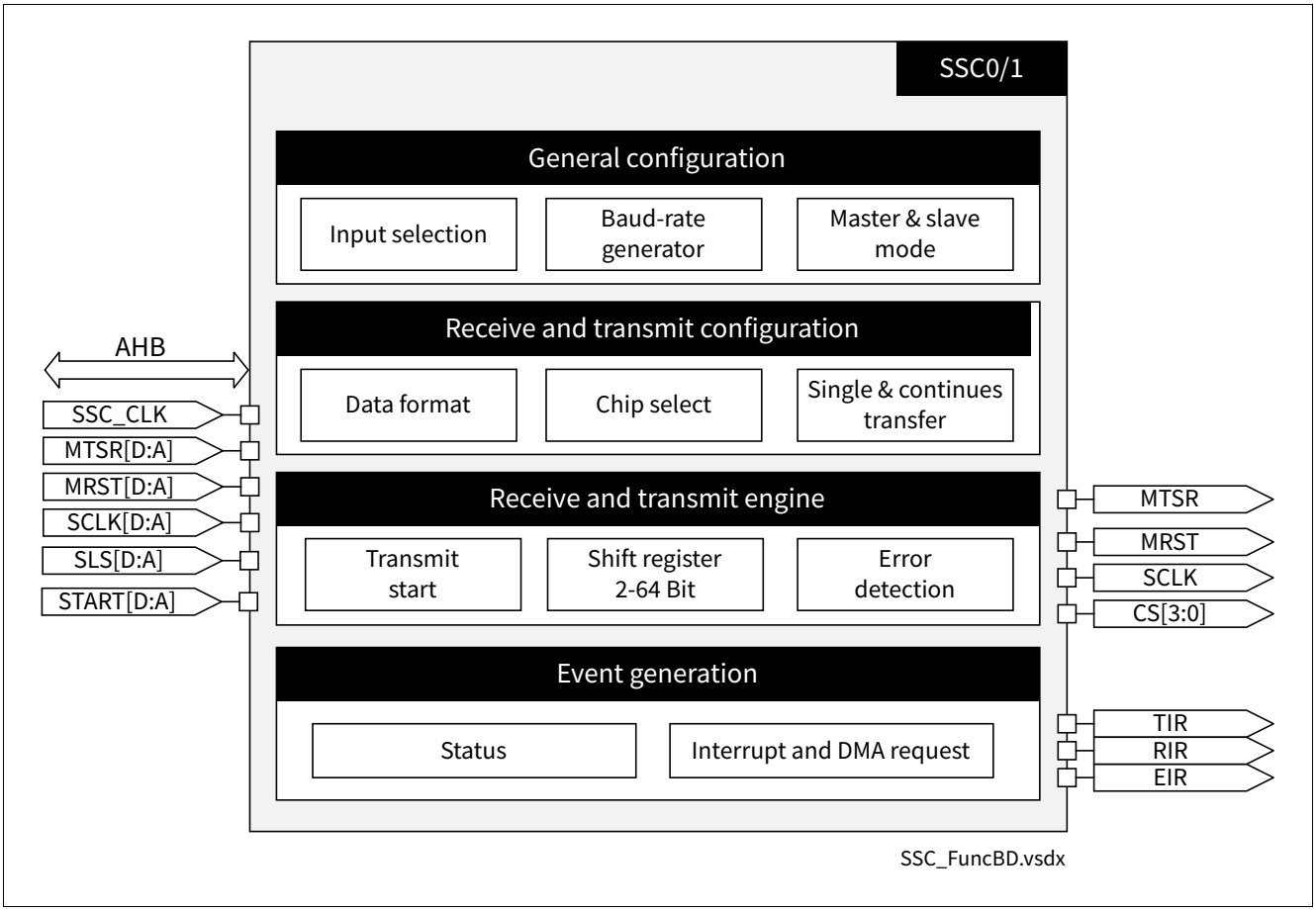


Figure 121 Block diagram SSC

10.3 Toplevel signals

Table 119 Interfaces (input/output

Signal	Direction	Description	From/To
SSC_CLK	Input	Clock for SSC module	See **Product definitions**, **SSC0 interconnections**, **SSC1 interconnections**
MRST[D:A]	Input	Master receive/slave Transmit data input as master	
MTSR[D:A]	Input	Master transmit/slave Receive data input as slave	
SCLK[D:A]	Input	Serial shift clock as slave	
SLS[D:A]	Input	Chip select input as slave	
START[D:A]	Input	External transmit start event	
MTSR	Output	Master transmit/slave Receive data output as master	
MRST	Output	Master receive/slave Transmit data output as slave	
SCLK	Output	Serial shift clock output as master	
CS[3:0]	Output	Chip select output as master	
TIR	Output	Transmit interrupt request	
RIR	Output	Receive interrupt request	
EIR	Output	Error interrupt request	

10.4 Interrupts

Interrupt requests

The SSC0/1has following events which can request an interrupt:

- Error interrupt request EIR (ored REIR, PEIR, TEIR, BEIR)
- Receive interrupt request RIR
- Transmit interrupt request TIR

DMA service requests

- Receive interrupt request RIR
- Transmit interrupt request TIR

10.5 Operation mode behavior

Table 120 Operation mode behavior SSC0/1

Reset	The SSC0/1 module is reset via RESET_TYPE_5: • All SFRs are reset • Receive and transmit state machines are in idle state • All events are reset
Power-up / Power-down	• The SSC0/1 is kept in reset state as long as supply is not in specified operating range • The SSC0/1 is released from reset state when supply is in specified operating range
Active mode	• The SSC0/1 module can be initialized and is operational
Stop mode	• The SSC0/1 module is off powered
Sleep mode	• The SSC0/1 module is off powered
Fail-sleep mode	• The SSC0/1 module is off powered

10.6 Application use cases

The SSC is an on-board synchronous communication interface. It can be used in different setups.

Example 1: SSC as master to one or several slave devices (full-duplex)

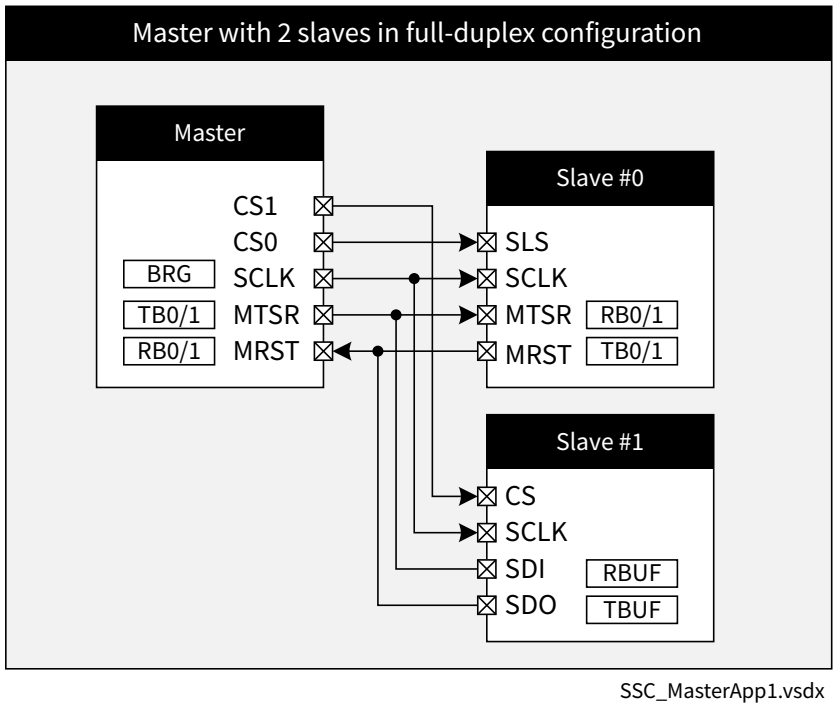


Figure 122 Master and two slaves full-duplex

Example 2: SSC as master to one or several slave devices (half-duplex)

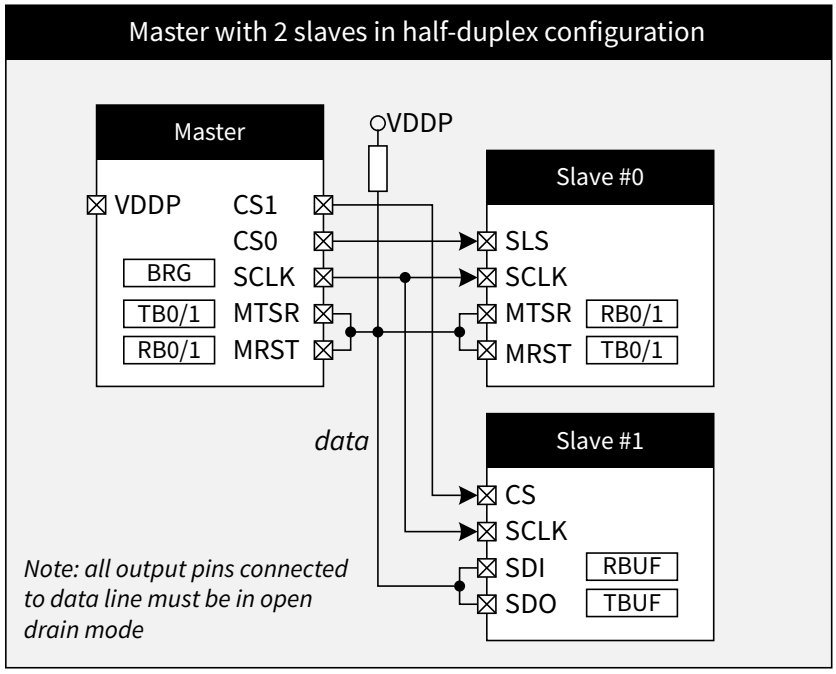


Figure 123 Master and two slaves half-duplex

Notes

1. *In half-duplex mode the output pins (GPIOs) have to be configured to open-drain mode and an external pull-up device is required.*
2. *Half-duplex mode has to respect a time-multiplexed communication scheme. This means that only one participant (sender) is allowed to transmit data on the shared MTSR/MTSR line at a time. Due to simultaneous receive and transmit, the transmit buffer (TB0/1) content of the receiver may corrupt the receiving data. Therefore the receiver's TB0/1 should contain ONES.*

Example 3: Slave to one master device (full duplex)

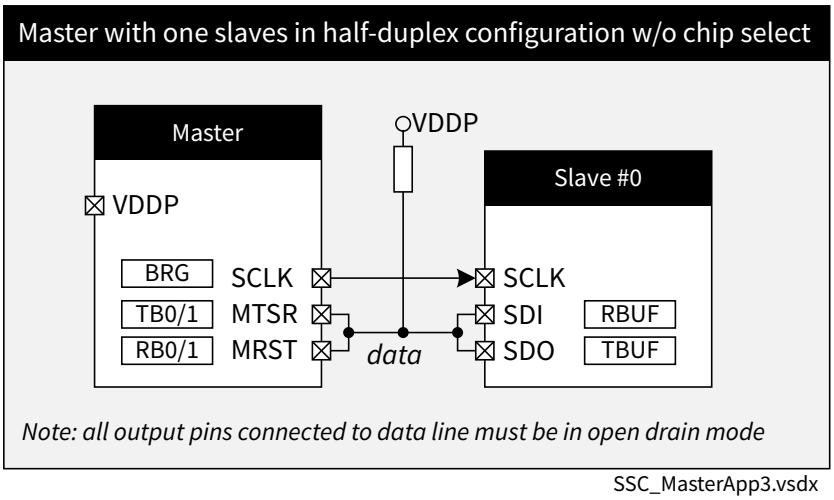


Figure 124 Master and one slave half-duplex w/o chip select

10.7 General configuration

10.7.1 Input selection (INSEL)

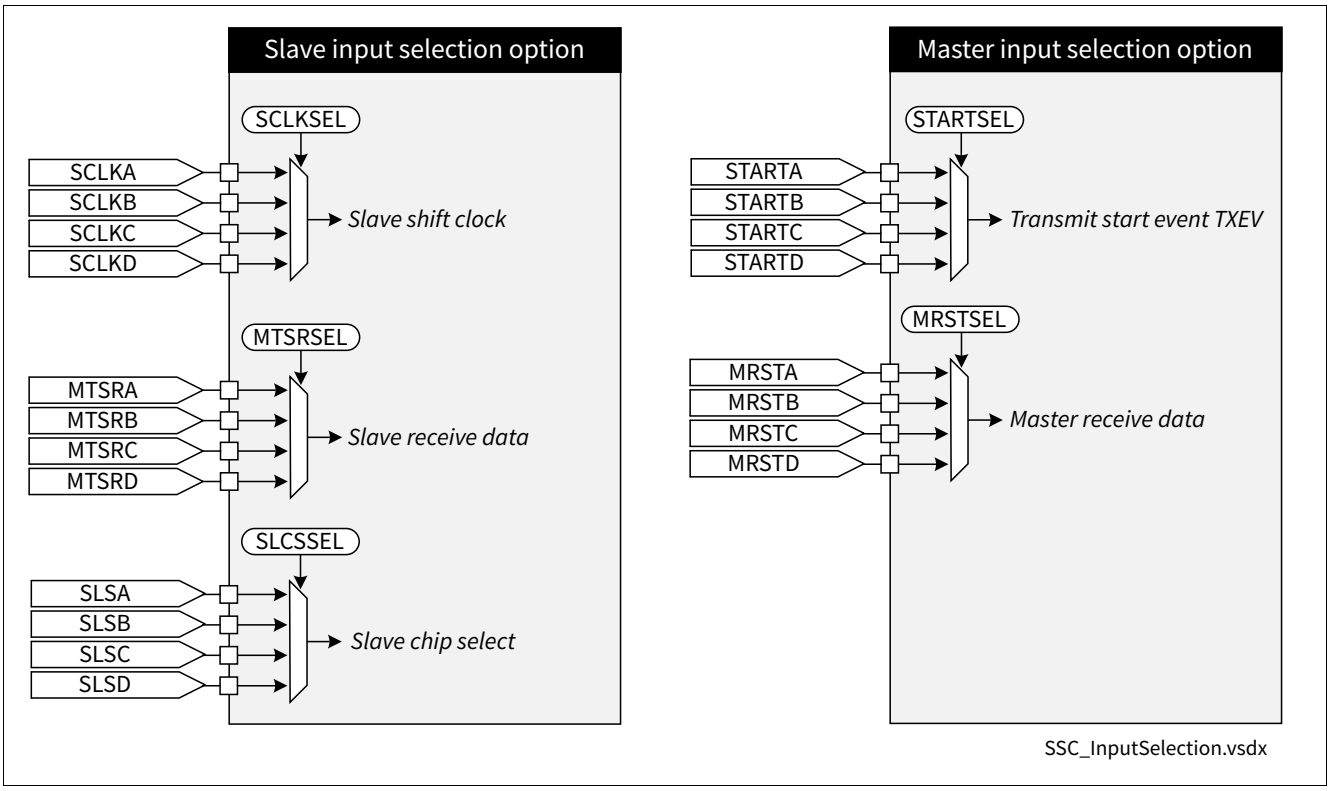


Figure 125 Input selection options

Up to four input options are available for slave receive data MTSR[D:A], shift clock SCLK[D:A] and chip select SLS[D:A].

Up to four input options are available for master receive data MRST [D:A] and start event START[D:A].

The selection is done via the corresponding bit in the INSEL register.

10.7.2 Mode selection (CON)

Module enable (EN)

- The module enable is done via setting bit CON.EN
- Clearing bit CON.EN resets the baud-rate generator, the sets the shift engine to idle and disables the shift clock

Master mode (MS)

CON.MS = 0: slave mode is selected and following configuration is used		
	SLS	As chip select input (if SLCSEN = 1)
	SCLK	As slave clock input for shifting and latching data
	MRST	As data output from shift register
	MTSR	As data input to shift register
ON.MS = 1: master mode is selected and following configuration is used		
	CS	As chip select output (if MSCSEN = 1)
	SCLK	As master clock output for shifting data
	MRST	As data input from shift register
	MTSR	As data output to shift register

Debug support - loop back mode (LB)

For debugging the data lines MRST and MTSR can be internally connected and do not leave the SSC module (CON.LB = 1).

Error handling support - automatic reset (AREN)

When enabled (CON.AREN=1) and a baud-rate error (BEIR) is detected, the shift registers are reset and the shift engine is set to idle.

10.7.3 Baud-rate configuration (BR)

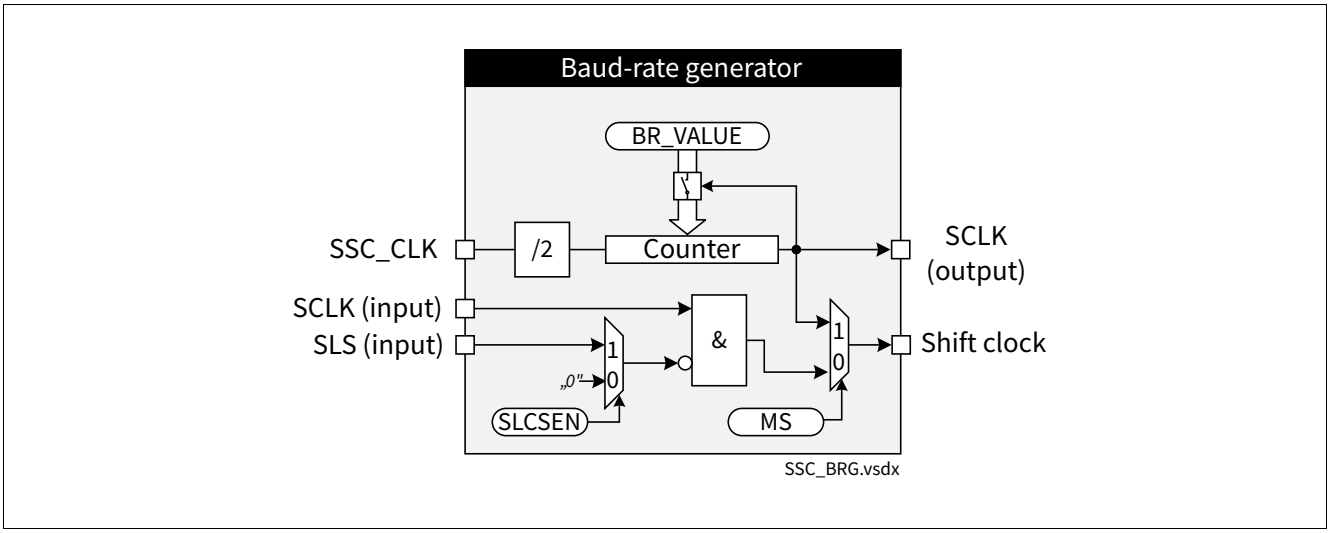


Figure 126 Baud-rate generator

- The baud-rate is generated with a 16-bit auto-reload counter with a programmable reload value
- The baud-rate is used as shift and latch clock for the receive and transmit engine
- In master mode the baud-rate clock is propagated to the SCLK output
- In slave mode the baud-rate clock is used for error detection
- In slave mode the slave chip select (SLS) gates the shift clock

- The baud-rate can be adjusted via BR.BR_VALUE according to formula:

(10.1)

$$\text{Baud-rate} = \frac{f_{ssc\_clk}}{2 \times (BR\_VALUE + 1)}$$

Notes

1. *The maximum baud-rate $f_{SCLKmaster} = f_{SSC\_CLK} / 2$*
2. *The maximum baud-rate $f_{SCLKslave} = f_{SSC\_CLK} / 4$*

10.8 Receive and transmit configuration

10.8.1 Data format configuration (CON)

Data width (BM)

The data width is programmable from 2 bits to 64 bits via CON.BM.

Heading bit (HB)

The heading control defines if the LSB or MSB is shifted first (CON.HB):

- 0: LSB first
- 1: MSB first

Clock polarity (PO)

The clock polarity defines the idle level of the SCLK (CON.PO):

- 0: idle level is low
- 1: idle level is high

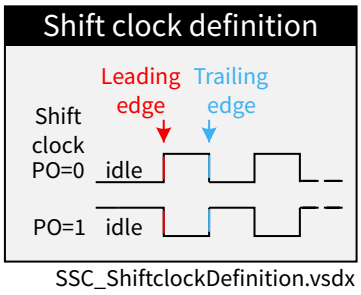


Figure 127 Shift clock definition

Shift clock definition:

- Idle clock is low (PO = 0): leading clock edge is low-to-high
- Idle clock is high (PO = 1): leading clock edge is high-to-low

Clock phase (PH)

The clock phase control defines if leading or trailing SCLK edge is used for shifting (CON.PH).

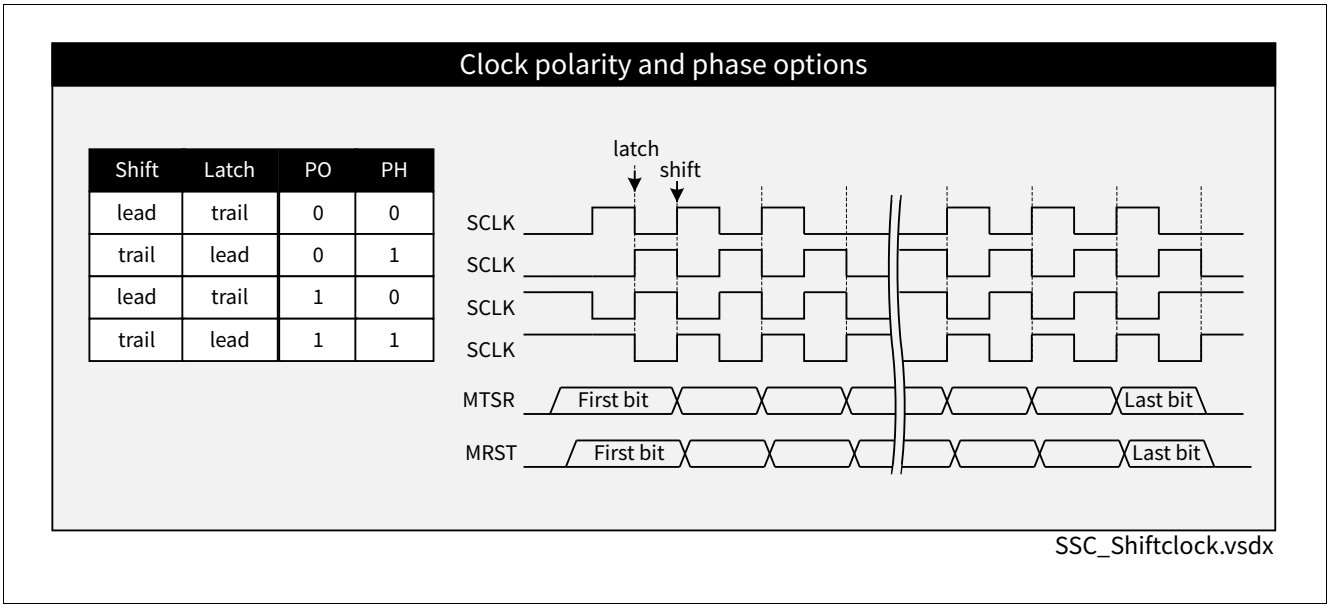


Figure 128 Clock polarity and phase options

10.8.2 Chip select and timing configuration

Master mode

In master mode the master chip select is configurable:

- Master chip select enable (CON.MSCSEN) enables/disables the generation of the CSx output line
- Master chip select selection (CON.MSCSEL) selects one out of four CSx lines for a transmission. The MSCSEL must not be reprogrammed while a transmission is ongoing.
- The chip select timing is configurable in numbers of SSC_CLKs. The timing depends on the values programmed in CSTIM.ST, CSTIM.END, CSTIM.HIGH, BR.BR_VALUE, CON.PH and the SSC_CLK
- The CS setup time (CS falling edge to first edge of SCLK) can be adjusted according to following table:

PH	CS setup time
0	SSC_CLK × (CSTIM.ST + BR.BR_VALUE)
1	SSC_CLK × (CSTIM.ST + 1 + 2 × BR.BR_VALUE)

Examples:

BR at f_{SSC_CLK} = 60 MHz	PH	ST	BR_VALUE	Setup time in SSC_CLKs
30 Mbit	0	7	0	7
15 Mbit	0	7	1	8
7.5 Mbit	0	8	2	10
30 Mbit	1	7	0	8
15 Mbit	1	7	1	10
7.5 Mbit	1	8	2	13

- The CS hold time (SCLK last edge to CS rising edge) can be adjusted according following table:

PH	CS setup time
0	SSC_CLK × (CSTIM.END + BR.BR_VALUE + 2)
1	SSC_CLK × (CSTIM.END + 1)

Examples:

BR at f_{SSC_CLK} = 60 MHz	PH	ST	BR_VALUE	Setup time in SSC_CLKs
30 Mbit	0	2	0	4
15 Mbit	0	2	1	5
7.5 Mbit	0	3	2	7
30 Mbit	1	2	0	2
15 Mbit	1	2	1	3
7.5 Mbit	1	3	2	4

- The CS idle time (CS rising edge to next CS falling edge) can be adjusted via CSTIM.HIGH:
  CS idle time: SSC_CLK × (CSTIM.HIGH)

Slave mode

In slave mode the slave chip select line is programmable:

- Slave chip select (SLCSEN) can be enabled or disabled
- If enabled, the SLS gates the slave shift clock

10.8.3 Single and continues transfer

Single transfer

The transfer of a data frame is initiated with a start event (TXEV), e.g. write to TB0. The chip select surrounds the data frame (see master mode timing diagram). The next data frame is initiated with next start event (TXEV).

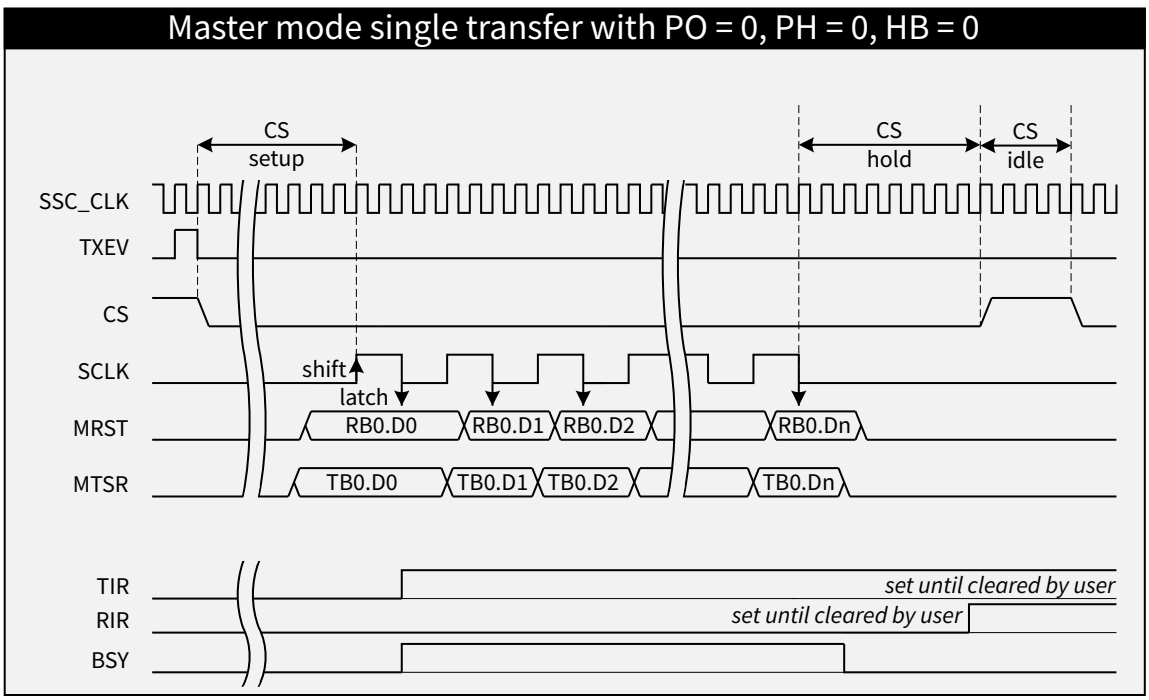


Figure 129 Master mode timing diagram (single transfer)

Continues transfer (streaming data)

A streaming data transfer is started with the first start event (TXEV), e.g. write to TB0. Before the last bit is shifted (bit counter BC > 0), TB0 has to be written with the next data to be transmitted. Otherwise a gap is introduced by the chip select sequence (see continues transfer timing diagram).

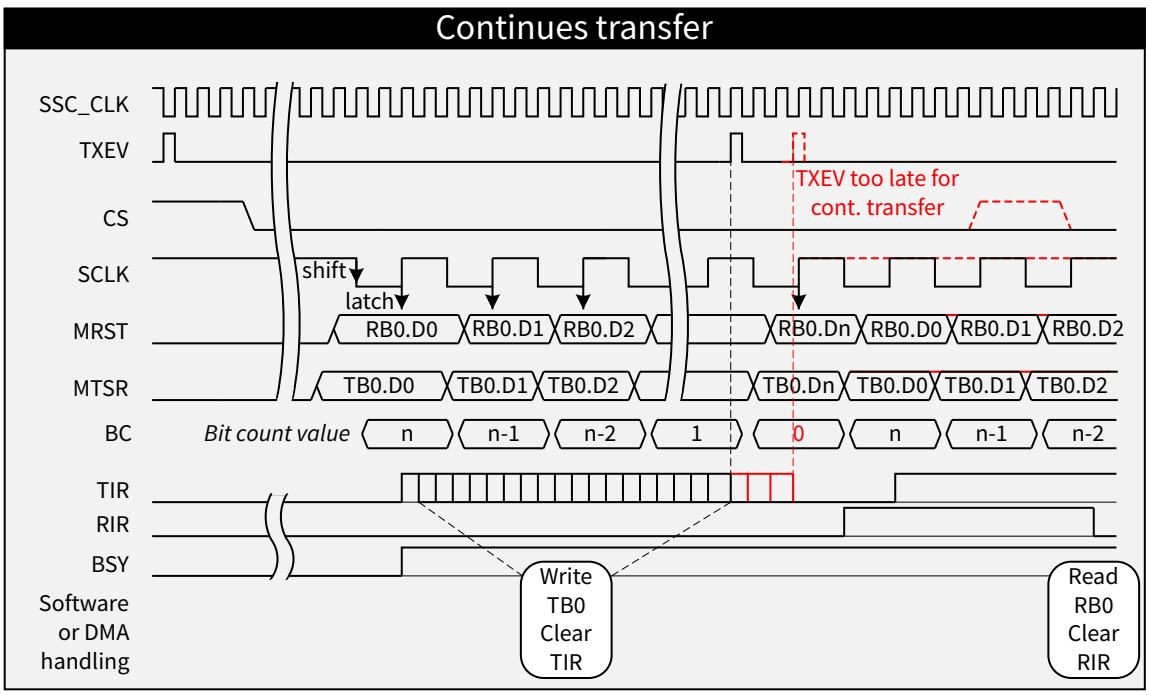


Figure 130 Continues transfer

Note: The setup and hold timing parameters for SCLK to MRST/MTSR are specified in the datasheet.

10.9 Receive and transmit engine

The shift register of the SSC is connected to both the transmit lines and the receive lines. Transmission and reception of serial data are synchronized and take place at the same time, i.e. the same number of transmitted bits is also received.

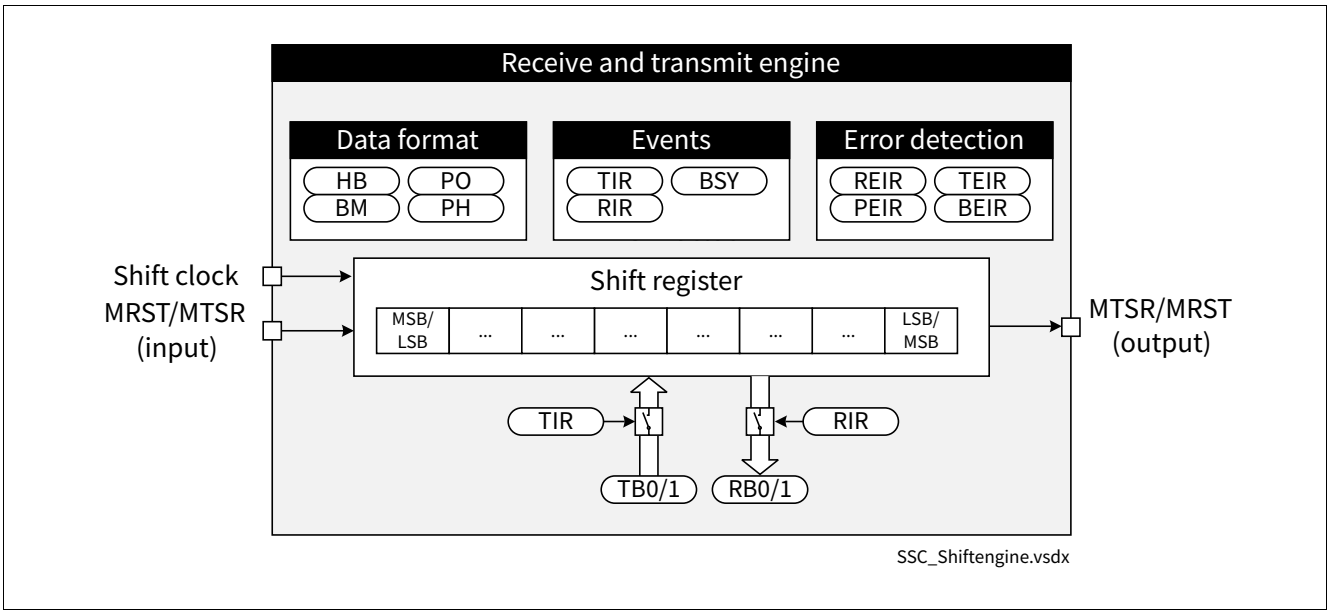


Figure 131 Receive and transmit engine

The data to be transmitted have to be written into the transmit buffer TB0/1. As soon as the shift register is empty (TIR = 1), the TB0/1 is copied to the shift register.

A SSC master (CON.MS = 1) immediately begins transmitting, while a SSC slave (CON.MS = 0) waits for an active shift clock.

When the transfer starts, the busy flag STAT.BSY is set and the transmit interrupt request (TIR) is set to indicate that the register TB0/1 may be reloaded again. When the programmed number of bits (2…64) has been shifted out, the content of the shift register is moved to the receive buffer RB0/1 and the receive interrupt request (RIR) is set. If no further transfer is to take place (TB0/1 is empty), STAT.BSY is cleared.

10.9.1 Slave mode state machine

Idle

- Error detection is inactive
- STAT.BSY and STAT.BC are cleared
- The baud-rate generator is inactive
- SSC0/1 waits on a SCLK edge to change to active state

Note: TB0 must be written with data before the first shift clock edge, otherwise a transmit error (TEIR) is set. Once TB0/1 is written the selected first bit is immediately visible at the MRST output.

Active

- Error detection is active
- SCLK is used as latch and shift clock
- The baud-rate generator is started

- STAT.BSY is set
- STAT.BC is loaded with the BM value and counts SCLKs
- The shift register output is shifted to MTSR output based on SCLK
- The MRST input is latched and shifted into the shift register input based on SCLK
- IS.RIR is set when the bit counter BC counts to zero
- The shift register is copied to RB0/1
- Change to idle state

10.9.2 Master mode state machine

Idle

- Error detection is inactive
- STAT.BSY is cleared and STAT.BC is loaded with the BM value
- The baud-rate generator is inactive
- SSC0/1 waits on a transmit start event

Note: TB0 must be written with data before the start transmit start event, otherwise a transmit error (TEIR) is set. Once TB0/1 is written the selected first bit is immediately visible at the MRST output.

Active

- CSx is generated (if MSCSEN = 1)
- SCLK is generated as master clock
- The baud-rate generator is started
- STAT.BSY is set
- STAT.BC is loaded with the BM value and counts SCLKs
- The shift register output is shifted to MTSR output based on SCLK
- The MRST input is latched and shifted into the shift register input based on SCLK Error detection is active
- IS.TIR and IS.RIR are set accordingly
- CSx is set to inactive
- Change to idle state

10.9.3 Transmission start options

The transmission starts on a transmit start event. Following sources can be selected via CON.MSTXENSEL:

- Write to TB0 buffer
- Set bit CON.MSTXSTART
- External start event TXEV selected via INSEL.TXEVSEL

10.10 Error detection

Master and slave mode

Following errors are detected in master and slave mode:

- Receive error (REIR)
  - A receive error is indicated when the shift register is copied to RB0/1 but RB0/1 has not been read since the previous transfer, i.e. valid data are overwritten
- Phase error (PEIR)
  - A phase error is indicated when the incoming data at MRST (master mode) or MTSR (slave mode) changes between one cycle before and two cycles after the latching edge of the shift clock signal
  - Note: phase errors always occur if BR = 0

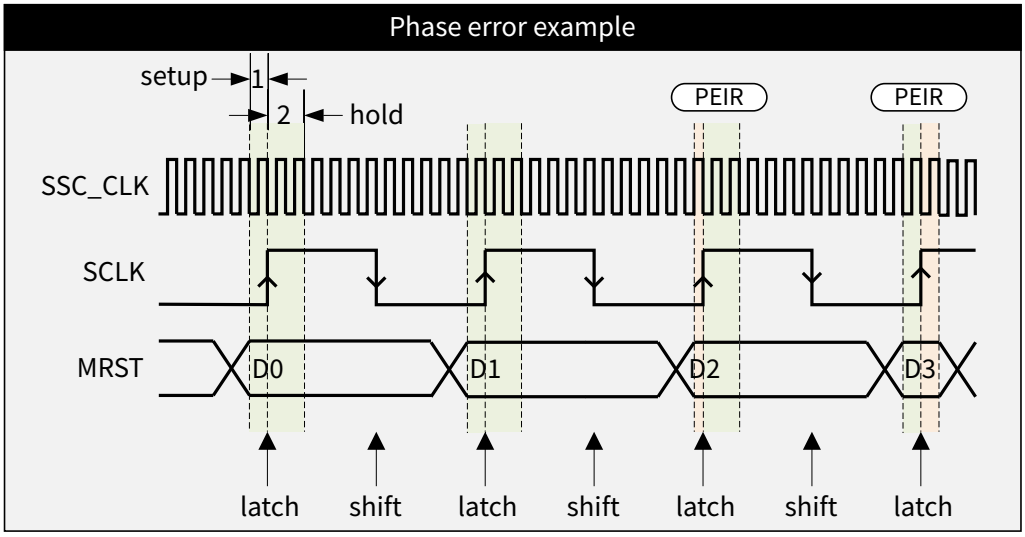


Figure 132 Phase error

Slave mode

- Transmit error (TEIR)
  - A transmit error is indicated when data are received at MTSR but the transmit buffers TB0/TB1 was not updated since the last transfer, i.e. “old” data are shifted out
- Baud-rate error (BEIR)
  - A baud-rate error is indicated if the incoming SCLK is either more than double or less than half the expected baud-rate
  - A baud-rate error is indicated in case of additional or missing SCLK pulses occur
  - Note: the slave state machine can be optionally reset to idle state with IEN.BEIREN = 1 and with CON.AREN = 1
- The error indication bits have to be cleared by user software

10.11 Event generation

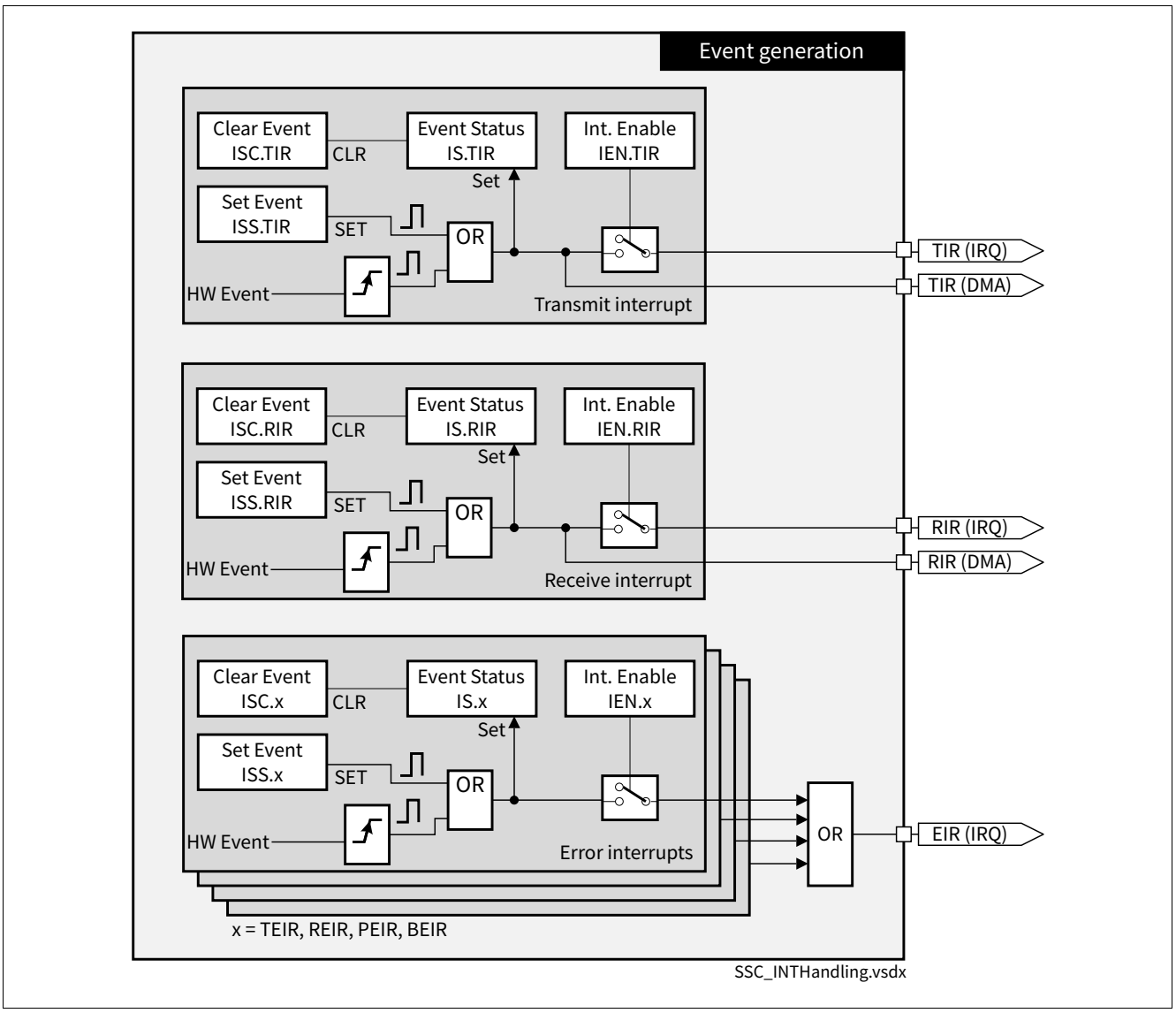


Figure 133 Interrupt and DMA request

Following events can request an interrupt:

- EIR (ored REIR, PEIR, TEIR, BEIR)
- RIR
- TIR
- The events are indicated in the corresponding bits in the IS register
- The events can be alternatively set by software via the corresponding bit in the ISS register
- The events can be cleared by the corresponding bit in the ISC register
- The interrupt request can be individually enabled individually by the corresponding bit in the IEN register

Following events can request a DMA:

- RIR
- TIR

10.12 Programmer's guide

The example shows how to configure SSC0 as master and SSC1 as slave. In the example SSC0 is connected to SSC1 via external wires. Both SSCs exchange data. Two separate interrupt service routine handlers are used. The master transmission is initiated periodically in the main loop (e.g. ~1 ms).

Following configuration is used:

- SSC0 config: as master with CS, 16 bits, 10 Mbaud
- SSC1 config: as slave with SLS, 16 bits, 10 Mbaud
- SSC0 <--> SSC1 connection

Following pins are configured and connected:

SSC0		Connected to	SSC1	
CS	P1.2	-->	P0.2	SLSA
SCLK	P0.5	-->	P0.7	SCLKB
MTSR	P0.4	-->	P0.8	MTSRB
MRSTB	P0.6	<--	P0.9	MRST

Initialization SSC0 as master

```
/* GPIO alternate input function MRSTB --> P0.6 */
SSC0->INSEL.reg = 0x0001;

/* Baudrate = 10 Mbaud */
SSC0->BR.reg = 0x0005;

/* CS timing ST=0, END=0, HIGH=0 */
SSC0->CSTIM.reg = 0x0000;

/* CON setup BM=15, HB=0, PH=0, PO=0, LB=0, AREN=0, MS=1, EN=1, */
/* SLCSEN=0, MSCSSEL=CS1, MSTXSTART=0, MSTXENSEL=write TB0      */
SSC0->CON.reg = 0x0006C00F;

/* GPIO alternate output function */
/*    CS0  --> P1.2 (Out_3)       */
/*    MTSR --> P0.4 (Out_3)       */
/*    SCLK --> P0.5 (Out_3)       */
GPIO->P1_ALTSEL0.reg |= 0x00000300;
GPIO->P1_DIR.reg |= 0x0004;
GPIO->P0_ALTSEL0.reg |= 0x00330000;
GPIO->P0_DIR.reg |= 0x0030;

/* Interrupt assignment and enable */
SSC0->ISC.reg |= 0x0F;              /* Clr all interrupt requests */
SCU->INP6.reg |= 0x00;              /* Select IRQ24               */
CPU->NVIC_IPR6.reg |= 0x00000038;   /* PRI_N24 = 7                */
CPU->NVIC_ICPR.reg |= 0x01000000;   /* IRQCLRPEND24               */
CPU->NVIC_ISER.reg |= 0x01000000;   /* IRQEN24                    */
SSC0->IEN.reg |= 0x02;              /* Enable RIR                 */
```

Initialization SSC1 as slave

```
/* GPIO alternate input function */
/*    SLSA  --> P0.2             */
/*    SCLKB --> P0.7             */
/*    MTSRB --> P0.9             */
SSC1->INSEL.reg = 0x0014;

/* Baudrate = 10 Mbaud */
SSC1->BR.reg = 0x0005;

/* CS timing ST=0, END=0, HIGH=0 */
SSC1->CSTIM.reg = 0x0000;

/* CON setup BM=15, HB=0, PH=0, PO=0, LB=0, AREN=0, MS=0, EN=1, */
/* SLCSEN=1, MSCSSEL=0, MSTXSTART=0, MSTXENSEL=write TB0        */
SSC1->CON.reg = 0x0001800F;

/* GPIO alternate output function */
/*    MRST --> P0.9 (Out_3)       */
GPIO->P0_ALTSEL1.reg |= 0x00000030;
GPIO->P0_DIR.reg |= 0x0200;

/* Interrupt assignment and enable */
SSC0->ISC.reg |= 0x0F;             /* Clr all interrupt requests */
SSC0->INP6.reg |= 0x10;            /* Select IRQ25               */
CPU->NVIC_IPR6.reg |= 0x00004000;  /* PRI_N25 = 9                */
CPU->NVIC_ICPR.reg |= 0x02000000;  /* IRQCLRPEND25               */
CPU->NVIC_ISER.reg |= 0x02000000;  /* IRQEN25                    */
SSC1->IEN.reg |= 0x02;             /* Enable RIR                 */
```

Main loop (e.g. every 1 ms)

```
SSC0->TB0.reg = 0xAA55;    /* Initiate master transmission */
```

SSC0 master receive interrupt service routine

```
SSC0->ISC.reg |= 0x02;     /* Clr RIR             */
recdata0 = SSC0->RB0.reg;  /* Read received value */
```

SSC1 slave receive interrupt service routine

```
SSC1->ISC.reg |= 0x08;     /* Clr RIR             */
recdata1 = SSC1->RB0.reg;  /* Read received value */
SSC1->TB0.reg = 0x8181;
```

10.13 Register description SSC0/1

10.13.1 SSC0/1 Address Maps

Table 121 Register Address Space

Module	Base Address	End Address	Note
SSC0	48020000_H	$48023FFF_H$	
SSC1	48024000_H	$48027FFF_H$	

Table 122 Register Overview - SSCREG (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CON	Control Register	0000_H	475
INSEL	Port Input Select Register	0004_H	476
CSTIM	Master Mode Chip Select Timings Register	0008_H	477
IEN	Interrupt Enable Register	$000C_H$	478
IS	Interrupt Status Register	0010_H	479
ISS	Interrupt Status Set Register	0014_H	480
ISC	Interrupt Status Clear Register	0018_H	481
BR	Baud Rate Timer Register	$001C_H$	482
TB0	Transmitter Buffer Register Bits [31:0]	0020_H	482
TB1	Transmitter Buffer Register Bits [63:32]	0024_H	483
RB0	Receiver Buffer Register Bits [31:0]	0028_H	483
RB1	Receiver Buffer Register Bits [63:32]	$002C_H$	483
STAT	Status Register	0030_H	484

10.13.2 SSCREG Registers

Control Register

CON
Control Register **(0000_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES							MSTXENSEL		MSTX START	MSCSSEL				MSCS EN	SLCSE N
r							rw		w	rw				rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	MS	RES	AREN	RES		LB	PO	PH	HB	BM					
rw	rw	r	rw	r		rw	rw	rw	rw	rw					

Field	Bits	Type	Description
BM	5:0	rw	**Data Width Selection - Number of bits per transfer** 00_H **Reserved**, Do not use this setting. 01_H **BITS_2**, (BM+1) Data Bits … $3F_H$ **BITS_64**, (BM+1) Data Bits
HB	6	rw	**Heading Control** 0_B **LSB**, Transmit/Receive LSB First. 1_B **MSB**, Transmit/Receive MSB First.
PH	7	rw	**Clock Phase Control** 0_B **SHIFTL**, Shift on leading SCLK edge 1_B **SHIFTT**, Shift on trailing SCLK edge
PO	8	rw	**Clock Polarity Control** 0_B **LOW**, SCLK idle level is low 1_B **HIGH**, SCLK idle level is high
LB	9	rw	**Loop Back Control** 0_B **NORMAL**, Normal Operation 1_B **LB**, Internal Loopback Mode (Rx to Tx)
RES	11:10, 13, 31:25	r	**Reserved** Always read as 0
AREN	12	rw	**Automatic Reset Enable** 0_B **NoRESET**, No automatic reset upon baud-rate error 1_B **RESET**, Automatic reset upon baud-rate error
MS	14	rw	**Master Select** 0_B **SLAVE**, Slave mode 1_B **MASTER**, Master mode
EN	15	rw	**Enable Bit** 0_B **Disable**, SSC Module disabled 1_B **Enable**, SSC Module enabled

Register description SSC0/1

Field	Bits	Type	Description
SLCSEN	16	rw	**Slave Chip Select Enable** 0_B **NoCS**, No slave chip select is used 1_B **CS**, SLS input is used as slave chip select
MSCSEN	17	rw	**Master Chip Select Enable** 0_B **NoCS**, No master chip select is used 1_B **CS**, The selected CSx is used as master chip select
MSCSSEL	21:18	rw	**Master Chip Select Output Selection** 1_H **CS0**, Chip Select Output 0 selected 2_H **CS1**, Chip Select Output 1 selected 4_H **CS2**, Chip Select Output 2 selected 8_H **CS3**, Chip Select Output 3 selected
MSTXSTART	22	w	**Master Mode Transmit Start Bit** 0_B **IDLE**, No Data Transmission 1_B **START**, Software Start Event
MSTXENSEL	24:23	rw	**Master Mode Transmit Start Trigger Select** 00_B **TB0WR**, Write to TB0 01_B **MSTXSTART**, Set MSTXSTART 10_B **TXEVR**, Rising edge on selected input 11_B **RES**, Rising edge on selected input

Port Input Select Register

INSEL

Port Input Select Register **(0004_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						**TXEVSEL**		**SLCSSEL**		**MTSRSEL**		**SLCLKSEL**		**MRSTSEL**	
r						rw		rw		rw		rw		rw	

Field	Bits	Type	Description
MRSTSEL	1:0	rw	**Master Mode Data Input Select** 00_B **INP0**, MRSTA input 01_B **INP1**, MRSTB input 10_B **INP2**, MRSTC input 11_B **INP3**, MRSTD input
SLCLKSEL	3:2	rw	**Slave Mode Clock Input Select** 00_B **INP0**, SCLKA input 01_B **INP1**, SCLKB input 10_B **INP2**, SCLKC input 11_B **INP3**, SCLKD input

Register description SSC0/1

Field	Bits	Type	Description
MTSRSEL	5:4	rw	**Slave Mode Data Input Select** 00_B **INP0**, MTSRA input 01_B **INP1**, MTSRB input 10_B **INP2**, MTSRC input 11_B **INP3**, MTSRD input
SLCSSEL	7:6	rw	**Slave Mode Chip Select Input Select** 00_B **INP0**, SLSA input 01_B **INP1**, SLSB input 10_B **INP2**, SLSC input 11_B **INP3**, SLSD input
TXEVSEL	9:8	rw	**Master Mode TX Start Event Input Select** 00_B **INP0**, STARTA input 01_B **INP1**, STARTB input 10_B **INP2**, STARTC input 11_B **INP3**, STARTD input
RES	31:10	r	**Reserved** Always read as 0

Master Mode Chip Select Timings Register

CSTIM
Master Mode Chip Select Timings Register (0008_H) RESET_TYPE_5 Value: $0000\ 0000_H$

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

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

Field	Bits	Type	Description
ST	5:0	rw	**CS Setup Time** 00_H **RES_0**, 7 … 06_H **RES_6**, 7 07_H **CYC_7**, 7 … $3F_H$ **CYC_63**, 63
RES	7:6, 15:14, 31:22	r	**Reserved** Always read as 0

Register description SSC0/1

Field	Bits	Type	Description
END	13:8	rw	**CS Hold Time** 00_H **RES**, 2 01_H **CYC_2**, 2 … $3F_H$ **CYC_64**, 64
HIGH	21:16	rw	**CS High Time** 00_H **RES_0**, 3 01_H **RES_1**, 3 02_H **CYC_3**, 3 … $3F_H$ **CYC_64**, 64

Interrupt Enable Register

IEN
Interrupt Enable Register **($000C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										BEIREN	PEIREN	REIREN	TEIREN	RIREN	TIREN
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
TIREN	0	rw	**Transmit Buffer Empty Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
RIREN	1	rw	**Receive Buffer Full Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
TEIREN	2	rw	**Transmit Error Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
REIREN	3	rw	**Receive Error Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
PEIREN	4	rw	**Phase Error Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled
BEIREN	5	rw	**Baud Rate Error Interrupt Enable** 0_B **Disable**, Interrupt disabled 1_B **Enable**, Interrupt enabled

Register description SSC0/1

Field	Bits	Type	Description
RES	31:6	r	**Reserved** Always read as 0

Interrupt Status Register

IS
Interrupt Status Register **(0010_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										**BEIR**	**PEIR**	**REIR**	**TEIR**	**RIR**	**TIR**
r										rhxre	rhxre	rhxre	rhxre	rhxre	rhxre

Field	Bits	Type	Description
TIR	0	rhxre	**Transmit Buffer Empty Interrupt Flag** Indicates that Transmit Buffer can be reloaded. This bit is set by hardware and can only be cleared by software. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
RIR	1	rhxre	**Receive Buffer Full Interrupt Flag** Indicates that Receive Buffer is full and can be read. This bit is set by hardware and can only be cleared by software. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
TEIR	2	rhxre	**Transmit Error Interrupt Flag** This bit is set by hardware and can only be cleared by software. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
REIR	3	rhxre	**Receive Error Interrupt Flag** This bit is set by hardware and can only be cleared by software. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
PEIR	4	rhxre	**Phase Error Interrupt Flag** This bit is set by hardware and can only be cleared by software. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
BEIR	5	rhxre	**Baud Rate Error Interrupt Flag** This bit is set by hardware and can only be cleared by software. 0_B **NoINT**, Interrupt not occurred 1_B **INT**, Interrupt occurred
RES	31:6	r	**Reserved** Always read as 0

Register description SSC0/1

Interrupt Status Set Register

ISS
Interrupt Status Set Register **(0014_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										BEIRSET	PEIRSET	REIRSET	TEIRSET	RIRSET	TIRSET
r										w	w	w	w	w	w

Field	Bits	Type	Description
TIRSET	0	w	**Transmit Buffer Empty Interrupt Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not SET 1_B **SET**, Interrupt SET
RIRSET	1	w	**Receive Buffer Full Interrupt Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not set 1_B **SET**, Interrupt set
TEIRSET	2	w	**Transmit Error Interrupt Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not set 1_B **SET**, Interrupt set
REIRSET	3	w	**Receive Error Interrupt Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not set 1_B **SET**, Interrupt set
PEIRSET	4	w	**Phase Error Interrupt Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not set 1_B **SET**, Interrupt set
BEIRSET	5	w	**Baud Rate Error Interrupt Set** This bit is set by software and automatically cleared by hardware 0_B **NoSET**, Interrupt not set 1_B **SET**, Interrupt set
RES	31:6	r	**Reserved** Always read as 0

Register description SSC0/1

Interrupt Status Clear Register

ISC
Interrupt Status Clear Register **(0018_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

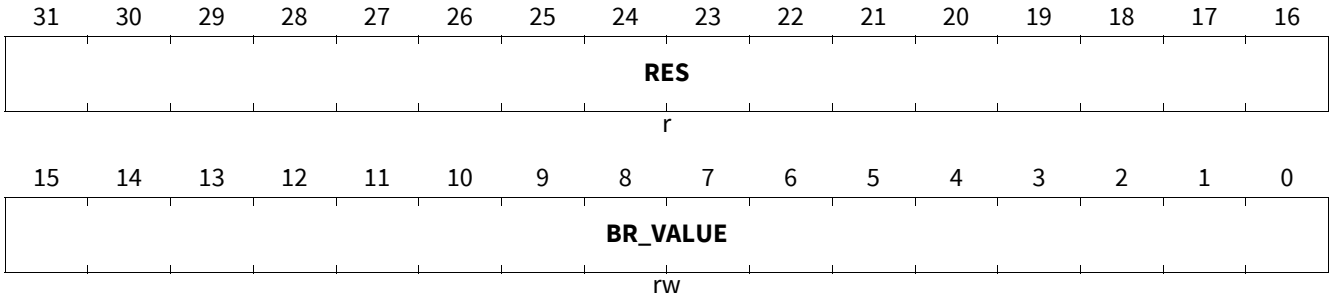
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										BEIRCLR	PEIRCLR	REIRCLR	TEIRCLR	RIRCLR	TIRCLR
r										w	w	w	w	w	w

Field	Bits	Type	Description
TIRCLR	0	w	**Transmit Buffer Empty Interrupt Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
RIRCLR	1	w	**Receive Buffer Full Interrupt Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
TEIRCLR	2	w	**Transmit Error Interrupt Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
REIRCLR	3	w	**Receive Error Interrupt Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
PEIRCLR	4	w	**Phase Error Interrupt Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
BEIRCLR	5	w	**Baud Rate Error Interrupt Clear** This bit is set by software and automatically cleared by hardware 0_B **NotCleared**, Interrupt not cleared 1_B **Cleared**, Interrupt cleared
RES	31:6	r	**Reserved** Always read as 0

Register description SSC0/1

Baud Rate Timer Register

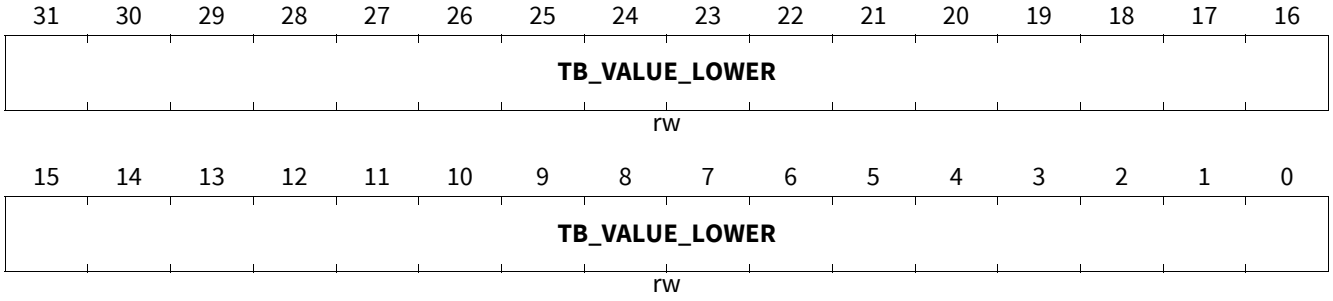
BR
Baud Rate Timer Register **($001C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
BR_VALUE	15:0	rw	**Baud Rate Timer Value** Baud Rate = SSC_CLK / 2 x (BR_VALUE + 1)
RES	31:16	r	**Reserved** Always read as 0

Transmitter Buffer Register Bits [31:0]

TB0
Transmitter Buffer Register Bits [31:0] **(0020_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
TB_VALUE_LOWER	31:0	rw	**Transmit Data [31:0]**

Register description SSC0/1

Transmitter Buffer Register Bits [63:32]

TB1
Transmitter Buffer Register Bits [63:32] (0024_H) RESET_TYPE_5 Value: $0000\ 0000_H$

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

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

Field	Bits	Type	Description
TB_VALUE_UPPER	31:0	rw	Transmit Data [63:32]

Receiver Buffer Register Bits [31:0]

RB0
Receiver Buffer Register Bits [31:0] (0028_H) RESET_TYPE_5 Value: $0000\ 0000_H$

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

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

Field	Bits	Type	Description
RB_VALUE_LOWER	31:0	r	Receive Data [31:0]

Receiver Buffer Register Bits [63:32]

RB1
Receiver Buffer Register Bits [63:32] ($002C_H$) RESET_TYPE_5 Value: $0000\ 0000_H$

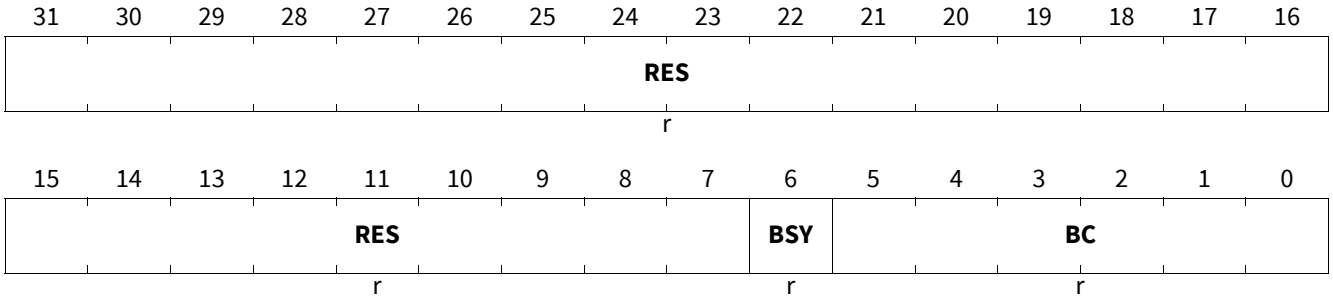
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RB_VALUE_UPPER															
r															

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

Field	Bits	Type	Description
RB_VALUE_UPPER	31:0	r	**Receive Data [63:32]**

Status Register

STAT
Status Register **(0030_H)** **RESET_TYPE_5 Value: 0000 0000$_H$**



Field	Bits	Type	Description
BC	5:0	r	**Bit Count Field** Contains the current value of the bit counter.
BSY	6	r	**Busy Flag** 0_B **NotBUSY**, No Data Transfer Ongoing 1_B **BUSY**, Data Transfer Ongoing
RES	31:7	r	**Reserved** Always read as 0

11 CAN Controller (MultiCAN+)

11.1 Features overview

The MultiCAN+ provides a communication interface which is compliant to the CAN specification CAN FD ISO11898-1 (non-ISO CAN FD format and ISO CAN FD), providing communications at up to 1 Mbit/s in classical CAN (ISO 11898-1:2003(E) mode and/or CAN FD until 2 MBaud data speed, dependent on frequency and nodes).

The MultiCAN+ for the TLE989x/TLE988x consists of 1 module (i.e. MultiCAN with 1 CAN nodes), representing 1 serial communication interfaces. All nodes are CAN FD capable. Each CAN node communicates over two pins (TXD and RXD). The device ports which are used for TXD and RXD may be individually configured within the GPIO block. Several port configuration options are available to provide application-specific flexibility.

The MultiCAN+ contains 1 independently operating CAN node with Full-CAN functionality that is able to exchange Data and Remote Frames via a gateway function. Each CAN node can receive and transmit standard frames with 11-bit identifiers as well as extended frames with 29-bit identifiers.

All CAN nodes share a common set of 32 message objects. Each message object can be individually allocated to one of the CAN nodes. Besides serving as a storage container for incoming and outgoing frames, message objects can be combined to build gateways between the CAN nodes or to setup a FIFO buffer.

The message objects are organized in double-chained linked lists, where each CAN node has its own list of message objects. A CAN node stores frames only into message objects that are allocated to the message object list of the CAN node, and it transmits only messages belonging to this message object list. A powerful, command-driven list controller performs all message object list operations.

The bit timings for the CAN nodes are derived from the module timer clock (f_{CAN}) and are programmable up to a data rate of 1 Mbit/s in Classical CAN (ISO 11898-1:2003(E) mode or up to 2 MBaud in CAN FD mode. External bus transceivers are connected to a CAN node via a pair of receive and transmit pins.

The MultiCAN+ provides the following features:

- Compliant with ISO 11898 and SAE J 1939
- Supports CAN with Flexible Data-Rate Specification CAN FD (non-ISO CAN FD format and ISO CAN FD) with max. 64 data bytes
- Data transfer rates up to 1 Mbit/s when operating in Classical CAN mode per ISO 11898-1:2003(E)
- Supports up to 2 MBaud, when operating in CAN FD mode.
- Support for asynchronous clock sources for baud-rate generation
- Flexible and powerful message transfer control and error handling capabilities
- Advanced CAN bus bit timing analysis and baud-rate detection for each CAN node via a frame counter
- Full-CAN functionality: A set of 32 message objects can be individually
  - Configured as transmit or receive object
  - Setup to handle frames with 11-bit or 29-bit identifier
  - Identified by a timestamp via a frame counter
  - Configured to remote monitoring mode
- Advanced Acceptance Filtering
  - Each message object provides an individual acceptance mask to filter incoming frames
  - A message object can be configured to accept standard or extended frames or to accept both standard and extended frames
  - Message objects can be grouped into different priority classes for transmission and reception

  - The selection of the message to be transmitted first can be based on frame identifier, IDE bit and RTR bit according to CAN arbitration rules, or on its order in the list
- Advanced CAN node features
  - Analyzer mode supports monitoring of bus traffic without actively participating on the bus
  - Internal Loop-Back mode is available for test purposes
  - Data transmission from a node can be stopped without affecting reception
  - Programmable minimum delay between two consecutive messages
- Advanced message object functionality
  - Message objects can be combined to build FIFO message buffers of arbitrary size, limited only by the total number of message objects
- Advanced data management
  - The message objects are organized in double-chained lists
  - up to 8 lists can be used for message objects
  - List reorganizations can be performed at any time, even during full operation of the CAN nodes
  - A powerful, command-driven list controller manages the organization of the list structure and ensures consistency of the list
  - Message FIFOs are based on the list structure and can easily be scaled in size during CAN operation
- Advanced interrupt handling
  - Message interrupts, node interrupts can be generated
  - Interrupt requests can be routed individually to one of the 3 interrupt output lines
  - Message post-processing notifications can be combined flexibly into a dedicated register field of 256 notification bits

11.2 Block diagram

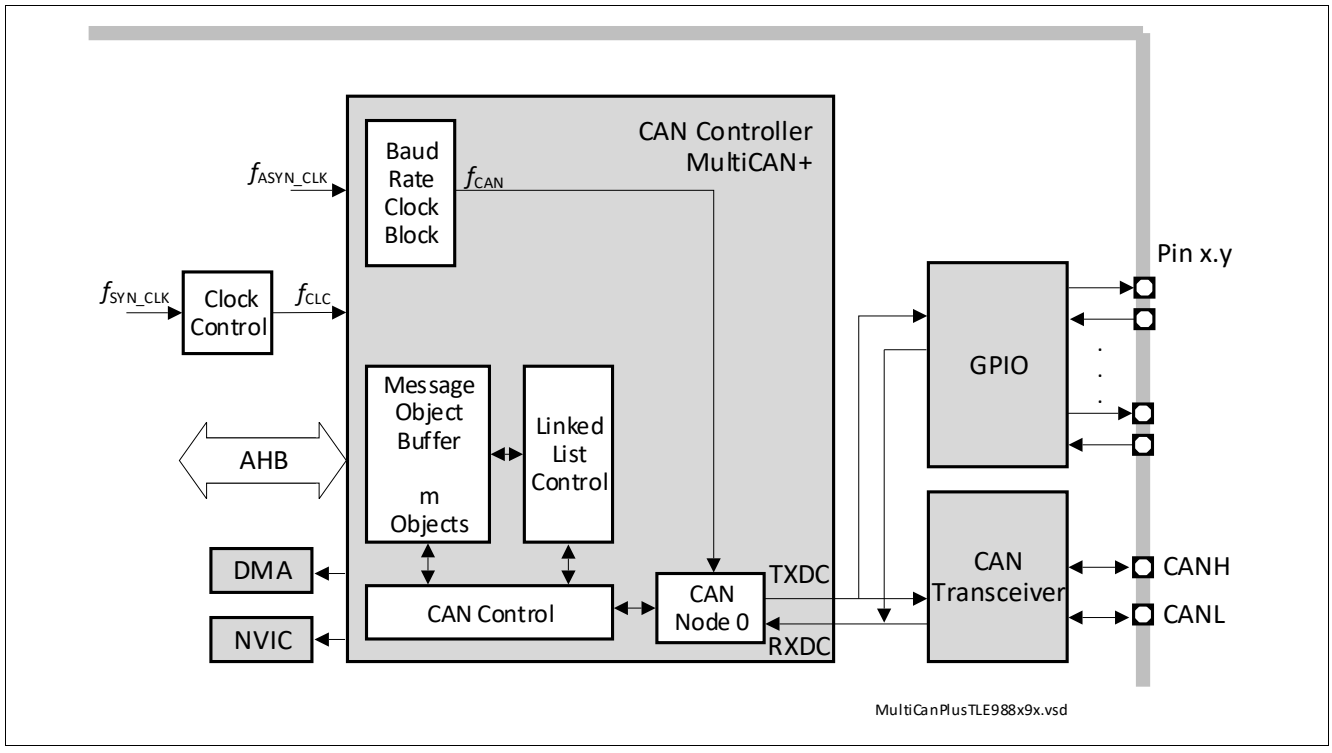


Figure 134 Block diagram MultiCAN+

11.3 Toplevel signals

Table 123 MultiCAN+ toplevel block

Signal	Direction	Description	From/To
TXDC	output	CAN transmit signal	see **Product definitions**, **MultiCAN interconnections**
RXDC	input	CAN receive signal	see **Product definitions**, **MultiCAN interconnections**
fSYN_CLK	input	Bus clock for SRF access	see **Product definitions**, **MultiCAN interconnections**
fASYN_CLK	input	Clock for baud-rate generation	see **Product definitions**, **MultiCAN interconnections**
AHB	bidirectional	Bus interface	CPU/DMA

11.4 Interrupts

The interrupt control logic in the MultiCAN+ uses an interrupt compressing scheme that allows high flexibility in interrupt processing. There are hardware and software interrupt sources available:

- CAN node interrupts:
  - Four different interrupt sources for each of the CAN nodes = 4 × 1 interrupt sources
- Message object interrupts:
  - Two interrupt source for each message object = 2 × 32 interrupt sources
- One register (MITR) to initiate 3 interrupts via software

Each of the hardware-initiated interrupt sources is controlled by a 4-bit interrupt pointer that directs the interrupt source to one of the 3 interrupt outputs INT_Om (m = 0-2). This makes it possible to connect more than one interrupt source (between one and all) to one interrupt output line. The interrupt wiring matrix shown in **Figure 135** is built up according to the following rules:

- Each output of the 4-bit interrupt pointer demultiplexer is connected to exactly one OR-gate input of the INT_Om line. The number “m” of the corresponding selected INT_Om interrupt output line is defined by the interrupt pointer value.
- Each INT_Om output line has an input OR gate which is connected to all interrupt pointer demultiplexer outputs which are selected by an identical 4-bit pointer value.

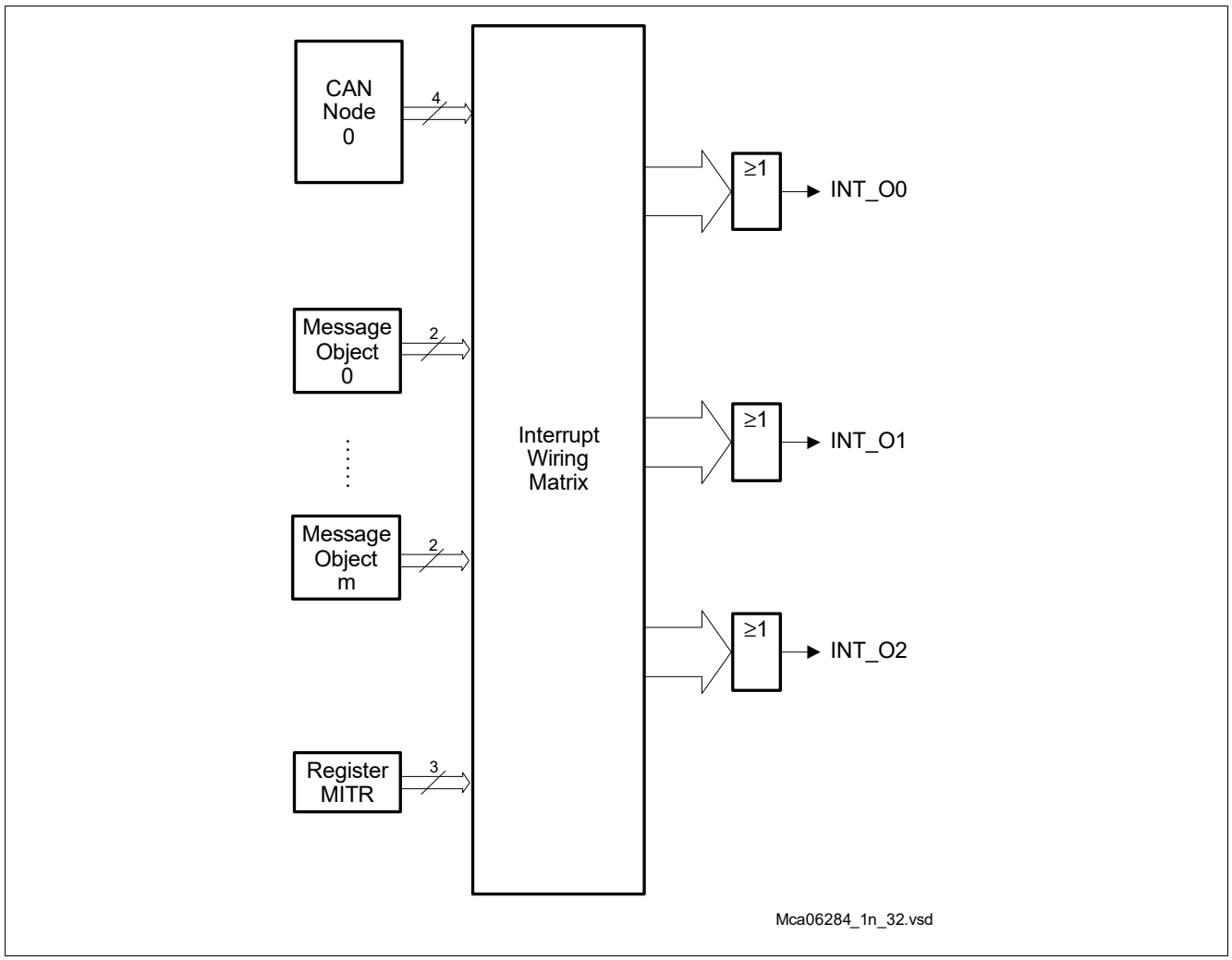


Figure 135 Interrupt compressor, m = 31

Connections to interrupt router inputs

The interrupt output lines INTx are connected to the interrupt router module, see **Table 124**.

Table 124 Interrupt router inputs

Interrupt router input	Connected to CAN interrupt output
SRC_CANINT0	INT_O0
SRC_CANINT1	INT_O1
SRC_CANINT2	INT_O2

11.5 Operation mode behavior

Table 125 Operation mode behavior MultiCAN+

Reset	• RESET_SYSx resets the MultiCAN+ and all its SFRs
Power-up/ Power-down	• The MultiCAN+ is held in reset until power supply is stable at power up • The MultiCAN+ is reset when power supply falls below threshold
Active mode	• The MultiCAN+ is by default off
Stop mode	• The module and peripheral clocks for MultiCAN+ are stopped • MultiCAN+ is powered, MultiCAN+ memories and SFRs keep its content • The user has to ensure that ongoing message reception/ transmission is completed and must follow a command sequence for a defined transition to stop mode • After “stop-to-active transition without reset” the module is clocked, the configuration is unchanged compared to the entry state • A “stop-to-active transition with reset” resets the module
Sleep mode	• The MultiCAN+ is unpowered in sleep mode and loses its configuration • The sleep-to-active mode transition is handled like a power-up with reset for the MultiCAN+
Fail-safe mode	• None

11.6 CAN basics

CAN is an asynchronous serial bus system with one logical bus line. It has an open, linear bus structure with equal bus participants called nodes. A CAN bus consists of two or more nodes.

The bus logic corresponds to a “wired-AND” mechanism. Recessive bits (equivalent to the logic 1 level) are overwritten by dominant bits (logic 0 level). As long as no bus node is sending a dominant bit, the bus is in the recessive state. In this state, a dominant bit from any bus node generates a dominant bus state. The maximum CAN bus speed is, by definition, 1 Mbit/s. This speed limits the CAN bus to a length of up to 40 m. For bus lengths longer than 40 m, the bus speed must be reduced.

The binary data of a CAN frame is coded in NRZ code (Non-Return-to-Zero). To ensure re-synchronization of all bus nodes, bit stuffing is used. This means that during the transmission of a message, a maximum of five consecutive bits can have the same polarity. Whenever five consecutive bits of the same polarity have been transmitted, the transmitter will insert one additional bit (stuff bit) of the opposite polarity into the bit stream before transmitting further bits. The receiver also checks the number of bits with the same polarity and removes the stuff bits from the bit stream (= destuffing).

In CAN FD format frames, the CAN bit stuffing method is changed for the CRC Sequence. The stuff bits will be inserted at fixed positions.

11.6.1 Addressing and bus arbitration

In the CAN protocol, address information is defined in the identifier field of a message. The identifier indicates the contents of the message and its priority. The lower the binary value of the identifier, the higher is the priority of the message.

For bus arbitration, CSMA/CD with NDA (Carrier Sense Multiple Access/Collision Detection with Non-Destructive Arbitration) is used. If bus node A attempts to transmit a message across the network, it first checks that the bus is in the idle state (“Carrier Sense”) i.e. no node is currently transmitting. If this is the case (and no other node wishes to start a transmission at the same moment), node A becomes the bus master and sends its message. All other nodes switch to receive mode during the first transmitted bit (Start-Of-Frame bit). After correct reception of the message (acknowledged by each node), each bus node checks the message identifier and stores the message, if required. Otherwise, the message is discarded.

If two or more bus nodes start their transmission at the same time (“Multiple Access”), bus collision of the messages is avoided by bit-wise arbitration (“Collision Detection / Non-Destructive Arbitration” together with the “Wired-AND” mechanism, dominant bits override recessive bits). Each node that sends also reads back the bus level. When a recessive bit is sent but a dominant one is read back, bus arbitration is lost and the transmitting node switches to receive mode. This condition occurs for example when the message identifier of a competing node has a lower binary value and therefore sends a message with a higher priority. In this way, the bus node with the highest priority message wins arbitration without losing time by having to repeat the message. Other nodes that lost arbitration will automatically try to repeat their transmission once the bus returns to idle state. Therefore, the same identifier can be sent in a data frame only by one node in the system. There must not be more than one node programmed to send data frames with the same identifier.

Standard message identifier has a length of 11 bits. CAN specification 2.0B extended the message identifier lengths to 29 bits, i.e. the extended identifier. Both frame formats are part of the ISO 11898-1. The identifier are available for Classical CAN as well as for CAN FD.

11.6.2 CAN frame formats

Four different data frame formats are supported which differ in the length of the arbitration field and the control field:

- Classical CAN base format: 11-bit long identifier, constant bit rate
- Classical CAN extended format: 29-bit long identifier, constant bit rate
- CAN FD base format: 11-bit long identifier, dual bit rate
- CAN FD extended format: 29-bit long identifier, dual bit rate

In addition for Classical CAN remote frames exist, for 11-bit and 29-bit identifiers.

11.6.3 CAN frame types

There are three types of CAN frames:

- Data frames
- Remote frames
- Error frames

A data frame for classical CAN contains a data field of 0 to 8 bytes in length. A remote frame contains no data field and is typically generated as a request for data (e.g. from a sensor). Data frames and remote frames can use an 11-bit “Standard” identifier or a 29-bit “Extended” identifier. An error frame can be generated by any node that detects a CAN bus error. For CAN FD a data frame can be up to 64 bytes long. Remote frames do not exist in CAN FD.

11.6.3.1 Data frames

There are following types of data frames defined (see **Figure 136**):

- 11-bit ID data frame classical CAN format
- 29-bit ID data frame classical CAN format
- 11-bit ID data frame ISO CAN FD format
- 29-bit ID data frame ISO CAN FD format
- 11-bit ID data frame non-ISO CAN FD format
- 29-bit ID data frame non-ISO CAN FD format

11.6.3.1.1 11-bit data frame (classical CAN format)

A data frame begins with the Start-Of-Frame bit (SOF = dominant level) for hard synchronization of all nodes. The SOF is followed by the arbitration field consisting of 12 bits, the 11-bit identifier (reflecting the contents and priority of the message), and the RTR (Remote Transmission Request for Classical CAN) bit. With RTR at dominant level, the frame is marked as data frame. With RTR at recessive level, the frame is defined as a remote frame.

The next field is the control field consisting of 6 bits. The first bit of this field is the IDE (Identifier Extension) bit and is at dominant level for the standard data frame. The following bit is reserved and defined as a dominant bit. The remaining 4 bits of the control field are the Data Length Code (DLC) that specifies the number of bytes in the data field. The data field can be 0 to 8 bytes wide. The Cyclic Redundancy (CRC) field that follows the data bytes is used to detect possible transmission errors. It consists of a 15-bit CRC sequence completed by a recessive CRC delimiter bit.

The final field is the acknowledge field. During the ACK slot, the transmitting node sends out a recessive bit. Any node that has received an error free frame acknowledges the correct reception of the frame by sending back a dominant bit, regardless of whether or not the node is configured to accept that specific message. This

behavior assigns the CAN protocol to the “in-bit-response” group of protocols. The recessive ACK delimiter bit, which must not be overwritten by a dominant bit, completes the acknowledge field.

Seven recessive End-of-Frame (EOF) bits finish the data frame. Between any two consecutive frames, the bus must remain in the recessive state for at least 3-bit times (called inter frame space). If after the inter frame space, no other nodes attempt to transmit the bus remains in idle state with a recessive level.

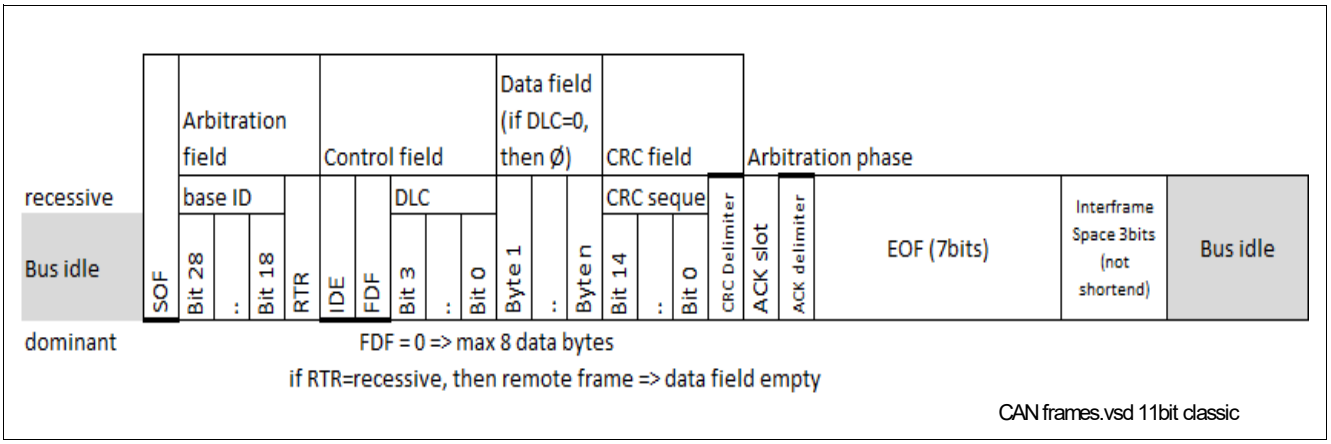


Figure 136 Classical 11-bit ID CAN data frame

11.6.3.1.2 Extended data frame (classical CAN format)

In the extended CAN data frame, the message identifier of the standard frame has been extended to 29-bit. A split of the extended identifier into two parts, an 11-bit least significant section (as in classical CAN frame) and an 18-bit most significant section, ensures that the identifier extension bit (IDE) can remain at the same bit position in both standard and extended frames.

In the extended CAN data frame, the SOF bit is followed by the 32-bit arbitration field. The first 11 bits are the least significant bits of the 29-bit Identifier (“Base-ID”). These 11 bits are followed by the recessive Substitute Remote Request (SRR) bit. The SRR is further followed by the recessive IDE bit, which indicates the frame to be an extended CAN frame. If arbitration remains unresolved after transmission of the first 11 bits of the identifier, and if one of the nodes involved in arbitration is sending a classical CAN frame, then the CAN frame will win arbitration due to the assertion of its dominant IDE bit. Therefore, the SRR bit in an extended CAN frame is recessive to allow the assertion of a dominant RTR bit by a node that is sending a CAN remote frame. The SRR and IDE bits are followed by the remaining 18 bits of the extended identifier and the RTR bit.

Control field and frame termination is identical to the classical data frame.

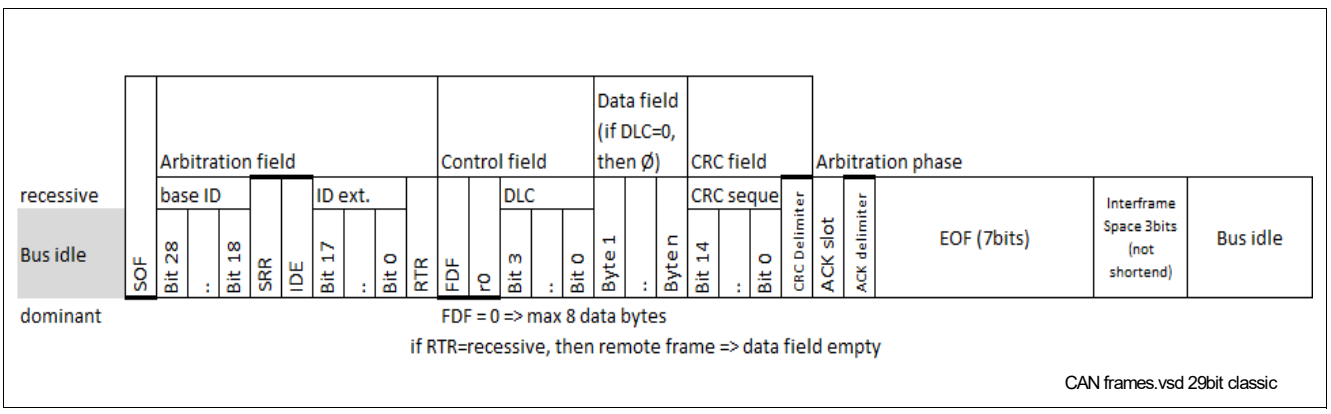


Figure 137 Classical 29-bit ID CAN data frame

11.6.3.1.3 Standard and extended data frame (ISO and non-ISO CAN FD format)

Data frames for CAN FD with 11-bit identifier are shown in **Figure 138**.

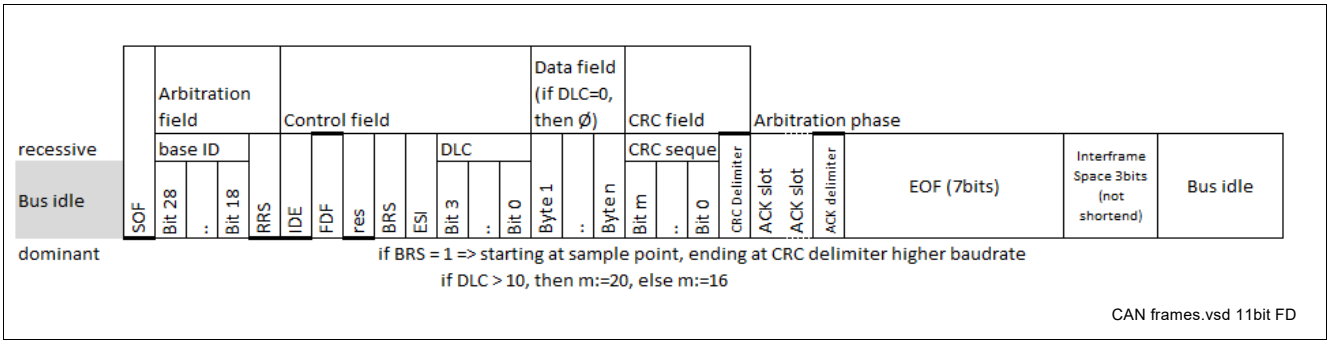


Figure 138 Non-ISO CAN FD 11-bit ID data frames

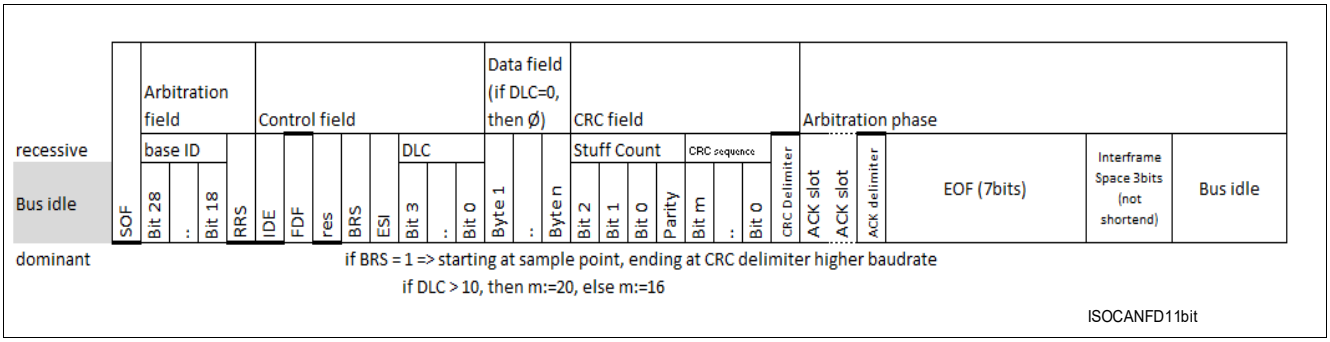


Figure 139 ISO CAN FD 11-bit ID data frames

Extended data frames for CAN FD are shown in **Figure 140**.

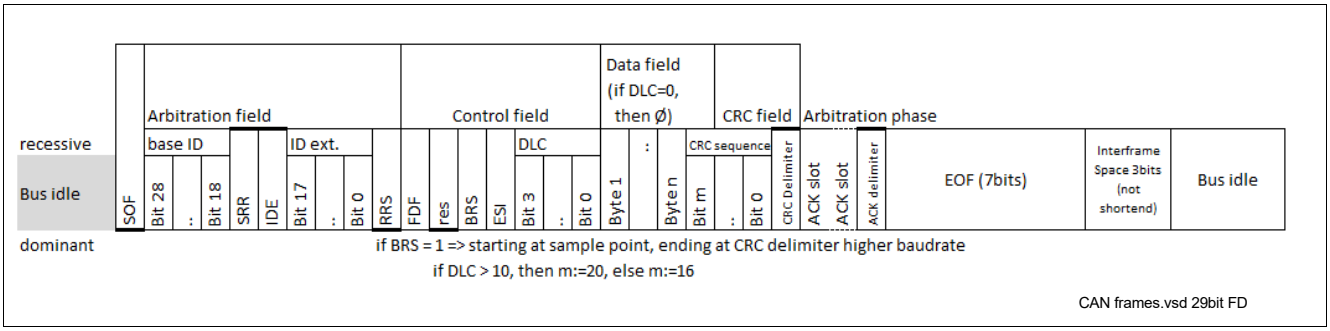


Figure 140 Non-ISO CAN FD 29-bit ID data frames

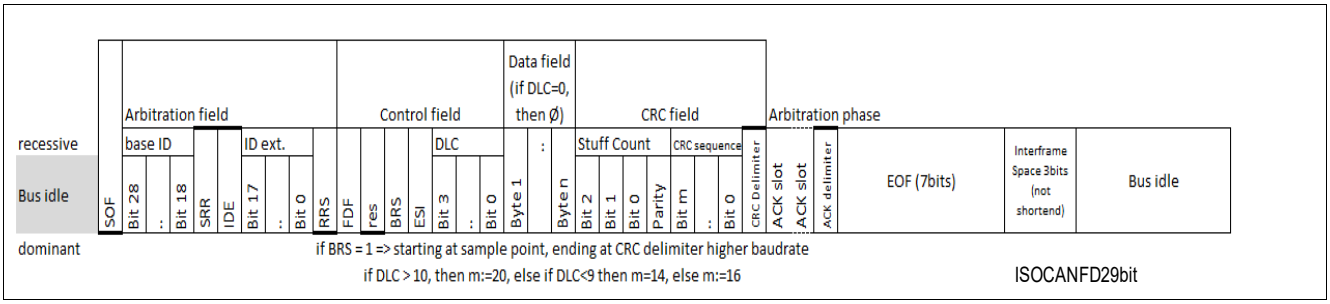


Figure 141 ISO CAN FD 29-bit ID data frames

The difference between standard and extended data frames in CAN and CAN FD format are highlighted below:

- In the arbitration field:
  - CAN format frames contain the RTR bit, where in CAN FD Format it is replaced with the dominant r1(reserved) bit
  - The reserved r0, r1 (bits) are sent dominant. Receivers accept dominant and recessive bits
- In the control field:
  - CAN FD format frames consists of the additional (FDF) CAN FD format bit, (BRS) bit rate switch bit and (ESI) error state indicator bit
  - CAN FD format (FDF) bit comes after the IDE bit for frames with 11-bit identifier and it comes after the r1 (reserved) bit with 29-bit identifier. FDF is the new name for the previous r0 bit
  - Bit Rate Switch (BRS) bit switches the bit rate from standard bit rate of arbitration phase to preconfigured bit rate of the data phase when the bit is transmitted recessive. The bit rate is not switched when BRS bit is transmitted dominant
  - Error State Indicator (ESI) bit is transmitted dominant by error active nodes and recessive by error passive nodes
  - Data Length Code (DLC) bits indicates the number of bytes in the data field
- In the CRC field:
  - A stuff bit counter is included, within the frame module 8, which limits the counter to 3 bits. The stuff bit counter has a parity bit as forth bit
  - CRC sequence bits for CAN FD uses CRC_17 for data field up to sixteen bytes long and CRC_21 for data field longer than sixteen bytes. CRC calculation consists of the SOF, arbitration field, control field and (if present) data field, supplemented with nCRC bits of ‘0’
    Stuff-bits are included in CRC calculation. All CRC sequences (CRC_15, 17, 21) are calculated for all nodes, where the node that wins the arbitration send the CRC sequence selected by values of the FDF bit and DLC. Receivers only consider the selected CRC polynomial to check for CRC error. The CRC starts with a fixed stuff bit
  - CRC delimiter for CAN FD consists of one or two recessive bits that has the function of switching the data phase to arbitration phase when the sample point reaches the first bit of the CRC delimiter. Transmitter sends only one recessive bit as CRC delimiter, but accepts two recessive bits before the edge from recessive to dominant of the Acknowledge slot. Receiver sends acknowledge bit after the first CRC delimiter
- In the ACK field:
  - For ACK slot, CAN FD nodes accept a two bit long dominant phase of overlapping ACK bits as a valid ACK, to compensate for phase shifts between receivers

11.6.3.2 Remote frames

Normally, data transmission is performed on an autonomous basis with the data source node (e.g. a sensor) sending out a data frame. It is also possible, however, for a destination node (or nodes) to request the data from the source. For this purpose, the destination node sends a remote frame with an identifier that matches the identifier of the required data frame. The appropriate data source node will then send a data frame as a response to this remote request.

There are two differences between a remote frame and a data frame:

- The RTR bit is in the recessive state in a remote frame
- There is no data field in a remote frame

If a data frame and a remote frame with the same identifier are transmitted at the same time, the data frame wins arbitration due to the dominant RTR bit following the identifier. In this way, the node that transmitted the remote frame receives the requested data immediately.

Remote frames are only defined in classical CAN format. Remote frames and the corresponding RTR bit do not exist for CAN FD format. Remote frame requests do not change the format of the preconfigured reply data frames (i.e. FDF and BRS bits of preconfigured data frames are not changed on remote frame requests, reply data frames may be in CAN base/extended or CAN FD base/extended).

11.6.3.3 Error frames

An error frame is generated by any node that detects a bus error. An error frame consists of two fields, an error flag field followed by an error delimiter field. The error delimiter field consists of 8 recessive bits and allows the bus nodes to restart bus communications after an error. There are, however, two forms of error flag fields. The form of the error flag field depends on the error status of the node that detects the error.

When an error-active node detects a bus error, the node generates an error frame with an active-error flag. The error-active flag is composed of 6 consecutive dominant bits that actively violate the bit-stuffing rule. All other stations recognize a bit-stuffing error and generate error frames themselves. The resulting error flag field on the CAN bus therefore consists of 6 to 12 consecutive dominant bits (generated by one or more nodes). The error delimiter field completes the error frame. After completion of the error frame, bus activity returns to normal and the interrupted node attempts to re-send the aborted message.

If an error-passive node detects a bus error, the node transmits an error-passive flag followed, again, by the error delimiter field. The error-passive flag consists of six consecutive recessive bits, and therefore the error frame (for an error-passive node) consists of 14 recessive bits (i.e. no dominant bits). Therefore, the transmission of an error frame by an error-passive node will not affect any other node on the network, unless the bus error is detected by the node that is actually transmitting (i.e. the bus master). If the bus master node generates an error-passive flag, this may cause other nodes to generate error frames due to the resulting bit-stuffing violation. After transmission of an error frame an error-passive node must wait for 6 consecutive recessive bits on the bus before attempting to rejoin bus communications.

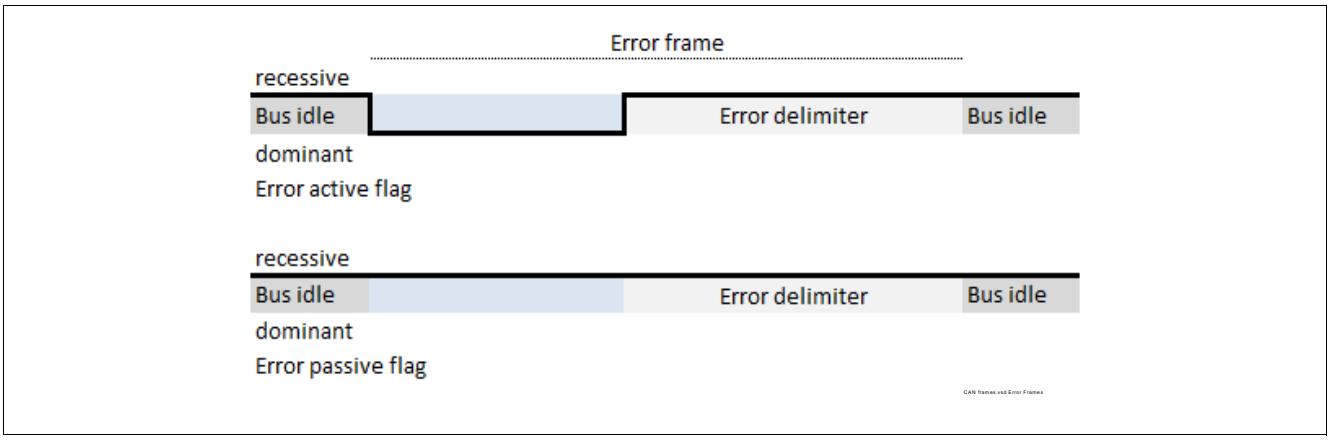


Figure 142 CAN error frames

A CAN FD node operating in the data phase will switch back to the arbitration phase when starting an error flag.

11.6.3.4 Overload frame

The overload frame consists of two bitfields which are overload flag and overload delimiter.

Overload conditions

If a CAN FD node samples a dominant bit at the eight bit (last bit) of an error delimiter or overload delimiter, or if a CAN FD receiver samples a dominant bit at the last bit of end of frame, it will start transmitting an overload frame (not an error frame). The error counters will not be incremented.

11.6.4 Nominal bit time

One bit cell (this means one high or low pulse of the NRZ code) is composed by four segments. Each segment is an integer multiple of time quanta t_Q. The time quanta is the smallest discrete timing resolution used by a CAN node. The nominal bit time definition with its segments is shown in **Figure 143**.

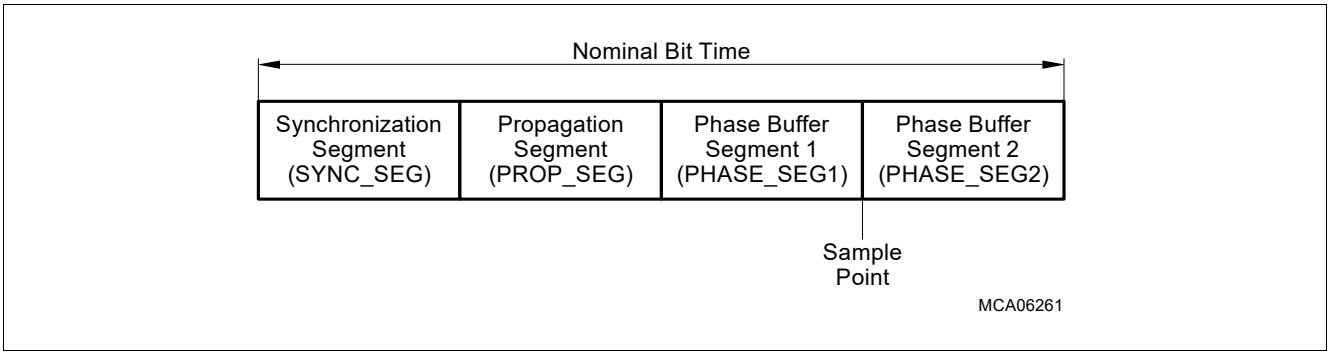


Figure 143 Partition of nominal bit time

The synchronization segment (SYNC_SEG) is used to synchronize the various bus nodes. If there is a bit state change between the previous bit and the current bit, then the bus state change is expected to occur within this segment. The length of this segment is always 1 t_Q.

The propagation segment (PROP_SEG) is used to compensate for signal delays across the network. These delays are caused by signal propagation delay on the bus line and through the electronic interface circuits of the bus nodes.

The phase segments 1 and 2 (PHASE_SEG1, PHASE_SEG2) are used to compensate for edge phase errors. These segments can be lengthened or shortened by re-synchronization. PHASE_SEG2 is reserved for calculation of the subsequent bit level, and is ≥ 2 t_Q. At the sample point, the bus level is read and interpreted as the value of the bit cell. It occurs at the end of PHASE_SEG1.

The total number of t_Q in a bit time is between 8 and 25.

As a result of re-synchronization, PHASE_SEG1 can be lengthened or PHASE_SEG2 can be shortened. The amount of lengthening or shortening the phase buffer segments has an upper limit given by the re-synchronization jump width. The re-synchronization jump width may be between 1 and 4 t_Q, but it may not be longer than PHASE_SEG1.

11.6.4.1 CAN FD bit timing

The first part of a CAN FD frame until the BRS bit is transmitted with the nominal bit rate. The bit rate is switched if the BRS bit is recessive, until the CRC delimiter is reached or until the CAN FD controller sees an error condition that results in the starting of an error frame. CAN FD error frames, as well as ACK field, End-of-Frame, overload frames and all in CAN format are transmitted with the nominal bit rate.

Synchronization rules

Hard synchronization and re-synchronization are the two forms of synchronization. They obey the following rules,

- Hard synchronization is performed whenever there is a recessive to dominant edge during bus Idle, suspend transmission, and second or third bits of Intermission. Hard synchronization is also performed at the recessive to the dominant edge from FDF to r0 in CAN FD format frames
- A transmitter shall not re-synchronize while it transmits in the CAN FD data phase

11.6.5 Error detection and error handling

The CAN protocol has sophisticated error detection mechanisms. The following errors can be detected:

- Cyclic redundancy check (CRC) error
  With the CRC, the transmitter calculates special check bits for the bit sequence from the start of a frame until the end of the data field. This CRC sequence is transmitted in the CRC field. The receiving node also calculates the CRC sequence using the same formula, and performs a comparison to the received sequence. If a mismatch is detected, a CRC error has occurred and an error frame is generated. The message is repeated.
- Acknowledge error
  In the acknowledge field of a message, the transmitter checks whether a dominant bit is read during the acknowledge slot (that is sent out as a recessive bit). If not, no other node has received the frame correctly, an acknowledge error has occurred, and the message must be repeated. No error frame is generated.
- Form error
  If a transmitter detects a dominant bit in one of the four segments End-of-Frame, interframe space, acknowledge delimiter, or CRC delimiter, a form error has occurred, and an error frame is generated. The message is repeated.
- Bit error
  A bit error occurs if In case:
  - A transmitter sends a dominant bit and detects a recessive bit or
  - If the transmitter sends a recessive bit and detects a dominant bit when monitoring the actual bus level and comparing it to the just transmitted bit.
    In this case, no error occurs during the arbitration field (ID, RTR, IDE) and the acknowledge slot.
- Stuff error
  If between Start-of-Frame and CRC delimiter, 6 consecutive bits with the same polarity are detected, the bit-stuffing rule has been violated. A stuff error occurs and an error frame is generated. The message is repeated.

Detected errors are made public to all other nodes via error frames (except acknowledge errors). The transmission of the erroneous message is aborted and the frame is repeated as soon as possible. Furthermore, each CAN node is in one of the three error states (error-active, error-passive or bus-off) according to the value of the internal error counters. The error-active state is the usual state where the bus node can transmit messages and active-error frames (made of dominant bits) without any restrictions. In the error-passive state, messages and passive-error frames (made of recessive bits) may be transmitted. The bus-off state makes it

temporarily impossible for the node to participate in the bus communication. During this state, messages can be neither received nor transmitted.

11.6.5.1 Basic-CAN, Full-CAN

There is one more CAN characteristic that is related to the interface of a MultiCAN+ (CAN controller) and the host CPU: Basic-CAN and Full-CAN functionality.

In Basic-CAN devices, only basic functions of the protocol are implemented in hardware, such as the generation and the check of the bit stream. The decision, whether a received message has to be stored or not (acceptance filtering), and the complete message management must be done by software.

Full-CAN devices (this is the case for the MultiCAN+ controller as implemented in TLE989x/TLE988x) manage the whole bus protocol in hardware, including the acceptance filtering and message management. Full-CAN devices contain message objects that handle autonomously the identifier, the data, the direction (receive or transmit) and the information of CAN operation. During the initialization of the device, the host CPU determines which messages are to be sent and which are to be received. The host CPU is informed by interrupt if the identifier of a received message matches with one of the programmed (receive-) message objects. The CPU load of Full-CAN devices is greatly reduced. When using Full-CAN devices, high baud-rates and high bus loads with many messages can be handled.

Normally, the CAN device also provides only one transmit buffer and one or two receive buffers. Therefore, the host CPU load is quite high when using Basic-CAN modules. The main advantage of Basic-CAN is a reduced chip size leading to low costs of these devices.

11.7 CAN flexible data-rate (CAN FD)

CAN flexible data-rate (CAN FD) builds on existing CAN (ISO 11898-1) specifications allowing higher data rates and larger payloads. This is achieved with a new CAN FD frame format different from existing classical CAN format, both frame formats can coexist within the same network. classical CAN nodes and CAN FD nodes can communicate with each other as long as CAN FD frame format is not being used.

CAN FD functionality is available on all nodes of the MultiCAN+.

11.7.1 Transmitter delay compensation

The CAN protocol requires that transmitted data be compared with the received data from its local CAN transceiver to determine if there are any transmit errors. In the case of CAN FD data phase when a faster bit rate is used, resulting in shorter bit timing, the delay caused by the local transmitting loop delay will be greater than TSEG1 (time segment before sample point) causing a bit error to be detected. The transmitter loop delay limits the bit rate in the data phase of a CAN FD frame.

Thus to overcome this limitation, a transmitter delay compensation feature is introduced where a new sample point (secondary sample point) shall be used by transmitters in the data phase of a CAN FD frame, the sample point which does not account for the transceiver loop delay is ignored.

The secondary sample point consists of the transmitter loop delay and a configurable transmitter delay compensation offset (NTDCRx.TDCO). (i.e. the secondary sample point is reached by counting the total delay consisting of the compensation offset and measured transceiver loop delay.)

The transmitter loop delay is measured in each transmitted frame at the edge from the FDF bit to the following bit r0, between the edge of the transmitted bit and the edge of the received bit. (see **Figure 144**) The count down is started with the begin of the bit time (transmit point).

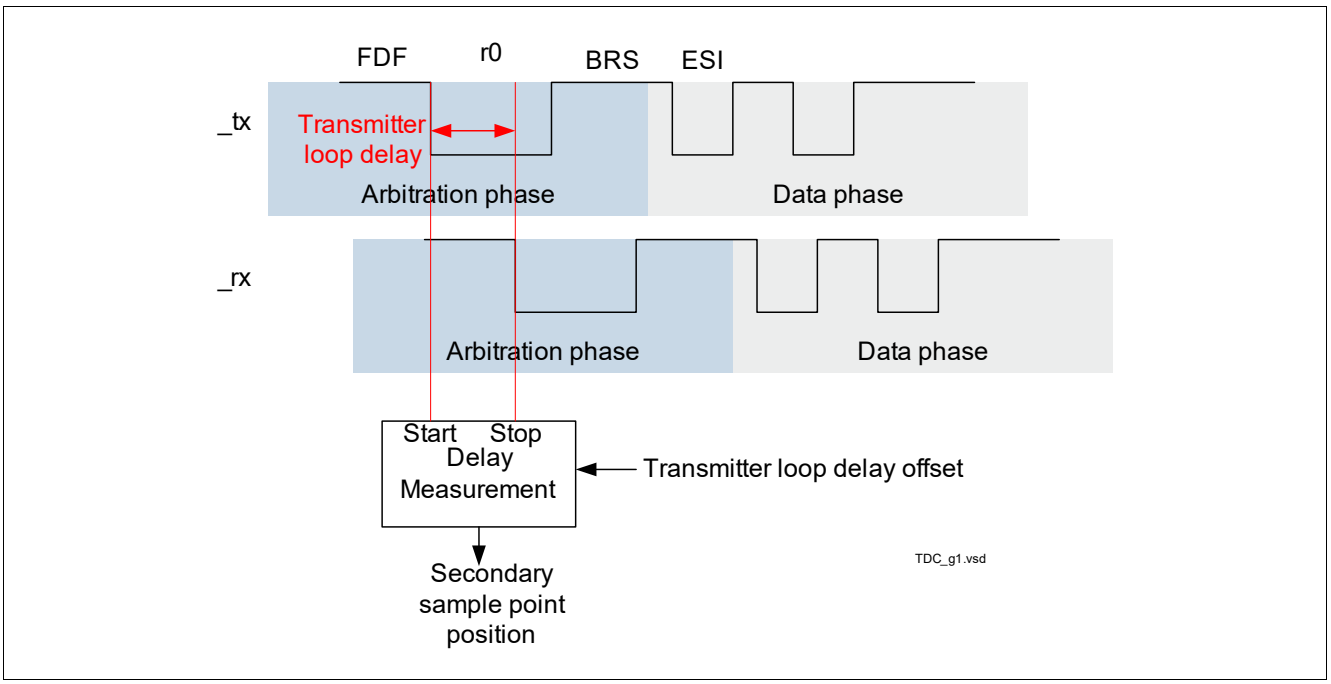


Figure 144 Transmitter delay loop measurement

Transmitter delay compensation offset (NTDCRx.TDCO) is used to adjust the secondary sample point inside the bit time (e.g. half of the bit time in the data phase). Finally the resulting secondary sample point is rounded down to the next integer number of time quanta t_q and placed after the end of the transmitted bit. If a bit error is detected at the Secondary Sample Point, the transmitter will react to this bit error at the next sample point.

(i.e. when a bit error is detected by the Transmitter Delay compensation unit, the error is reported at the next sample point and becomes visible (in error active case) at the transmit point that follows the regular sample point.)

Secondary sample point is used in the data phase of CAN FD and sample point is used in the arbitration phase of CAN FD. The transmitter delay compensation which determines the secondary sample point is able to handle a total delay (measured transceiver loop delay + compensation offset) which goes beyond the current bit time.

The MultiCAN+ allows the secondary sample point to be placed anywhere within the current and next 3 bit times (i.e. covers up to 4 data phase bit rate). The maximum delay which can be compensated by MultiCAN+ delay compensation during the data phase is 4 bit times.

Notes

1. *CAN receive input line contributes to the measured loop delay.*
2. *Measurement granularity of the transmitter delay compensation is the fast time quantum given by the fast baud-rate prescaler.*

11.8 MultiCAN+ kernel

This chapter describes the functionality of the MultiCAN+.

11.8.1 Module structure

Figure 145 shows the general structure of the MultiCAN+.

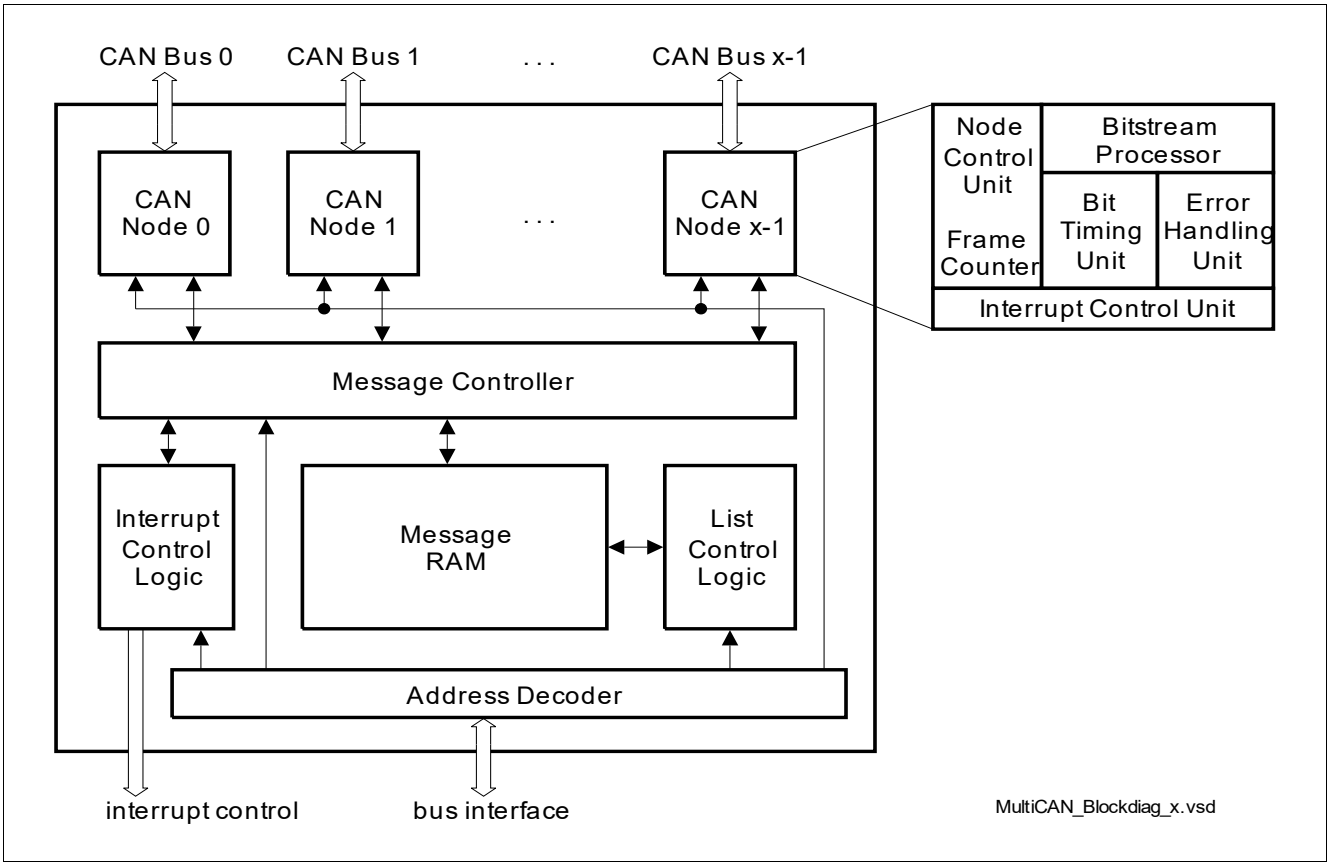


Figure 145 Module structure MultiCAN+

11.8.1.1 CAN nodes

Each CAN node consists of several sub-units:

- Bitstream processor
  The bitstream processor performs data, remote, error and overload frame processing according to the ISO 11898 standard. This includes conversion between the serial data stream and the input/output registers.
- Bit timing unit
  The bit timing unit determines the length of a bit time and the location of the sample point according to the user settings, taking into account propagation delays and phase shift errors. The bit timing unit also performs re-synchronization.
- Error handling unit
  The error handling unit manages the receive and transmit error counter. Depending on the contents of both counters, the CAN node is set into an error-active, error passive or bus-off state.
- Node control unit
  The node control unit coordinates the operation of the CAN node:
  – Enable/disable CAN transfer of the node

  - Enable/disable and generate node-specific events that lead to an interrupt request (CAN bus errors, successful frame transfers etc.)
  - Administration of the frame counter
- Interrupt control unit
  The interrupt control unit in the CAN node controls the interrupt generation for the different conditions that can occur in the CAN node.

11.8.1.2 Message controller

The message controller handles the exchange of CAN frames between the CAN nodes and the message objects that are stored in the message RAM. The message controller performs several functions:

- Receive acceptance filtering to determine the correct message object for storing of a received CAN frame
- Transmit acceptance filtering to determine the message object to be transmitted first, individually for each CAN node
- Transfer contents between message objects and the CAN nodes, taking into account the status/control bits of the message objects
- Handling of the FIFO buffering and gateway functionality
- Aggregation of message-pending notification bits

11.8.1.3 List controller

The list controller performs all operations that lead to a modification of the double- chained message object lists. Only the list controller is allowed to modify the list structure. The allocation/de-allocation or reallocation of a message object can be requested via a user command interface (command panel). The list controller state machine then performs the requested command autonomously.

11.8.1.4 Interrupt control

The general interrupt structure is shown in **Figure 146**. The interrupt event can trigger the interrupt generation. The interrupt pulse is generated independently of the interrupt flag in the interrupt status register. The interrupt flag can be reset by software by writing a 0 to it.

If enabled by the related interrupt enable bit in the interrupt enable register, an interrupt pulse can be generated at one of the 3 interrupt output lines INT_Om of the MultiCAN+. If more than one interrupt source is connected to the same interrupt node pointer (in the interrupt node pointer register), the requests are combined to one common line.

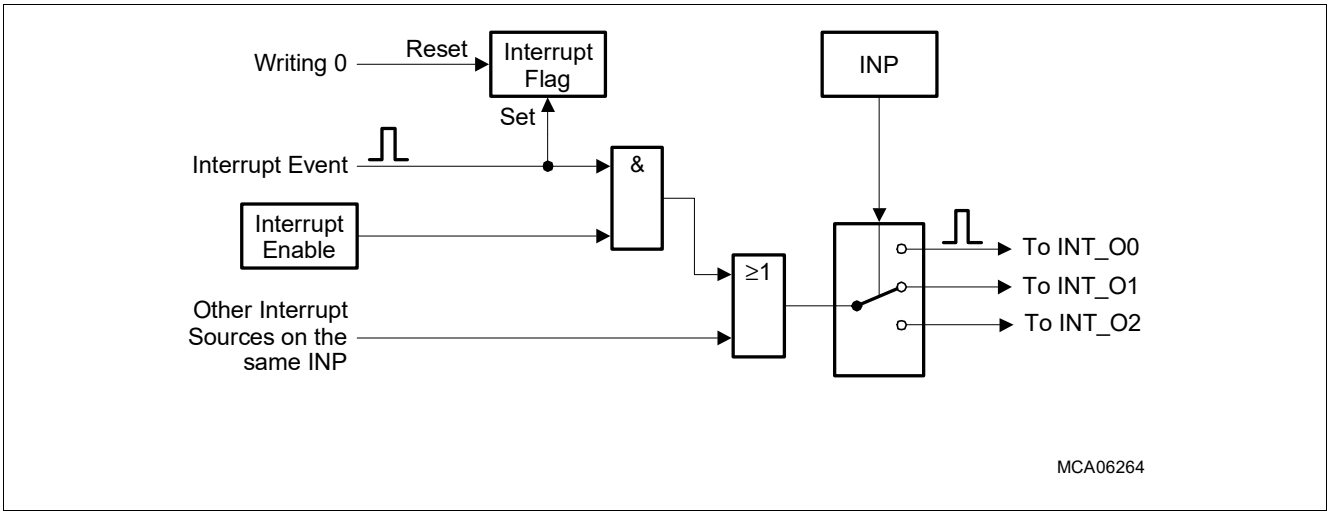


Figure 146 General interrupt structure

11.8.2 Clock control

The MultiCAN+ timer clock f_{CAN} of the functional blocks of the MultiCAN+ is derived from the asynchronous clock source f_A. The fractional divider is used to generate f_{CAN} used for bit timing calculation, The frequency of f_{CAN} is identical for all CAN nodes. The register file operates with the module control clock f_{CLC}. See also **Figure 147**.

The output clock f_{CAN} of the fractional divider is based on the clock f_A, but only every nth clock pulse is taken.

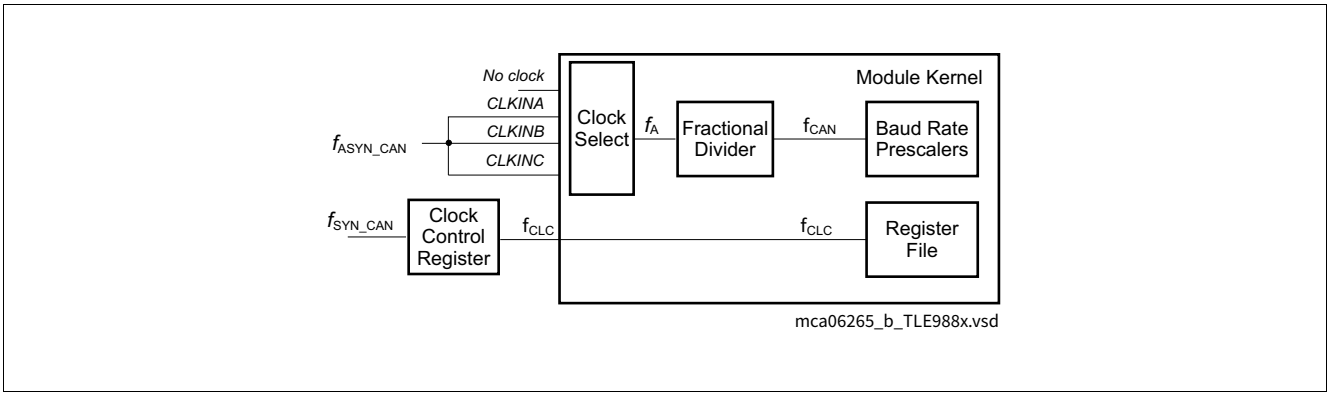


Figure 147 MultiCAN+ clock generation

The f_{SYN_CLK} is identical to f_{PER_CLK}. CLKINx please refer to **Product definitions**, **MultiCAN interconnections**.

Table 126 indicates the minimum operating frequencies in MHz for f_{CLC} that are required for a baud-rate of 1 Mbit/s for the active CAN nodes. If a lower baud-rate is desired, the values can be scaled linearly (e.g. for a maximum of 500 Kbit/s, 50% of the indicated value are required).

For CAN FD operations, please refer to **Table 127** for the minimum operating frequency in MHz for f_{CLC} that are required.

The values imply that the CPU executes maximum accesses to the MultiCAN+. The values may contain rounding effects.

Table 126 Minimum operating frequencies[1] in MHz

Number of allocated message objects MO[2],	Number of active CAN
	1
16 MO	12
32 MO	15

1) In the case of 15 time quanta, the minimum operating frequency required is 15 MHz.
2) Only those message objects have to be taken into account that are allocated to a CAN node. The unallocated message objects have no influence on the minimum operating frequency.

Table 127 Minimum operating frequencies for CAN FD in MHz

No. of allocated MO[1]	No. of active CAN nodes	Acceleration factor[2][3]				
		1	2	3		
16 MO	1	**20**	**40**	**60**		
32 MO	1	**20**	**40**	**60**		

1) Only those message objects have to be taken into account that are allocated to a CAN node. The unallocated message objects have no influence on the minimum operating frequency.
When message objects are configured as transmit or receive FIFO structures i.e. MOFCRn.MMC = 0001/0010, only the base object of 1 is counted towards the minimum frequency requirement, all other slave objects on the FIFO do not count towards the minimum frequency requirement. See **Chapter 11.8.10.5**
2) Acceleration Factor is the ratio of the Data Bit Rate to Nominal Bit Rate. The Nominal Bit Rate is taken as 1Mbit/s in table above. As an example, an acceleration factor (A.F) of 4 refers to a Data Bit Rate of 4 Mbit/s and Nominal Bit Rate is 1 Mbit/s.
Only the node operating with the highest baud-rate need to be considered when several nodes operate at different acceleration factor. e.g. When 1 node operate at A.F of 3 and 3 nodes operate at A.F of 2, the minimum operating frequency required is determined by the node operate at A.F of 3.
3) Please note that other combinations of acceleration factor are possible, values here are for illustrative purposes only. Values with dash '-' indicates required f_{CLC} exceeding the maximum frequency capability off product SCU typically at 100 MHz.

The baud-rate generation of the MultiCAN+ being based on f_A, this frequency has to be chosen carefully to allow correct CAN bit timing. The required value of f_A is given by an integer multiple (n) of the CAN baud-rate multiplied by the number of time quanta per CAN bit time. For example, to reach 1 Mbit/s with 20 tq per bit time, possible values of f_A are given by formula [n × 20] MHz, with n being an integer value, starting at 1.

It is not advised to use fractional divider mode.

Additionally, for correct operation of the MultiCAN, the following conditions have to be fulfilled:

$$\text{Baudrate}_{max} = [(8 \times T_{CAN}) + (8 \times T_{CLC}) + (4 \times \text{No. of active CAN nodes} \times T_{CLC})] \quad (11.1)$$

also

$$\text{NBTR.SJW} < \text{NBTR.TSEG1}$$

As an example, when f_{CLC} = 10 MHz, f_{CAN} = 20 MHz, No of active CAN nodes =2,

Baudratemax=[(8 x 50 ns) + (8 x 100 ns) + (4 x 2 x 100 ns)] = 2000 ns = 500 kBaud

Table 128 illustrates the minimum MultiCAN+ timer clock f_{CAN} and control clock f_{CLC} that is required to support a baud-rate generation of 500 kBaud. If a higher baud-rate is desired, the values need to be calculated as per **Equation (11.1)**.

Table 128 Minimum operating frequencies in MHz required for 500 kBaud

No. of active CAN nodes	$f_{CAN} = f_{CLC}$ (MHz)	$f_{CAN} \neq f_{CLC}$ (MHz)	
		f_{CAN}	f_{CLC}
1	**10**	**16**	**8**
		20	**8**
		24	**8**
		80	**7**

11.8.3 Port input control

It is possible to select the input lines for the RXDCx inputs for the CAN nodes. The selected input is connected to the CAN node and is also available to wake-up the system. More details are defined in **Chapter 11.10.2.2**.

11.8.4 CAN node control

Each CAN node may be configured and run independently of the other CAN node. Each CAN node is equipped with its own node control logic to configure the global behavior and to provide status information.

Note: In the following descriptions, index "x" stands for the node number and index "n" represents the message object number.

Configuration mode is activated when bit NCRx.CCE is set to 1. This mode allows CAN bit timing parameters and the error counter registers to be modified.

11.8.4.1 CAN analyzer mode

CAN analyzer mode is activated when bit NCRx.CALM is set to 1. In this operation mode, data and remote frames are monitored without active participation in any CAN transfer (CAN transmit pin is held on recessive level). Incoming remote frames are stored in a corresponding transmit message object, while arriving data frames are saved in a matching receive message object.

In CAN analyzer mode, the entire configuration information of the valid (including ACK) received frame is stored in the corresponding message object, and can be evaluated by the CPU to determine their identifier, IDE bit information and data length code (ID and DLC optionally if the remote monitoring mode is active, bit MOFCRn.RMM = 1). Incoming frames are not acknowledged, and no error frames are generated. If CAN analyzer mode is enabled, remote frames are not responded to by the corresponding data frame, and data frames cannot be transmitted by setting the transmit request bit MOSTATn.TXRQ. Receive interrupts are generated in CAN analyzer mode (if enabled) for all error free received frames.

The node-specific interrupt configuration is also defined by the node control logic via the CAN_NCR0 register bits TRIE, ALIE and LECIE:

- If control bit TRIE is set to 1, a transfer interrupt is generated when the CAN_NSR0 register has been updated (after each successfully completed message transfer)
- If control bit ALIE is set to 1, an alert interrupt is generated when a "bus-off" condition has been recognized or the error warning level has been exceeded or under-run. Additionally, list or object errors lead to this type of interrupt
- If control bit LECIE is set to 1, a last error code interrupt is generated when an error code > 0 is written into bitfield NSRx.LEC by hardware

Setting bit TXDIS in register NCRx stops the transmit activity of this node without affecting reception; bit CANDIS disables the node completely.

The Node x Status register NSRx provides an overview about the current state of the respective CAN node x, comprising information about CAN transfers, CAN node status, and error conditions.

The CAN frame counter can be used to check the transfer sequence of message objects or to obtain information about the instant a frame has been transmitted or received from the associated CAN bus. CAN frame counting is performed by a 16-bit counter, controlled by register NFCRx. Bitfields NFCRx.CFMOD and NFCRx.CFSEL determine the operation mode and the trigger event incrementing the frame counter.

11.8.4.2 Bit timing unit

According to the ISO 11898 standard, a CAN bit time is subdivided into different segments (**Figure 148**). Each segment consists of multiples of a time quantum t_q. The magnitude of t_q is adjusted by Node x Bit Timing register bitfields NBTRx.BRP and NBTRx.DIV8, both controlling the baud-rate prescaler (see CAN_NBTR0 register). The baud-rate prescaler is driven by the module timer clock f_{CAN} (generation and control of f_{CAN} is described in **Chapter 11.10.1**).

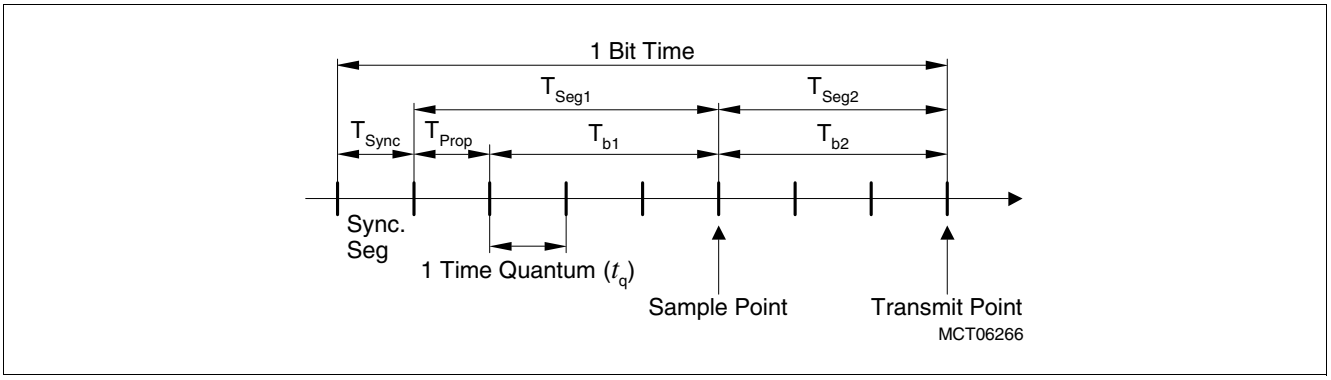


Figure 148 CAN bus bit timing standard

The synchronization segment (T_{Sync}) allows phase synchronization between transmitter and receiver time base. The synchronization segment length is always one t_q. The propagation time segment (T_{Prop}) takes into account the physical propagation delay in the transmitter output driver on the CAN bus line and in the transceiver circuit. For a working collision detection mechanism, T_{Prop} must be two times the sum of all propagation delay quantities rounded up to a multiple of t_q. The phase buffer segments 1 and 2 (T_{b1}, T_{b2}) before and after the signal sample point are used to compensate for a mismatch between transmitter and receiver clock phases detected in the synchronization segment.

The maximum number of time quanta allowed for re-synchronization is defined by bitfield NBTRx.SJW. The propagation time segment and the phase buffer segment 1 are combined to parameter T_{Seg1}, which is defined by the value NBTRx.TSEG1. A minimum of 3 time quanta is demanded by the ISO standard. Parameter T_{Seg2}, which is defined by the value of NBTRx.TSEG2, covers the phase buffer segment 2. A minimum of 2 time quanta is demanded by the ISO standard. According to ISO standard, a CAN bit time, calculated as the sum of T_{Sync}, T_{Seg1} and T_{Seg2}, must not fall below 8 time quanta.

CAN Controller (MultiCAN+)

Calculation of the bit time:

t_q	= (BRP + 1) / f_{CAN}	if DIV8 = 0
	= 8 × (BRP + 1) / f_{CAN}	if DIV8 = 1
T_{Sync}	= 1 × t_q	
T_{Seg1}	= (TSEG1 + 1) × t_q	(min. 3 t_q)
T_{Seg2}	= (TSEG2 + 1) × t_q	(min. 2 t_q)
bit time	= T_{Sync} + T_{Seg1} + T_{Seg2}	(min. 8 t_q)

To compensate phase shifts between clocks of different CAN controllers, the CAN controller must synchronize on any edge from the recessive to the dominant bus level. The hard synchronization is enabled (at the start of frame), the bit time is restarted at the synchronization segment. Otherwise, the re-synchronization jump width T_{SJW} defines the maximum number of time quanta, a bit time may be shortened or lengthened by one re-synchronization. The value of SJW is defined by bitfield NBTRx.SJW.

T_{SJW}	= (SJW + 1) × t_q
T_{Seg1}	≥ T_{SJW} + T_{prop}
T_{Seg2}	≥ T_{SJW}

The maximum relative tolerance for f_{CAN} in classical CAN format and CAN FD format depends on the phase buffer segments, re-synchronization jump width and the bit time.

Classical CAN format

df_{CAN}	≤ min (T_{b1}, T_{b2}) / [2 × (13 × bit time - T_{b2})]	AND
df_{CAN}	≤ T_{SJW} / 20 × bit time	

CAN FD format

df_{CAN}	≤ min ($T_{b1(N)}$, $T_{b2(N)}$) / [2 × (13 × bit time(N) - $T_{b2(N)}$)]	AND
df_{CAN}	≤ $T_{SJW(N)}$ / 20 × bit time(N)	AND
df_{CAN}	≤ $T_{SJW(D)}$ / 20 × bit time(D)	AND
df_{CAN}	≤ min ($T_{b1(D)}$, $T_{b2(D)}$) / (2 × [(6 × bit time(D) - $T_{b2(D)}$) × $BRP_{(D)}$ / $BRP_{(N)}$ + (7 × bit time(N))] (	AND
df_{CAN}	≤ [$T_{SJW(D)}$ - ($BRP_{(N)}$ / $BRP_{(D)}$ - 1)] / (2 × [(2 × bit time(N) - $T_{b2(N)}$) × $BRP_{(N)}$ / $BRP_{(D)}$ + $T_{b2(D)}$ + 4 × bit time(D)])	

A valid CAN bit timing must be written to the CAN Node Bit Timing register NBTR and Fast Node Bit Timing register before resetting the INIT bit in the Node Control register, i.e. before enabling the operation of the CAN node.

The Node Bit Timing register may be written only if bit CCE (Configuration Change Enable) is set in the corresponding Node Control register.

11.8.4.3 Bitstream processor

Based on the message objects in the message buffer, the bitstream processor generates the remote and data frames to be transmitted via the CAN bus. It controls the CRC generator and adds the checksum information

to the new remote or data frame. After including the SOF bit and the EOF field, the bitstream processor starts the CAN bus arbitration procedure and continues with the frame transmission when the bus was found in idle state. While the data transmission is running, the bitstream processor continuously monitors the I/O line. If (outside the CAN bus arbitration phase or the acknowledge slot) a mismatch is detected between the voltage level on the I/O line and the logic state of the bit currently sent out by the transmit shift register, a CAN LEC error interrupt request is generated, and the error code is indicated by the Node x Status register bit field NSRx.LEC.

The data consistency of an incoming frame is verified by checking the associated CRC field. When an error has been detected, a CAN LEC error interrupt request is generated and the associated error code is presented in the Node x Status register NSRx. Furthermore, an error frame is generated and transmitted on the CAN bus. After decomposing a faultless frame into identifier and data portion, the received information is transferred to the message buffer executing remote and data frame handling, interrupt generation and status processing.

11.8.4.4 Error handling unit

The error handling unit of a CAN node x is responsible for the fault confinement of the CAN device. Its two counters, the Receive Error Counter REC and the Transmit Error Counter TEC (see NECNT0 register) are incremented and decremented by commands from the bitstream processor. If the bitstream processor itself detects an error while a transmit operation is running, the Transmit Error Counter is incremented by 8. An increment of 1 is used when the error condition was reported by an external CAN node via an error frame generation. For error analysis, the transfer direction of the disturbed message and the node that recognizes the transfer error are indicated for the respective CAN node x in register NECNTx. Depending on the values of the error counters, the CAN node is set into error- active, error-passive, or bus-off state.

The CAN node is in error-active state if both error counters are below the error-passive limit of 128. The CAN node is in error-passive state, if at least one of the error counters is equal to or greater than 128.

The bus-off state is activated if the Transmit Error Counter is equal to or greater than the bus-off limit of 256. This state is reported for CAN node x by the Node x Status register flag NSRx.BOFF. The device remains in this state, until the “bus-off” recovery sequence is finished. Additionally, Node x Status register flag NSRx.EWRN is set when at least one of the error counters is equal to or greater than the error warning limit defined by the Node x Error Count register bitfield NECNTx.EWRNLVL. Bit NSRx.EWRN is reset if both error counters fall below the error warning limit again (see CAN_NCR0 register).

11.8.4.5 CAN frame counter

Each CAN node is equipped with a frame counter that counts transmitted/received CAN frames or obtains information about the time when a frame has been started to transmit or be received by the CAN node. CAN frame counting/bit time counting is performed by a 16-bit counter that is controlled by the CAN_NFCR0 register. Bitfield CAN_NFCR0.CFSEL determines the operation mode of the frame counter:

- Frame count mode
  After the successful transmission and/or reception of a CAN frame, the frame counter is copied into the CFCVAL bitfield of the MOIPRn register of the message object involved in the transfer. Afterwards, the frame counter is incremented.
- Time stamp mode
  The frame counter is incremented (internally) with the beginning of a new bit time. Its value is permanently sampled in the CFC field while the bus is idle. The value sampled just before the SOF bit of a new frame is detected is written to the corresponding message object. When the treatment of a message object is finished, the sampling continues.
- Bit timing mode
  Used for baud-rate detection and analysis of the bit timing (**Chapter 11.8.6.3**).

- Error count mode
  The frame counter is incremented when an error frame is received or an error is detected by the node (001_B to 110_B) (see **Table 141** for Encoding of the LEC Bit field). If the NFCRx.CFCIE interrupt bit is enabled, the NFCRx.CFCOV overflow flag will be set when the frame counter overflows. Configuration of CFSEL has no influence.

11.8.4.6 CAN node interrupts

Each CAN node has four hardware triggered interrupt request types that are able to generate an interrupt request upon:

- The successful transmission or reception of a frame
- A CAN protocol error with a last error code
- An alert condition: Transmit/receive error counters reach the warning limit, bus-off state changes, a list length error occurs, or a list object error occurs
- An overflow of the frame counter

Besides the hardware generated interrupts, software-initiated interrupts can be generated using the Module Interrupt Trigger register MITR. Writing a 1 to bit n of bitfield MITR.IT generates an interrupt request signal on the corresponding interrupt output line INT_On. When writing MITR.IT more than one bit can be set resulting in activation of multiple INT_On interrupt output lines at the same time. See also **Chapter 11.4** for further processing of the CAN node interrupts.

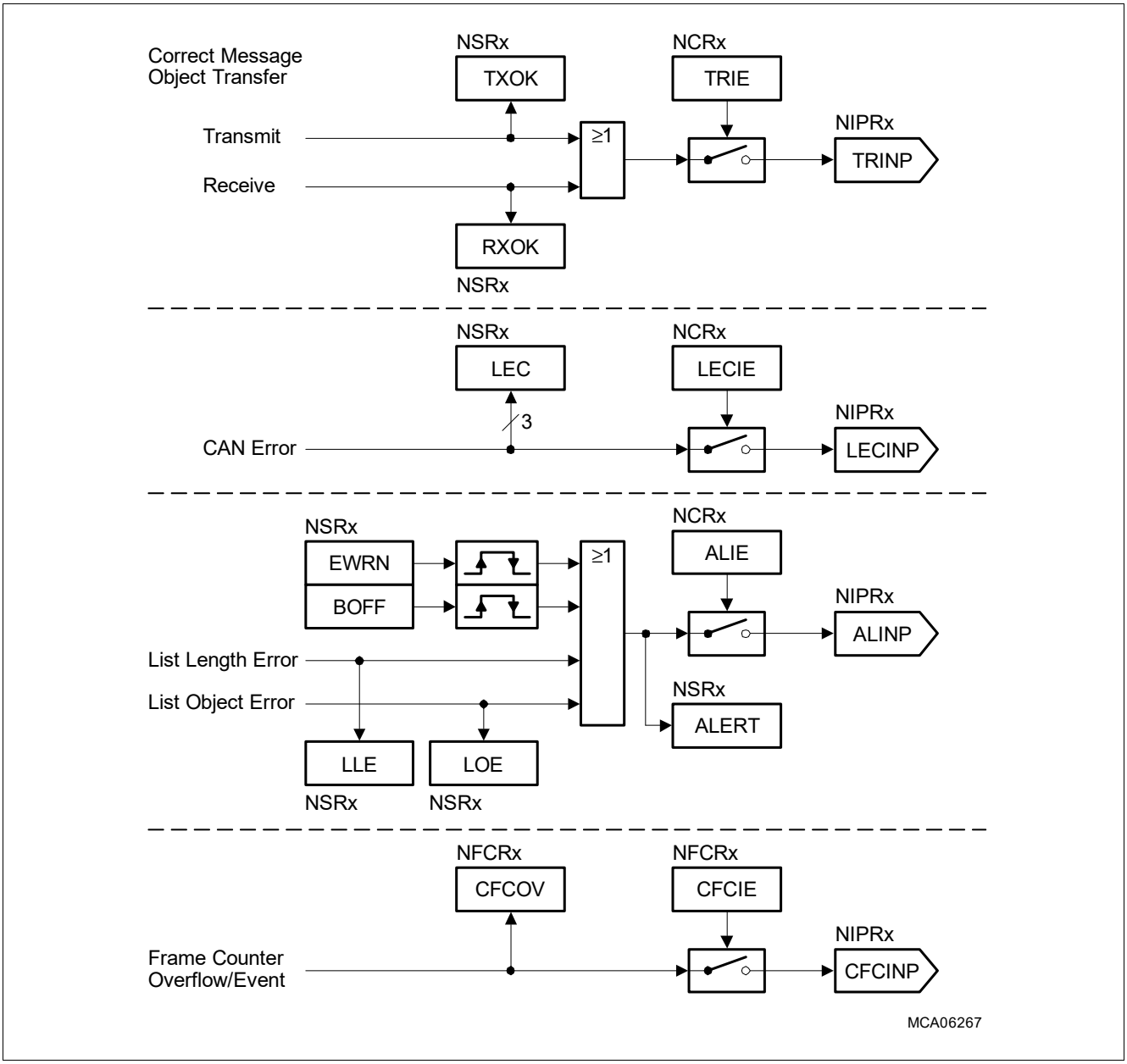


Figure 149 CAN node interrupts

11.8.5 Message object list structure

This chapter describes the structure of the message object lists in the MultiCAN+.

11.8.5.1 Basics

The message objects of the MultiCAN+ are organized in double-chained lists, where each message object has a pointer to the previous message object in the list as well as a pointer to the next message object in the list. The MultiCAN+ provides 8 lists. Each message object is allocated to one of these lists. In the example in **Figure 150**, the three message objects (3, 5, and 16) are allocated to the list with index 2 (List register LIST2).

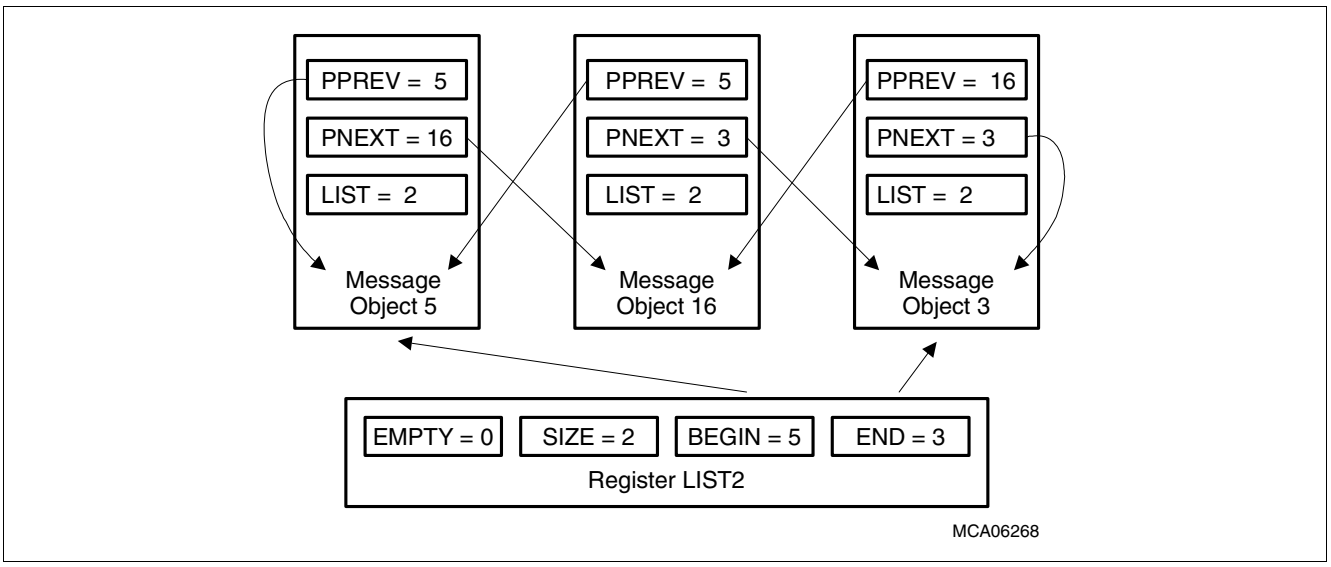


Figure 150 Example allocation of message objects to a list

Bitfield BEGIN in the LISTn register points to the first element in the list (object 5 in the example), and bitfield END points to the last element in the list (object 3 in the example). The number of elements in the list is indicated by bitfield SIZE of the List register (SIZE = number of list elements - 1, thus SIZE = 2 for the 3 elements in the example). The EMPTY bit of the List register indicates whether or not a list is empty (EMPTY = 0 in the example, because list 2 is not empty).

Each message object n has a pointer PNEXT in its CAN_MOSTATn register that points to the next message object in the list, and a pointer PPREV that points to the previous message object in the list. PPREV of the first message object points to the message object itself because the first message object has no predecessor (in the example message object 5 is the first message object in the list, indicated by PPREV = 5). PNEXT of the last message object also points to the message object itself because the last message object has no successor (in the example, object 3 is the last message object in the list, indicated by PNEXT = 3).

Bitfield MOSTATn.LIST indicates the list index number to which the message object is currently allocated to. The message objects of the example are allocated to list 2. Therefore, all LIST bitfields for the message objects assigned to list 2 are set to LIST = 2.

11.8.5.2 List of unallocated elements

The list with list index 0 has a special meaning: it is the list of all unallocated elements. An element is called unallocated if it belongs to list 0 (MOSTATn.LIST = 0). It is called allocated if it belongs to a list with an index not equal to 0 (MOSTATn.LIST > 0).

After reset, all message objects are unallocated. This means that they are assigned to the list of unallocated elements with MOSTATn.LIST = 0. After this initial allocation of the message objects caused by reset, the list of all unallocated message objects is ordered by message number (predecessor of message object n is object n-1, successor of object n is object n+1).

11.8.5.3 Connection to the CAN nodes

Each CAN node is linked to one unique list of message objects. A CAN node performs message transfer only with the message objects that are allocated to the list of the CAN node. This is illustrated in **Figure 151**. Frames that are received on a CAN node may only be stored in one of the message objects that belongs to the CAN node; frames to be transmitted on a CAN node are selected only from the message objects that are allocated to that node, as indicated by the vertical arrows.

There are more lists (8) than CAN nodes (1). This means that some lists are not linked to one of the CAN nodes. A message object that is allocated to one of these unlinked lists cannot receive messages directly from a CAN node and it may not transmit messages.

FIFO and gateway mechanisms refer to message numbers and not directly to a specific list. The user must take care that the message objects targeted by FIFO/gateway belong to the desired list. The mechanisms make it possible to work with lists that do not belong to the CAN node.

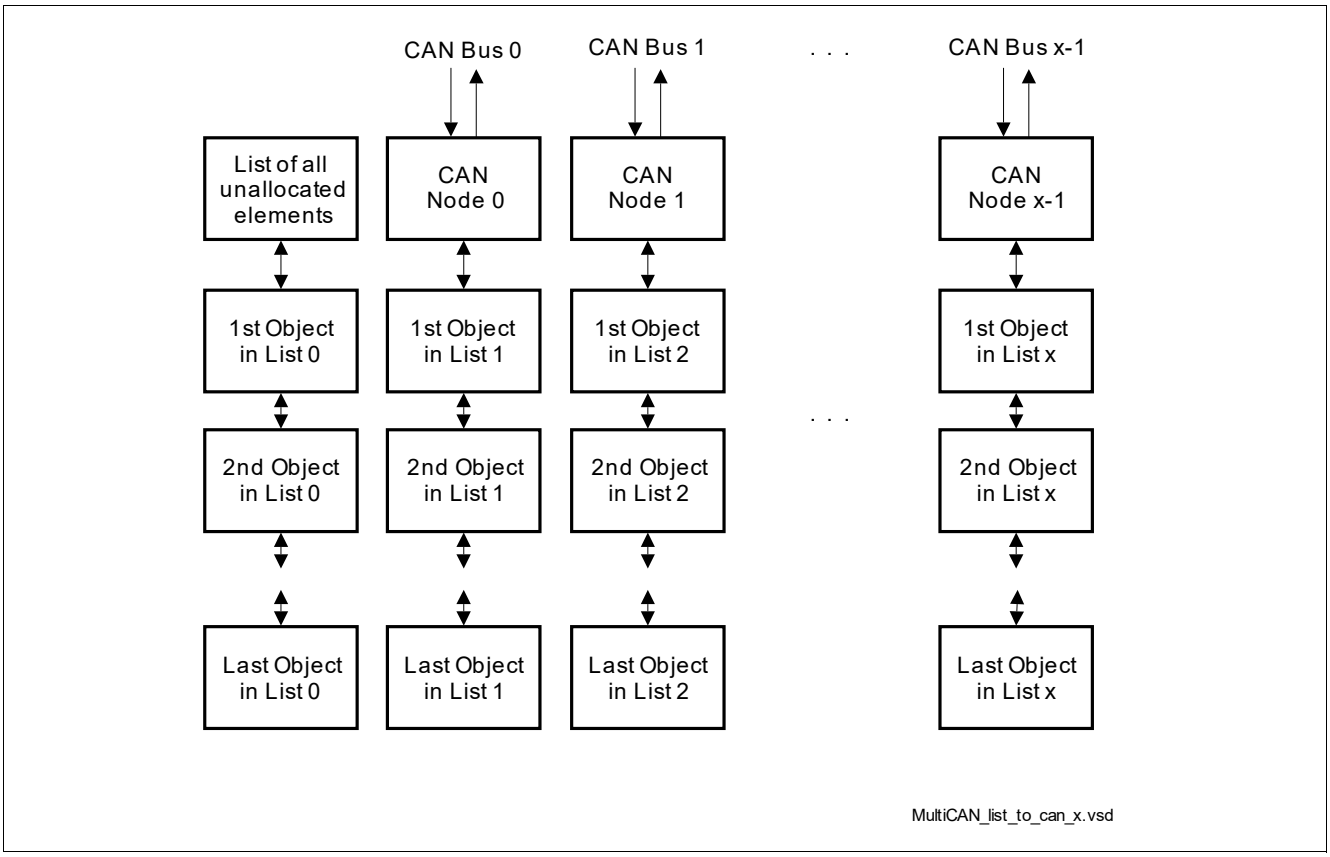


Figure 151 Message objects linked to CAN nodes

11.8.5.4 List command panel

The list structure cannot be modified directly by write accesses to the LIST registers and the PPREV, PNEXT and LIST bitfields in the Message Object Status registers, as they are read only. The list structure is managed by and limited to the list controller inside the MultiCAN+. The list controller is controlled via a command panel allowing the user to issue list allocation commands to the list controller. The list controller has two main purposes:

1. Ensure that all operations that modify the list structure result in a consistent list structure
2. Present maximum ease of use and flexibility to the user

The list controller and the associated command panel allows the programmer to concentrate on the final properties of the list, which are characterized by the allocation of message objects to a CAN node, and the ordering relation between objects that are allocated to the same list. The process of list (re-)building is done in the list controller.

Table 129 gives an overview on the available panel commands while **Table 140** describes the panel commands in more detail.

Table 129 Panel commands overview

Command Name	Description
No Operation	No new command is started.
Initialize Lists	Run the initialization sequence to reset the CTRL and LIST field of all message objects.
Static Allocate	Allocate message object to a list.
Dynamic Allocate	Allocate the first message object of the list of unallocated objects to the selected list.
Static Insert Before	Remove a message object (source object) from the list that it currently belongs to, and insert it before a given destination object into the list structure of the destination object.
Dynamic Insert Before	Insert a new message object before a given destination object.
Static Insert Behind	Remove a message object (source object) from the list that it currently belongs to, and insert it behind a given destination object into the list structure of the destination object.
Dynamic Insert Behind	Insert a new message object behind a given destination object.

A panel command is started by writing the respective command code into the PANCTR register bitfield PANCTR.PANCMD. The corresponding command arguments must be written into bitfields PANCTR.PANAR1 and PANCTR.PANAR2 before writing the command code, or latest along with the command code in a single 32-bit write access to the Panel Control register.

With the write operation of a valid command code, the PANCTR.BUSY flag is set and further write accesses to the Panel Control register are ignored. The BUSY flag remains active and the control panel remains locked until the execution of the requested command has been completed. After a reset and resetting the CLC.DISR bitfield, the list controller builds up list 0. Afterwards the BUSY bit is set, dependent on core speed this might be visible. During list controller initialization, BUSY is set and other accesses to the CAN RAM are forbidden. The CAN RAM can be accessed again when BUSY becomes inactive.

Note: The CAN RAM is automatically initialized after enabling the clocks of the module, by the list controller in order to ensure correct list pointers in each message object. The operation is indicated by automatically setting the BUSY bit. The end of this CAN RAM initialization is indicated by bit

PANCTR.BUSY becoming inactive. It is advised to initialize some registers within the CAN controller before polling the PANCTR.BUSY the first time.

In case of a dynamic allocation command that takes an element from the list of unallocated objects, the PANCTR.RBUSY bit is also set along with the BUSY bit (RBUSY = BUSY = 1). This indicates that bitfields PANAR1 and PANAR2 are going to be updated by the list controller in the following way:

1. The message number of the message object taken from the list of unallocated elements is written to PANAR1.
2. If ERR (bit 7 of PANAR2) is set to 1, the list of unallocated elements was empty and the command is aborted. If ERR is 0, the list was not empty and the command will be performed successfully.

The results of a dynamic allocation command are written before the list controller starts the actual allocation process. As soon as the results are available, RBUSY becomes inactive (RBUSY = 0) again, while BUSY still remains active until completion of the command. This allows the user to set up the new message object while it is still in the process of list allocation. The access to message objects is not limited during ongoing list operations. However, any access to a register resource located inside the RAM delays the ongoing allocation process by one access cycle.

As soon as the command is finished, the BUSY flag becomes inactive (BUSY = 0) and write accesses to the Panel Control register are enabled again. Also, the “No Operation” command code is automatically written to the PANCTR.PANCMD field. A new command may be started any time when BUSY = 0.

All fields of the Panel Control register PANCTR except BUSY and RBUSY may be written by the user. This makes it possible to save and restore the Panel Control register if the Command Panel is used within independent (mutually interruptible) interrupt service routines. If this is the case, any task that uses the Command Panel and that may interrupt another task that also uses the Command Panel should poll the BUSY flag until it becomes inactive and save the whole PANCTR register to a memory location before issuing a command. At the end of the interrupt service routine, the task should restore PANCTR from the memory location.

Before a message object that is allocated to the list of an active CAN node shall be moved to another list or to another position within the same list, bit MOSTATn.MSGVAL (“Message Valid”) of message object n must be cleared.

11.8.6 CAN node analyzer mode

The chapter describes the CAN node analyzer capabilities of the MultiCAN+.

11.8.6.1 Analyzer mode

The CAN analyzer mode makes it possible to monitor the CAN traffic for each CAN node individually without affecting the logical state of the CAN bus. The CAN analyzer mode for CAN node x is selected by setting Node x Control register bit NCRx.CALM.

In CAN analyzer mode, the transmit pin of a CAN node is held at a recessive level permanently. The CAN node may receive frames (data, remote, and error frames) but is not allowed to transmit. Received data/remote frames are not acknowledged (i.e. acknowledge slot is sent recessive) but will be received and stored in matching message objects as long as there is any other node that acknowledges the frame. The complete message object functionality is available, but no transmit request will be executed. CAN analyzer mode works for both classical CAN format and CAN FD format.

11.8.6.2 Loop-Back mode

The MultiCAN+ provides a Loop-Back mode to enable an in-system test of the MultiCAN+ as well as the development of CAN driver software without access to an external CAN bus.

The loop-back feature consists of an internal CAN bus (inside the MultiCAN+) and a bus select switch for each CAN node (see **Figure 152**). With the switch, each CAN node can be connected either to the internal CAN bus (Loop-Back mode activated) or the external CAN bus, respectively to transmit and receive pins (normal operation). The CAN bus that is not currently selected is driven recessive; this means the transmit pin is held at 1, and the receive pin is ignored by the CAN nodes that are in Loop-Back mode.

The Loop-Back mode is selected for CAN node x by setting the Node x Port Control register bit NPCRx.LBM. All CAN nodes that are in Loop-Back mode may communicate together via the internal CAN bus without affecting the normal operation of the other CAN nodes that are not in Loop-Back mode.

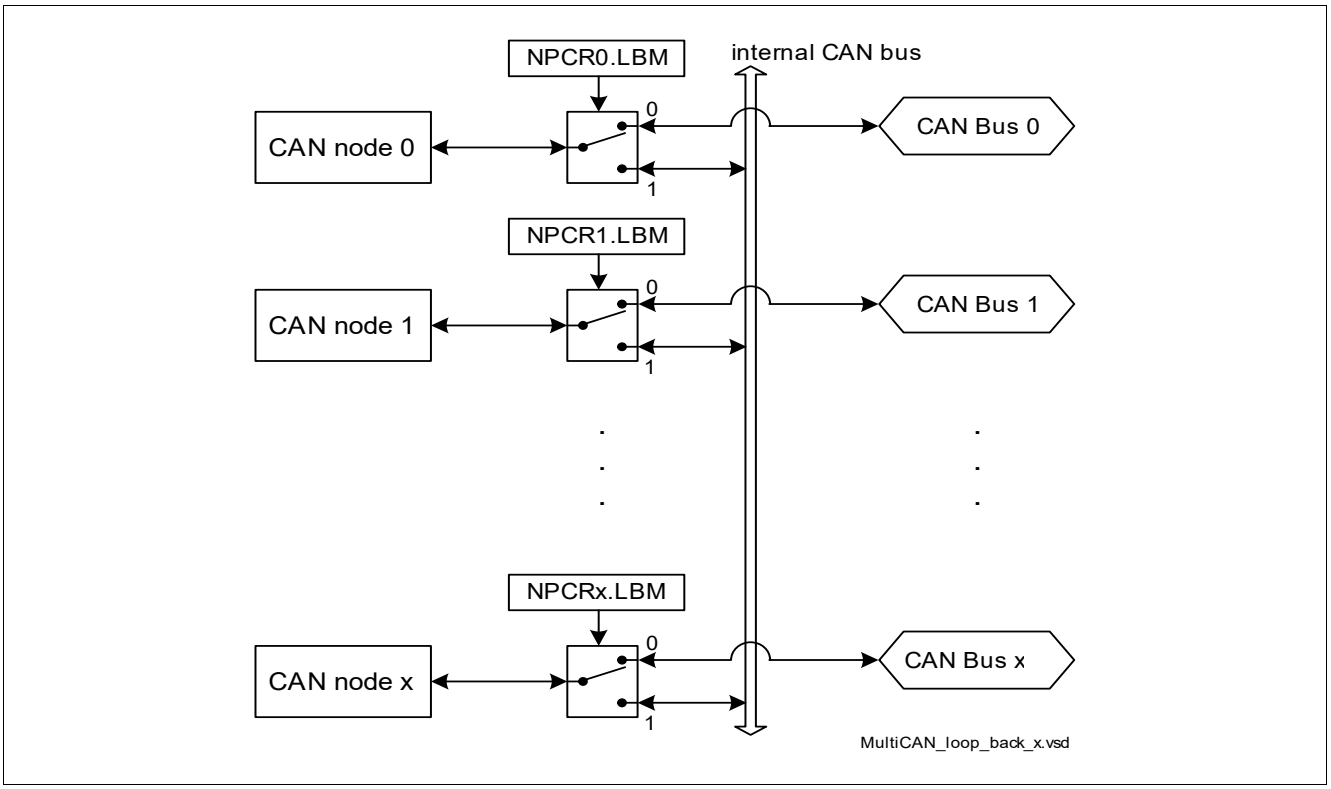


Figure 152 Loop-Back mode

11.8.6.3 Bit timing analysis

Detailed analysis of the bit timing can be performed for each CAN node using the analysis modes of the CAN frame counter. The bit timing analysis functionality of the frame counter may be used for automatic detection of the CAN baud-rate, as well as to analyze the timing of the CAN network.

Bit timing analysis for CAN node x is selected when bitfield NFCRx.CFMOD = 10_B. Bit timing analysis does not affect the operation of the CAN node.

The bit timing measurement results are written into the NFCRx.CFC bitfield. Whenever NFCRx.CFC is updated in bit timing analysis mode, bit NFCRx.CFCOV is also set to indicate the CFC update event. The value of NFCRx.CFC is valid one module cycle later when NFCRx.CFCOV is set. If NFCRx.CFCIE is set, an interrupt request can be generated (see **Figure 149**).

Automatic baud-rate detection

For automatic baud-rate detection, the time between the observation of subsequent dominant edges on the CAN bus must be measured. This measurement is automatically performed if bitfield NFCRx.CFSEL = 000_B. With each dominant edge monitored on the CAN receive input line, the time (measured in f_{CLC} clock cycles) between this edge and the most recent dominant edge is stored in the NFCRx.CFC bitfield.

Synchronization analysis

The bit time synchronization is monitored if NFCRx.CFSEL = 010_B. The time between the first dominant edge and the sample point is measured and stored in the NFCRx.CFC bitfield. The bit timing synchronization offset may be derived from this time as the first edge after the sample point triggers synchronization and there is only one synchronization between consecutive sample points.

Synchronization analysis can be used, for example, for fine tuning of the baud-rate during reception of the first CAN frame with the measured baud-rate.

Driver delay measurement

The delay between a transmitted edge and the corresponding received edge is measured when NFCRx.CFSEL = 011_B (dominant to dominant) and NFCRx.CFSEL = 100_B (recessive to recessive). These delays indicate the time needed to represent a new bit value on the physical implementation of the CAN bus.

11.8.7 Message acceptance filtering

The chapter describes the message acceptance filtering capabilities of the MultiCAN+.

11.8.7.1 Receive acceptance filtering

When a CAN frame is received by a CAN node, a unique message object is determined in which the received frame is stored after successful frame reception. A message object is qualified for reception of a frame if the following six conditions are met.

- The message object is allocated to the message object list of the CAN node by which the frame is received.
- Bit MOSTATn.MSGVAL is set.
- Bit MOSTATn.RXEN is set.
- Bit MOSTATn.DIR is equal to bit RTR of the received frame.
  If bit MOSTATn.DIR = 1 (transmit object), the message object accepts only Remote Frames. If bit MOSTATn.DIR = 0 (receive object), the message object accepts only Data Frames.
- If bit MOAMRn.MIDE = 1, the IDE bit of the received frame becomes evaluated in the following way: If MOARn.IDE = 1, the IDE bit of the received frame must be set (indicates extended identifier). If MOARn.IDE = 0, the IDE bit of the received frame must be cleared (indicates standard identifier).
  If bit MOAMRn.MIDE = 0, the IDE bit of the received frame is “don’t care”. In this case, message objects with standard and extended frames are accepted.
- The identifier of the received frame matches the identifier stored in the Arbitration register of the message object as qualified by the acceptance mask in the MOAMRn register. This means that each bit of the received message object identifier is equal to the bitfield MOARn.ID, except those bits for which the corresponding acceptance mask bits in bitfield MOAMRn.AM are cleared. These identifier bits are “don’t care” for reception. **Figure 153** illustrates this receive message identifier check.

Among all messages that fulfill all six qualifying criteria the message object with the highest receive priority wins receive acceptance filtering and becomes selected to store the received frame. All other message objects lose receive acceptance filtering.

The following priority scheme is defined for the message objects. A message object a (MOa) has higher receive priority than a message object b (MOb) if the following two conditions are fulfilled (see CAN_MOARn register):

1. MOa has a higher priority class than MOb. This means, the 2-bit priority bitfield MOARa.PRI must be equal or less than bitfield MOARb.PRI.
2. If both message objects have the same priority class (MOARa.PRI = MOARb.PRI), MOb is a list successor of MOa. This means that MOb can be reached by means of successively stepping forward in the list, starting from a.

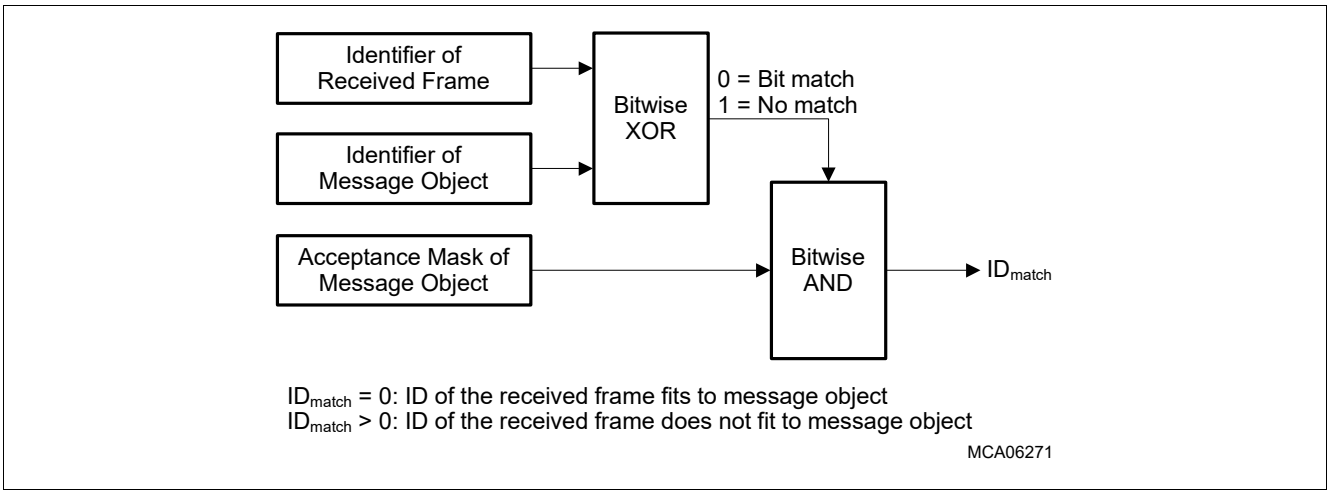


Figure 153 Received message identifier acceptance check

11.8.7.2 Transmit acceptance filtering

A message is requested for transmission by setting a transmit request in the message object that holds the message. If more than one message object have a valid transmit request for the same CAN node, one of these message objects is chosen for transmission, because only a single message object can be transmitted at one time on a CAN bus.

A message object is qualified for transmission on a CAN node if the following four conditions are met (see also **Figure 154**).

1. The message object is allocated to the message object list of the CAN node.
2. Bit MOSTATn.MSGVAL is set.
3. Bit MOSTATn.TXRQ is set.
4. Bit MOSTATn.TXEN0 and MOSTATn.TXEN1 are set.

A priority scheme determines which one of all qualifying message objects is transmitted first. It is assumed that message object a (MOa) and message object b (MOb) are two message objects qualified for transmission. MOb is a list successor of MOa. For both message objects, CAN messages CANa and CANb are defined (identifier, IDE, and RTR are taken from the message-specific bitfields and bits MOARn.ID, MOARn.IDE and MOSTATn.DIR).
If both message objects belong to the same priority class (identical PRI bitfield in register MOARn), MOa has a higher transmit priority than MOb if one of the following conditions is fulfilled.

- PRI = 10_B and CAN message MOa has higher or equal priority than CAN message MOb with respect to CAN arbitration rules (see **Table 146**).
- PRI = 01_B or PRI = 11_B (priority by list order).
- PRI = 00_B is reserved and makes the message object to have no function.

The message object that is qualified for transmission and has highest transmit priority wins the transmit acceptance filtering, and will be transmitted first. All other message objects lose the current transmit acceptance filtering round. They get a new chance in subsequent acceptance filtering rounds.

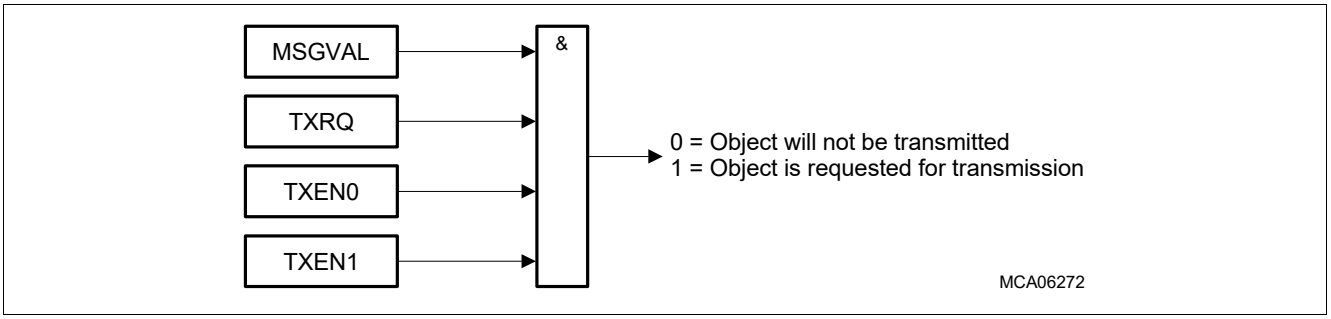


Figure 154 Effective transmit request of message object

11.8.8 Message postprocessing

After a message object has successfully received or transmitted a frame, the CPU can be notified to perform a postprocessing on the message object. The postprocessing of the MultiCAN+ consists of two elements:

- Message interrupts to trigger postprocessing
- Message pending registers to collect pending message interrupts into a common structure for postprocessing

11.8.8.1 Message object interrupts

When the storage of a received frame into a message object or the successful transmission of a frame is completed, a message interrupt can be issued. For each message object, a transmit and a receive interrupt can be generated and routed to one of the sixteen CAN interrupt output lines (see **Figure 155**). A receive interrupt occurs also after a frame storage event that has been induced by a FIFO or a gateway action. The status bits TXPND and RXPND in the Message Object n Status register are always set after a successful transmission/reception, whether or not the respective message interrupt is enabled.

A third FIFO full interrupt condition of a message object is provided. If bitfield MOFCRn.OVIE (overflow interrupt enable) is set, the FIFO full interrupt will be activated depending on the actual message object type.

In case of a Receive FIFO Base Object (MOFCRn.MMC = 0001_B), the FIFO full interrupt is routed to the interrupt output line INT_Om as defined by the transmit interrupt node pointer MOIPRn.TXINP.

In case of a Transmit FIFO Base Object (MOFCRn.MMC = 0010_B), the FIFO full interrupt becomes routed to the interrupt output line INT_Om as defined by the receive interrupt node pointer MOIPRn.RXINP.

See also **Chapter 11.4** for further processing of the message object interrupts.

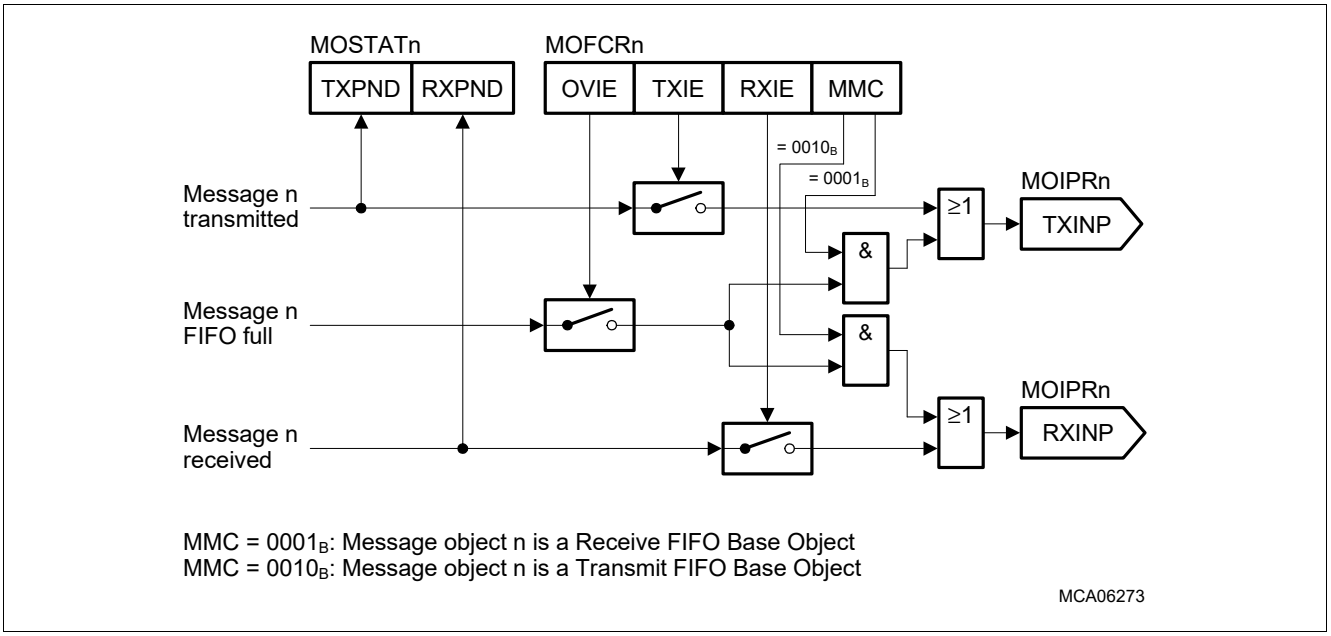


Figure 155 Message interrupt request routing

11.8.8.2 Pending messages

When a message interrupt request is generated, a message pending bit is set in one of the Message Pending registers. There are 2 Message Pending registers, MSPNDk (k = 0-1) with 32 pending bits available each. The general **Figure 156** shows the allocation of the message pending bits in case that the maximum possible number of eight Message Pending registers are implemented and available on the chip.

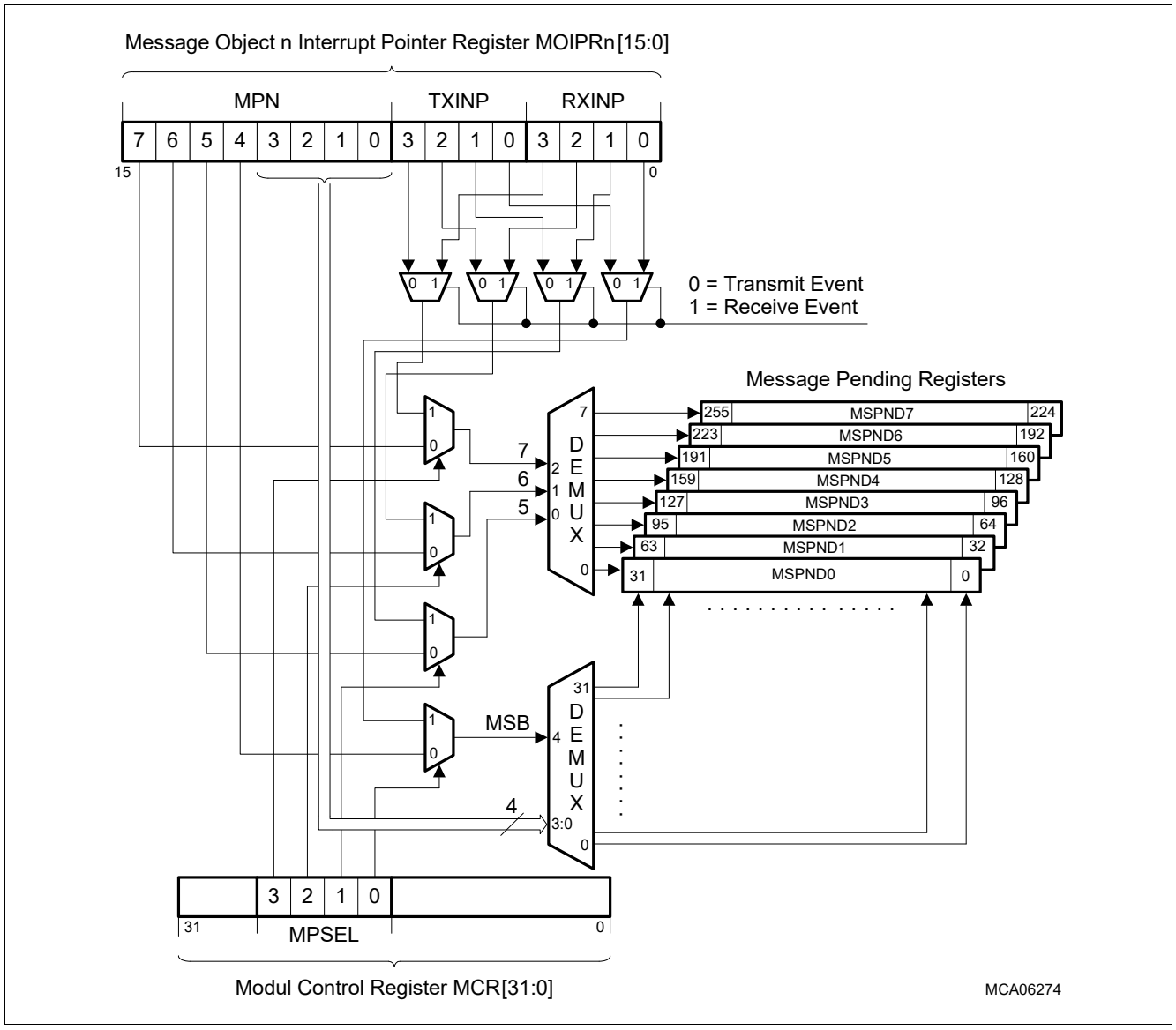


Figure 156 Message pending bit allocation

The location of a pending bit is defined by two demultiplexers selecting the number k of the MSPNDk registers (3-bit demux), and the bit location within the corresponding MSPNDk register (5-bit demux).

Allocation case 1

In this allocation case, bitfield MCR.MPSEL = 0000_B. The location selection consists of 2 parts:

- The upper three bits of MOIPRn.MPN (MPN[7:5]) select the number k of a Message Pending register MSPNDk in which the pending bit will be set
- The lower five bits of MOIPRn.MPN (MPN[4:0]) select the bit position (0-31) in MSPNDk for the pending bit to be set

Allocation case 2

In this allocation case, bitfield MCR.MPSEL is taken into account for pending bit allocation. Bitfield MCR.MPSEL makes it possible to include the interrupt request node pointer for reception (MOIPRn.RXINP) or transmission (MOIPRn.TXINP) for pending bit allocation in such a way that different target locations for the pending bits are used in receive and transmit case. If MPSEL = 1111_B, the location selection operates in the following way:

- At a transmit event, the upper 3 bits of TXINP determine the number k of a Message Pending register MSPNDk in which the pending bit will be set. At a receive event, the upper 3 bits of RXINP determine the number k
- The bit position (0-31) in MSPNDk for the pending bit to be set is selected by the lowest bit of TXINP or RXINP (selects between low and high half-word of MSPNDk) and the four least significant bits of MPN

General hints

The Message Pending registers MSPNDk can be written by software. Bits that are written with 1 are left unchanged, and bits which are written with 0 are cleared. This makes it possible to clear individual MSPNDk bits with a single register write access. Therefore, access conflicts are avoided when the MultiCAN+ (hardware) sets another pending bit at the same time when software writes to the register.

Each Message Pending register MSPNDk is associated with a MSIDk register which indicates the lowest bit position of all set (1) bits in Message Pending register k. The MSIDk register is a read-only register that is updated immediately when a value in the corresponding Message Pending register k is changed by software or hardware.

11.8.9 Message object data handling

This chapter describes the handling capabilities for the Message Object Data of the MultiCAN+.

11.8.9.1 Frame reception

After the reception of a message, it is stored in a message object according to the scheme shown in **Figure 157**. The MultiCAN+ not only copies the received data into the message object, and it provides advanced features to enable consistent data exchange between MultiCAN+ and CPU.

11.8.9.1.1 MSGVAL

Bit MSGVAL (Message Valid) in the Message Object n Status register MOSTATn is the main switch of the message object. During the frame reception, information is stored in the message object only when MSGVAL = 1. If bit MSGVAL is reset by the CPU, the MultiCAN+ stops all ongoing write accesses to the message object. Now the message object can be re-configured by the CPU with subsequent write accesses to it without being disturbed by the MultiCAN+.

11.8.9.1.2 RTSEL

When the CPU re-configures a message object during CAN operation (for example, clears MSGVAL, modifies the message object and sets MSGVAL again), the following scenario can occur:

1. The message object wins receive acceptance filtering
2. The CPU clears MSGVAL to re-configure the message object
3. The CPU sets MSGVAL again after re-configuration
4. The end of the received frame is reached. As MSGVAL is set, the received data is stored in the message object, a message interrupt request is generated, gateway and FIFO actions are processed, etc.

After the re-configuration of the message object (after step 3 above) the storage of further received data may be undesirable. This can be achieved through bit MOSTATn.RTSEL (receive/transmit selected) that makes it possible to disconnect a message object from an ongoing frame reception.

When a message object wins the receive acceptance filtering, its RTSEL bit is set by the MultiCAN+ to indicate an upcoming frame delivery. The MultiCAN+ checks RTSEL whether it is set on successful frame reception to verify that the object is still ready for receiving the frame. The received frame is then stored in the message object (along with all subsequent actions such as message interrupts, FIFO & gateway actions, flag updates) only if RTSEL = 1.

When a message object is invalidated during CAN operation (resetting bit MSGVAL), RTSEL should be cleared before setting MSGVAL again (latest with the same write access that sets MSGVAL) to prevent the storage of a frame that belongs to the old context of the message object. Therefore, a message object re-configuration should consist of the following steps:

1. Clear MSGVAL bit
2. Re-configure the message object while MSGVAL = 0
3. Clear RTSEL bit and set MSGVAL again

11.8.9.1.3 RXEN

Bit MOSTATn.RXEN enables a message object for frame reception. A message object can receive CAN messages from the CAN bus only if RXEN = 1. The MultiCAN+ evaluates RXEN only during receive acceptance filtering. After receive acceptance filtering, RXEN is ignored and has no further influence on the actual storage of a received message in a message object.

Bit RXEN enables the “soft phase out” of a message object: after clearing RXEN, a currently received CAN message for which the message object has won acceptance filtering is still stored in the message object but for subsequent messages the message object no longer wins receive acceptance filtering.

11.8.9.1.4 RXUPD, NEWDAT and MSGLST

An ongoing frame storage process is indicated by the RXUPD (receive updating) flag in the MOSTATn register. RXUPD is set with the start and cleared with the end of a message object update, which consists of frame storage as well as flag updates.

After storing the received frame (identifier, IDE bit, DLC; including the data field for data frames), the NEWDAT (new data) bit of the message object is set. If NEWDAT was already set before it becomes set again, bit MSGLST (message lost) is set to indicate a data loss condition.

The RXUPD and NEWDAT flags can help to read consistent frame data from the message object during an ongoing CAN operation. The following steps are recommended to be executed:

1. Clear NEWDAT bit
2. Read message content (identifier, data etc.) from the message object
3. Check that both, NEWDAT and RXUPD, are cleared. If this is not the case, go back to step 1
4. When step 3 was successful, the message object contents is consistent and has not been updated by the MultiCAN+ while reading

Bits RXUPD, NEWDAT and MSGLST have the same behavior for the reception of data as well as remote frames.

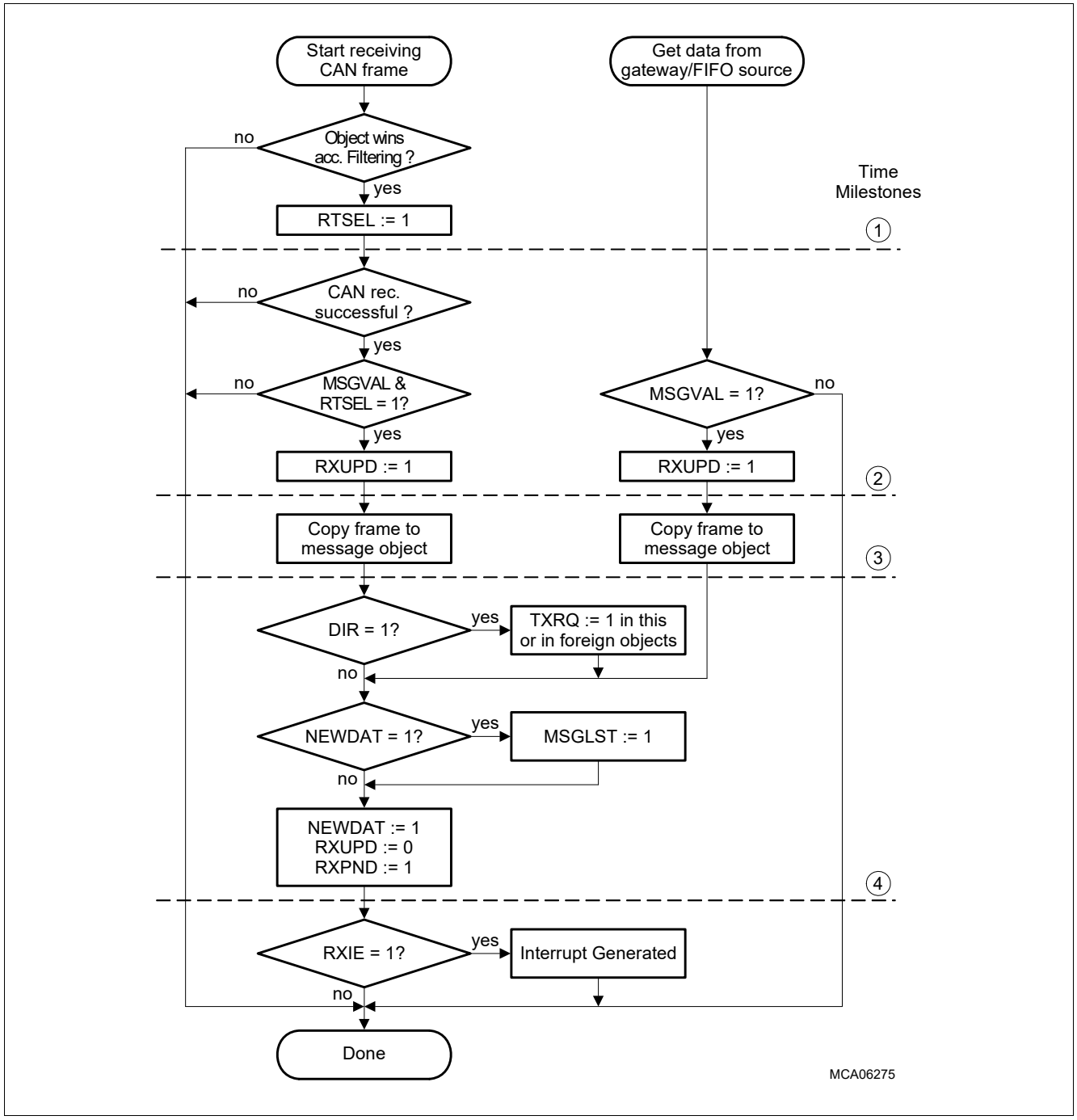


Figure 157 Reception of a message object

11.8.9.2 Frame transmission

The process of a message object transmission is shown in **Figure 158**. Along with the copy of the message object content to be transmitted (identifier, IDE bit, RTR = DIR bit, DLC, including the data field for data frames) into the internal transmit buffer of the assigned CAN node, several status flags are also served and monitored to control consistent data handling.

The transmission process of a message object starting after the transmit acceptance filtering is identical for remote and data frames.

11.8.9.2.1 MSGVAL, TXRQ, TXEN0, TXEN1

A message can only be transmitted if all four bits in registers MOSTATn, MSGVAL (message valid), TXRQ (transmit request), TXEN0 (transmit enable 0), TXEN1 (transmit enable 1) are set as shown in **Figure 154**. Although these bits are equivalent with respect to the transmission process, they have different semantics:

Table 130 Message transmission bit definitions

Bit	Description
MSGVAL	Message valid This is the main switch bit of the message object.
TXRQ	Transmit request This is the standard transmit request bit. This bit must be set whenever a message object should be transmitted. TXRQ is cleared by hardware at the end of a successful transmission, except when there is new data (indicated by NEWDAT = 1) to be transmitted. When bit MOFCRn.STT (“Single Transmit Trial”) is set, TXRQ becomes already cleared when the contents of the message object are copied into the transmit frame buffer of the CAN node. A received remote request (after a remote frame reception) sets bit TXRQ to request the transmission of the requested data frame.
TXEN0	Transmit enable 0 This bit can be temporarily cleared by software to suppress the transmission of this message object when it writes new content to the data field. This avoids transmission of inconsistent frames that consist of a mixture of old and new data. Remote requests are still accepted when TXEN0 = 0, but transmission of the data frame is suspended until transmission is re-enabled by software (setting TXEN0).
TXEN1	Transmit enable 1 This bit is used in transmit FIFOs to select the message object that is transmit active within the FIFO structure. For message objects that are not transmit FIFO elements, TXEN1 can either be set permanently to 1 or can be used as a second independent transmission enable bit.

11.8.9.2.2 RTSEL

When a message object has been identified to be transmitted next after transmission acceptance filtering, bit MOSTATn.RTSEL (receive/transmit selected) is set.

When the message object is copied into the internal transmit buffer, bit RTSEL is checked, and the message is transmitted only if RTSEL = 1. After the successful transmission of the message, bit RTSEL is checked again and the message postprocessing is only executed if RTSEL = 1.

For a complete re-configuration of a valid message object, the following steps should be executed:

1. Clear MSGVAL bit
2. Re-configure the message object while MSGVAL = 0
3. Clear RTSEL and set MSGVAL

Clearing of RTSEL ensures that the message object is disconnected from an ongoing/scheduled transmission and no message object processing (copying message to transmit buffer including clearing NEWDAT, clearing TXRQ, time stamp update, message interrupt, etc.) within the old context of the object can occur after the message object becomes valid again, but within a new context.

11.8.9.2.3 NEWDAT

When the contents of a message object have been transferred to the internal transmit buffer of the CAN node, bit MOSTATn.NEWDAT (new data) is cleared by hardware to indicate that the transmit message object data is no longer new.

When the transmission of the frame is successful and NEWDAT is still cleared (if no new data has been copied into the message object meanwhile), TXRQ (transmit request) is cleared automatically by hardware.

If, however, the NEWDAT bit has been set again by the software (because a new frame should be transmitted), TXRQ is not cleared to enable the transmission of the new data.

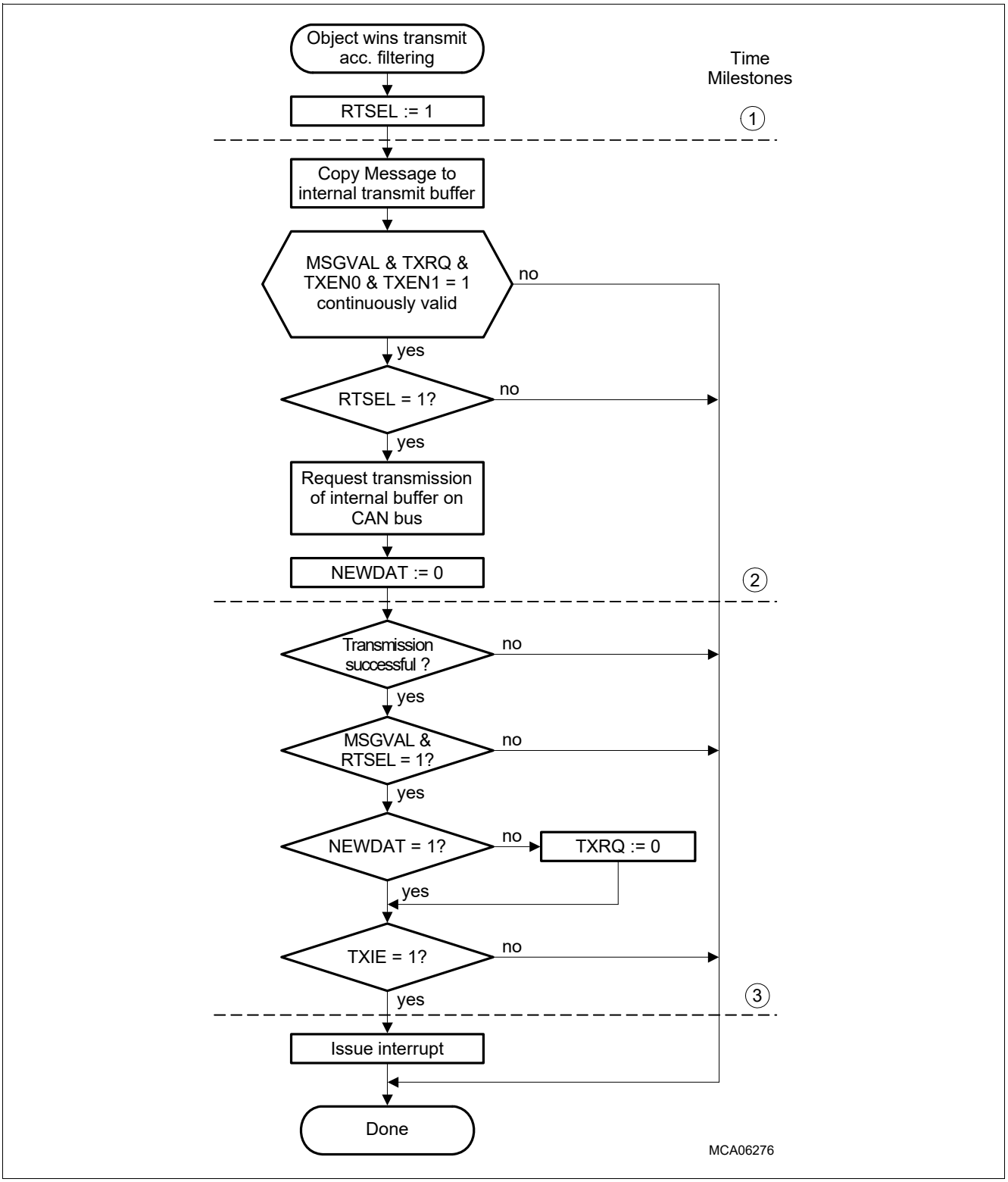


Figure 158 Transmission of a message object

11.8.10 Message object functionality

This chapter describes the functionality of the message objects in the MultiCAN+.

11.8.10.1 Standard message object

A message object is selected as standard message object when bitfield MOFCRn.MMC = 0000_B (see CAN_NFCR0 register). The standard message object can transmit and receive CAN frames according to the basic rules described in the previous sections. Additional services such as single data transfer mode or single transmit trial (see following sections) are available and can be individually selected.

11.8.10.2 Single data transfer mode

Single data transfer mode is a useful feature in order to broadcast data over the CAN bus without unintended duplication of information. Single data transfer mode is selected via bit MOFCRn.SDT.

Message reception

When a received message stored in a message object is overwritten by a new received message, the contents of the first message are lost and replaced with the contents of the new received message (indicated by MSGLST = 1).

If SDT is set (single data transfer mode activated), bit MSGVAL of the message object is automatically cleared by hardware after the storage of a received Data or Remote Frame. This prevents the reception of further messages.

Message transmission

When a message object receives a series of multiple remote requests, it transmits several data frames in response to the remote requests. If the data within the message object has not been updated in the time between the transmissions, the same data can be sent more than once on the CAN bus.

In single data transfer mode (SDT = 1), this is avoided because MSGVAL is automatically cleared after the successful transmission of a data or remote frame.

11.8.10.3 Single transmit trial

If the bit STT in the message object function register is set (STT = 1), the transmission request is cleared (TXRQ = 0) when the frame contents of the message object have been copied to the internal transmit buffer of the CAN node. Thus, the transmission of the message object is not tried again when it fails due to CAN bus errors.

11.8.10.4 Message object format (classical CAN & CAN FD)

Message objects are used for transmission or reception of CAN data frames. The transmit and receive behavior of a classical CAN node in classical CAN (ISO 11898-1:2003(E)) and the chosen CAN FD format are determined by a node control register bit (i.e. NCRx.FDEN) and two message object function control register bits (i.e. MOFCRn.FDF and MOFCRn.BRS) as shown in **Table 145**.

A CAN node which has been set to work in classical CAN mode i.e. (NCRx.FDEN = 0) would be able to transmit message objects programmed for classical CAN frames (i.e. MOFCRn.FDF = 0, MOFCRn.BRS = 0/1) only. When receiving classical CAN data frames in a node that works i.e. classical CAN mode the message object function control register bits (i.e. MOFCRn.FDF and MODFCRn.BRS) are not used and not updated.

In the case when a CAN node set to work in a classical CAN mode and receives a message object request for transmission or reception that does not correspond with the CAN mode (i.e. MOFCRn.FDF = 1, MOFCRn.BRS = 0/1), the transmission would be cancelled (i.e. corresponding MOSTATn.TXRQ cleared) and

CAN frame not received in message object. The CAN node is not blocked and able to transmit or receive other message objects. This is due to the fact that the Receive Error Counter is incremented and not the Transmit Error Counter, therefore no bus-off will occur.

However a CAN node that has been set to work in CAN FD mode i.e. (NCRx.FDEN = 1) is able to transmit and receive classical CAN data frames, long data frames or long + fast data frames. For message object transmission, message object function control bits (i.e. MOFCRn.FDF & MOFCRn.BRS) determines the CAN frame format to be used and functions as status register bits during message reception.

Note: Remote frame requests do not change MOFCRn.FDF and MOFCRn.BRS bits.

11.8.10.5 Message object FIFO structure

In case of high CPU load it may be difficult to process a series of CAN frames in time. This may happen if multiple messages are received or must be transmitted in short time.

Therefore, a FIFO buffer structure is available to avoid loss of incoming messages and to minimize the setup time for outgoing messages. The FIFO structure can also be used to automate the reception or transmission of a series of CAN messages and to generate a single message interrupt when the whole CAN frame series is done.

There can be several FIFOs in parallel. The number of FIFOs and their size are limited only by the number of available message objects. A FIFO can be installed, resized and de-installed at any time, even during CAN operation.

The basic structure of a FIFO is shown in **Figure 159**. A FIFO consists of one base object and n slave objects. The slave objects are chained together in a list structure (similar as in message object lists). The base object may be allocated to any list. Although **Figure 159** shows the base object as a separate part beside the slave objects, it is also possible to integrate the base object at any place into the chain of slave objects. This means that the base object is slave object, too (not possible for gateways). The absolute object numbers of the message objects have no impact on the operation of the FIFO.

The base object does not need to be allocated to the same list as the slave objects. Only the slave object must be allocated to a common list (as they are chained together). Several pointers (BOT, CUR and TOP) that are located in the Message Object n FIFO/Gateway Pointer register MOFGPRn link the base object to the slave objects, regardless whether the base object is allocated to the same or to another list than the slave objects.

The smallest FIFO would be a single message object which is both, FIFO base and FIFO slave (not very useful). The biggest possible FIFO structure would include all message objects of the MultiCAN+ module. Any FIFO sizes between these limits are possible.

In the FIFO base object, the FIFO boundaries are defined. Bitfield MOFGPRn.BOT of the base object points to (includes the number of) the bottom slave object in the FIFO structure. The MOFGPRn.TOP bitfield points to (includes the number of) the top slave object in the FIFO structure. The MOFGPRn.CUR bitfield points to (includes the number of) the slave object that is actually selected by the MultiCAN+ for message transfer. When a message transfer takes place with this object, CUR is set to the next message object in the list structure of the slave objects (CUR = PNEXT of current object). If CUR was equal to TOP (top of the FIFO reached), the next update of CUR will result in CUR = BOT (wrap-around from the top to the bottom of the FIFO). This scheme represents a circular FIFO structure where the bitfields BOT and TOP establish the link from the last to the first element.

Bitfield MOFGPRn.SEL of the base object can be used for monitoring purposes. It makes it possible to define a slave object within the list at which a message interrupt is generated whenever the CUR pointer reaches the value of the SEL pointer. Thus SEL makes it possible to detect the end of a predefined message transfer series or to issue a warning interrupt when the FIFO becomes full.

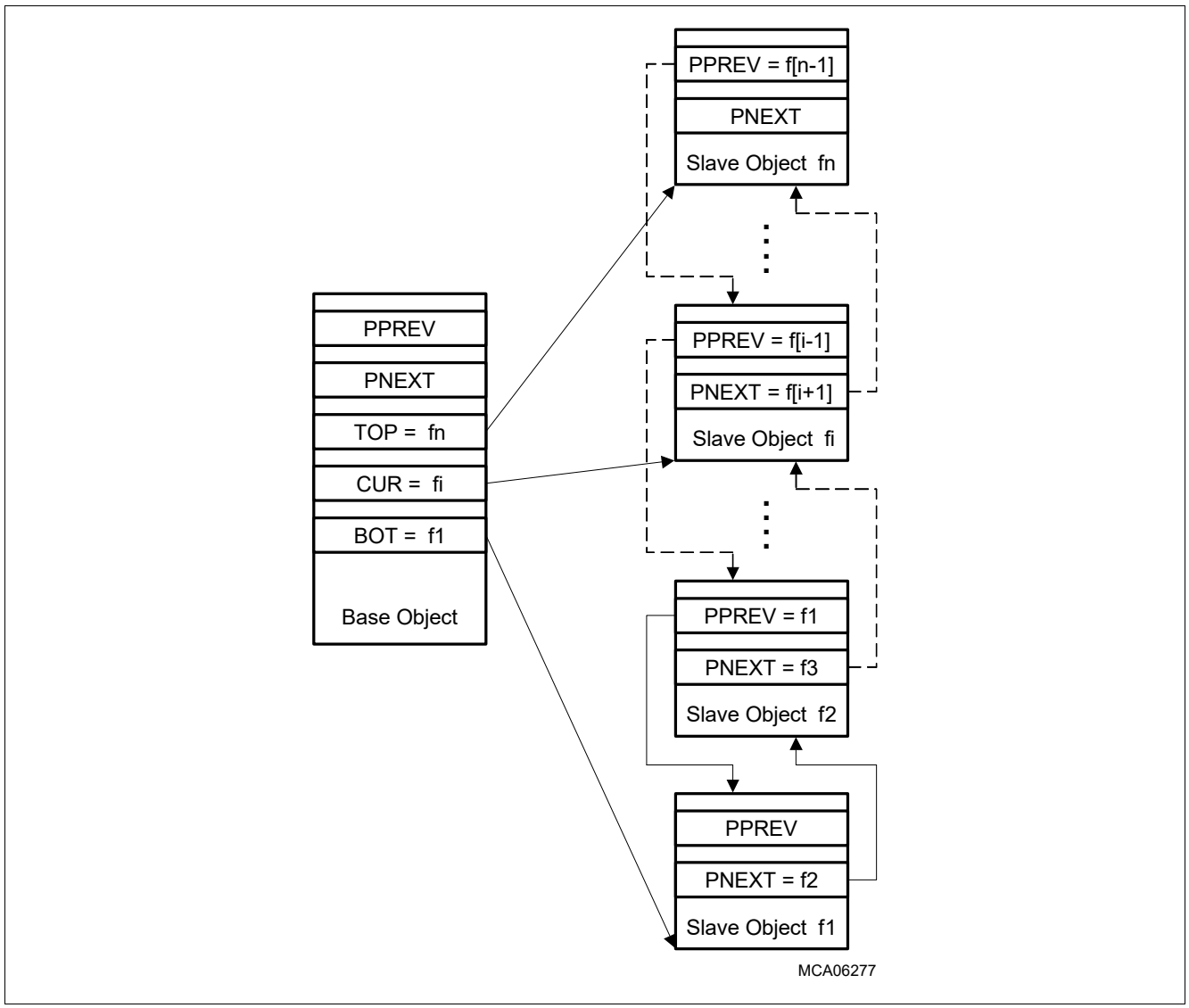


Figure 159 FIFO structure with FIFO base object and n FIFO slave objects

11.8.10.6 Receive FIFO

The receive FIFO structure is used to buffer incoming (received) remote or data frames.

A receive FIFO is selected by setting MOFCRn.MMC = 0001_B in the FIFO base object. This MMC code automatically designates a message object as FIFO base object. The message modes of the FIFO slave objects are not relevant for the operation of the receive FIFO.

When the FIFO base object receives a frame from the CAN node it belongs to, the frame is not stored in the base object itself but in the message object that is selected by the base object's MOFGPRn.CUR pointer. This message object receives the CAN message as if it is the direct receiver of the message. However, MOFCRn.MMC = 0000_B is implicitly assumed for the FIFO slave object, and a standard message delivery is performed. The actual message mode (MMC setting) of the FIFO slave object is ignored. For the slave object, no acceptance filtering takes place that checks the received frame for a match with the identifier, IDE bit, and DIR bit.

With the reception of a CAN frame, the current pointer CUR of the base object is set to the number of the next message object in the FIFO structure. This message object will then be used to store the next incoming message.

If bitfield MOFCRn.OVIE ("Overflow Interrupt Enable") of the FIFO base object is set and the current pointer MOFGPRn.CUR becomes equal to MOFGPRn.SEL, a FIFO overflow interrupt request is generated. This interrupt request is generated on interrupt node TXINP of the base object immediately after the storage of the received frame in the slave object. Transmit interrupts are still generated if TXIE is set.

A CAN message is stored in FIFO base and slave object only if MSGVAL = 1.

In order to avoid direct reception of a message by a slave message object, as if it was an independent message object and not a part of a FIFO, the bit RXEN of each slave object must be cleared. The setting of the bit RXEN is "don't care" only if the slave object is located in a list not assigned to a CAN node.

A connection with an object having MMC=5 is not possible.

11.8.10.7 Transmit FIFO

The transmit FIFO structure is used to buffer a series of data or remote frames that must be transmitted. A transmit FIFO consists of one base message object and one or more slave message objects.

A transmit FIFO is selected by setting MOFCRn.MMC = 0010_B in the FIFO base object. Unlike the receive FIFO, slave objects assigned to the transmit FIFO must explicitly set their bitfields MOFCRn.MMC = 0011_B. The CUR pointer in all slave objects must point back to the Transmit FIFO Base Object (to be initialized by software).

The MOSTATn.TXEN1 bits (transmit enable 1) of all message objects except the one which is selected by the CUR pointer of the base object must be cleared by software. TXEN1 of the message (slave) object selected by CUR must be set. CUR (of the base object) may be initialized to any FIFO slave object.

When tagging the message objects of the FIFO as valid to start the operation of the FIFO, then the base object must be tagged valid (MSGVAL = 1) first.

Before a transmit FIFO becomes de-installed during operation, its slave objects must be tagged invalid (MSGVAL = 0).

The transmit FIFO uses the TXEN1 bit in the Message Object Status register of all FIFO elements to select the actual message for transmission. Transmit acceptance filtering evaluates TXEN1 for each message object and a message object can win transmit acceptance filtering only if its TXEN1 bit is set. When a FIFO object has transmitted a message, the hardware clears its TXEN1 bit in addition to standard transmit postprocessing (clear TXRQ, transmit interrupt etc.), and moves the CUR pointer in the next FIFO base object to be transmitted. TXEN1 is set automatically (by hardware) in the next message object. Thus, TXEN1 moves along the Transmit FIFO structure as a token that selects the active element.

If bitfield MOFCRn.OVIE (“Overflow Interrupt Enable”) of the FIFO base object is set and the current pointer CUR becomes equal to MOFGPRn.SEL, a FIFO overflow interrupt request is generated. The interrupt request is generated on interrupt node RXINP of the base object after postprocessing of the received frame. Receive interrupts are still generated for the Transmit FIFO base object if bit RXIE is set.

A connection with an object having MMC=5 is not possible.

11.8.10.8 Gateway mode

The Gateway mode makes it possible to establish an automatic information transfer between two independent CAN buses without CPU interaction.

The Gateway mode operates on message object level. In Gateway mode, information is transferred between two message objects, resulting in an information transfer between the two CAN nodes to which the message objects are allocated. A gateway may be established with any pair of CAN nodes, and there can be as many gateways as there are message objects available to build the gateway structure.

The Gateway mode is selected by setting MOFCRs.MMC = 0100_B for the gateway source object s. The gateway destination object d is selected by the MOFGPRs.CUR=d pointer of the source object. The gateway destination object only needs to be valid (its MSGVAL = 1). All other settings are not relevant for the information transfer from the source object to the destination object.

Gateway source object behaves as a standard message object with the difference that some additional actions are performed by the MultiCAN+ when a CAN frame has been received and stored in the source object (see **Figure 160**):

- If bit MOFCRs.DLCC is set, the data length code MOFCRs.DLC is copied from the gateway source object to the gateway destination object
- If bit MOFCRs.IDC is set, the identifier MOARs.ID and the identifier extension MOARs.IDE are copied from the gateway source object to the gateway destination object
- If bit MOFCRs.DATC is set, the data bytes stored in the two data registers MODATALs and MODATAHs are copied from the gateway source object to the gateway destination object. All 8 data bytes are copied, even if MOFCRs.DLC indicates less than 8 data bytes
- If bit MOFCRs.GDFS is set, the transmit request flag MOSTATd.TXRQ is set in the gateway destination object
- The receive pending bit MOSTATd.RXPND and the new data bit MOSTATd.NEWDAT are set in the gateway destination object
- A message interrupt request is generated for the gateway destination object if its MOSTATd.RXIE is set
- The current object pointer MOFGPRs.CUR of the gateway source object is moved to the next destination object according to the FIFO rules as described in **Chapter 11.8.10.5**. A gateway with a single (static) destination object is obtained by setting MOFGPRs.TOP = MOFGPRs.BOT = MOFGPRs.CUR = destination object

The link from the gateway source object to the gateway destination object works in the same way as the link from a FIFO base to a FIFO slave. This means that a gateway with an integrated destination FIFO may be created; in **Figure 159**, the object on the left is the gateway source object and the message object on the right side is the gateway destination objects.

The gateway operates equivalent for the reception of data frames (source object is receive object, i.e. DIR = 0) as well as for the reception of Remote Frames (source object is transmit object).

A connection with an object having MMC=5 is not possible.

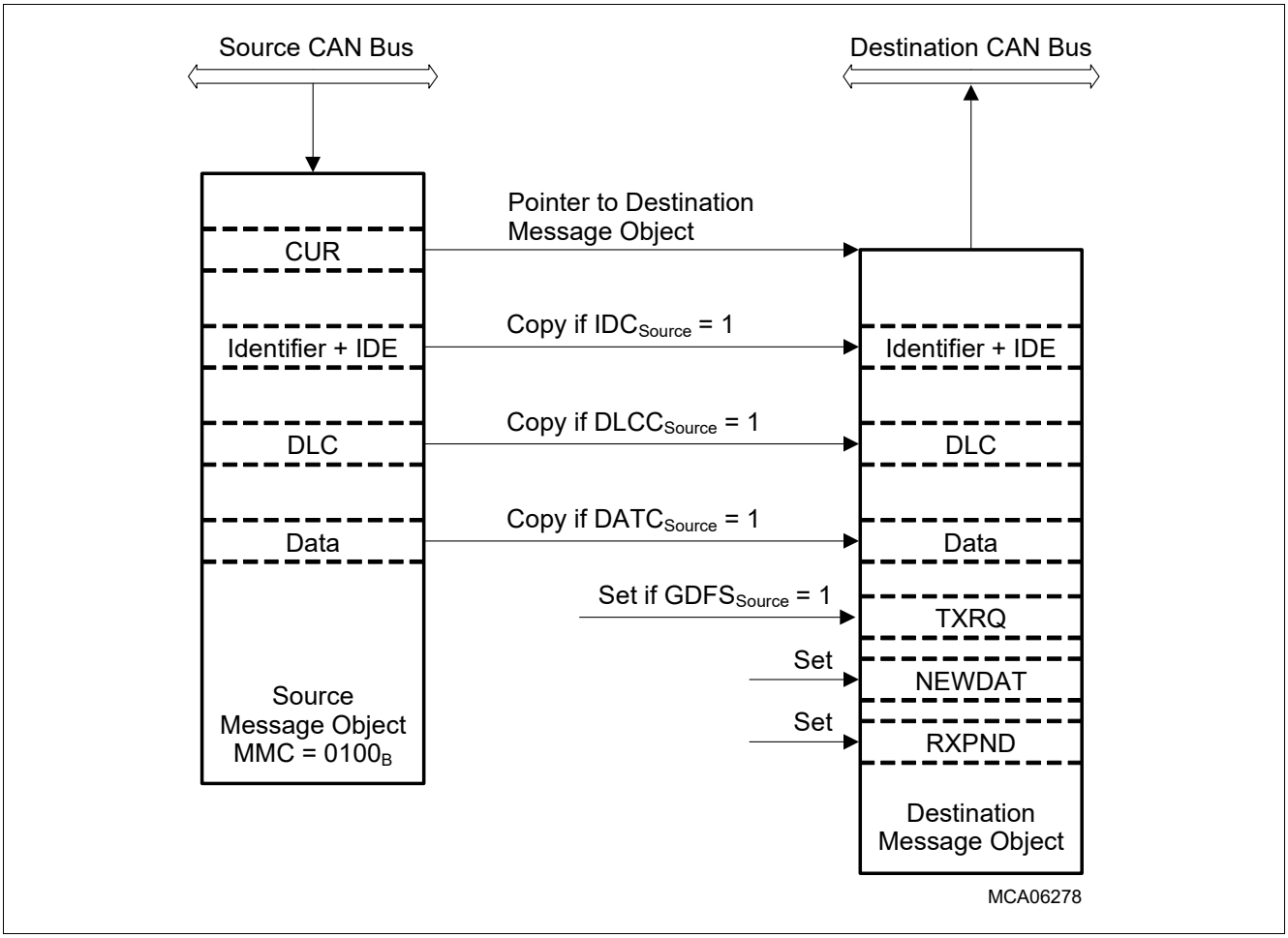


Figure 160 Gateway transfer from source to destination

11.8.10.9 Foreign remote requests

When a remote frame has been received on a CAN node and is stored in a message object, a transmit request is set to trigger the answer (transmission of a data frame) to the request or to automatically issue a secondary request. If the foreign remote request enable bit MOFCRn.FRREN is cleared in the message object in which the remote request is stored, MOSTATn.TXRQ is set in the same message object.

If bit FRREN is set (FRREN = 1: foreign remote request enabled), TXRQ is set in the message object that is referenced by pointer MOFGPRn.CUR. The value of CUR is, however, not changed by this feature.

Although the foreign remote request feature works independently of the selected message mode, it is especially useful for gateways to issue a remote request on the source bus of a gateway after the reception of a remote request on the gateway destination bus. According to the setting of FRREN in the gateway destination object, there are two capabilities to handle remote requests that appear on the destination side (assuming that the source object is a receive object and the destination is a transmit object, i.e. $DIR_{source} = 0$ and $DIR_{destination} = 1$):

FRREN = 0 in the gateway destination object

1. A remote frame is received by gateway destination object
2. TXRQ is set automatically in the gateway destination object
3. A data frame with the current data stored in the destination object is transmitted on the destination bus

FRREN = 1 in the gateway destination object

1. A remote frame is received by gateway destination object
2. TXRQ is set automatically in the gateway source object (must be referenced by CUR pointer of the destination object)
3. A remote request is transmitted by the source object (which is a receive object) on the source CAN bus
4. The receiver of the remote request responds with a data frame on the source bus
5. The data frame is stored in the source object
6. The data frame is copied to the destination object (gateway action)
7. TXRQ is set in the destination object (assuming $GDFS_{source} = 1$)
8. The new data stored in the destination object is transmitted on the destination bus, in response to the initial remote request on the destination bus

11.8.10.10CAN FD - 64-byte messages

In order to support higher than 8 data bytes payload (e.g. 64 bytes data payload) using current message object structure the data byte buffer (MODATALn and MODATAHn) has to be extended.

Thus an additional CAN FD 64-bytes message mode, MOFCR.MMC=5 is added. When CAN FD 64-bytes message mode is selected, additional message objects are used to store the extra data bytes.

The additional message objects used are specified by the pointer on MOFGPR.BOT and MOFGPR.TOP register bits. (i.e. those message object registers (MOFCR, MOFGPR, MOIPR, MOAMR, MODATAL, MODATAH and MOAR) are used to store the extra data bytes with their alternate register view on EMOzDATAn at the same address location. MOCTR is the only register that is not used for data storage it retains its original function.

As an example for a 64-byte message, data bytes 0-7 are stored in the message object, similar to a standard message object, data bytes 8-35 are stored in the message object to which MOFGPR.BOT refers to and data bytes 36-63 are stored in the message object to which MOFGPR.TOP refers to. Data byte 0 refers to the first byte transferred within a CAN frame. For shorter than 64-bytes messages, unused data bytes are padded upon reception and ignored upon transmission.

The additional message objects chosen to store the extra data bytes must not take part in CAN communication i.e. either they are allocated to a list that does not belong to an active CAN node or have that particular message object MOCTR register bits MOCTRn.RXEN, MOCTRn.TXEN0 (and/or TXEN1, TXRQ) cleared, i.e. (RXEN=0, TXEN0=0, TXEN1=0 and TXRQ=0). MOCTR.MSGVAL remains the same, data is stored in the extra message objects when MSGVAL is set. When MSGVAL of the base object is cleared, an ongoing data transfer to TOP and BOT objects are aborted, thereby freeing objects BOT and TOP. Clearing MSGVAL bits of BOT, TOP objects and the extra message objects do not stop an ongoing data transfer and are not considered at all.

For optimum performance it is recommended to place the extra message objects outside active CAN node lists, so they do not take part in CAN acceptance filtering.

11.9 MultiCAN+ kernel registers

This chapter describes the kernel registers of the MultiCAN+ module. All MultiCAN+ kernel register names described in this chapter are also referenced in other parts of the TLE989x/TLE988x User manual, Z8F69685615 by the MultiCAN+ prefix “CAN_”.

The MultiCAN+ kernel include three blocks of registers:

- Global module registers
- Node registers, for each CAN node x
- Message object registers, for each message object n

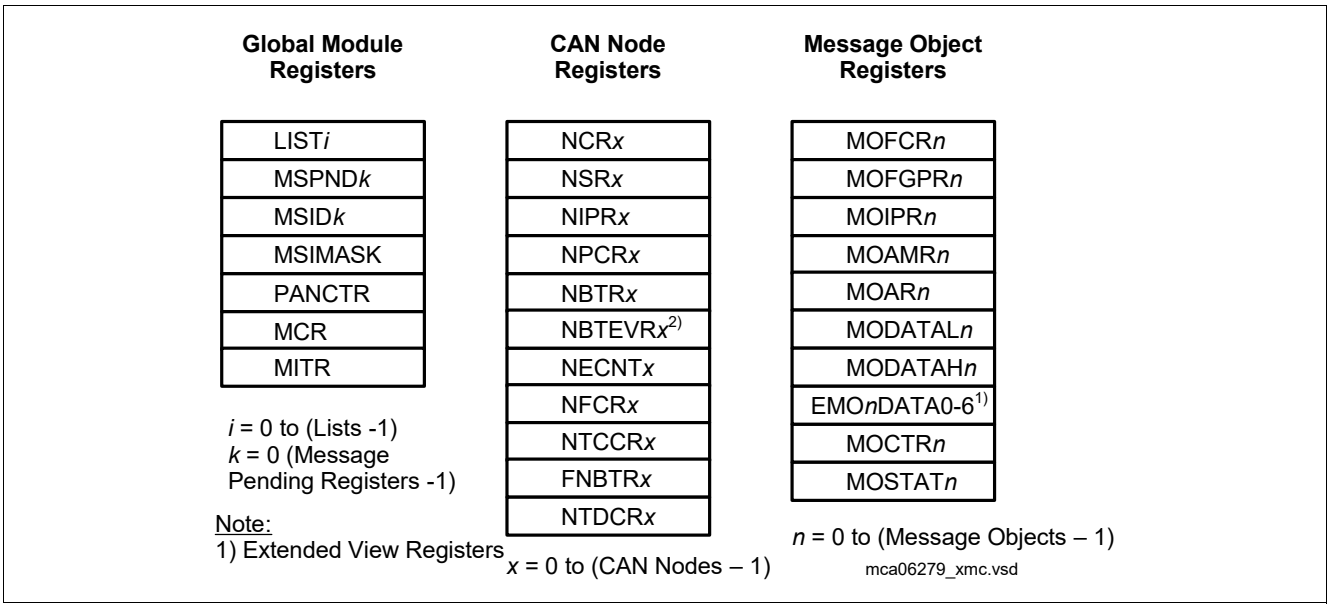


Figure 161 MultiCAN+ kernel registers

Figure 162 shows the MultiCAN+ register address map.

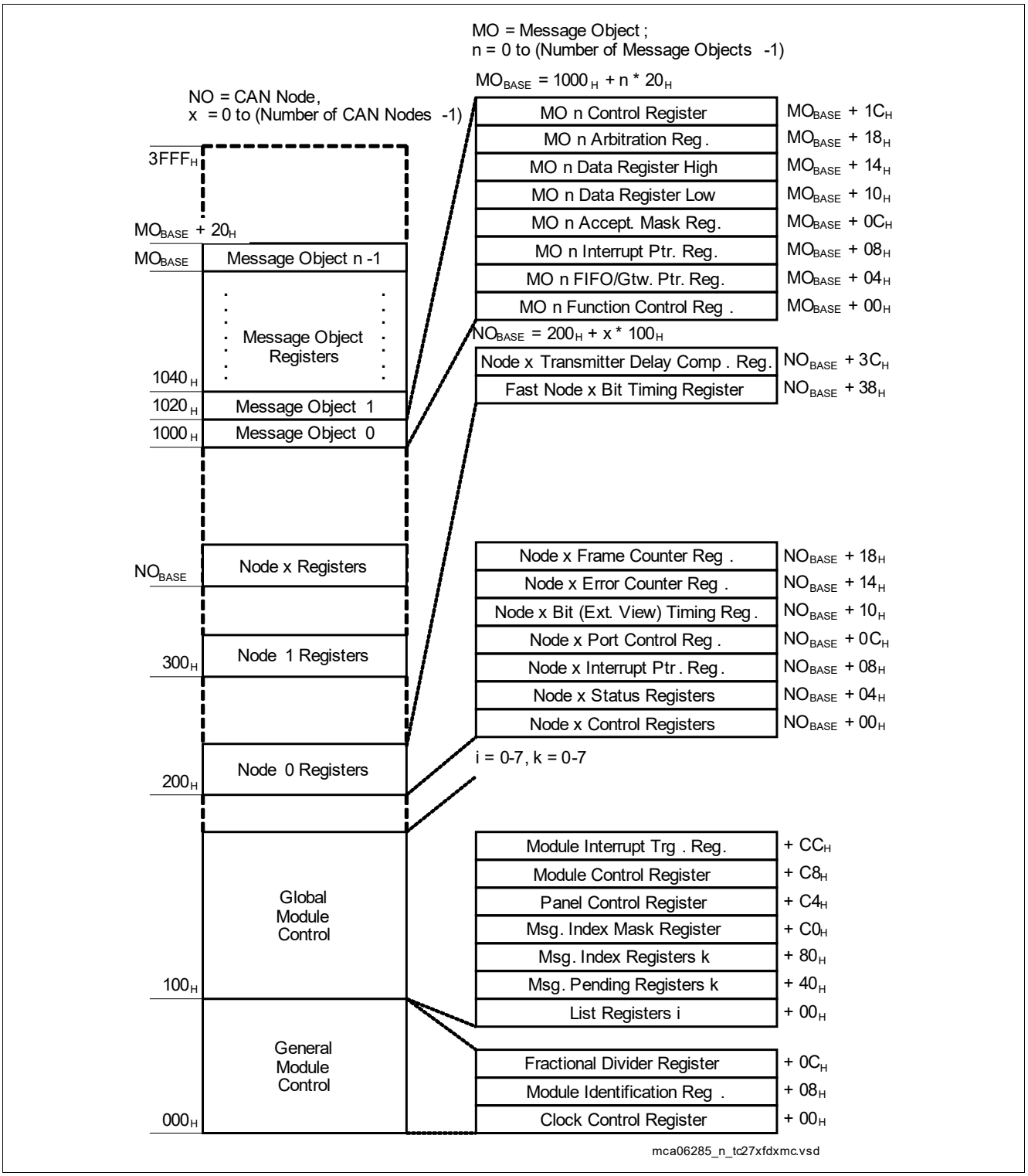


Figure 162 MultiCAN+ register address map XMC with CAN FD

Offset address: The absolute register address is calculated as follows:
Module Base Address (see **Chapter 11.12.1**) + Offset Address (shown in the figure)
Further, the following ranges for parameters i, k, x, and n are valid: i = 0-7, k = 0-1, x = 0-0, n = 0-31.

Access mode: Accesses to empty addresses: nBE.

11.9.1 Global module registers

All list operations such as allocation, de-allocation and relocation of message objects within the list structure are performed via the command panel. It is not possible to modify the list structure directly by software by writing to the message objects and the LIST registers.

Table 131 Register Overview - Global_Module_Registers (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
LISTn	List Register n	0100_H+n*4	552
MSPNDk	Message Pending Register k	0140_H+k*4	552
MSIDk	Message Index Register k	0180_H+k*4	553
MSIMASK	Message Index Mask Register	$01C0_H$	554
PANCTR	Panel Control Register	$01C4_H$	554
MCR	Module Control Register	$01C8_H$	557
MITR	Module Interrupt Trigger Register	$01CC_H$	558

11.9.2 CAN node registers

The CAN node registers are built in for each CAN node of the MultiCAN+. They contain information that is directly related to the operation of the CAN nodes and are shared among the nodes.

Table 132 Register Overview - CAN_Node_Registers (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CAN_NCRx	Node x Control Register	0200_H	559
CAN_NSRx	Node x Status Register	0204_H	562
CAN_NIPRx	Node x Interrupt Pointer Register	0208_H	564
CAN_NPCRx	Node x Port Control Register	$020C_H$	565
CAN_NBTRx	Node x Bit Timing Register	0210_H	566
CAN_NBTEVRx	Node x Bit Timing Extended View Register	0210_H	567
CAN_NECNTx	Node x Error Counter Register	0214_H	568
CAN_NFCRx	Node x Frame Counter Register	0218_H	569
CAN_FNBTRx	Fast Node x Bit Timing Register	0238_H	572
CAN_NTDCRx	Node x Transmitter Delay Compensation Register	$023C_H$	573

11.9.3 Message object registers

Table 133 Register Overview - Message_Object_Registers (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CAN_MOFCRn	Message Object n Function Control Register	1000_H+n*20_H	574
CAN_EMOnDATA0	Extended Message Object n Data 0 Register	1000_H+n*20_H	578
CAN_MOFGPRn	Message Object n FIFO/Gateway Pointer Register	1004_H+n*20_H	578
CAN_EMOnDATA1	Extended Message Object n Data 1 Register	1004_H+n*20_H	579
CAN_MOIPRn	Message Object n Interrupt Pointer Register	1008_H+n*20_H	580
CAN_EMOnDATA2	Extended Message Object n Data 2 Register	1008_H+n*20_H	581
CAN_MOAMRn	Message Object n Acceptance Mask Register	$100C_H$+n*20_H	581
CAN_EMOnDATA3	Extended Message Object n Data 3 Register	$100C_H$+n*20_H	582
CAN_MODATALn	Message Object n Data Register Low	1010_H+n*20_H	583
CAN_EMOnDATA4	Extended Message Object n Data 4 Register	1010_H+n*20_H	583
CAN_MODATAHn	Message Object n Data Register High	1014_H+n*20_H	584
CAN_EMOnDATA5	Extended Message Object n Data 5 Register	1014_H+n*20_H	584
CAN_MOARn	Message Object n Arbitration Register	1018_H+n*20_H	585
CAN_EMOnDATA6	Extended Message Object n Data 6 Register	1018_H+n*20_H	586
CAN_MOCTRn	Message Object n Control Register	$101C_H$+n*20_H	587
CAN_MOSTATn	Message Object n Status Register	$101C_H$+n*20_H	589

11.10 MultiCAN+ interfaces

Figure 163 shows the specific implementation details and interconnections of the MultiCAN+. The I/O lines of the MultiCAN+ are connected to the ports and CAN transceiver. The MultiCAN+ is also supplied by clock control, interrupt control, and address decoding logic. MultiCAN+ interrupts can be directed to the CPU (see **Chapter 11.4**).

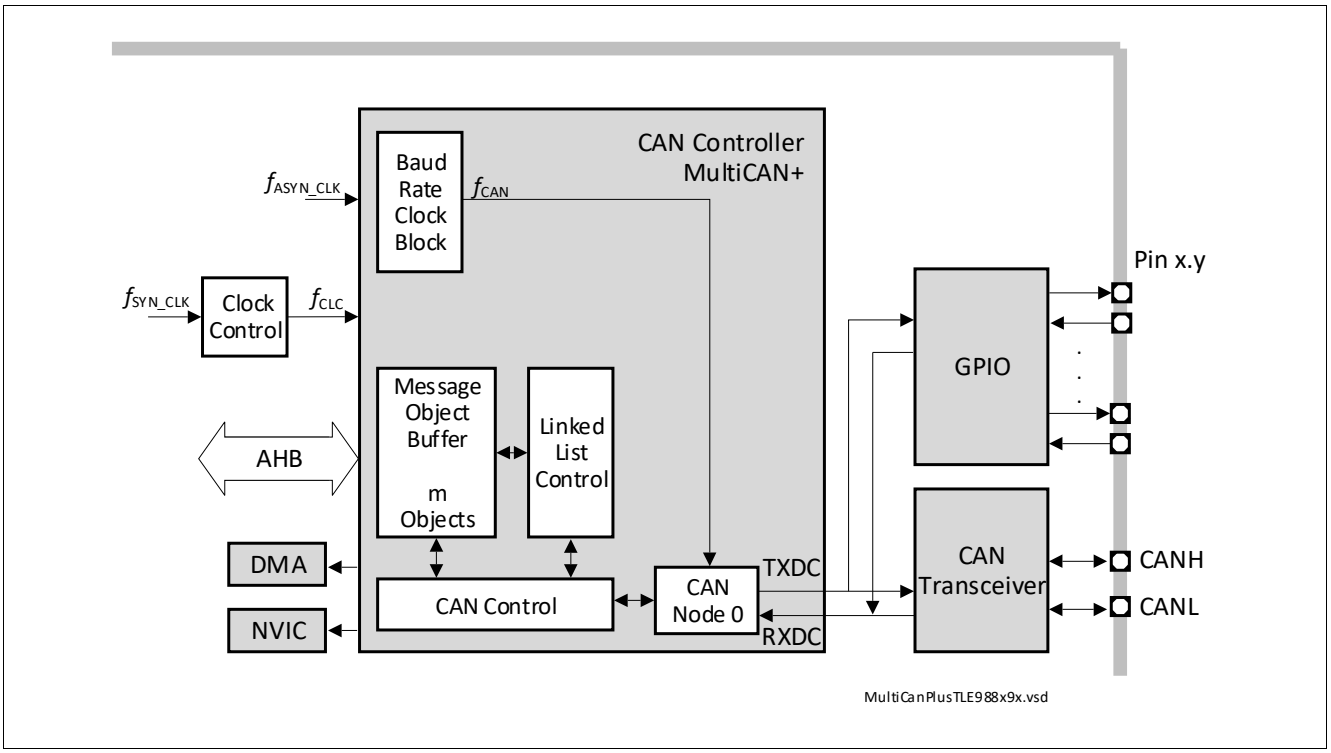


Figure 163 MultiCAN+ implementation and interconnection

11.10.1 Module clock generation

This chapter describes the way the module gets its clock.

11.10.1.1 Clock selection

The bit timing machine and the rest of the MultiCAN+ are separate frequency domains and can be driven by separate independent frequencies. The bit timing unit can be driven by the AHB bus clock or with the direct oscillator clock, and the rest of the chip is driven only by the AHB bus clock.

The purpose of supplying the bit timing unit with a direct oscillator clock is to avoid the clock jitter added by the PLL, necessary when the chip is driven by a low-cost ceramic resonator instead of a high precision quartz crystal.

Selecting the clock source for the bit timing unit is done by programming the bit-field MCR.CLKSEL.

Enabling and disabling the clock of the module by using CLC.DISR affects always both frequency domains, so that when f_{CLC} is switched off, f_A is also switched off.

11.10.1.2 Fractional divider

As shown in **Figure 164**, the clock signals for the MultiCAN+ are generated and controlled by a clock control unit. This clock generation unit is responsible for the enable/disable control, the clock frequency adjustment, and the debug clock control. This unit includes two registers:

- CAN_CLC: generation of the module control clock f_{CLC}

- CAN_FDR: frequency control of the module timer clock f_{CAN}

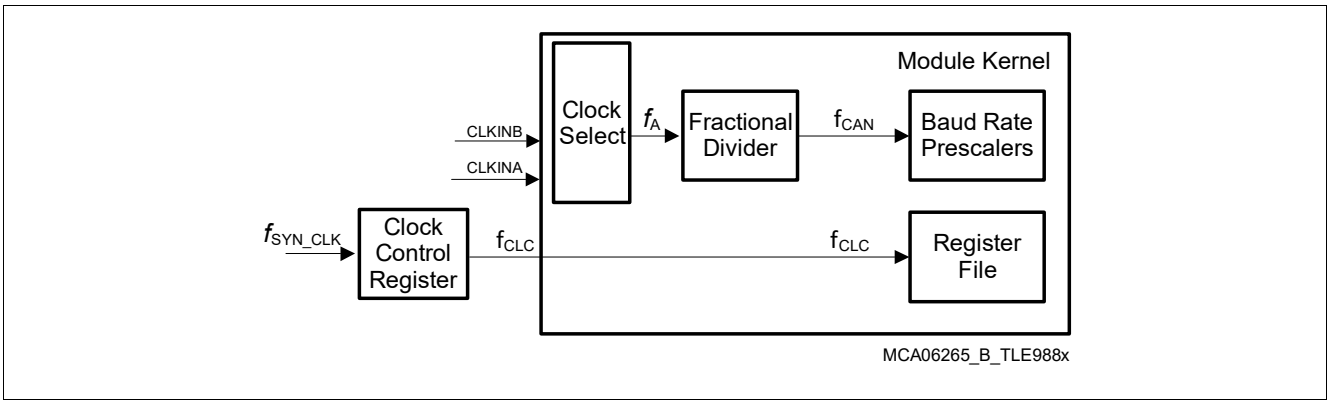


Figure 164 Module clock generation

The f_{SYN_CLK} is identical to f_{PER_CLK}. CLKINx please refer to **Product definitions**, **MultiCAN interconnections**.

The module control clock f_{CLC} is used inside the MultiCAN+ for control purposes such as clocking of control logic and register operations. The frequency of f_{CLC} is identical to the system clock frequency. The clock control register CAN_CLC makes it possible to enable/disable f_{CLC} under certain conditions.

The module timer clock f_{CAN} is used inside the MultiCAN+ as input clock for all timing relevant operations (e.g. bit timing). The settings in the CAN_FDR register determine the frequency of the module timer clock f_{CAN} according the following two formulas:

$$f_{CAN} = f_A \times \frac{1}{n} \quad \text{with n = 1024 - CAN\_FDR.STEP} \tag{11.2}$$

$$f_{CAN} = f_A \times \frac{n}{1024} \quad \text{with n = 0-1023} \tag{11.3}$$

Equation (11.2) applies to normal divider mode (CAN_FDR.DM = 01_B) of the fractional divider. **Equation (11.3)** applies to fractional divider mode (CAN_FDR.DM = 10_B).

Note: The MultiCAN+ is disabled after reset. In general, after reset, the module control clock f_{CLC} must be switched on (writing to register CAN_CLC) before the frequency of the module timer clock f_{CAN} is defined (writing to register CAN_FDR).

11.10.2 Port and I/O line control

The interconnections between the MultiCAN+ and the port I/O lines are controlled in the port logic. Additionally to the port input selection, the following port control operations must be executed:

- Input/output direction control selection (DIR registers)
- Alternate output selection (ALTSELx registers)

11.10.2.1 Input/output function selection in ports

The port input/output control registers contain the bitfields that select the digital output and input driver characteristics such as pull-up/down devices, port direction (input/output), open-drain, and alternate output selections. The I/O lines for the MultiCAN+ are controlled by the port input/output control registers, which are described in the datasheet. In case of discrepancies between datasheet and CAN chapter, the description in the datasheet is correct.

Figure 134 shows the corresponding pins, sorted by node. Even though the table is pair wise, it is possible to select a different pairing for RXD and TXD. In addition the RXSEL value to be programmed for the Receive Pins is part of this table. For more information on RXSEL see **Chapter 11.10.2.2**.

Table 134 MultiCAN+ I/O control selection

Node	RXD	NPCRx.RXSEL
CAN0	RXDCA	000B
	RXDCB	001B
	RXDCC	010B
	RXDCD	011B

11.10.2.2 Node receive input selection

Additionally to the I/O control selection, as defined in the datasheet, the selection of a CAN node's receive input line requires that bitfield RXSEL in its node port control register NPCRx must be set according to. As a hint A results in 0_H, B in 1_H until resulting in the value of 3_H.

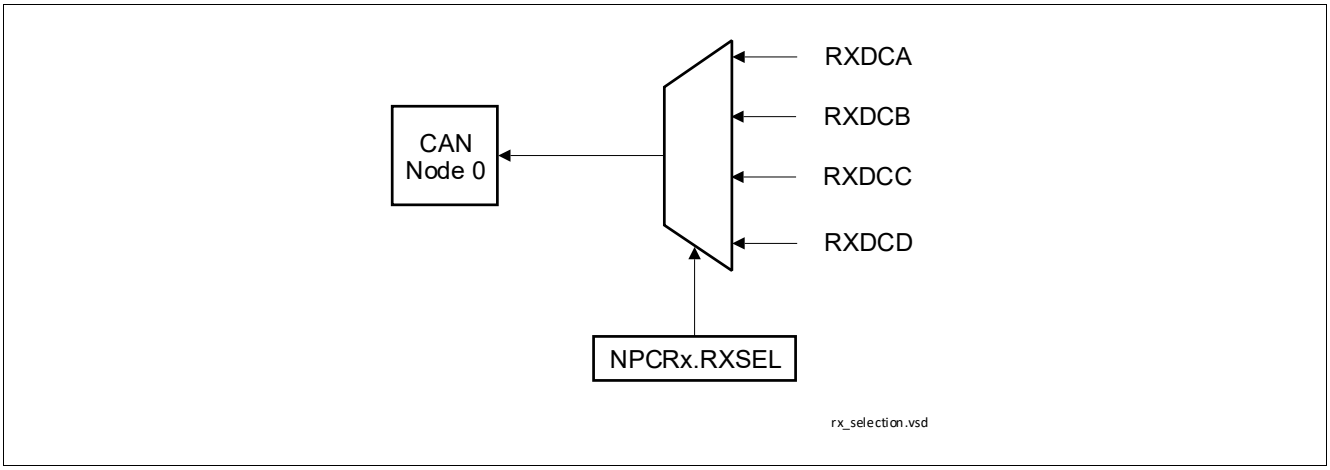


Figure 165 Node receive input selection

11.10.3 MultiCAN+ external registers

The registers listed in **Figure 166** are not included in the MultiCAN+ kernel, some registers must be programmed for proper operation of the MultiCAN+.

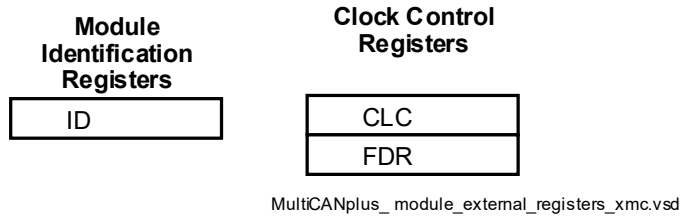


Figure 166 CAN implementation-specific special function registers

Table 135 Register Overview - Module_Identification_Registers (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
ID	Module Identification Register	0008_H	592

Table 136 Register Overview - Clock_Control_Registers (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CLC	CAN Clock Control Register	0000_H	592
FDR	CAN Fractional Divider Register	$000C_H$	593

11.11 Programmer's guide

This chapter shows an initialization example with CAN/CAN-FD transmit and receive objects.

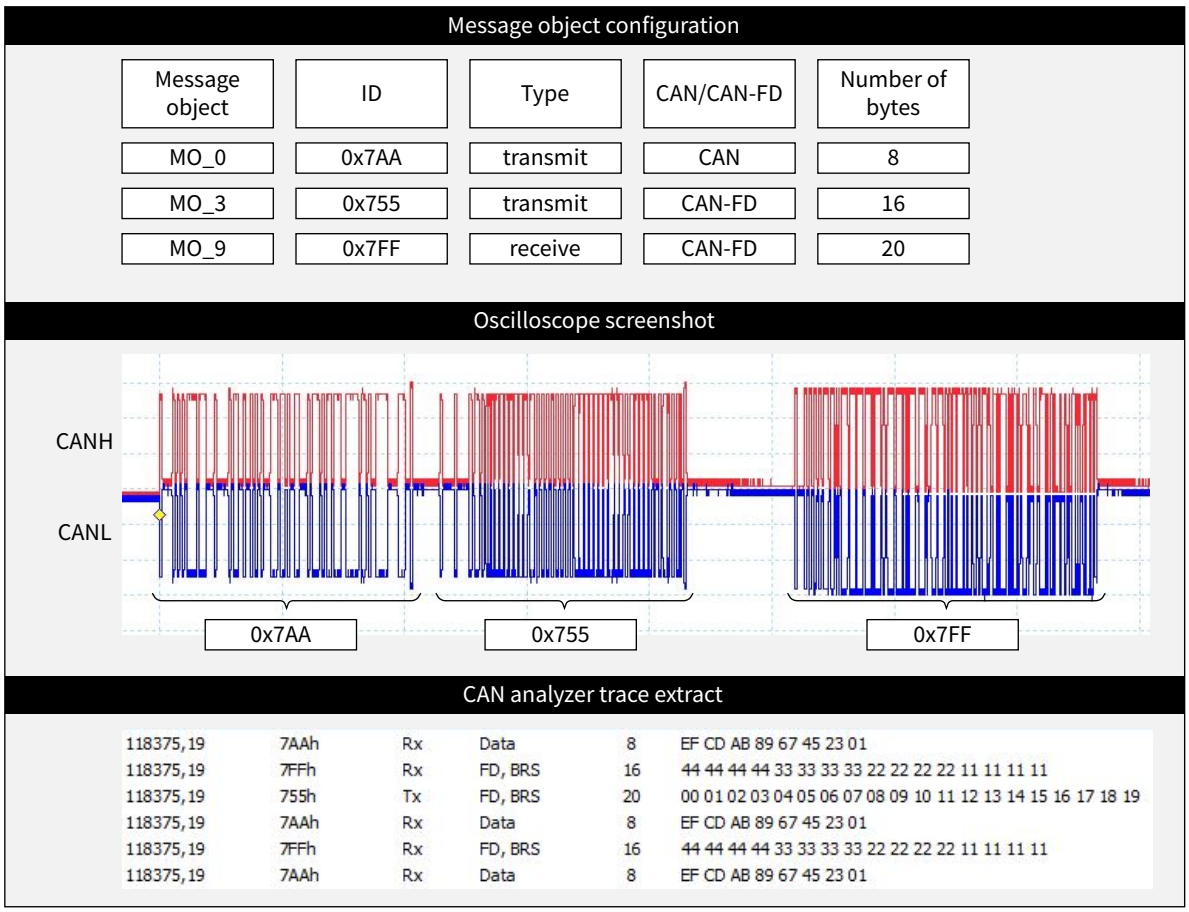


Figure 167 Message object configuration

Prerequisites

- Clock control with PLL1 at 80 MHz as clock source for MultiCAN
- CAN transceiver is enabled

Notes

For debugging purpose, it may be helpful to use the alternate GPIO input/output function:

- SCU.CLKOUT feature for PLL1 and XTAL clock
- MultiCAN.TXD output
- CANTRX.RXD output

11.11.1 Initialization

```
/* Load global MultiCAN+ registers */
CANNODE->CLC.reg=0;  // enable MultiCAN module

/* Configure fractional divider fcan = fa */
CANNODE->FDR.bit.STEP=0x3FF;
CANNODE->FDR.bit.DM=0x01;

/* Select clock for fa = CLKINB (from PLL1) */
CANNODE->MCR.bit.CLKSEL=0x02;  // CLKSEL = 2

/* Init CAN node 0 */
CANNODE->CAN_NCR0.bit.CCE=1;   // CCE  = 1
CANNODE->CAN_NCR0.bit.INIT=1;  // INIT = 1

/* CAN-FD enabled */
CANNODE->CAN_NCR0.bit.FDEN=1;  // CAN classic and CAN-FD is supported

/* MultiCAN input select */
CANNODE->CAN_NPCR0.bit.RXSEL=0x00;  // CANTRX.RXD -> MultiCAN.RXDCA

/* Set normal node bit timing with FDEN = 1 - 1 MBit         */
/* fcan = 80 MHz, tq = 50 ns, TSEG1 = 750 ns, TSEG2 = 200 ns */
CANNODEFD->CAN_NBTEVR0.bit.BRP=0x03;
CANNODEFD->CAN_NBTEVR0.bit.SJW=0x3;
CANNODEFD->CAN_NBTEVR0.bit.TSEG1=0xE;
CANNODEFD->CAN_NBTEVR0.bit.TSEG2=0x3;
CANNODEFD->CAN_NBTEVR0.bit.DIV8=0;

/* Set fast node bit timing with FDEN = 1 - 2 MBit */
CANNODEFD->CAN_FNBTR0.bit.FBRP=0x01;
CANNODEFD->CAN_FNBTR0.bit.FSJW=0x3;
CANNODEFD->CAN_FNBTR0.bit.FTSEG1=0xD;
CANNODEFD->CAN_FNBTR0.bit.FTSEG2=0x4;

/* Allocate message objects to CAN nodes          */
/* Note: list 1 is assigned to node 0 in MulitCAN */

// Wait until PANCTR is not busy
while ((CANNODE->PANCTR.bit.BUSY)||(CANNODE->PANCTR.bit.RBUSY));
// Map MO_0 to list 1, static allocation
CANNODE->PANCTR.bit.PANAR1=0;
CANNODE->PANCTR.bit.PANAR2=1;
CANNODE->PANCTR.bit.PANCMD=0x02;

// Wait until PANCTR is not busy
while ((CANNODE->PANCTR.bit.BUSY)||(CANNODE->PANCTR.bit.RBUSY));
// Map MO_3 (base object) to list 1, static allocation
CANNODE->PANCTR.bit.PANAR1=3;
CANNODE->PANCTR.bit.PANAR2=1;
CANNODE->PANCTR.bit.PANCMD=0x02;
```

```
// Wait until PANCTR is not busy
while ((CANNODE->PANCTR.bit.BUSY)||(CANNODE->PANCTR.bit.RBUSY));
// Map MO_4 (slave object) to list 2, static allocation
CANNODE->PANCTR.bit.PANAR1=4;
CANNODE->PANCTR.bit.PANAR2=2;
CANNODE->PANCTR.bit.PANCMD=0x02;

// Wait until PANCTR is not busy
while ((CANNODE->PANCTR.bit.BUSY)||(CANNODE->PANCTR.bit.RBUSY));
// Map MO_5 (slave object) to list 2, static allocation
CANNODE->PANCTR.bit.PANAR1 = 5;
CANNODE->PANCTR.bit.PANAR2 = 2;
CANNODE->PANCTR.bit.PANCMD = 0x02;

// Wait until PANCTR is not busy
while ((CANNODE->PANCTR.bit.BUSY)||(CANNODE->PANCTR.bit.RBUSY));
// Map MO_9 to list 1, static allocation
CANNODE->PANCTR.bit.PANAR1 = 9;
CANNODE->PANCTR.bit.PANAR2 = 1;
CANNODE->PANCTR.bit.PANCMD = 0x02;

// Wait until PANCTR is not busy
while ((CANNODE->PANCTR.bit.BUSY)||(CANNODE->PANCTR.bit.RBUSY));
// Map MO_10 to list 2, static allocation
CANNODE->PANCTR.bit.PANAR1 = 10;
CANNODE->PANCTR.bit.PANAR2 = 2;
CANNODE->PANCTR.bit.PANCMD = 0x02;

// Wait until PANCTR is not busy
while ((CANNODE->PANCTR.bit.BUSY)||(CANNODE->PANCTR.bit.RBUSY));
// Map MO_11 to list 2, static allocation
CANNODE->PANCTR.bit.PANAR1 = 11;
CANNODE->PANCTR.bit.PANAR2 = 2;
CANNODE->PANCTR.bit.PANCMD = 0x02;

/* Init transmit message object MO_0 - CAN classic & standard - INT_O0 */
CANMSGOBJ0->CAN_MOFCR0.bit.DLC=8;   // length DLC = 8
CANMSGOBJ0->CAN_MOFCR0.bit.TXIE=1;  // transmit interrupt enabled
// SETDIR = 1, SETTXEN1 = 1, SETTXEN0 = 1
CANMSGOBJ0->OBJ0.CAN_MOCTR0.reg=(1<<27)|(1<<26)|(1<<25);
// MO_0 arbitration: PRI = 1 (acc to list), IDE = 0 (11 bit), ID = 0x07AA
CANMSGOBJ0->CAN_MOAR0.reg=((0x00000001<<30)|(0x000007AA<<18));
// MO_0 interrupt: select INT_O0, set MPN bitpos to 0
CANMSGOBJ0->CAN_MOIPR0.bit.MPN=0x0;
CANMSGOBJ0->CAN_MOIPR0.bit.TXINP=0x0;

/* Init transmit message object MO_3 - CAN-FD & standard - INT_O0  */
/* MO_3 baseobj  CUR (d = 0-7)                                     */
/* MO_4 slaveobj BOT (d = 8-35)                                    */
/* MO_5 slaveobj TOP (d = 36-63)                                   */
/* MO_3 function: MMC = CAN - FD, BRS = 1, FDF = 1, DLC = 16 byte  */
CANMSGOBJ0->CAN_MOFCR3.bit.MMC=0x5;
CANMSGOBJ0->CAN_MOFCR3.bit.DLC=0xA;
CANMSGOBJ0->CAN_MOFCR3.bit.FDF=1;
```

```
CANMSGOBJ0->CAN_MOFCR3.bit.BRS=1;
// MO_3 function: Transmit interrupt enabled
CANMSGOBJ0->CAN_MOFCR3.bit.TXIE=1;
// MO_3 FIFO pointer MO_3 -> MO_4 -> MO_5
CANMSGOBJ0->CAN_MOFGPR3.bit.BOT=0x04;  // MO_4
CANMSGOBJ0->CAN_MOFGPR3.bit.CUR=0x03;  // MO_3
CANMSGOBJ0->CAN_MOFGPR3.bit.TOP=0x05;  // MO_5
// MO_3 mode: SETDIR = 1, SETTXEN1 = 1, SETTXEN0 = 1
CANMSGOBJ0->OBJ3.CAN_MOCTR3.reg=(1<<27)|(1<<26)|(1<<25);
// MO_3 arbitration: PRI = 1 (acc to list), IDE = 0 (11 bit), ID = 0x07FF
CANMSGOBJ0->CAN_MOAR3.reg=((0x00000001<<30)|(0x000007FF<<18));
// MO_3 interrupt: select INT_O0, set MPN bitpos to 3
CANMSGOBJ0->CAN_MOIPR3.bit.MPN=0x3;
CANMSGOBJ0->CAN_MOIPR3.bit.TXINP=0x0;

/* Init receive message object MO_9 - CAN-FD & standard - INT_O1       */
/* MO_9 function: CAN-FD long MO, fast br, DLC = 20 bytes, rec int ena */
/* MO_9 baseobj   CUR (d = 0-7)                                        */
/* MO_10 slaveobj BOT (d = 8-35)                                       */
/* MO_11 slaveobj TOP (d = 36-63)                                      */
CANMSGOBJ0->CAN_MOFCR9.bit.MMC=5;
CANMSGOBJ0->CAN_MOFCR9.bit.FDF=1;
CANMSGOBJ0->CAN_MOFCR9.bit.BRS=1;
CANMSGOBJ0->CAN_MOFCR9.bit.DLC=0xB;
CANMSGOBJ0->CAN_MOFCR9.bit.RXIE=1;
// MO_9 FIFO pointer: MO_9 -> MO_10 -> MO_11
CANMSGOBJ0->CAN_MOFGPR9.bit.BOT=10;  // MO_10
CANMSGOBJ0->CAN_MOFGPR9.bit.CUR=9;   // MO_9
CANMSGOBJ0->CAN_MOFGPR9.bit.TOP=11;  // MO_11
// MO_9 mode: SETRXEN = 1, SETMSGVAL = 1
CANMSGOBJ0->OBJ9.CAN_MOCTR9.reg=(1<<23)|(1<<21);
// MO_9 arbitration: PRI = 1 (acc to list), IDE = 0 (11 bit), ID = 0x0755
CANMSGOBJ0->CAN_MOAR9.reg=((0x00000001<<30)|(0x00000755<<18));
// MO_9 interrupt: select INT_O1, set MPN bitpos to 9
CANMSGOBJ0->CAN_MOIPR9.bit.MPN=0x9;
CANMSGOBJ0->CAN_MOIPR9.bit.RXINP=0x1;

/* MultiCAN errors interrupt enable - INT_O2 */
// ALINP = 2, LCINP = 2 -> INT_O2
CANNODE->CAN_NIPR0.reg=(2<<0)|(2<<4);
// ALIE = 1, LCIE = 1
CANNODE->CAN_NCR0.reg|=(1<<2)|(1<<3);

/* NVIC MultiCAN node interrupt enable (transmit) */
CPU->NVIC_ISER.bit.IRQEN26=1;
/* NVIC MultiCAN node interrupt enable (receive) */
CPU->NVIC_ISER.bit.IRQEN27=1;
/* NVIC MultiCAN node interrupt enable (error) */
CPU->NVIC_ISER.bit.IRQEN28=1;

/* Node init finished */
CANNODE->CAN_NCR0.bit.CCE=0;  // CCE  = 0
CANNODE->CAN_NCR0.bit.INIT=0; // INIT = 0
```

11.11.2 Main loop

Set transmit request for MO_0 and MO_3

```
/* Set transmit request for MO_0 */
if (!(CANMSGOBJ0->OBJ0.CAN_MOSTAT0.reg&0x02u))  // check TXPND
{
    // Load transmit data for MO_0
    CANMSGOBJ0->CAN_MODATAL0.reg=0x01234567;
    CANMSGOBJ0->CAN_MODATAH0.reg=0x89ABCDEF;
    // MO_0 mode: SETTXRQ = 1, SETMSGVAL = 1, SETNEWDAT = 1
    CANMSGOBJ0->OBJ0.CAN_MOCTR0.reg=(1<<24)|(1<<21)|(1<<19);
}
/* Set transmit request for MO_3 */
if (!(CANMSGOBJ0->OBJ3.CAN_MOSTAT3.reg&0x02u))  // check TXPND
{
    // Load transmit data for MO_3
    // MO_3 is slaveobj with data2/3 in EMO4DATA0/1
    CANMSGOBJ0->CAN_MODATAL3.reg=0x11111111;
    CANMSGOBJ0->CAN_MODATAH3.reg=0x22222222;
    CANMSGOBJ1->CAN_EMO4DATA0.reg=0x33333333;
    CANMSGOBJ1->CAN_EMO4DATA1.reg=0x44444444;
    // MO_3 mode: SETTXRQ = 1, SETMSGVAL = 1, SETNEWDAT = 1
    CANMSGOBJ0->OBJ3.CAN_MOCTR3.reg=(1<<24)|(1<<21)|(1<<19);
}
```

Transmit interrupt service routine

```
void NVIC_IRQ26_Handler(void)
{
    /* Check if MO_0 pending */
    if (CANNODE->MSPND0.reg&0x01u)
    {
        CANNODE->MSPND0.reg=~(0x01u);  // clr MO_0 msg pending bit
        // clr MO_0 TXPND - TXPND indicates a successful transmitted msgobj
        CANMSGOBJ0->OBJ0.CAN_MOCTR0.reg=0x02u;
    }
    /* Check if MO_3 pending */
    if (CANNODE->MSPND0.reg&0x08u)
    {
        CANNODE->MSPND0.reg=~(0x08u);  // clr MO_3 msg pending bit
        // clr MO_3 TXPND - TXPND indicates a successful transmitted msgobj
        CANMSGOBJ0->OBJ3.CAN_MOCTR3.reg=0x02u;
    }
}
```

Receive interrupt service routine

```
void NVIC_IRQ27_Handler(void)
{
    /* Check which MO has been received */
    if (CANNODE->MSPND0.reg&0x200u)  // check if MO_9 pending
    {
        CANNODE->MSPND0.reg=~(0x200u);  // clr MO_9 msg pending bit
        // reset MO_9 RXPND and NEWDAT
        CANMSGOBJ0->OBJ9.CAN_MOCTR9.reg=(1<<3)|(1<<0);
        // read receive data of MO_9
        MO_9_data0=CANMSGOBJ0->CAN_MODATAL9.reg;
        MO_9_data1=CANMSGOBJ0->CAN_MODATAH9.reg;
        MO_9_data2=CANMSGOBJ1->CAN_EMO10DATA0.reg;
        MO_9_data3=CANMSGOBJ1->CAN_EMO10DATA1.reg;
        MO_9_data4=CANMSGOBJ1->CAN_EMO10DATA2.reg;

        /* Check for data consistency */
        if ((CANMSGOBJ0->OBJ9.CAN_MOSTAT9.bit.MSGLST==1)|
          (CANMSGOBJ0->OBJ9.CAN_MOSTAT9.bit.NEWDAT==1)
        {
            // data is lost - MO has been overwritten by new msg but not read
            // reset MO_ MSGLST
            CANMSGOBJ0->OBJ9.CAN_MOCTR9.reg=(1<<4);
            // do message lost handling accordingly
        }
    }
}
```

Error interrupt service routine

```
void NVIC_IRQ28_Handler(void)
{
    /* Check errors */
    if ((CANNODE->CAN_NSR0.bit.LEC)!=0)  // LEC
    {
        // do something
    }
    if ((CANNODE->CAN_NSR0.bit.ALERT)==1)  // ALERT
    {
        // do something
        CANNODE->CAN_NSR0.bit.ALERT=0;  // clear ALERT
    }
    if ((CANNODE->CAN_NSR0.bit.EWRN)==1)  // EWRN
    {
        // do something
    }
    if ((CANNODE->CAN_NSR0.bit.BOFF)==1)  // BOFF
    {
        // do something
    }
    if ((CANNODE->CAN_NSR0.bit.LLE)==1)  // LLE
    {
        // do something
```

```
    }
    if ((CANNODE->CAN_NSR0.bit.LOE)==1)  // LOE
    {
        // do something
    }
    if ((CANNODE->CAN_NSR0.bit.RESI)==1)  // RESI
    {
        // do something
    }
    if ((CANNODE->CAN_NSR0.bit.FLEC)!=1)  // FLEC
    {
        // do something
    }
}
```

11.12 Register description MultiCAN+

11.12.1 MultiCAN+ address maps

Table 137 Register Address Space - CANCONTR

Module	Base Address	End Address	Note
MULTICAN	48010000_H	$48013FFF_H$	

Table 138 Register Overview - CANCONTR (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CLC	CAN Clock Control Register	0000_H	592
ID	Module Identification Register	0008_H	592
FDR	CAN Fractional Divider Register	$000C_H$	593
LISTn	List Register n	0100_H+n*4	552
MSPNDk	Message Pending Register k	0140_H+k*4	552
MSIDk	Message Index Register k	0180_H+k*4	553
MSIMASK	Message Index Mask Register	$01C0_H$	554
PANCTR	Panel Control Register	$01C4_H$	554
MCR	Module Control Register	$01C8_H$	557
MITR	Module Interrupt Trigger Register	$01CC_H$	558
CAN_NCRx	Node x Control Register	0200_H	559
CAN_NSRx	Node x Status Register	0204_H	562
CAN_NIPRx	Node x Interrupt Pointer Register	0208_H	564
CAN_NPCRx	Node x Port Control Register	$020C_H$	565
CAN_NBTRx	Node x Bit Timing Register	0210_H	566
CAN_NBTEVRx	Node x Bit Timing Extended View Register	0210_H	567
CAN_NECNTx	Node x Error Counter Register	0214_H	568
CAN_NFCRx	Node x Frame Counter Register	0218_H	569
CAN_FNBTRx	Fast Node x Bit Timing Register	0238_H	572
CAN_NTDCRx	Node x Transmitter Delay Compensation Register	$023C_H$	573
CAN_MOFCRn	Message Object n Function Control Register	1000_H+n*20_H	574
CAN_EMOnDATA0	Extended Message Object n Data 0 Register	1000_H+n*20_H	578
CAN_MOFGPRn	Message Object n FIFO/Gateway Pointer Register	1004_H+n*20_H	578
CAN_EMOnDATA1	Extended Message Object n Data 1 Register	1004_H+n*20_H	579
CAN_MOIPRn	Message Object n Interrupt Pointer Register	1008_H+n*20_H	580
CAN_EMOnDATA2	Extended Message Object n Data 2 Register	1008_H+n*20_H	581
CAN_MOAMRn	Message Object n Acceptance Mask Register	$100C_H$+n*20_H	581

Table 138 Register Overview - CANCONTR (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Page Number
CAN_EMOnDATA3	Extended Message Object n Data 3 Register	$100C_H$+n*20_H	582
CAN_MODATALn	Message Object n Data Register Low	1010_H+n*20_H	583
CAN_EMOnDATA4	Extended Message Object n Data 4 Register	1010_H+n*20_H	583
CAN_MODATAHn	Message Object n Data Register High	1014_H+n*20_H	584
CAN_EMOnDATA5	Extended Message Object n Data 5 Register	1014_H+n*20_H	584
CAN_MOARn	Message Object n Arbitration Register	1018_H+n*20_H	585
CAN_EMOnDATA6	Extended Message Object n Data 6 Register	1018_H+n*20_H	586
CAN_MOCTRn	Message Object n Control Register	$101C_H$+n*20_H	587
CAN_MOSTATn	Message Object n Status Register	$101C_H$+n*20_H	589

The MultiCAN+ register address map of **Figure 162** also shows the general implementation-specific registers for clock control, module identification, interrupt service request control and the absolute address information.

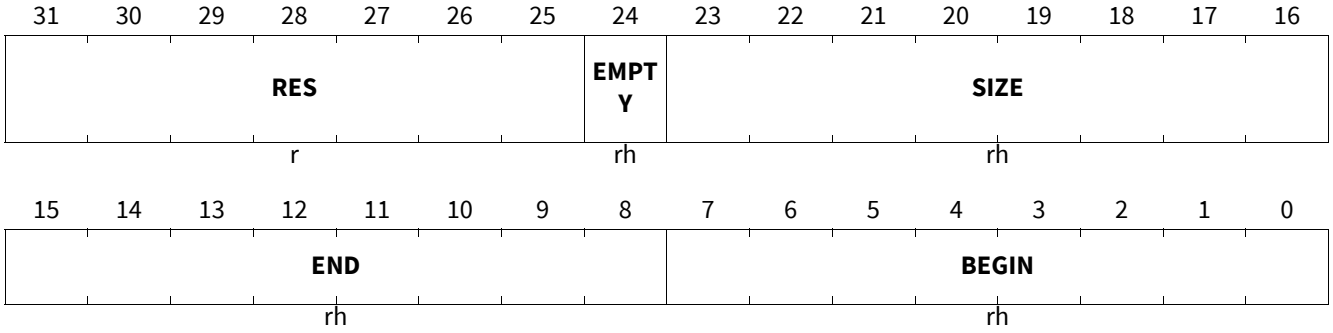
11.12.2 CANCONTR Registers

List Register n

Each CAN node has a list that determines the allocated message objects. Additionally, a list of all unallocated objects is available. Furthermore, general purpose lists are available which are not associated to a CAN node. The List Registers are assigned in the following way:

- LIST0 provides the list of all unallocated objects
- LIST1 provides the list for CAN node 0

LISTn (n=0-7)
List Register n **(0100_H+n*4)** **Reset Value: Table 139**



Field	Bits	Type	Description
BEGIN	7:0	rh	**List Begin** BEGIN indicates the number of the first message object in list i.
END	15:8	rh	**List End** END indicates the number of the last message object in list i.
SIZE	23:16	rh	**List Size** SIZE indicates the number of elements in the list i. SIZE = number of list elements - 1 SIZE = 0 indicates that list i is empty.
EMPTY	24	rh	**List Empty Indication** 0_B At least one message object is allocated to list i. 1_B No message object is allocated to the list i. List i is empty.
RES	31:25	r	**Reserved** Read as 0.

Table 139 Reset Values of LISTn (n=0-7)

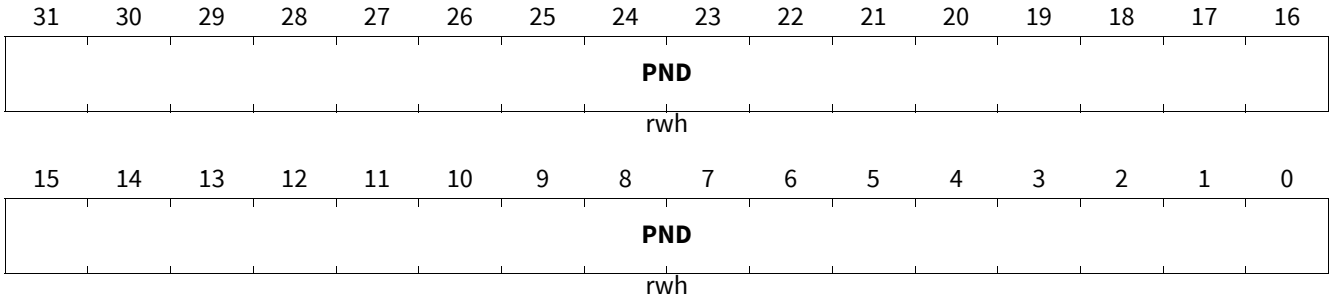
Reset Type	Reset Value	Note
RESET_TYPE_5	$0100\ 0000_H$	LISTn (n>0)
RESET_TYPE_5	$001F\ 1F00_H$	LIST0

Message Pending Register k

The Message Pending Register MSPNDk contains the pending interrupt notification of list i.

MSPNDk (k=0-1)

Message Pending Register k **(0140_H+k*4)** **RESET_TYPE_5 Value: 0000 0000_H**



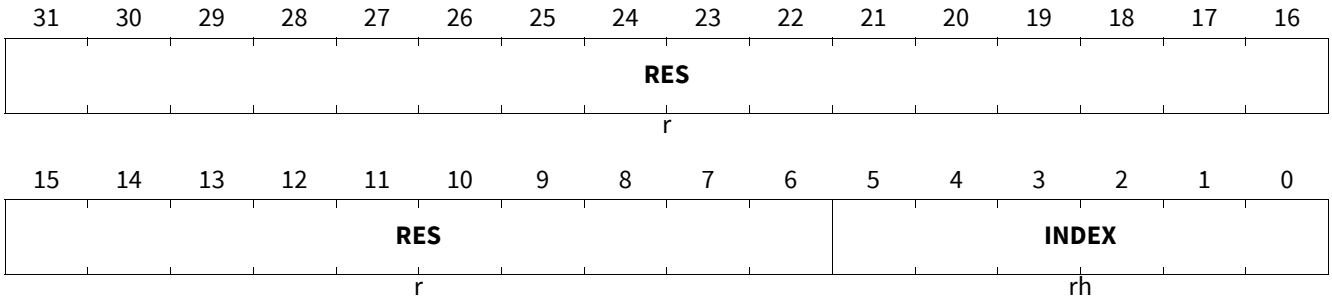
Field	Bits	Type	Description
PND	31:0	rwh	**Message Pending** When a message interrupt occurs, the message object sets a bit in one of the MSPND register, where the bit position is given by the MPN[4:0] field of the IPR register of the message object. The register selection n is given by the higher bits of MPN. The register bits can be cleared by software (write 0). Writing a 1 has no effect.

Message Index Register k

Each Message Pending Register has a Message Index Register MSIDk associated with it. The Message Index Register shows the active (set) pending bit with lowest bit position within groups of pending bits.

MSIDk (k=0-1)

Message Index Register k **(0180_H+k*4)** **RESET_TYPE_5 Value: 0000 0020_H**



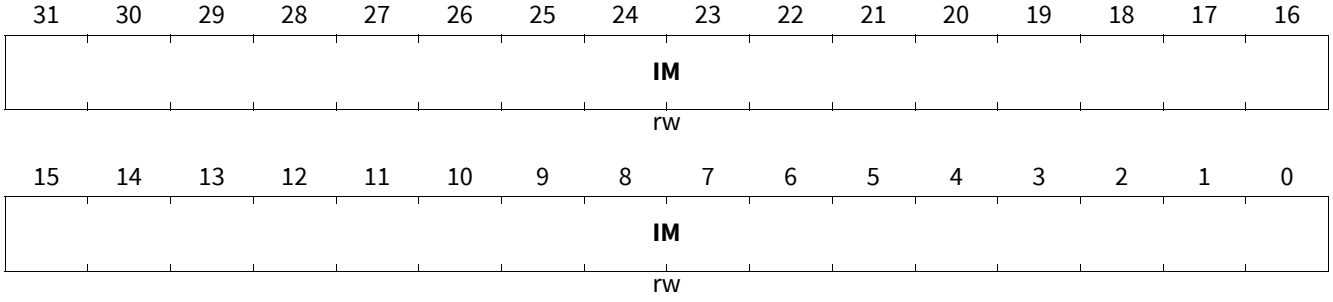
Field	Bits	Type	Description
INDEX	5:0	rh	**Message Pending Index** The value of INDEX is given by the bit position i of the pending bit of MSPNDk with the following properties: 1. MSPNDk[i] & IM[i] = 1 2. i = 0 or MSPNDk[i-1:0] & IM[i-1:0] = 0 If no bit of MSPNDk satisfies these conditions then INDEX reads 100000_B. Thus INDEX shows the position of the first pending bit of MSPNDk, in which only those bits of MSPNDk that are selected in the Message Index Mask Register are taken into account.

Field	Bits	Type	Description
RES	31:6	r	**Reserved** Read as 0; should be written with 0.

Message Index Mask Register

The Message Index Mask Register MSIMASK selects individual bits for the calculation of the Message Pending Index. The Message Index Mask Register is used commonly for all Message Pending registers and their associated Message Index registers.

MSIMASK
Message Index Mask Register **($01C0_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

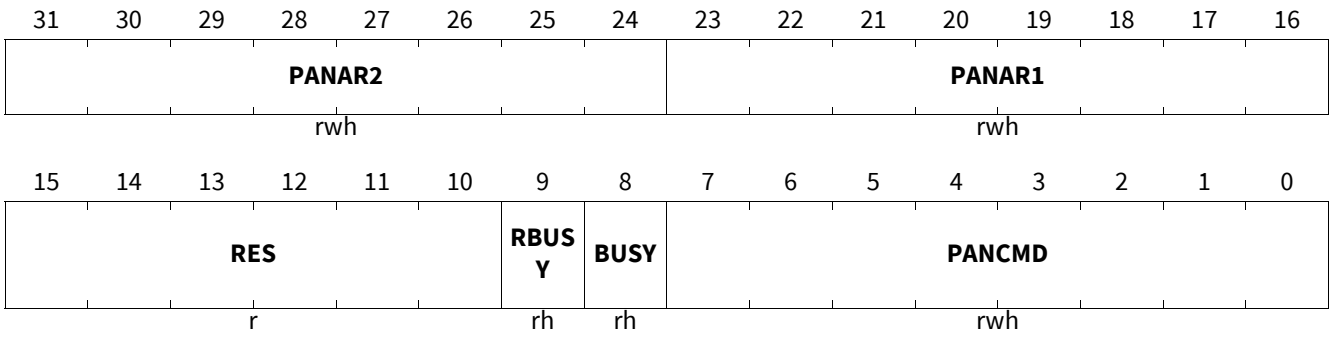


Field	Bits	Type	Description
IM	31:0	rw	**Message Index Mask** Only those bits in MSPNDk for which the corresponding Index Mask bits are set contribute to the calculation of the Message Index.

Panel Control Register

The Panel Control Register PANCTR is used to start a new command by writing the command arguments and the command code into its bit fields.

PANCTR
Panel Control Register **($01C4_H$)** **RESET_TYPE_5 Value: $0000\ 0301_H$**



Field	Bits	Type	Description
PANCMD	7:0	rwh	**Panel Command** This bit field is used to start a new command by writing a panel command code into it. At the end of a panel command, the NOP (no operation) command code is automatically written into PANCMD. The coding of PANCMD is defined in **Table 140**.
BUSY	8	rh	**Panel Busy Flag** Initial list controller initialization must be finalized, when INIT bit is reset. 0_B Panel has finished command and is ready to accept a new command. 1_B Panel operation is in progress.
RBUSY	9	rh	**Result Busy Flag** 0_B No update of PANAR1 and PANAR2 is scheduled by the list controller. 1_B A list command is running (BUSY = 1) that will write results to PANAR1 and PANAR2, but the results are not yet available.
RES	15:10	r	**Reserved** Read as 0; should be written with 0.
PANAR1	23:16	rwh	**Panel Argument 1** See **Table 140**.
PANAR2	31:24	rwh	**Panel Argument 2** See **Table 140**.

Panel Commands

A panel operation consists of a command code (PANCMD) and up to two panel arguments (PANAR1, PANAR2). Commands that have a return value deliver it to the PANAR1 bit field. Commands that return an error flag deliver it to bit 31 of the Panel Control Register, this means bit 7 of PANAR2.

Table 140 Panel Commands

PANCMD	PANAR2	PANAR1	Command Description
00_H	–	–	**No Operation** Writing 00_H to PANCMD has no effect. No new command is started.
01_H	**Result:** Bit 7: ERR Bit 6-0: undefined	–	**Initialize Lists** Run the initialization sequence to reset the CTRL and LIST fields of all message objects. List registers LIST[7:0] are set to their reset values. This results in the de-allocation of all message objects. The initialization command requires that bits NCRx.INIT and NCRx.CCE are set for all CAN nodes. Bit 7 of PANAR2 (ERR) reports the success of the operation: 0_B Initialization was successful 1_B Not all NCRx.INIT and NCRx.CCE bits are set. Therefore, no initialization is performed. The initialize lists command is automatically performed with each reset of the module, but with the exception that all message object registers are reset, too.
02_H	**Argument:** List Index	**Argument:** Message Object Number	**Static Allocate** Allocate message object to a list. The message object is removed from the list that it currently belongs to, and appended to the end of the list, given by PANAR2. This command is also used to deallocate a message object. In this case, the target list is the list of unallocated elements (PANAR2 = 0).
03_H	**Argument:** List Index **Result:** Bit 7: ERR Bit 6-0: undefined	**Result:** Message Object Number	**Dynamic Allocate** Allocate the first message object of the list of unallocated objects to the selected list. The message object is appended to the end of the list. The message number of the message object is returned in PANAR1. An ERR bit (bit 7 of PANAR2) reports the success of the operation: 0_B Success. 1_B The operation has not been performed because the list of unallocated elements was empty.

Table 140 Panel Commands (cont'd)

PANCMD	PANAR2	PANAR1	Command Description
04_H	**Argument:** Destination Object Number	**Argument:** Source Object Number	**Static Insert Before** Remove a message object (source object) from the list that it currently belongs to, and insert it before a given destination object into the list structure of the destination object. The source object thus becomes the predecessor of the destination object.
05_H	**Argument:** Destination Object Number **Result:** Bit 7: ERR Bit 6-0: undefined	**Result:** Object Number of inserted object	**Dynamic Insert Before** Insert a new message object before a given destination object. The new object is taken from the list of unallocated elements (the first element is chosen). The number of the new object is delivered as a result to PANAR1. An ERR bit (bit 7 of PANAR2) reports the success of the operation: 0_B Success. 1_B The operation has not been performed because the list of unallocated elements was empty.
06_H	**Argument:** Destination Object Number	**Argument:** Source Object Number	**Static Insert Behind** Remove a message object (source object) from the list that it currently belongs to, and insert it behind a given destination object into the list structure of the destination object. The source object thus becomes the successor of the destination object.
07_H	**Argument:** Destination Object Number **Result:** Bit 7: ERR Bit 6-0: undefined	**Result:** Object Number of inserted object	**Dynamic Insert Behind** Insert a new message object behind a given destination object. The new object is taken from the list of unallocated elements (the first element is chosen). The number of the new object is delivered as result to PANAR1. An ERR bit (bit 7 of PANAR2) reports the success of the operation: 0_B Success. 1_B The operation has not been performed because the list of unallocated elements was empty.
08_H - FF_H	–	–	**Reserved**

Module Control Register

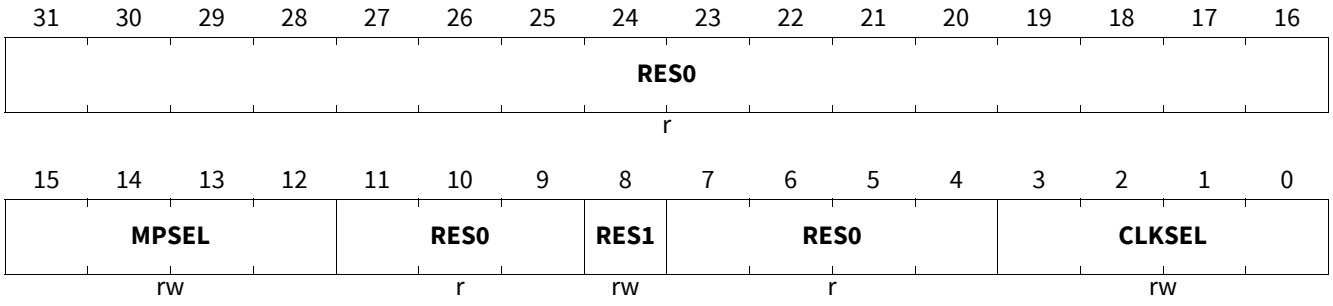
The Module Control Register MCR contains basic settings that determine the operation of the module.

The write access to the lowest byte of the MCR register is possible only if the CCE bits of all CAN nodes are set (NCRx.CCE bits). The NCRx.INIT bits will be automatically set when the lowest byte of the MCR register is written, independent of the setting of the CCE bits. The INIT bits have to be reset by software in order to activate the CAN nodes.

The reconfiguration of the clock source has to be done by using two writes: first a write of zero to the CLKSEL bit field, and then a second write defining the new clock source. Between the first and the second write a delay of 4 / f_A + 2 / f_{CAN} number of cycles must be inserted by software, where f_A is the frequency being switched off with the first write. Exception: in case that is selected as the baud rate logic clock (MCR.CLKSEL = 1), no delay cycles between the writes are necessary. In both cases, simply using one write defining the new clock source is not allowed.

Note: If the baud rate logic is supplied from an unstable clock source, or no clock at all, the CAN functionality is not guaranteed.

MCR
Module Control Register **($01C8_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CLKSEL	3:0	rw	**Baud Rate Logic Clock Select** 0_H no clock selected 1_H CLKINA selected 2_H CLKINB selected 3_H CLKINC selected 4_H not allowed - no_clock … F_H not allowed - no_clock
RES0	7:4, 11:9, 31:16	r	**Reserved** Read as 0; should be written with 0.
RES1	8	rw	**Reserved** Read as 0; should be written with 0.
MPSEL	15:12	rw	**Message Pending Selector** Bit field MPSEL makes it possible to select the bit position of the message pending bit after a message reception/transmission by a mixture of the MOIPRn register bit fields RXINP, TXINP, and MPN. Selection details are given in **Figure 156** on **Page 520**.

Module Interrupt Trigger Register

The Interrupt Trigger Register ITR is used to trigger interrupt requests on each interrupt output line by software.

MITR

Module Interrupt Trigger Register **($01CC_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								IT							
r								w							

Field	Bits	Type	Description
IT	7:0	w	**Interrupt Trigger** Writing a 1 to IT[m] (m = 0-2) generates an interrupt request on interrupt output line INT_O[m]. Writing a 0 to IT[m] has no effect. Bit field IT is always read as 0. Multiple interrupt requests can be generated with a single write operation to MITR by writing a 1 to several bit positions of IT. All 16 interrupts are existing, even if the interrupt request unit is not connected.
RES	31:8	r	**Reserved** Read as 0; should be written with 0.

Node 0 Control Register

The Node Control Register contains basic settings that determine the operation of the CAN node.

CAN_NCR0

Node 0 Control Register **(0200_H)** **RESET_TYPE_5 Value: $0000\ 0041_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES1															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NISO	RES1				PED	FDEN	RES0	CALM	CCE	TXDIS	CANDIS	ALIE	LECIE	TRIE	INIT
rw	r				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rwh

Field	Bits	Type	Description
INIT	0	rwh	**Node Initialization** Bit INIT is automatically set when the CAN node enters the bus-off state (see **Page 508**). 0_B Resetting bit INIT enables the participation of the node in the CAN traffic. If the CAN node is in the bus-off state, the ongoing bus-off recovery (which does not depend on the INIT bit) is continued. With the end of the bus-off recovery sequence the CAN node is allowed to take part in the CAN traffic. If the CAN node is not in the bus-off state, a sequence of 11 consecutive recessive bits must be detected before the node is allowed to take part in the CAN traffic. 1_B Setting this bit terminates the participation of this node in the CAN traffic. Any ongoing frame transfer is cancelled and the transmit line goes recessive. If the CAN node is in the bus-off state, then the running bus-off recovery sequence is continued. If the INIT bit is still set after the successful completion of the bus-off recovery sequence, i.e. after detecting 128 sequences of 11 consecutive recessive bits (11 × 1), then the CAN node leaves the bus-off state but remains inactive as long as INIT remains set.
TRIE	1	rw	**Transfer Interrupt Enable** TRIE enables the transfer interrupt of CAN node x. This interrupt is generated after the successful reception or transmission of a CAN frame in node x. Bit field NIPRx.TRINP selects the interrupt output line which becomes activated at this type of interrupt. 0_B Transfer interrupt is disabled. 1_B Transfer interrupt is enabled.
LECIE	2	rw	**LEC Indicated Error Interrupt Enable** LECIE enables the last error code interrupt of CAN node x. This interrupt is generated with each hardware update of bit field NSRx.LEC with LEC > 0 (CAN protocol error). Bit field NIPRx.LECINP selects the interrupt output line which becomes activated at this type of interrupt. 0_B Last error code interrupt is disabled. 1_B Last error code interrupt is enabled.
ALIE	3	rw	**Alert Interrupt Enable** ALIE enables the alert interrupt of CAN node x. This interrupt is generated by any one of the following events: • A change of bit NSRx.BOFF • A change of bit NSRx.EWRN • A List Length Error, which also sets bit NSRx.LLE • A List Object Error, which also sets bit NSRx.LOE Bit field NIPRx.ALINP selects the interrupt output line which becomes activated at this type of interrupt. 0_B Alert interrupt is disabled. 1_B Alert interrupt is enabled.

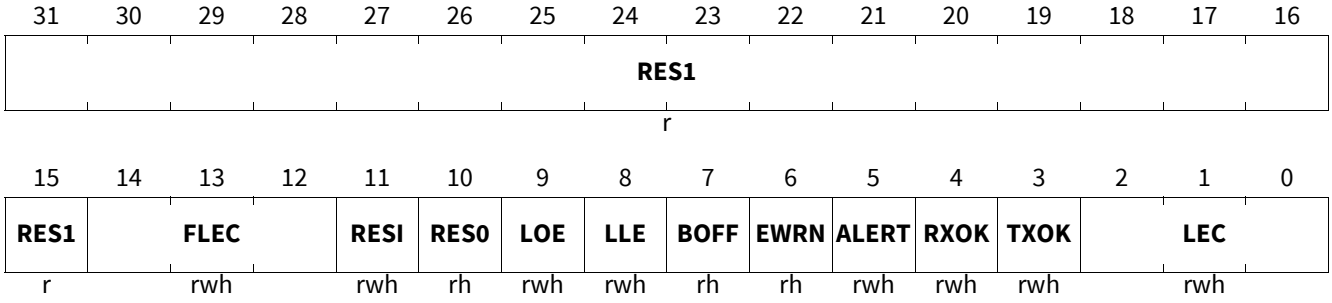
Field	Bits	Type	Description
CANDIS	4	rw	**CAN Disable** Setting this bit disables the CAN node. The CAN node first waits until it is bus-idle or bus-off. Then bit NCRx.INIT is automatically set, and an alert interrupt is generated if bit ALIE is set.
TXDIS	5	rw	**Transmit Disable** Setting this bit disables the transmission on CAN node x as soon as bus-idle is reached. Reception and bits in MOSTATn, e.g. TXRQ, will not be influenced.
CCE	6	rw	**Configuration Change Enable** 0_B The Bit Timing Register, the Port Control Register, Error Counter Register and NCRx.FDEN bit may only be read. All attempts to modify them are ignored. 1_B The Bit Timing Register, the Port Control Register, Error Counter Register and NCRx.FDEN bit may be read and written.
CALM	7	rw	**CAN Analyzer Mode** If this bit is set, then the CAN node operates in Analyzer Mode. This means that messages may be received, but not transmitted. No acknowledge is sent on the CAN bus upon frame reception. Active-error flags are sent recessive instead of dominant. The transmit line is continuously held at recessive (1) level. Bit CALM can be written only while bit INIT is set.
RES0	8	rw	**Reserved** Read as 0; should be written with 0.
FDEN	9	rw	**CAN Flexible Data-Rate Enable** This bit enables the CAN FD feature: Bit FDEN is protected by CCE and INIT bit. Please see **Section 11.8.10.4** and **Table 145** 0_B CAN FD disabled. Message transfer for CAN frames using classical CAN Format. 1_B CAN FD enabled. Message transfer for CAN frames using CAN FD Format[1)].
PED	10	rw	**Protocol Exception Disable** The protocol exception event is described in the ISO11898-1 as option. The error frame on the res bit can be controlled with this option. This bit is CCE and INIT protected. 0_B Protocol Exception Event Enabled 1_B Protocol Exception Event Disabled
RES1	14:11, 31:16	r	**Reserved** Read as 0; should be written with 0.
NISO	15	rw	**Non-ISO Operation** If this bit is set, the MultiCAN+ uses the CAN FD frame format as specified by the ISO11898-1 DIS 2014 format. The bit is CCE and INIT protected. 0_B CAN FD frame format according to ISO11898-1 1_B CAN FD frame format none ISO. If this bit is set, the CRC software workaround is recommended.

1) Message transmission in Classical CAN, Long Frame or Long + Fast CAN FD Frames. Message Reception according to CAN FD Protocol Specification, (i.e. Able to Receive Classical CAN Format Frames ISO11898-1 Long Frame and Long + Fast CAN FD Frames)

Node 0 Status Register

The Node Status Register NSRx reports errors as well as successfully transferred CAN frames.

CAN_NSR0
Node 0 Status Register **(0204_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
LEC	2:0	rwh	**Last Error Code** This bit field indicates the type of the last (most recent) CAN error. The encoding of this bit field is described in **Table 141**.
TXOK	3	rwh	**Message Transmitted Successfully** TXOK must be reset by software (write 0). Writing 1 has no effect. 0_B No successful transmission since last (most recent) flag reset. 1_B A message has been transmitted successfully (error-free and acknowledged by at least another node).
RXOK	4	rwh	**Message Received Successfully** RXOK must be reset by software (write 0). Writing 1 has no effect. 0_B No successful reception since last (most recent) flag reset. 1_B A message has been received successfully.
ALERT	5	rwh	**Alert Warning** The ALERT bit is set upon the occurrence of one of the following events (the same events which also trigger an alert interrupt if ALIE is set): ALERT must be reset by software (write 0). Writing 1 has no effect. • A change of bit NSRx.BOFF • A change of bit NSRx.EWRN • A List Length Error, which also sets bit NSRx.LLE • A List Object Error, which also sets bit NSRx.LOE
EWRN	6	rh	**Error Warning Status** 0_B No warning limit exceeded. 1_B One of the error counters REC or TEC reached the warning limit EWRNLVL.
BOFF	7	rh	**Bus-off Status** 0_B CAN controller is not in the bus-off state. 1_B CAN controller is in the bus-off state.

Field	Bits	Type	Description
LLE	8	rwh	**List Length Error** LLE must be reset by software (write 0). Writing 1 has no effect. 0_B No List Length Error since last (most recent) flag reset. 1_B A List Length Error has been detected during message acceptance filtering. The number of elements in the list that belongs to this CAN node differs from the list SIZE given in the list termination pointer.
LOE	9	rwh	**List Object Error** LOE must be reset by software (write 0). Writing 1 has no effect. 0_B No List Object Error since last (most recent) flag reset. 1_B A List Object Error has been detected during message acceptance filtering. A message object with wrong LIST index entry in the Message Object Status Register has been detected.
RES0	10	rh	**Reserved**
RESI	11	rwh	**Received Error State Indicator Flag** This bit is an error flag that is set when the ESI flag in a received CAN FD frame is set. The flag can be reset, by writing a 0 to it. 0_B Last received CAN FD message did not have its ESI flag set. 1_B Last received CAN FD message had its ESI flag set.
FLEC	14:12	rwh	**Fast Last Error Code** This bit field indicates the type of the last (most recent) CAN error that occurred in the data phase of a CAN FD format frame with its BRS flag set. Coding is the same as for LEC. The encoding of this bit field is described in **Table 141**. This field will be cleared to zero when a CAN FD frame with its BRS flag set has been transferred (reception or transmission) without error.
RES1	31:15	r	**Reserved** Read as 0; should be written with 0.

Encoding of the LEC Bit field

Table 141 Encoding of the LEC Bit field

LEC Value	Signification
000_B	**No Error:** No error was detected for the last (most recent) message on the CAN bus.
001_B	**Stuff Error:** More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.
010_B	**Form Error:** A fixed format part of a received frame has the wrong format.
011_B	**Ack Error:** The transmitted message was not acknowledged by another node.
100_B	**Bit1 Error:** During a message transmission, the CAN node tried to send a recessive level (1) outside the arbitration field and the acknowledge slot, but the monitored bus value was dominant.

Table 141 Encoding of the LEC Bit field (cont'd)

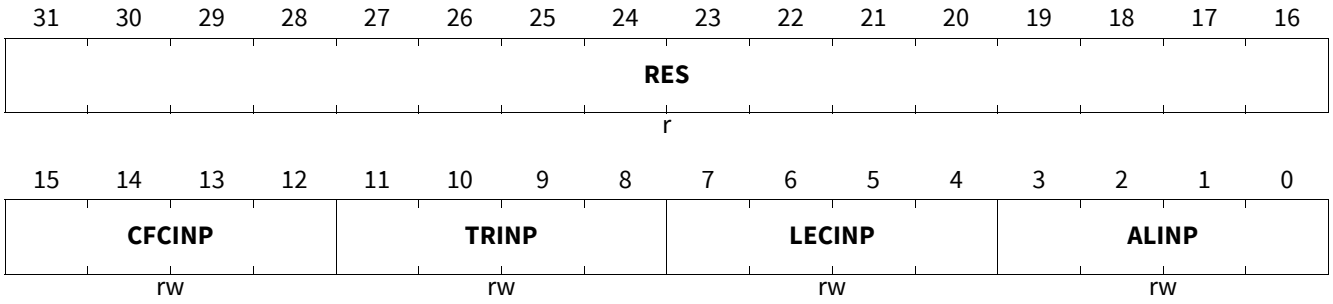
LEC Value	Signification
101_B	**Bit0 Error:** Two different conditions are signaled by this code: 1. During transmission of a message (or acknowledge bit, active-error flag, overload flag), the CAN node tried to send a dominant level (0), but the monitored bus value was recessive. 2. During bus-off recovery, this code is set each time a sequence of 11 recessive bits has been monitored. The CPU may use this code as indication that the bus is not continuously disturbed.
110_B	**CRC Error:** The CRC checksum of the received message was incorrect.
111_B	**CPU write to LEC:** Whenever the CPU writes the value 111 to LEC, it takes the value 111. Whenever the CPU writes another value to LEC, the written LEC value is ignored.

Note: When a frame in CAN FD format reached data phase with BRS flag set, the next CAN event (error/valid frame) will be shown in FLEC instead of LEC. An error in a fixed stuff bit of CAN FD CRC will be shown as Form and not Stuff Error. Correspondingly CAN event (error/valid frame) will be shown back at LEC after the first bit of the CRC delimiter when CAN FD switches from Data bit rate to Nominal bit rate.

Node 0 Interrupt Pointer Register

The four interrupt pointers in the Node Interrupt Pointer Register NIPRx select one out of the sixteen interrupt outputs individually for each type of CAN node interrupt. See also **Page 509** for more CAN node interrupt details.

CAN_NIPR0
Node 0 Interrupt Pointer Register **(0208_H)** **RESET_TYPE_5 Value: 0000 0000$_H$**



Field	Bits	Type	Description
ALINP	3:0	rw	**Alert Interrupt Node Pointer** ALINP selects the interrupt output line INT_Om (m = 0-2) for an alert interrupt of CAN Node x. 0_H Interrupt output line INT_O0 is selected. … 2_H Interrupt output line INT_O2 is selected. 3_H not allowed … F_H not allowed
LECINP	7:4	rw	**Last Error Code Interrupt Node Pointer** LECINP selects the interrupt output line INT_Om (m = 0-2) for an LEC interrupt of CAN Node x. 0_H Interrupt output line INT_O0 is selected. … 2_H Interrupt output line INT_O2 is selected. 3_H not allowed … F_H not allowed
TRINP	11:8	rw	**Transfer OK Interrupt Node Pointer** TRINP selects the interrupt output line INT_Om (m = 0-2) for a transfer OK interrupt of CAN Node x. 0_H Interrupt output line INT_O0 is selected. … 2_H Interrupt output line INT_O2 is selected. 3_H not allowed … F_H not allowed
CFCINP	15:12	rw	**Frame Counter Interrupt Node Pointer** CFCINP selects the interrupt output line INT_Om (m = 0-2) for a frame counter overflow interrupt of CAN Node x. 0_H Interrupt output line INT_O0 is selected. … 2_H Interrupt output line INT_O2 is selected. 3_H not allowed … F_H not allowed
RES	31:16	r	**Reserved** Read as 0; should be written with 0.

Node 0 Port Control Register

The Node Port Control Register NPCRx configures the CAN bus transmit/receive ports. NPCRx can be written only if bit NCRx.CCE is set.

CAN_NPCR0

Node 0 Port Control Register **(020C$_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

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

Field	Bits	Type	Description
RXSEL	2:0	rw	**Receive Select** RXSEL selects one out of 8 possible receive inputs. The CAN receive signal is performed only through the selected input. *Note: In TLE989x/TLE988x, only specific combinations of RXSEL are available (see also **"Node receive input selection" on Page 541** for description and the page before for RXSEL selections).*
RES	7:3, 31:9	r	**Reserved** Read as 0; should be written with 0.
LBM	8	rw	**Loop-Back Mode** 0_B Loop-Back Mode is disabled. 1_B Loop-Back Mode is enabled. This node is connected to an internal (virtual) loop-back CAN bus. All CAN nodes which are in Loop-Back Mode are connected to this virtual CAN bus so that they can communicate with each other internally. The external transmit line is forced recessive in Loop-Back Mode.

Node 0 Bit Timing Register

The Node Bit Timing Register NBTRx contains all parameters to set up the bit timing for the CAN transfer. NBTRx can be written only if bit NCRx.CCE is set. Please note that NBTRx is a register with two views, depending on NCRx.FDEN settings different view applies. (i.e. When NCRx.FDEN=0, CAN_NBTRx view applies and when NCRx.FDEN=1, CAN_NBTEVRx applies)

CAN_NBTR0

Node 0 Bit Timing Register **(0210$_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV8	TSEG2			TSEG1				SJW		BRP					
rw	rw			rw				rw		rw					

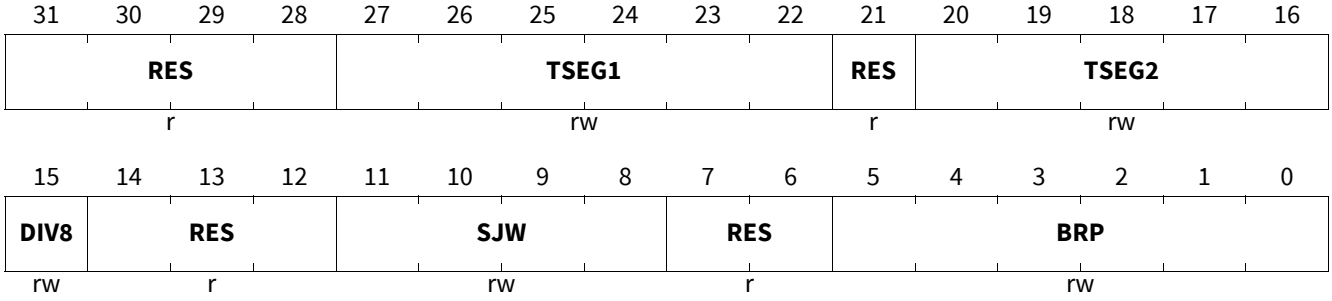
Field	Bits	Type	Description
BRP	5:0	rw	**Baud Rate Prescaler** The duration of one time quantum is given by (BRP + 1) clock cycles if DIV8 = 0. The duration of one time quantum is given by 8 × (BRP + 1) clock cycles if DIV8 = 1.
SJW	7:6	rw	**(Re) Synchronization Jump Width** (SJW + 1) time quanta are allowed for re-synchronization.
TSEG1	11:8	rw	**Time Segment Before Sample Point** (TSEG1 + 1) time quanta is the user-defined nominal time between the end of the synchronization segment and the sample point. It includes the propagation segment, which takes into account signal propagation delays. The time segment may be lengthened due to re-synchronization. Valid values for TSEG1 are 2 to 15.
TSEG2	14:12	rw	**Time Segment After Sample Point** (TSEG2 + 1) time quanta is the user-defined nominal time between the sample point and the start of the next synchronization segment. It may be shortened due to re-synchronization. Valid values for TSEG2 are 1 to 7.
DIV8	15	rw	**Divide Prescaler Clock by 8** 0_B A time quantum lasts (BRP+1) clock cycles. 1_B A time quantum lasts 8 × (BRP+1) clock cycles.
RES	31:16	r	**Reserved** Read as 0; should be written with 0.

Node 0 Bit Timing Extended View Register

The Node Bit Timing Extended View Register NBTEVRx is applicable only when NCRx.FDEN=1 (CAN FD enabled). NBTRx register can be written only if bit NCRx.CCE is set.

CAN_NBTEVR0

Node 0 Bit Timing Extended View Register (0210_H) RESET_TYPE_5 Value: $0000\ 0000_H$

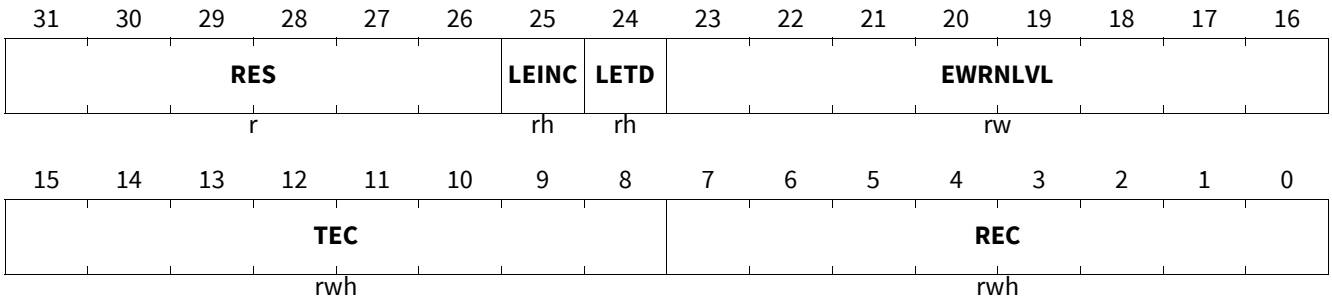


Field	Bits	Type	Description
BRP	5:0	rw	**Baud Rate Prescaler** The duration of one time quantum is given by (BRP + 1) clock cycles if DIV8 = 0. The duration of one time quantum is given by 8 × (BRP + 1) clock cycles if DIV8 = 1.
RES	7:6, 14:12, 21, 31:28	r	**Reserved** Read as 0; should be written with 0.
SJW	11:8	rw	**(Re) Synchronization Jump Width** (SJW + 1) time quanta are allowed for re-synchronization.
DIV8	15	rw	**Divide Prescaler Clock by 8** 0_B A time quantum lasts (BRP+1) clock cycles. 1_B A time quantum lasts 8 × (BRP+1) clock cycles.
TSEG2	20:16	rw	**Time Segment After Sample Point** (TSEG2 + 1) time quanta is the user-defined nominal time between the sample point and the start of the next synchronization segment. It may be shortened due to re-synchronization. Valid values for TSEG2 are 1 to 31.
TSEG1	27:22	rw	**Time Segment Before Sample Point** (TSEG1 + 1) time quanta is the user-defined nominal time between the end of the synchronization segment and the sample point. It includes the propagation segment, which takes into account signal propagation delays. The time segment may be lengthened due to re-synchronization. Valid values for TSEG1 are 2 to 63.

Node 0 Error Counter Register

The Node Error Counter Register NECNTx contains the CAN receive and transmit error counter as well as some additional bits to ease error analysis. NECNTx can be written only if bit NCRx.CCE is set.

CAN_NECNT0
Node 0 Error Counter Register **(0214_H)** **RESET_TYPE_5 Value: $0060\ 0000_H$**



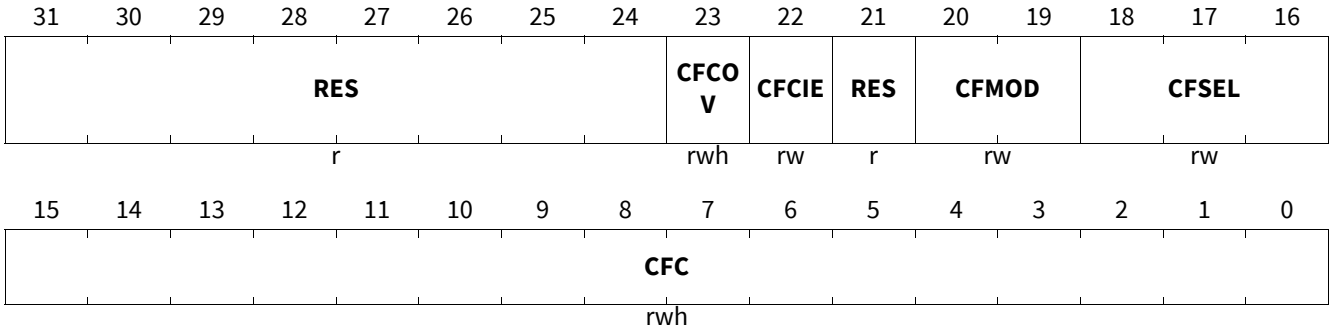
Field	Bits	Type	Description
REC	7:0	rwh	**Receive Error Counter** Bit field REC contains the value of the receive error counter of CAN node x.

Field	Bits	Type	Description
TEC	15:8	rwh	**Transmit Error Counter** Bit field TEC contains the value of the transmit error counter of CAN node x.
EWRNLVL	23:16	rw	**Error Warning Level** Bit field EWRNLVL determines the threshold value (warning level, default 96) to be reached in order to set the corresponding error warning bit EWRN.
LETD	24	rh	**Last Error Transfer Direction** 0_B The last error occurred while the CAN node x was receiver (REC has been incremented). 1_B The last error occurred while the CAN node x was transmitter (TEC has been incremented).
LEINC	25	rh	**Last Error Increment** 0_B The last error led to an error counter increment of 1. 1_B The last error led to an error counter increment of 8.
RES	31:26	r	**Reserved** Read as 0; should be written with 0.

Node 0 Frame Counter Register

The Node Frame Counter Register NFCRx contains the actual value of the frame counter as well as control and status bits of the frame counter.

CAN_NFCR0
Node 0 Frame Counter Register **(0218_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CFC	15:0	rwh	**CAN Frame Counter** In Frame Count Mode (CFMOD = 00_B), this bit field contains the frame count value. In Time Stamp Mode (CFMOD = 01_B), this bit field contains the captured bit time count value, captured with the start of a new frame. In all Bit Timing Analysis Modes[1] (CFMOD = 10_B), CFC always displays the number of f_{CLC} clock cycles (measurement result) minus 1. Example: a CFC value of 34 in measurement mode CFSEL = 000_B means that 35 f_{CLC} clock cycles have been elapsed between the most recent two dominant edges on the receive input. In Error Count Mode (CFMOD = 11_B), this bit field contains the total amount of error frames received or error detected by the node.
CFSEL	18:16	rw	**CAN Frame Count Selection** This bit field selects the function of the frame counter for the chosen frame count mode. **Frame Count Mode** Bit 0 If Bit 0 of CFSEL is set, then CFC is incremented each time a foreign frame (i.e. a frame not matching to a message object) has been received on the CAN bus. Bit 1 If Bit 1 of CFSEL is set, then CFC is incremented each time a frame matching to a message object has been received on the CAN bus. Bit 2 If Bit 2 of CFSEL is set, then CFC is incremented each time a frame has been transmitted successfully by the node. **Time Stamp Mode** The frame counter is incremented (internally) at the beginning of a new bit time. The value is sampled during the SOF bit of a new frame. The sampled value is visible in the CFC field. **Bit Timing Mode** The available bit timing measurement modes are shown in **Table 142**. If CFCIE is set, then an interrupt on request node x (where x is the CAN node number) is generated with a CFC update. **Error Count Mode** The frame counter is incremented when an error frame is received or an error is detected by the node. (001_B to 110_B) (see **Table 141** for **Encoding of the LEC Bit field**). The configuration is don't care, in this mode.

Field	Bits	Type	Description
CFMOD	20:19	rw	**CAN Frame Counter Mode** This bit field determines the operation mode of the frame counter. 00_B Frame Count Mode: The frame counter is incremented upon the reception and transmission of frames. 01_B Time Stamp Mode: The frame counter is used to count bit times. 10_B Bit Timing Mode: The frame counter is used for analysis of the bit timing. 11_B Error Count Mode: The frame counter is used for counting when an error frame is received or an error is detected by the node.
RES	21, 31:24	r	**Reserved** Read as 0; should be written with 0.
CFCIE	22	rw	**CAN Frame Count Interrupt Enable** CFCIE enables the CAN frame counter overflow interrupt of CAN node x. Bit field NIPRx.CFCINP selects the interrupt output line that is activated at this type of interrupt. 0_B CAN frame counter overflow interrupt is disabled. 1_B CAN frame counter overflow interrupt is enabled.
CFCOV	23	rwh	**CAN Frame Counter Overflow Flag** Flag CFCOV is set upon a frame counter overflow (transition from $FFFF_H$ to 0000_H). In bit timing analysis mode, CFCOV is set upon an update of CFC. An interrupt request is generated if CFCIE = 1. CFCOV must be reset by software. 0_B No overflow has occurred since last flag reset. 1_B An overflow has occurred since last flag reset.

1) The value of NFCRx.CFC is valid one module cycle later when NFCRx.CFCOV is set.

Bit Timing Analysis Modes

Table 142 Bit Timing Analysis Modes (CFMOD = 10)

CFSEL	Measurement
000_B	Whenever a dominant edge (transition from 1 to 0) is monitored on the receive input, the time (measured in clock cycles) between this edge and the most recent dominant edge is stored in CFC.
001_B	Whenever a recessive edge (transition from 0 to 1) is monitored on the receive input the time (measured in clock cycles) between this edge and the most recent dominant edge is stored in CFC.
010_B	Whenever a dominant edge is received as a result of a transmitted dominant edge, the time (clock cycles) between both edges is stored in CFC.
011_B	Whenever a recessive edge is received as a result of a transmitted recessive edge, the time (clock cycles) between both edges is stored in CFC.
100_B	Whenever a dominant edge that qualifies for synchronization is monitored on the receive input, the time (measured in clock cycles) between this edge and the most recent sample point is stored in CFC.

Table 142 Bit Timing Analysis Modes (CFMOD = 10) (cont'd)

CFSEL	Measurement
101_B	With each sample point, the time (measured in clock cycles) between the start of the new bit time and the start of the previous bit time is stored in CFC[11:0]. Additional information is written to CFC[15:12] at each sample point: CFC[15]: Transmit value of actual bit time CFC[14]: Receive sample value of actual bit time CFC[13:12]: CAN bus information (see **Table 143**)
110_B	Reserved, do not use this combination.
111_B	Reserved, do not use this combination.

Table 143 CAN Bus State Information

CFC[13:12]	CAN Bus State
00_B	**NoBit** The CAN bus is idle, performs bit (de-) stuffing or is in one of the following frame segments: SOF, SRR, CRC, delimiters, first 6 EOF bits, IFS.
01_B	**NewBit** This code represents the first bit of a new frame segment. The current bit is the first bit in one of the following frame segments: Bit 10 (MSB) of standard ID (transmit only), RTR, reserved bits, IDE, DLC(MSB), bit 7 (MSB) in each data byte and the first bit of the ID extension.
10_B	**Bit** This code represents a bit inside a frame segment with a length of more than one bit (not the first bit of those frame segments that is indicated by NewBit). The current bit is processed within one of the following frame segments: ID bits (except first bit of standard ID for transmission and first bit of ID extension), DLC (3 LSB) and bits 6-0 in each data byte.
11_B	**Done** The current bit is in one of the following frame segments: Acknowledge slot, last bit of EOF, active/passive-error frame, overload frame. Two or more directly consecutive Done codes signal an Error Frame.

Fast Node 0 Bit Timing Register

The Fast Node Bit Timing Register, FNBTRx contains all parameters to set up CAN bit timing for Data Bit Rate. FNBTRx can be written only if bit NCRx.CCE is set.

CAN_FNBTR0
Fast Node 0 Bit Timing Register **(0238_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

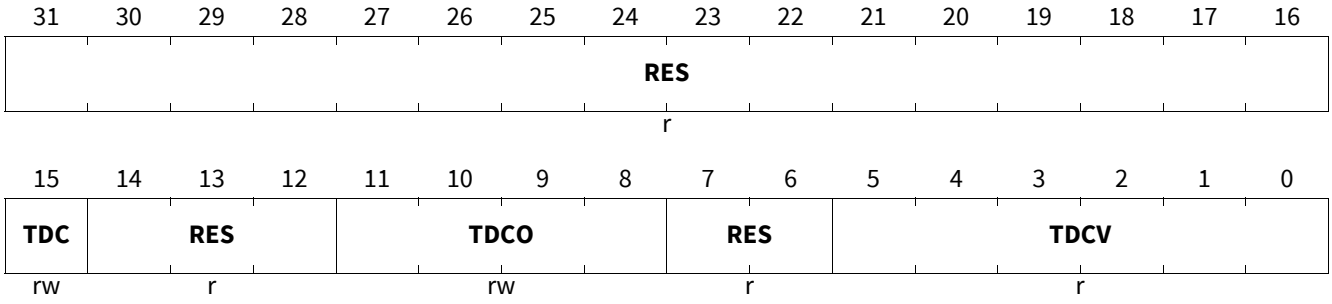
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	FTSEG2			FTSEG1				FSJW		FBRP					
r	rw			rw				rw		rw					

Field	Bits	Type	Description
FBRP	5:0	rw	**Fast Baud Rate Prescaler** The duration of one time quantum is given by (BRP + 1) clock cycles.
FSJW	7:6	rw	**Fast (Re) Synchronization Jump Width** (SJW + 1) time quanta are allowed for re-synchronization.
FTSEG1	11:8	rw	**Fast Time Segment Before Sample Point** (TSEG1 + 1) time quanta is the user-defined nominal time between the end of the synchronization segment and the sample point. It includes the propagation segment, which takes into account signal propagation delays. The time segment may be lengthened due to re-synchronization.
FTSEG2	14:12	rw	**Fast Time Segment After Sample Point** (TSEG2 + 1) time quanta is the user-defined nominal time between the sample point and the start of the next synchronization segment. It may be shortened due to re-synchronization.
RES	31:15	r	**Reserved** Read as 0; should be written with 0.

Node 0 Transmitter Delay Compensation Register

The Transmitter Delay Compensation Register, TDCRx contains all parameters to setup the Transmitter Delay Compensation Feature and the corresponding status bits. NTDCRx register can be written only if bit NCRx.CCE is set.

CAN_NTDCR0
Node 0 Transmitter Delay Compensation Register(023C$_H$) **RESET_TYPE_5 Value: 0000 0000$_H$**



Field	Bits	Type	Description
TDCV	5:0	r	**Transmitter Delay Compensation Value** This bit field shows the secondary sample point which is the sum of the measured Transmitter Delay (from CAN Transmit to Receive) and Transmitter Delay compensation offset, NTDCRz.TDCO. Valid values for TDCV are 0 to 63 times of time quanta t_Q.
RES	7:6, 14:12, 31:16	r	**Reserved** Read as 0; should be written with 0.

Field	Bits	Type	Description
TDCO	11:8	rw	**Transmitter Delay Compensation Offset** This bit field defines the Transmitter Delay compensation offset that is added to the measured Transmitter Delay (from CAN Transmit to Receive) which forms the secondary sample point. Valid values for TDCO are 0 to 15 times of time quanta t_Q.
TDC	15	rw	**Transmitter Delay Compensation Enable** This bit enables the Transmitter Delay Compensation feature: 0_B Transmitter Delay Compensation disabled. 1_B Transmitter Delay Compensation enabled.

Message Object n Function Control Register

The Message Object Function Control Register MOFCRn contains bits that select and configure the function of the message object. It also holds the CAN data length code.

CAN_MOFCRn (n=0-31)
Message Object n Function Control Register (1000_H+n*20_H) **RESET_TYPE_5 Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES1				DLC				STT	SDT	RMM	FRREN	RES1	OVIE	TXIE	RXIE
rw				rwh				rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES1				DATC	DLCC	IDC	GDFS	RES1	FDF	BRS	RES0	MMC			
rw				rw	rw	rw	rw	rw	rwh	rwh	rw	rw			

Field	Bits	Type	Description
MMC	3:0	rw	**Message Mode Control** MMC controls the message mode of message object n. 0_H Standard Message Object 1_H Receive FIFO Base Object 2_H Transmit FIFO Base Object 3_H Transmit FIFO Slave Object 4_H Gateway Source Object 5_H CANFD 64 bytes Message Mode 6_H Do not use … F_H Do not use
RES0	4	rw	**Reserved** Shall be written with 0_H.
BRS	5	rwh	**Bit Rate Switch** Please see **Section 11.8.10.4** and **Table 145** 0_B Message Object transmission/reception without bit rate switching. 1_B Message Object transmission/reception with bit rate switching.

Field	Bits	Type	Description
FDF	6	rwh	**CAN FD Frame Format** If CAN FD is not enabled for a node and FDF is set, transmission will not take place Please see **Section 11.8.10.4** and **Table 145** 0_B Message Object transmission/reception in Classical CAN Frame Format. 1_B Message Object transmission/reception in CAN FD Format (new DLC coding and CRC)
RES1	7, 15:12, 19, 31:28	rw	**Reserved** Read as 0 after reset; value last written is read back; should be written with 0.
GDFS	8	rw	**Gateway Data Frame Send** Applicable only to a gateway source object; ignored in other nodes. 0_B TXRQ is unchanged in the destination object. 1_B TXRQ is set in the gateway destination object after the internal transfer from the gateway source to the gateway destination object.
IDC	9	rw	**Identifier Copy** Applicable only to a gateway source object; ignored in other nodes. 0_B The identifier of the gateway source object is not copied. 1_B The identifier of the gateway source object (after storing the received frame in the source) is copied to the gateway destination object.
DLCC	10	rw	**Data Length Code Copy** Applicable only to a gateway source object; ignored in other nodes. 0_B Data length code is not copied. 1_B Data length code of the gateway source object (after storing the received frame in the source) is copied to the gateway destination object.
DATC	11	rw	**Data Copy** Applicable only to a gateway source object; ignored in other nodes. 0_B Data fields are not copied. 1_B Data fields in registers MODATALn and MODATAHn of the gateway source object (after storing the received frame in the source) are copied to the gateway destination.
RXIE	16	rw	**Receive Interrupt Enable** RXIE enables the message receive interrupt of message object n. This interrupt is generated after reception of a CAN message (independent of whether the CAN message is received directly or indirectly via a gateway action). Bit field MOIPRn.RXINP selects the interrupt output line which becomes activated at this type of interrupt. 0_B Message receive interrupt is disabled. 1_B Message receive interrupt is enabled.

Field	Bits	Type	Description
TXIE	17	rw	**Transmit Interrupt Enable** TXIE enables the message transmit interrupt of message object n. This interrupt is generated after the transmission of a CAN message. Bit field MOIPRn.TXINP selects the interrupt output line which becomes activated at this type of interrupt. 0_B Message transmit interrupt is disabled. 1_B Message transmit interrupt is enabled.
OVIE	18	rw	**Overflow Interrupt Enable** OVIE enables the FIFO full interrupt of message object n. This interrupt is generated when the pointer to the current message object (CUR) reaches the value of SEL in the FIFO/Gateway Pointer Register. If message object n is a Receive FIFO base object, bit field MOIPRn.TXINP selects the interrupt output line which becomes activated at this type of interrupt. If message object n is a Transmit FIFO base object, bit field MOIPRn.RXINP selects the interrupt output line which becomes activated at this type of interrupt. For all other message object modes, bit OVIE has no effect. 0_B FIFO full interrupt is disabled. 1_B FIFO full interrupt is enabled.
FRREN	20	rw	**Foreign Remote Request Enable** Specifies whether the TXRQ bit is set in message object n or in a foreign message object referenced by the pointer CUR. 0_B TXRQ of message object n is set on reception of a matching Remote Frame. 1_B TXRQ of the message object referenced by the pointer CUR is set on reception of a matching Remote Frame.
RMM	21	rw	**Transmit Object Remote Monitoring** Bit RMM applies only to transmit objects and has no effect on receive objects. 0_B Remote monitoring is disabled: Identifier, IDE bit, and DLC of message object n remain unchanged upon the reception of a matching Remote Frame. 1_B Remote monitoring is enabled: Identifier, IDE bit, and DLC of a matching Remote Frame are copied to transmit object n in order to monitor incoming Remote Frames.
SDT	22	rw	**Single Data Transfer** If SDT = 1 and message object n is not a FIFO base object, then MSGVAL is reset when this object has taken part in a successful data transfer (receive or transmit). If SDT = 1 and message object n is a FIFO base object, then MSGVAL is reset when the pointer to the current object CUR reaches the value of SEL in the FIFO/Gateway Pointer Register. With SDT = 0, bit MSGVAL is not affected.

Field	Bits	Type	Description
STT	23	rw	**Single Transmit Trial** If this bit is set, then TXRQ is cleared on transmission start of message object n. Thus, no transmission retry is performed in case of transmission failure.
DLC	27:24	rwh	**Data Length Code** Bit field determines the number of data bytes for message object n. In Classical CAN Format: A value of DLC > 8 results in a data length of 8 data bytes. If a frame with DLC > 8 is received, the received value is stored in the message object. In CAN FD format, valid values for DLC are 0 to 15. See **Table 144**

Coding of DLC in CAN FD

Table 144 Coding of DLC in CAN FD

DLC Value	Description
0000_B	0 Data Byte for Message Object n.
$\cdots_B$	...
1000_B	8 Data Byte for Message Object n.
1001_B	12 Data Byte for Message Object n.
1010_B	16 Data Byte for Message Object n.
1011_B	20 Data Byte for Message Object n.
1100_B	24 Data Byte for Message Object n.
1101_B	32 Data Byte for Message Object n.
1110_B	48 Data Byte for Message Object n.
1111_B	64 Data Byte for Message Object n.

CAN FD Transmit And Receive Behavior

Message transmission and reception behavior of CAN data frames is summarized at **Table 145** below.

Table 145 CAN FD Transmit And Receive Behavior

FDEN	FDF[1)]	BRS[1)]	Transmit Behavior
0_B	0_B	0_B	Classical CAN Frames (ISO 11898-1)
0_B	0_B	1_B	Classical CAN Frames (ISO 11898-1)
0_B	1_B	0_B	Transmission Cancelled[2)] (i.e. TXRQ bit cleared)
0_B	1_B	1_B	Transmission Cancelled[2)] (i.e. TXRQ bit cleared)
1_B	0_B	0_B	Classical CAN Frames (ISO 11898-1)
1_B	0_B	1_B	Classical CAN Frames (ISO 11898-1)
1_B	1_B	0_B	Long Frame (i.e. CAN FD frame with BRS = 0, whole frame transmitted with slow baudrate)
1_B	1_B	1_B	Long + Fast Frame (i.e. CAN FD frame with BRS = 1, switching fast baudrate for data phase)
FDEN	**FDF[3)]**	**BRS[3)]**	**Receive Behavior[4)]**

Table 145 CAN FD Transmit And Receive Behavior (cont'd)

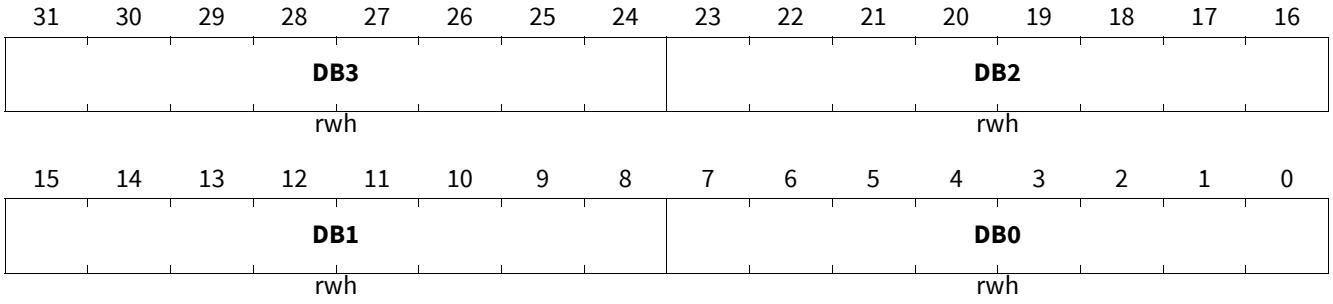
FDEN	FDF[1)]	BRS[1)]	Transmit Behavior
0_B	$0/1_B$	$0/1_B$	- Classical CAN Frames
1_B	0_B	$0/1_B$	- Classical CAN Frames
1_B	1_B	$0/1_B$	- CAN FD Frames - Long Frames possible (DLC dependent) - Fast Frames (dependent on BRS)

1) User writes.
2) TXRQ in message object is cleared upon transmit setup and transmit setup cancelled. CAN node is not blocked and able to transmit other message objects.
3) Hardware writes.
4) Please note that Remote Frames Request do not change BRS and FDF bits.

Extended Message Object n Data 0 Register

Extended Message Object n Data Register represents the alternate register view for MOFCR, MOFGPR, MOIPR, MOAMR, MODATAL, MODATAH and MOAR registers as pointed using (MOFGPR.TOP, MOFGPR.BOT) on the message object with message mode chosen in CAN FD 64 bytes message mode, (i.e. MOFCR.MMC = 5). (See **Section 11.8.10.10**)

CAN_EMOnDATA0 (n=0-31)
Extended Message Object n Data 0 Register (1000_H+n*20_H) **RESET_TYPE_5 Value: 0000 0000_H**



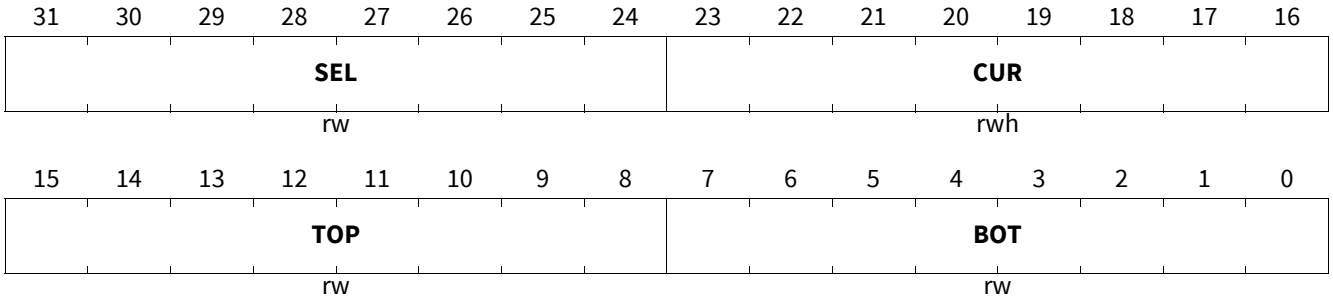
Field	Bits	Type	Description
DB0	7:0	rwh	**Data Byte 0 of Message Object n**
DB1	15:8	rwh	**Data Byte 1 of Message Object n**
DB2	23:16	rwh	**Data Byte 2 of Message Object n**
DB3	31:24	rwh	**Data Byte 3 of Message Object n**

Message Object n FIFO/Gateway Pointer Register

The Message Object FIFO/Gateway Pointer register MOFGPRn contains a set of message object link pointers that are used for FIFO and gateway operations.

CAN_MOFGPRn (n=0-31)

Message Object n FIFO/Gateway Pointer Register(1004_H+n*20_H) **RESET_TYPE_5 Value: 0000 0000_H**



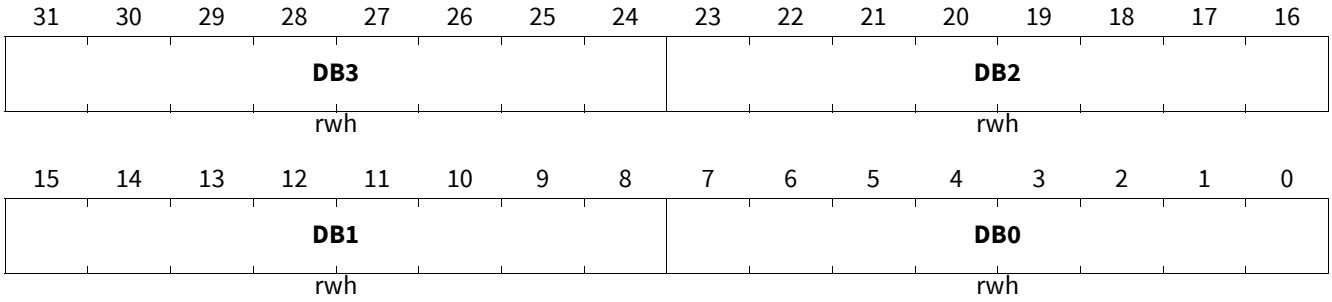
Field	Bits	Type	Description
BOT	7:0	rw	**Bottom Pointer** Bit field BOT points to the first element in a FIFO structure. Or in the case when MOFCR.MMC=5, CAN FD 64 bytes message mode, BOT points to where Data bytes 8-35 are stored in the message object.
TOP	15:8	rw	**Top Pointer** Bit field TOP points to the last element in a FIFO structure. Or in the case when MOFCR.MMC=5, CAN FD 64 bytes message mode, BOT points to where Data bytes 36-63 are stored in the message object.
CUR	23:16	rwh	**Current Object Pointer** Bit field CUR points to the actual target object within a FIFO/Gateway structure. After a FIFO/gateway operation CUR is updated with the message number of the next message object in the list structure (given by PNEXT of the Message Object Status Register) until it reaches the FIFO top element (given by TOP) when it is reset to the bottom element (given by BOT).
SEL	31:24	rw	**Object Select Pointer** Bit field SEL is the second (software) pointer to complement the hardware pointer CUR in the FIFO structure. SEL is used for monitoring purposes (FIFO interrupt generation).

Extended Message Object n Data 1 Register

Extended Message Object n Data Register represents the alternate register view for MOFCR, MOFGPR, MOIPR, MOAMR, MODATAL, MODATAH and MOAR registers as pointed using (MOFGPR.TOP, MOFGPR.BOT) on the message object with message mode chosen in CAN FD 64 bytes message mode, (i.e. MOFCR.MMC = 5). (See **Section 11.8.10.10**)

CAN_EMOnDATA1 (n=0-31)

Extended Message Object n Data 1 Register (1004_H+n*20_H) **RESET_TYPE_5 Value: 0000 0000_H**



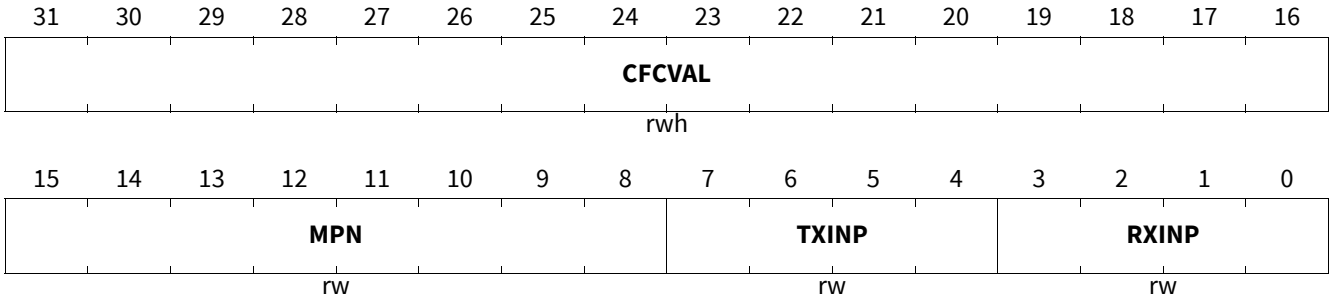
Field	Bits	Type	Description
DB0	7:0	rwh	**Data Byte 0 of Message Object n**
DB1	15:8	rwh	**Data Byte 1 of Message Object n**
DB2	23:16	rwh	**Data Byte 2 of Message Object n**
DB3	31:24	rwh	**Data Byte 3 of Message Object n**

Message Object n Interrupt Pointer Register

The Message Object Interrupt Pointer Register MOIPRn holds the message interrupt pointers, the message pending number, and the frame counter value of message object n.

CAN_MOIPRn (n=0-31)

Message Object n Interrupt Pointer Register (1008_H+n*20_H) **RESET_TYPE_5 Value: 0000 0000_H**



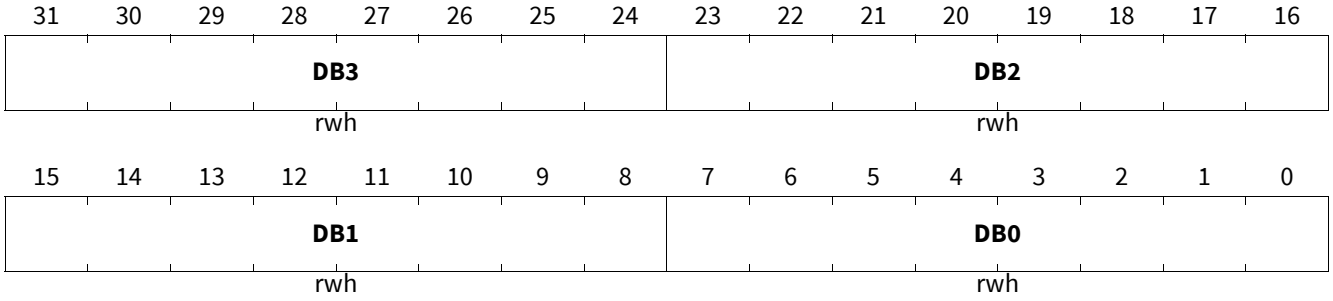
Field	Bits	Type	Description
RXINP	3:0	rw	**Receive Interrupt Node Pointer** RXINP selects the interrupt output line INT_Om (m = 0-2) for a receive interrupt event of message object n. RXINP can also be taken for message pending bit selection (see **Page 520**). 0_H Interrupt output line INT_O0 is selected. … 2_H Interrupt output line INT_O2 is selected. 3_H not allowed … F_H not allowed

Field	Bits	Type	Description
TXINP	7:4	rw	**Transmit Interrupt Node Pointer** TXINP selects the interrupt output line INT_Om (m = 0-2) for a transmit interrupt event of message object n. TXINP can also be taken for message pending bit selection (see **Page 520**). 0_H Interrupt output line INT_O0 is selected. … 2_H Interrupt output line INT_O2 is selected. 3_H not allowed … F_H not allowed
MPN	15:8	rw	**Message Pending Number** This bit field selects the bit position of the bit in the Message Pending Register that is set upon a message object n receive/transmit interrupt.
CFCVAL	31:16	rwh	**CAN Frame Counter Value** When a message is stored in message object n or message object n has been successfully transmitted, the CAN frame counter value NFCRx.CFC is then copied to CFCVAL.

Extended Message Object n Data 2 Register

Extended Message Object n Data Register represents the alternate register view for MOFCR, MOFGPR, MOIPR, MOAMR, MODATAL, MODATAH and MOAR registers as pointed using (MOFGPR.TOP, MOFGPR.BOT) on the message object with message mode chosen in CAN FD 64 bytes message mode, (i.e. MOFCR.MMC = 5). (See **Section 11.8.10.10**)

CAN_EMOnDATA2 (n=0-31)
Extended Message Object n Data 2 Register (1008_H+n*20_H) **RESET_TYPE_5 Value: 0000 0000_H**



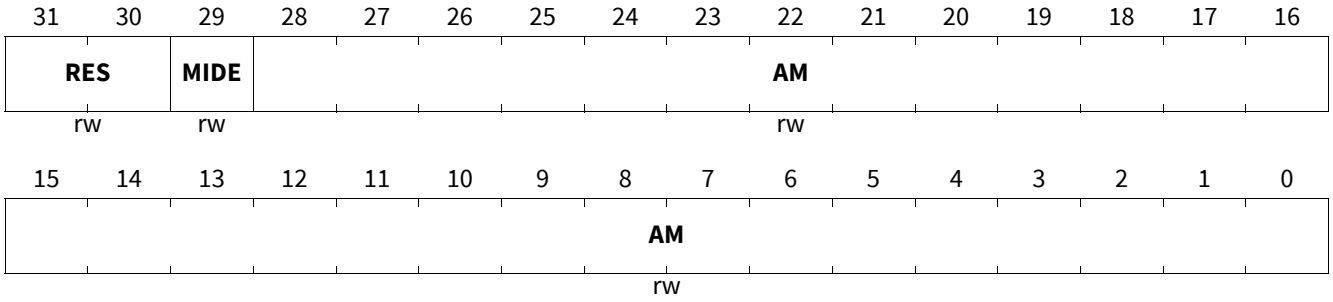
Field	Bits	Type	Description
DB0	7:0	rwh	**Data Byte 0 of Message Object n**
DB1	15:8	rwh	**Data Byte 1 of Message Object n**
DB2	23:16	rwh	**Data Byte 2 of Message Object n**
DB3	31:24	rwh	**Data Byte 3 of Message Object n**

Message Object n Acceptance Mask Register

Message Object n Acceptance Mask Register MOAMRn contains the mask bits for the acceptance filtering of the message object n.

CAN_MOAMRn (n=0-31)

Message Object n Acceptance Mask Register ($100C_H$+n*20_H) **RESET_TYPE_5 Value: 3FFF FFFF$_H$**



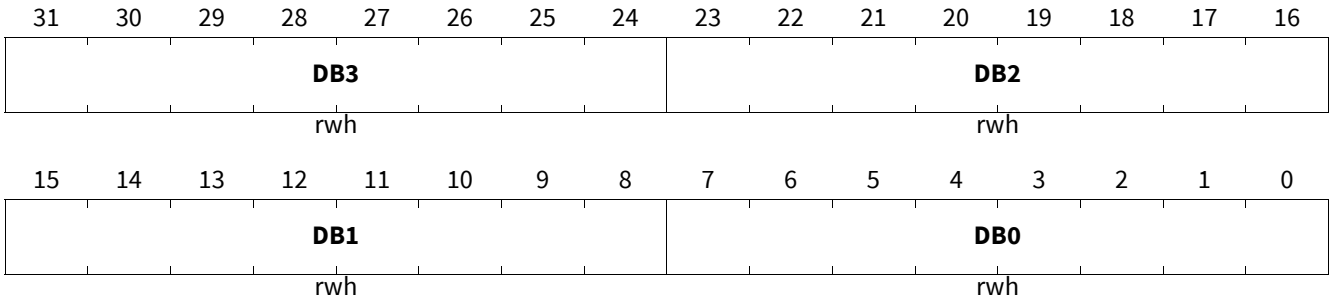
Field	Bits	Type	Description
AM	28:0	rw	**Acceptance Mask for Message Identifier** Bit field AM is the 29-bit mask for filtering incoming messages with standard identifiers (AM[28:18]) or extended identifiers (AM[28:0]). For standard identifiers, bits AM[17:0] are “don’t care”.
MIDE	29	rw	**Acceptance Mask Bit for Message IDE Bit** 0_B Message object n accepts the reception of both, standard and extended frames. 1_B Message object n receives frames only with matching IDE bit.
RES	31:30	rw	**Reserved** Read as 0 after reset; value last written is read back; should be written with 0.

Extended Message Object n Data 3 Register

Extended Message Object n Data Register represents the alternate register view for MOFCR, MOFGPR, MOIPR, MOAMR, MODATAL, MODATAH and MOAR registers as pointed using (MOFGPR.TOP, MOFGPR.BOT) on the message object with message mode chosen in CAN FD 64 bytes message mode, (i.e. MOFCR.MMC = 5). (See **Section 11.8.10.10**)

CAN_EMOnDATA3 (n=0-31)

Extended Message Object n Data 3 Register ($100C_H$+n*20_H) **RESET_TYPE_5 Value: 3FFF FFFF$_H$**



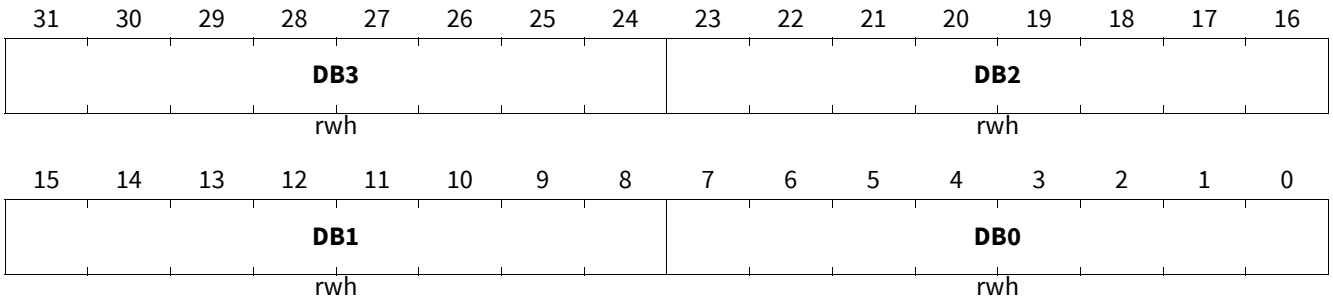
Field	Bits	Type	Description
DB0	7:0	rwh	**Data Byte 0 of Message Object n**
DB1	15:8	rwh	**Data Byte 1 of Message Object n**
DB2	23:16	rwh	**Data Byte 2 of Message Object n**

Field	Bits	Type	Description
DB3	31:24	rwh	**Data Byte 3 of Message Object n**

Message Object n Data Register Low

Message Object n Data Register Low MODATALn contains the lowest four data bytes of message object n. Unused data bytes are set to zero upon reception and ignored for transmission.

CAN_MODATALn (n=0-31)
Message Object n Data Register Low **(1010_H+n*20_H)** **RESET_TYPE_5 Value: 0000 0000_H**

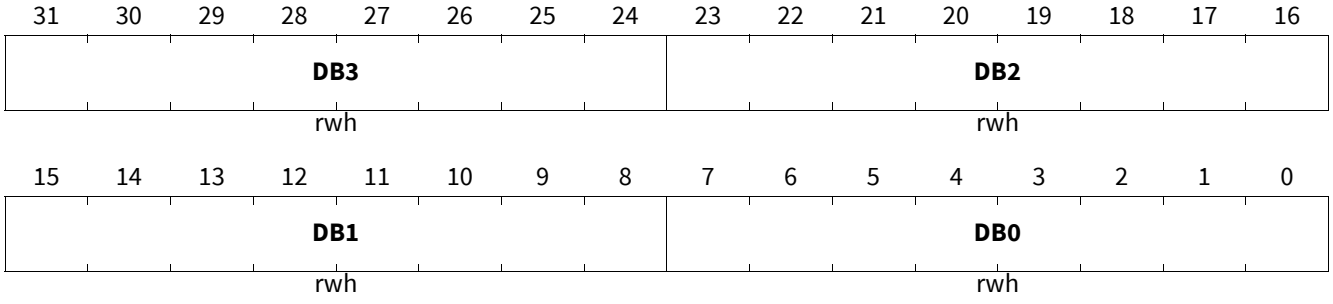


Field	Bits	Type	Description
DB0	7:0	rwh	**Data Byte 0 of Message Object n**
DB1	15:8	rwh	**Data Byte 1 of Message Object n**
DB2	23:16	rwh	**Data Byte 2 of Message Object n**
DB3	31:24	rwh	**Data Byte 3 of Message Object n**

Extended Message Object n Data 4 Register

Extended Message Object n Data Register represents the alternate register view for MOFCR, MOFGPR, MOIPR, MOAMR, MODATAL, MODATAH and MOAR registers as pointed using (MOFGPR.TOP, MOFGPR.BOT) on the message object with message mode chosen in CAN FD 64 bytes message mode, (i.e. MOFCR.MMC = 5). (See **Section 11.8.10.10**)

CAN_EMOnDATA4 (n=0-31)
Extended Message Object n Data 4 Register **(1010_H+n*20_H)** **RESET_TYPE_5 Value: 0000 0000_H**



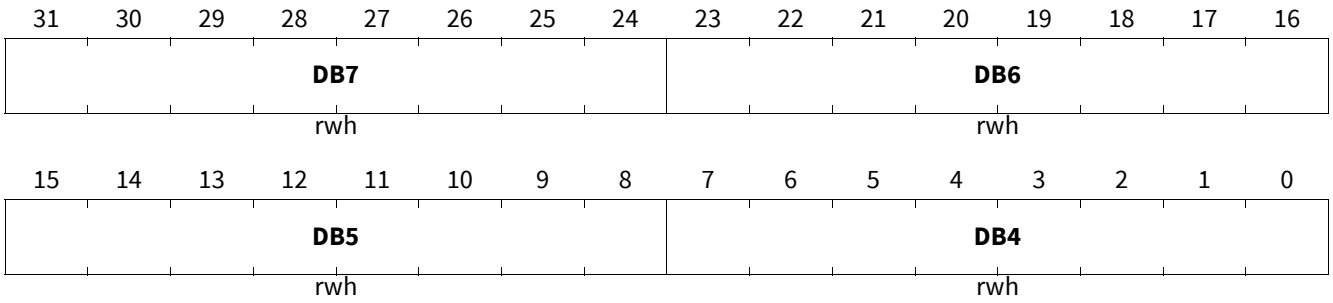
Field	Bits	Type	Description
DB0	7:0	rwh	**Data Byte 0 of Message Object n**
DB1	15:8	rwh	**Data Byte 1 of Message Object n**
DB2	23:16	rwh	**Data Byte 2 of Message Object n**

Field	Bits	Type	Description
DB3	31:24	rwh	**Data Byte 3 of Message Object n**

Message Object n Data Register High

Message Object n Data Register High MODATAH contains the highest four data bytes of message object n. Unused data bytes are set to zero upon reception and ignored for transmission.

CAN_MODATAHn (n=0-31)
Message Object n Data Register High **(1014_H+n*20_H)** **RESET_TYPE_5 Value: 0000 0000_H**

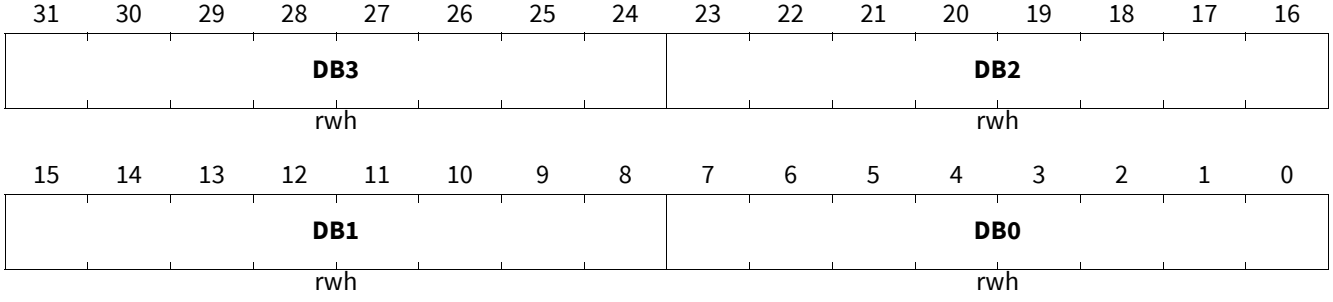


Field	Bits	Type	Description
DB4	7:0	rwh	**Data Byte 4 of Message Object n**
DB5	15:8	rwh	**Data Byte 5 of Message Object n**
DB6	23:16	rwh	**Data Byte 6 of Message Object n**
DB7	31:24	rwh	**Data Byte 7 of Message Object n**

Extended Message Object n Data 5 Register

Extended Message Object n Data Register represents the alternate register view for MOFCR, MOFGPR, MOIPR, MOAMR, MODATAL, MODATAH and MOAR registers as pointed using (MOFGPR.TOP, MOFGPR.BOT) on the message object with message mode chosen in CAN FD 64 bytes message mode, (i.e. MOFCR.MMC = 5). (See **Section 11.8.10.10**)

CAN_EMOnDATA5 (n=0-31)
Extended Message Object n Data 5 Register **(1014_H+n*20_H)** **RESET_TYPE_5 Value: 0000 0000_H**



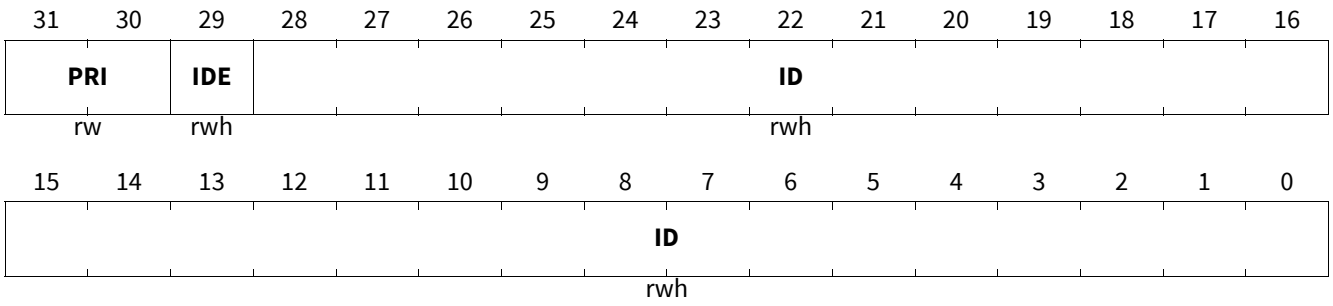
Field	Bits	Type	Description
DB0	7:0	rwh	**Data Byte 0 of Message Object n**
DB1	15:8	rwh	**Data Byte 1 of Message Object n**
DB2	23:16	rwh	**Data Byte 2 of Message Object n**

Field	Bits	Type	Description
DB3	31:24	rwh	**Data Byte 3 of Message Object n**

Message Object n Arbitration Register

Message Object n Arbitration Register MOARn contains the CAN identifier of the message object.

CAN_MOARn (n=0-31)
Message Object n Arbitration Register **(1018_H+n*20_H)** **RESET_TYPE_5 Value: 0000 0000_H**



Field	Bits	Type	Description
ID	28:0	rwh	**CAN Identifier of Message Object n** Identifier of a standard message (ID[28:18]) or an extended message (ID[28:0]). For standard identifiers, bits ID[17:0] are “don’t care”.
IDE	29	rwh	**Identifier Extension Bit of Message Object n** 0_B Message object n handles standard frames with 11-bit identifier. 1_B Message object n handles extended frames with 29-bit identifier.
PRI	31:30	rw	**Priority Class** PRI assigns one of the four priority classes 0, 1, 2, 3 to message object n. A lower PRI number defines a higher priority. Message objects with lower PRI value always win acceptance filtering for frame reception and transmission over message objects with higher PRI value. Acceptance filtering based on identifier/mask and list position is performed only between message objects of the same priority class. PRI also determines the acceptance filtering method for transmission: 00_B Reserved. 01_B Transmit acceptance filtering is based on the list order. This means that message object n is considered for transmission only if there is no other message object with valid transmit request (MSGVAL & TXEN0 & TXEN1 = 1) somewhere before this object in the list. 10_B Transmit acceptance filtering is based on the CAN identifier. This means, message object n is considered for transmission only if there is no other message object with higher priority identifier + IDE + DIR (with respect to CAN arbitration rules) somewhere in the list (see **Table 146**). 11_B Transmit acceptance filtering is based on the list order (as PRI = 01_B).

Transmit Priority of Msg. Objects based on CAN Arbitration Rules

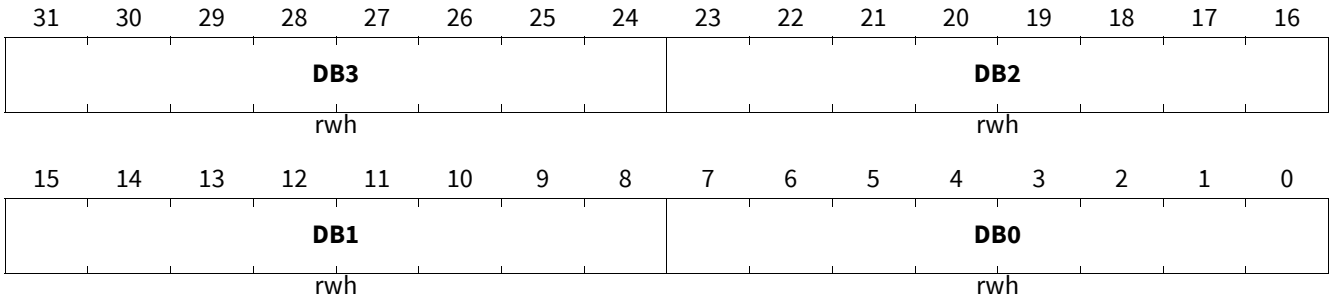
Table 146 Transmit Priority of Msg. Objects Based on CAN Arbitration Rules

Settings of Arbitrarily Chosen Message Objects A and B, (A has higher transmit priority than B)	Comment
A.MOAR[28:18] < B.MOAR[28:18] (11-bit standard identifier of A less than 11-bit standard identifier of B)	Messages with lower standard identifier have higher priority than messages with higher standard identifier. MOAR[28] is the most significant bit (MSB) of the standard identifier. MOAR[18] is the least significant bit of the standard identifier.
A.MOAR[28:18] = B.MOAR[28:18] A.MOAR.IDE = 0 (send Standard Frame) B.MOAR.IDE = 1 (send Extended Frame)	Standard Frames have higher transmit priority than Extended Frames with equal standard identifier.
A.MOAR[28:18] = B.MOAR[28:18] A.MOAR.IDE = B.MOAR.IDE = 0 A.MOSTAT.DIR = 1 (send Data Frame) B.MOSTAT.DIR = 0 (send Remote Fame)	Standard Data Frames have higher transmit priority than standard Remote Frames with equal identifier.
A.MOAR[28:0] = B.MOAR[28:0] A.MOAR.IDE = B.MOAR.IDE = 1 A.MOSTAT.DIR = 1 (send Data Frame) B.MOSTAT.DIR = 0 (send Remote Frame)	Extended Data Frames have higher transmit priority than Extended Remote Frames with equal identifier.
A.MOAR[28:0] < B.MOAR[28:0] A.MOAR.IDE = B.MOAR.IDE = 1 (29-bit identifier)	Extended Frames with lower identifier have higher transmit priority than Extended Frames with higher identifier. MOAR[28] is the most significant bit (MSB) of the overall identifier (standard identifier MOAR[28:18] and identifier extension MOAR[17:0]). MOAR[0] is the least significant bit (LSB) of the overall identifier.

Extended Message Object n Data 6 Register

Extended Message Object n Data Register represents the alternate register view for MOFCR, MOFGPR, MOIPR, MOAMR, MODATAL, MODATAH and MOAR registers as pointed using (MOFGPR.TOP, MOFGPR.BOT) on the message object with message mode chosen in CAN FD 64 bytes message mode, (i.e. MOFCR.MMC = 5). (See **Section 11.8.10.10**)

CAN_EMOnDATA6 (n=0-31)
Extended Message Object n Data 6 Register (1018_H+n*20_H) **RESET_TYPE_5 Value: 0000 0000_H**



Field	Bits	Type	Description
DB0	7:0	rwh	**Data Byte 0 of Message Object n**
DB1	15:8	rwh	**Data Byte 1 of Message Object n**
DB2	23:16	rwh	**Data Byte 2 of Message Object n**
DB3	31:24	rwh	**Data Byte 3 of Message Object n**

Message Object n Control Register

The Message Object Control Register MOCTRn and the Message Object Status Register MOSTATn are located at the same address offset within a message object address block (offset address $1C_H$). The MOCTRn is a write-only register that makes it possible to set/reset CAN transfer related control bits through software. The reset value is not relevant for this write-only register.

CAN_MOCTRn (n=0-31)
Message Object n Control Register **($101C_H+n*20_H$)** **Reset Value: Table 147**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES				SETDIR	SETTXEN1	SETTXEN0	SETTXRQ	SETRXEN	SETRTSEL	SETMSGVAL	SETMSGLST	SETNEWDAT	SETRXUPD	SETTXPND	SETRXPND
w				w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				RESDIR_SETDIR	RESTXEN1	RESTXEN0	RESTXRQ	RESRXEN	RESRTSEL	RESMSGVAL	RESMSGLST	RESNEWDAT	RESRXUPD	RESTXPND	RESRXPND
w				w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RESRXPND	0	w	**Reset Receive Pending** These bits control the reset condition for RXPND (see **Table 148**).
RESTXPND	1	w	**Reset Transmit Pending** These bits control the reset condition for TXPND (see **Table 148**).
RESRXUPD	2	w	**Reset Receive Updating** These bits control the reset condition for RXUPD (see **Table 148**).
RESNEWDAT	3	w	**Reset New Data** These bits control the reset condition for NEWDAT (see **Table 148**).
RESMSGLST	4	w	**Reset Message Lost** These bits control the reset condition for MSGLST (see **Table 148**).
RESMSGVAL	5	w	**Reset Message Valid** These bits control the reset condition for MSGVAL (see **Table 148**).
RESRTSEL	6	w	**Reset Receive/Transmit Selected** These bits control the reset condition for RTSEL (see **Table 148**).
RESRXEN	7	w	**Reset Receive Enable** These bits control the reset condition for RXEN (see **Table 148**).

Field	Bits	Type	Description
RESTXRQ	8	w	**Reset Transmit Request** These bits control the reset condition for TXRQ (see **Table 148**).
RESTXEN0	9	w	**Reset Transmit Enable 0** These bits control the reset condition for TXEN0 (see **Table 148**).
RESTXEN1	10	w	**Reset Transmit Enable 1** These bits control the reset condition for TXEN1 (see **Table 148**).
RESDIR_SETDIR	11	w	**Reset Message Direction** These bits control the reset condition for DIR (see **Table 148**).
RES	15:12, 31:28	w	**Reserved** Should be written with 0.
SETRXPND	16	w	**Set Receive Pending** These bits control the set condition for RXPND (see **Table 148**).
SETTXPND	17	w	**Set Transmit Pending** These bits control the set condition for TXPND (see **Table 148**).
SETRXUPD	18	w	**Set Receive Updating** These bits control the set condition for RXUPD (see **Table 148**).
SETNEWDAT	19	w	**Set New Data** These bits control the set condition for NEWDAT (see **Table 148**).
SETMSGLST	20	w	**Set Message Lost** These bits control the set condition for MSGLST (see **Table 148**).
SETMSGVAL	21	w	**Set Message Valid** These bits control the set condition for MSGVAL (see **Table 148**).
SETRTSEL	22	w	**Set Receive/Transmit Selected** These bits control the set condition for RTSEL (see **Table 148**).
SETRXEN	23	w	**Set Receive Enable** These bits control the set condition for RXEN (see **Table 148**).
SETTXRQ	24	w	**Set Transmit Request** These bits control the set condition for TXRQ (see **Table 148**).
SETTXEN0	25	w	**Set Transmit Enable 0** These bits control the set condition for TXEN0 (see **Table 148**).
SETTXEN1	26	w	**Set Transmit Enable 1** These bits control the set condition for TXEN1 (see **Table 148**).
SETDIR	27	w	**Set Message Direction** These bits control the set condition for DIR (see **Table 148**).

Table 147 Reset Values of CAN_MOCTRn (n=0-31)

Reset Type	Reset Value	Note
RESET_TYPE_5	$0100\ 0000_H$	CAN_MOSTAT0
RESET_TYPE_5	(n + 1) * 0x1000000 + (n - 1) * 0x10000	CAN_MOSTATn (n>0)
RESET_TYPE_5	$0000\ 0000_H$	CAN_MOCTRn

Table 148 Reset/Set Conditions for Bits in Register MOCTRn

RESy Bit[1)]	SETy Bit	Action on Write
Write 0	Write 0	Leave element unchanged
No write		
No write	Write 0	
Write 1	Write 1	
Write 1	Write 0	Reset element
No write		
Write 0	Write 1	Set element
No write		

1) The parameter “y” stands for the second part of the bit name (“RXPND”, “TXPND”, … up to “DIR”).

Message Object n Status Register

The MOSTATn is a read-only register that indicates message object list status information such as the number of the current message object predecessor and successor message object, as well as the list number to which the message object is assigned.

CAN_MOSTATn (n=0-31)
Message Object n Status Register **($101C_H+n*20_H$)** **Reset Value: Table 149**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PNEXT								PPREV							
rh								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LIST				DIR	TXEN1	TXEN0	TXRQ	RXEN	RTSEL	MSGVAL	MSGLST	NEWDAT	RXUPD	TXPND	RXPND
rh				rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RXPND	0	rh	**Receive Pending** RXPND is set by hardware and must be reset by software. 0_B No CAN message has been received. 1_B A CAN message has been received by the message object n, either directly or via gateway copy action.
TXPND	1	rh	**Transmit Pending** TXPND is set by hardware and must be reset by software. 0_B No CAN message has been transmitted. 1_B A CAN message from message object n has been transmitted successfully over the CAN bus.
RXUPD	2	rh	**Receive Updating** 0_B No receive update ongoing. 1_B Message identifier, DLC, and data of the message object are currently updated.

Field	Bits	Type	Description
NEWDAT	3	rh	**New Data** NEWDAT is set by hardware after a received CAN frame has been stored in message object n. NEWDAT is cleared by hardware when a CAN transmission of message object n has been started. NEWDAT should be set by software after the new transmit data has been stored in message object n to prevent the automatic reset of TXRQ at the end of an ongoing transmission. 0_B No update of the message object n since last flag reset. 1_B Message object n has been updated.
MSGLST	4	rh	**Message Lost** 0_B No CAN message is lost. 1_B A CAN message is lost because NEWDAT has become set again when it has already been set.
MSGVAL	5	rh	**Message Valid** Only a valid message object takes part in CAN transfers. 0_B Message object n is not valid. 1_B Message object n is valid.
RTSEL	6	rh	**Receive/Transmit Selected** **Frame Reception:** RTSEL is set by hardware when message object n has been identified for storage of a CAN frame that is currently received. Before a received frame becomes finally stored in message object n, a check is performed to determine if RTSEL is set. Thus the CPU can suppress a scheduled frame delivery to this message object n by clearing RTSEL by software. **Frame Transmission:** RTSEL is set by hardware when message object n has been identified to be transmitted next. A check is performed to determine if RTSEL is still set before message object n is actually set up for transmission and bit NEWDAT is cleared. It is also checked that RTSEL is still set before its message object n is verified due to the successful transmission of a frame. RTSEL needs to be checked only when the context of message object n changes, and a conflict with an ongoing frame transfer shall be avoided. In all other cases, RTSEL can be ignored. RTSEL has no impact on message acceptance filtering. RTSEL is not cleared by hardware. 0_B Message object n is not selected for receive or transmit operation. 1_B Message object n is selected for receive or transmit operation.
RXEN	7	rh	**Receive Enable** RXEN is evaluated for receive acceptance filtering only. 0_B Message object n is not enabled for frame reception. 1_B Message object n is enabled for frame reception.
TXRQ	8	rh	**Transmit Request** The transmit request becomes valid only if TXRQ, TXEN0, TXEN1 and MSGVAL are set. TXRQ is set by hardware if a matching Remote Frame has been received correctly. TXRQ is reset by hardware if message object n has been transmitted successfully and NEWDAT is not set again by software. 0_B No transmission of message object n is requested. 1_B Transmission of message object n on the CAN bus is requested.

Field	Bits	Type	Description
TXEN0	9	rh	**Transmit Enable 0** Message object n can be transmitted only if both bits, TXEN0 and TXEN1, are set. The user may clear TXEN0 in order to inhibit the transmission of a message that is currently updated, or to disable automatic response of Remote Frames. 0_B Message object n is not enabled for frame transmission. 1_B Message object n is enabled for frame transmission.
TXEN1	10	rh	**Transmit Enable 1** Message object n can be transmitted only if both bits, TXEN0 and TXEN1, are set. TXEN1 is used by the module for selecting the active message object in the Transmit FIFOs. 0_B Message object n is not enabled for frame transmission. 1_B Message object n is enabled for frame transmission.
DIR	11	rh	**Message Direction** 0_B Receive Object selected: With TXRQ = 1, a Remote Frame with the identifier of message object n is scheduled for transmission. On reception of a Data Frame with matching identifier, the message is stored in message object n. 1_B Transmit Object selected: If TXRQ = 1, message object n is scheduled for transmission of a Data Frame. On reception of a Remote Frame with matching identifier, bit TXRQ is set.
LIST	15:12	rh	**List Allocation** LIST indicates the number of the message list to which message object n is allocated. LIST is updated by hardware when the list allocation of the object is modified by a panel command.
PPREV	23:16	rh	**Pointer to Previous Message Object** PPREV holds the message object number of the previous message object in a message list structure.
PNEXT	31:24	rh	**Pointer to Next Message Object** PNEXT holds the message object number of the next message object in a message list structure.

Table 149 Reset Values of CAN_MOSTATn (n=0-31)

Reset Type	Reset Value	Note
RESET_TYPE_5	$0100\ 0000_H$	CAN_MOSTAT0
RESET_TYPE_5	(n + 1) * 0x1000000 + (n - 1) * 0x10000	CAN_MOSTATn (n>0)
RESET_TYPE_5	$0000\ 0000_H$	CAN_MOCTRn

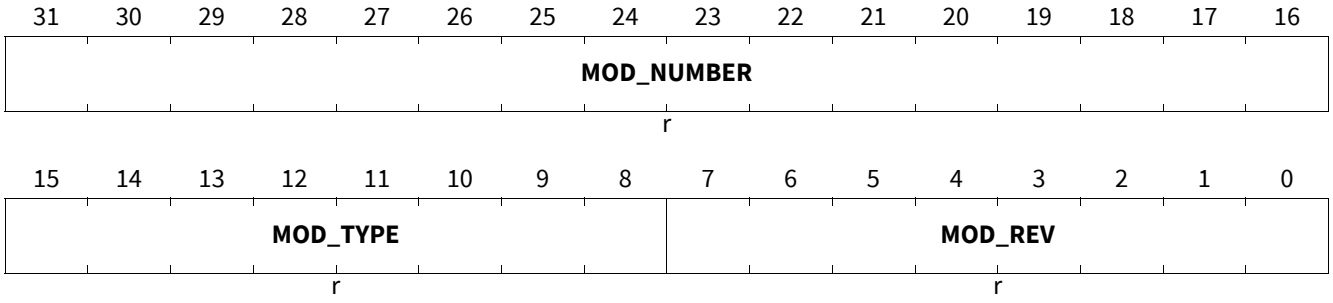
Table 150 MOSTATn Reset Values

Message Object	PNEXT	PPREV	Reset Value
0	1	0	$0100\ 0000_H$
1	2	0	$0200\ 0000_H$
2	3	1	$0301\ 0000_H$
3	4	2	$0402\ 0000_H$
...	...	...	...
31	31	30	$1F1E\ 0000_H$

Module Identification Register

ID

Module Identification Register (0008_H) RESET_TYPE_5 Value: $00B5\ C0XX_H$



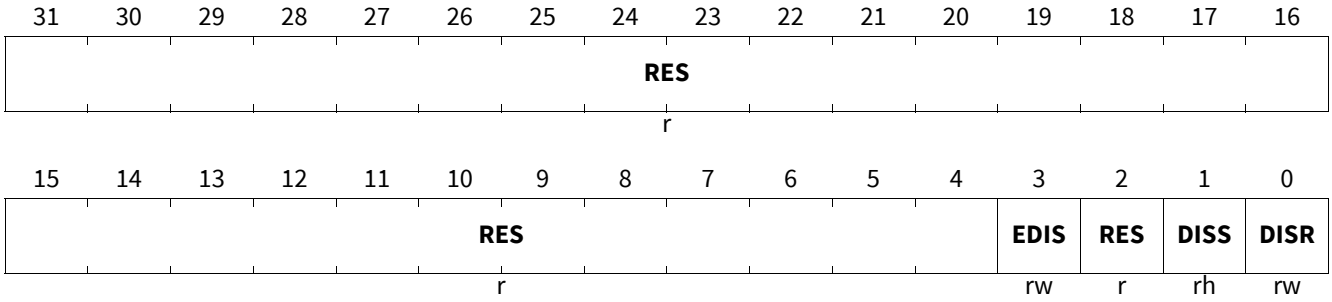
Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision Number** MOD_REV defines the revision number. The value of a module revision starts with 01H (first revision).
MOD_TYPE	15:8	r	**Module Type** $C0_H$ Define the module as a 32-bit module.
MOD_NUMBER	31:16	r	**Module Number Value** This bit field defines the module identification number (=00B5H)

CAN Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI for the module. CLC controls the f_{CAN} module clock signal.

CLC

CAN Clock Control Register **(0000_H)** **RESET_TYPE_5 Value: 0000 0003$_H$**



Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. Note that no register access is possible to any register while module is disabled.
DISS	1	rh	**Module Disable Status Bit** Bit indicates the current status of the module.
RES	2, 31:4	r	**Reserved** Read as 0; should be written with 0.
EDIS	3	rw	**Sleep Mode Enable Control** Used to control module's sleep mode.

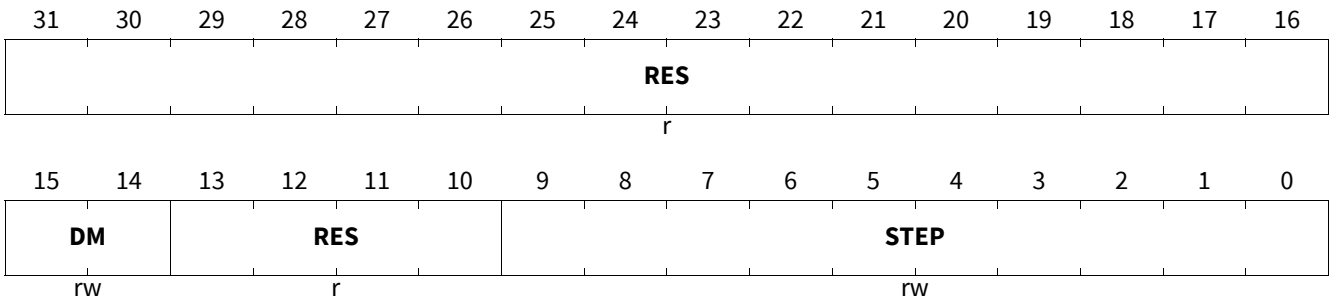
Note: The number of module clock cycles (wait states) which are required by the kernel to execute a read or write access depends on the selected CLC clock frequency.

CAN Fractional Divider Register

The fractional divider register allows the programmer to control the clock rate of the module timer clock f_{CAN}.

FDR

CAN Fractional Divider Register **($000C_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**



Field	Bits	Type	Description
STEP	9:0	rw	**Step Value** Reload or addition value for the result.
RES	13:10, 31:16	r	**Reserved** Read as 0; should be written with 0.

Field	Bits	Type	Description
DM	15:14	rw	**Divider Mode** This bit field selects normal divider mode, fractional divider mode, and off-state.

12 CAN Transceiver (CANTRX)

12.1 Features overview

The CAN transceiver (CANTRX) meets the physical layer requirements of the ISO11898-2:2016 High-Speed Controller Area Network (CAN) specification providing an interface between the CAN bus and the CAN protocol controller (MultiCAN+).

The CANTRX provides following features:

- Compliant to ISO11898-2:2016
- Compliant to classical CAN and CAN-FD up to 5 Mbps
- Fulfills CAN interfaces (v1.2) OEM hardware requirements
- Supports four operating modes:
  - Off mode
  - Normal mode (Rx, Tx)
  - Receive-only mode (Rx-only)
  - Sleep mode for low-power operation. Wake-up time: <100 µs typ. Wake-up pattern recognition
- Interfaces with multiple hosts:
  - MultiCAN+ protocol controller
  - UART
  - GPIO
  - Timer GPT12 and Timer2 (T2)
- CAN bus bias control. Ideal passive behavior when unpowered.
- Diagnostics:
  - CAN supply (VCAN) undervoltage supervision
  - CAN bus dominant timeout
  - CAN transceiver input (TxD) dominant timeout
- Overtemperature protection

12.2 Block diagram

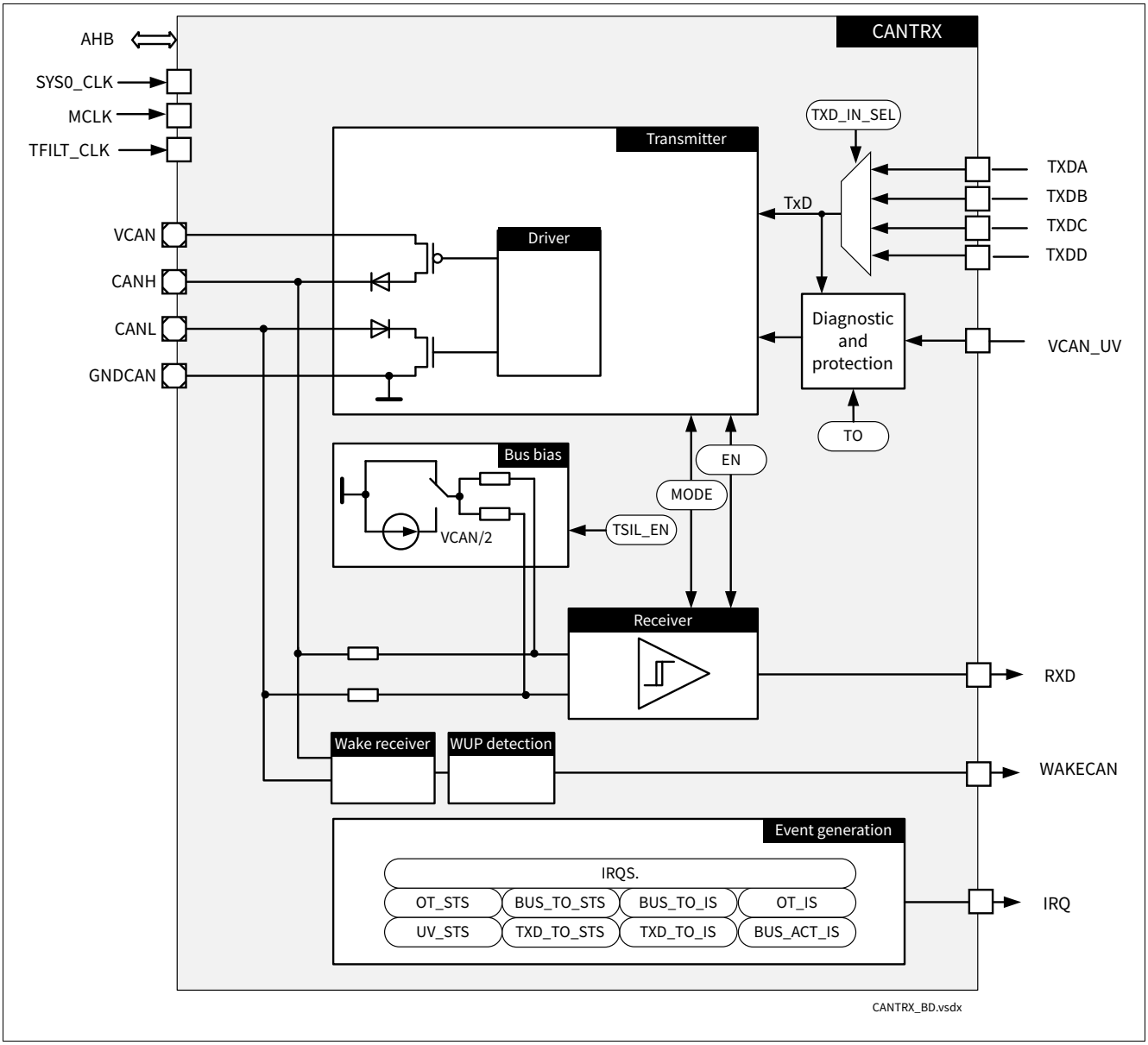


Figure 168 Block diagram CANTRX

12.3 Toplevel signals

The CANTRX module interconnects as specified in **Table 151** and **Table 152**.

Table 151 CANTRX power domain

Signal	Direction	Description	From/To
VCAN	Power	5 V CAN transceiver supply	PMU via external I/O
VMSUP	Power	1.5 V digital peripherals always-on supply	PMU
VDD5V_PD	Power	5 V peripherals always-on supply	PMU
GNDCAN	Power	VCAN supply ground	PMU via external I/O

Table 152 CANTRX control signals

Signal	Direction	Description	From/To
CANH	Input/output	CAN high bus signal	External I/O
CANL	Input/output	CAN low bus signal	External I/O
TxDA	Input	CAN transmit data	MultiCAN+
TxDB	Input	CAN transmit data	UART0
TxDC	Input	CAN transmit data	GPT12
TxDD	Input	CAN transmit data	GPIO
RxD	Output	CAN receive data	MultiCAN+, T2, GPT12, GPIO, UART0
WAKECAN	Output	Wake-up on CAN detected signal	PMU
VCAN_UV	Input	VCAN undervoltage signal	ADC2
SYS0_CLK	Input	Peripheral reference clock	SCU
TFILT_CLK	Input	Digital filter reference clock	SCU
MCLK	Input	Silence time reference clock	PMU
IRQ	Output	Interrupt service request signal	SCU
AHB slave bus I/F	Input/Output	Slave AHB-lite bus interface	AHB

Table 153 CANTRX reset signals

Signal	Direction	Description	From/To
RESET_TYPE_2	Input	Wake-up logic reset	PMU
RESET_TYPE_5	Input	Logic and SFRs reset	PMU

12.4 Interrupts

The CANTRX module provides one interrupt node (NVIC IRQ0 or IRQ1) to signal the following events:

- Bus dominant timeout (see IRQS register)
- TXD dominant timeout (see IRQS register)
- CAN overtemperature (see IRQS register)
- Bus active during CAN sleep mode (see IRQS register)

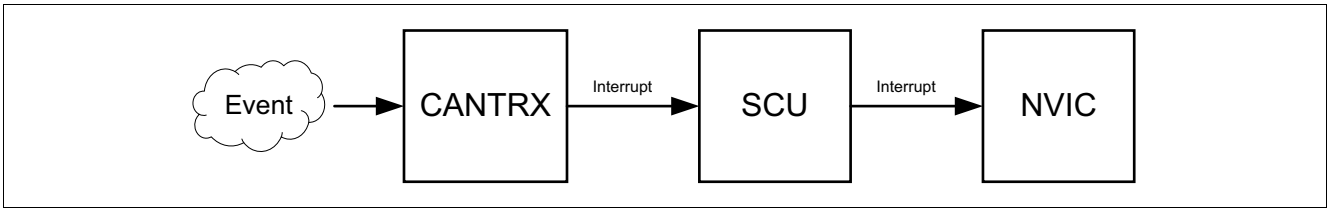


Figure 169 Event handling

To register the occurrence of an event, the CANTRX sets an Interrupt Status bit (xxx_IS). To enable an interrupt source, set high the corresponding Interrupt Enable bit (xxx_IEN). The interrupt events are propagated to the SCU for further processing and distribution. Interrupts will be generated (according to the event source) regardless of the Interrupt Status bit (xxx_IS). Conversely, an interrupt source can be masked by setting low the corresponding Interrupt Enable bit (xxx_IEN).

To clear an Interrupt Status bit (xxx_IS), set high the corresponding Interrupt Status Clear bit (xxx_ISC).

For debug and test purposes, the CANTRX offers the possibility to emulate an interrupt source. Set high the corresponding Interrupt Status Set bit (xxx_ISS).

In addition to the interrupt scheme, the CANTRX also provides event status bits (xxx_STS) indicating the occurrence of a trigger event (e.g. CAN module overtemperature).

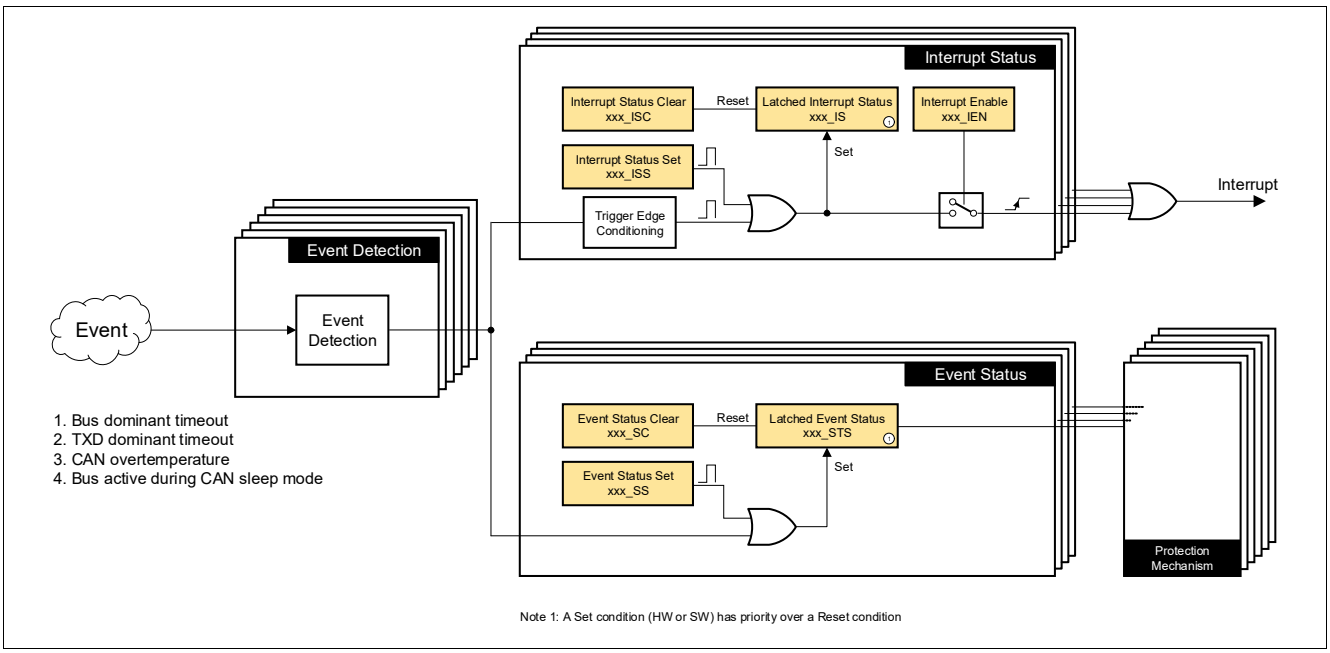


Figure 170 Interrupt and status registers

For more details, refer to the CANTRX register description at the end of the chapter (see IRQS, IRQCLR, IRQSET and IRQEN).

12.5 Operation mode behavior

Table 154 CANTRX state-of-operation vs. SoC system power modes

Reset	• RESET_TYPE_2 initializes the CAN wake configuration settings • RESET_TYPE_5 initializes the CANTRX configuration settings
Power-up **Power-down**	• After power-up the CANTRX is disabled • At power-down the CANTRX is reset
Active mode	• The CANTRX may be configured to off mode, CAN normal, CAN receive-only or CAN sleep mode
Stop mode	• The CANTRX is automatically set to CAN sleep mode or off mode when entering Stop SoC system power mode. The CANTRX wake capability depends on the PMU wake settings. For more details, please refer to **Figure 172** and **Figure 173**. When entering Stop mode, the bus bias and the CANTRX internal clock are active until $t_{SILENCE}$ has expired.
Sleep mode	• The CANTRX is automatically set to CAN sleep mode or off mode when entering Sleep SoC system power mode. The CANTRX wake capability depends on the PMU wake settings. For more details, please refer to **Figure 172** and **Figure 173**. When entering Sleep mode, the bus bias and the CANTRX internal clock are active until $t_{SILENCE}$ has expired.
Fail-sleep mode	• The CANTRX is automatically set to off mode when entering Fail-sleep SoC system power mode. Wake-up on CAN is not available

12.6 CANTRX state-of-operation

The Controller Area Network (CAN) transceiver provides High-Speed (up to 5 Mbps) differential mode data transmission and reception. The CAN transceiver provides a physical interface between a CAN protocol controller and the physical bus lines compatible to ISO 11898-2:2016 and SAE J2284.

The CAN transceiver offers low-power modes to reduce current consumption. To support software diagnostic functions, a CAN receive-only mode is implemented. The CAN transceiver is designed to provide excellent passive behavior when switched-off (mixed networks, clamp 15/30 applications).

A wake-up from CAN sleep mode is possible via a message on the bus. The MCU can be powered-down (or idling) and will be woken up by CAN bus activities. The transceiver can be configured to be wake capable in order to save current and to ensure a safe transition from normal to sleep mode (i.e. avoiding message loss).

The CAN transceiver is designed to withstand severe automotive application conditions.

12.6.1 CAN-FD support

CAN-FD stands for ‘CAN with Flexible Data Rate’. This standard is based on the well-established CAN protocol as specified in the ISO 11898-1. CAN-FD leverages the CAN bus arbitration method. The benefit being that the bit rate can be increased by switching to a shorter bit time at the end of the arbitration process and then returning to the longer bit time at the CRC delimiter, before the receivers transmit their acknowledge bits.

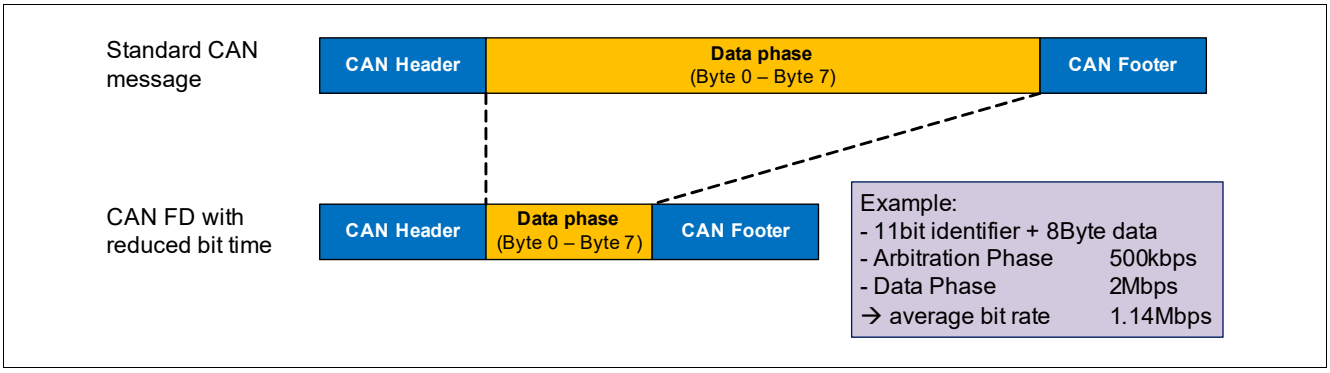


Figure 171 Bit rate increase with CAN-FD vs. standard CAN

In addition, the effective data rate is increased by allowing longer data fields, up to 64 data bytes compared to 8 data bytes in the standard CAN protocol.

12.7 CANTRX operating modes

The CAN transceiver (CANTRX) has four main operating modes: off mode, normal mode, receive-only mode and sleep mode. The selection of the operating mode is done via the CANTRX CTRL register and the PMU WAKE_CTRL.CAN_WAKE_EN bit. For more details, please refer to **Table 155**, **Figure 172** and **Figure 173**.

Table 155 CANTRX operating modes

CTRL.EN	CTRL.MODE	WAKE_CTRL.CAN_WAKE_EN	Description
0	XX	0	CAN off
0	XX	1	CAN sleep mode (wake capable)
1	X0	0	CAN off
1	X0	1	CAN sleep mode (wake capable)
1	01	X	CAN receive-only mode
1	11	X	CAN normal mode

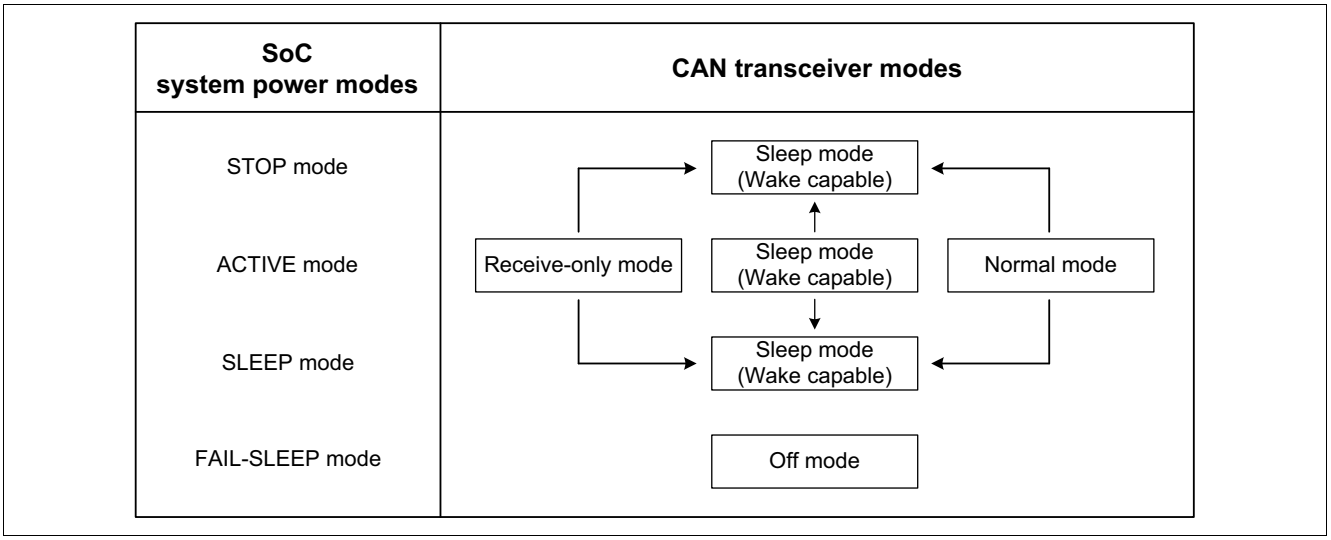


Figure 172 CANTRX modes vs. SoC system power modes (PMU WAKE_CTRL.CAN_WAKE_EN bit = 1)

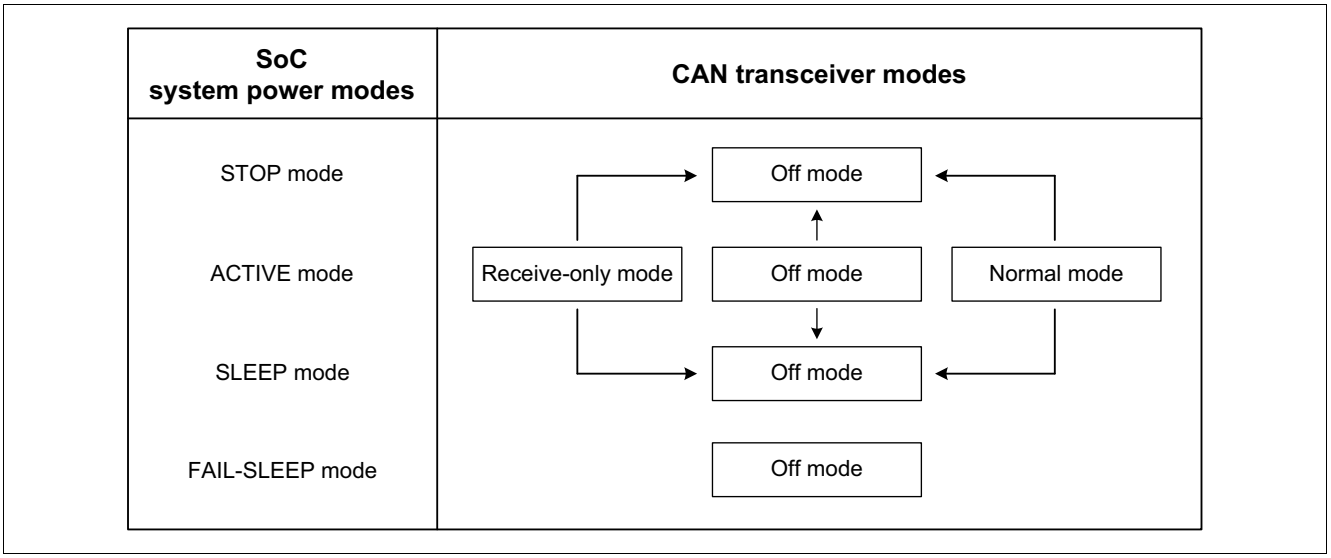


Figure 173 CANTRX modes vs. SoC system power modes (PMU WAKE_CTRL.CAN_WAKE_EN bit = 0)

12.7.1 CAN off mode

The CAN off mode is the default operating mode after the SoC powers up. It is available in all SoC system power modes. The CAN off mode terminates bus communication activities (CANTRX output stage in tristate). Any bus activity (e.g. a wake-up event) will be ignored.

12.7.2 CAN normal mode

The CANTRX is enabled via the SFR interface. The CAN normal mode is designed for normal data transmission and reception within a HS-CAN network. This mode is available in the ACTIVE SoC system power mode.

- Transmission:
  The bus driver switches the CANH/L output stage to transfer the TxD input signal on the CAN bus lines.
- Reception:
  Analog CAN bus signals are converted into digital signals via the differential input receiver. The received data are available on the RxD output.
- Enabling sequence:
  The CANTRX requires an enabling time ($t_{CAN,EN}$) before a message can be sent on the bus or received from the bus. The TxD signal can only be set to dominant after the enabling time. If this is not the case, the TxD signal needs to be set back to recessive until the enabling time is over. Only the next dominant bit will be transmitted on the bus. Conversely, the RxD output activates after the enabling time has expired.
  Figure 174 and **Figure 175** show different scenarios and details on how to enable CAN communication.
- Reduced electromagnetic emission:
  The bus driver controls CANH/L slopes symmetrically in order to reduce electromagnetic emissions (EME).

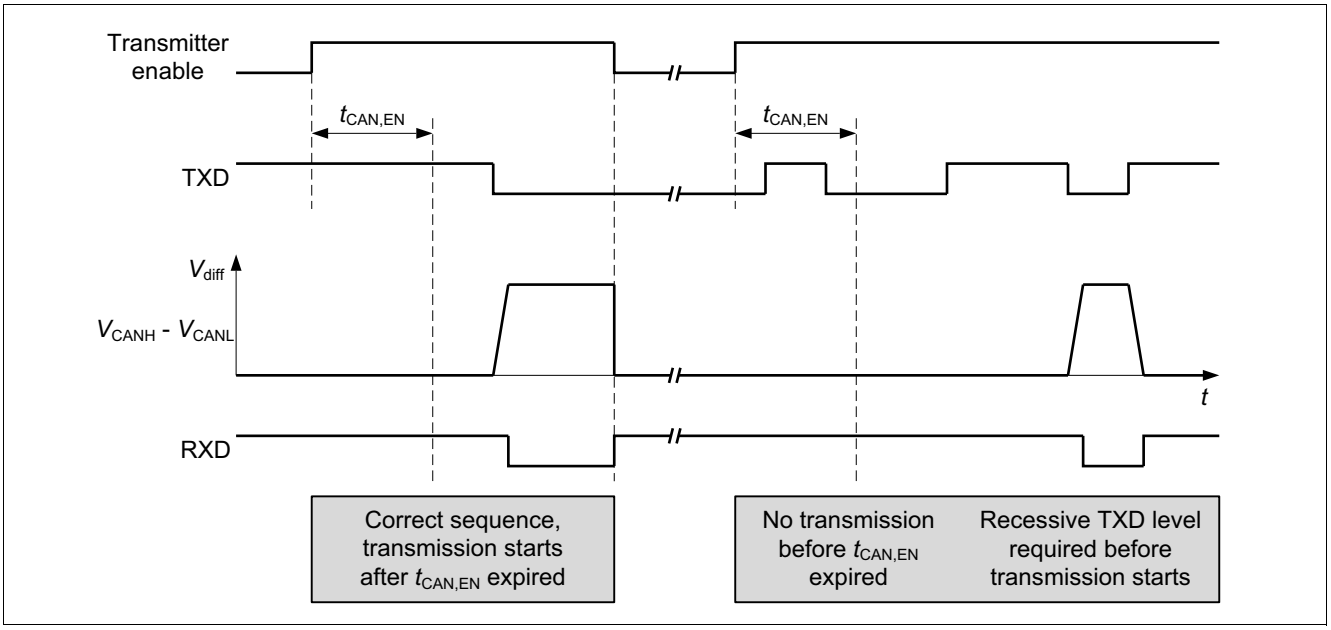


Figure 174 CANTRX activation sequence

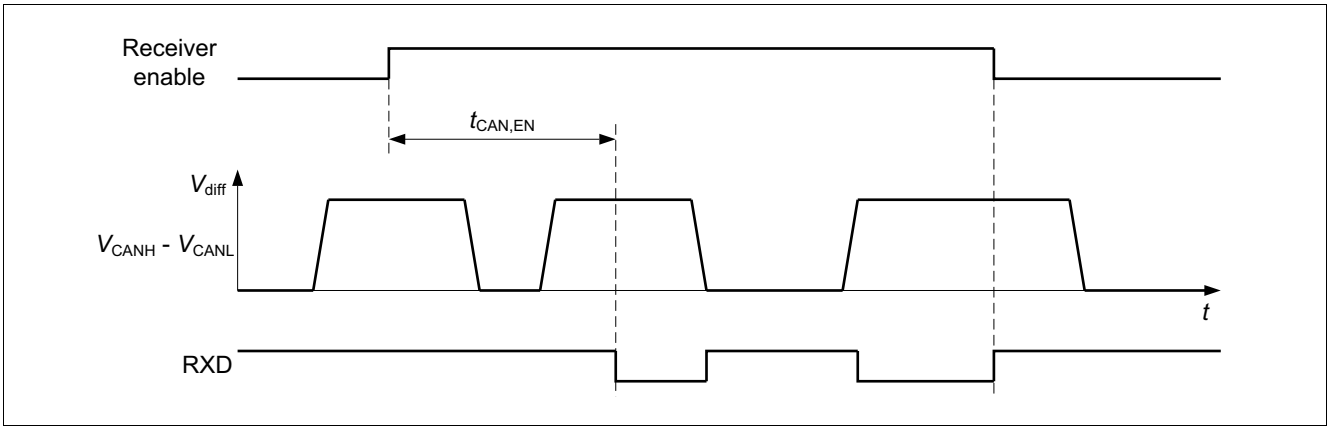


Figure 175 CANTRX activation sequence

12.7.3 CAN receive-only mode

In CAN receive-only mode (Rx-only), the driver stage is de-activated but reception is still operational. This mode is accessible via SFRs. This mode is available in the ACTIVE SoC system power mode.

Note: The transceiver is still working properly in receive-only mode even if VCAN is not available because of an independent receiver supply.

12.7.4 CAN sleep mode (wake capable)

The CAN sleep mode can be used to monitor bus activities with low-power consumption (CAN wake-up enabled, via PMU WAKE_CTRL.CAN_WAKE_EN bit = 1). This mode is available in the ACTIVE, STOP and SLEEP SoC system power mode.

Without wake-up capability the CAN transceiver is completely turned off and ignores all bus activities (output stage in tristate). The CAN transceiver wake-up capability is set and controlled by the PMU.

As shown in **Figure 176**, a wake-up pattern (WUP) on the bus is recognized by two consecutive dominant bus levels longer than a t_{Wake1} duration (i.e. 1.2 µs typ. wake-up time) and shorter than a t_{Wake2} time (2 ms typ.), separated by a recessive bus level longer than a t_{Wake1} duration and shorter than a t_{Wake2} time.

To avoid missing CAN wake-up events when transitioning into SLEEP system power mode, the transceiver starts monitoring for a valid CAN wake-up pattern (WUP) even though the transceiver may still be configured in CAN normal mode. For more details, please refer to **Figure 177**.

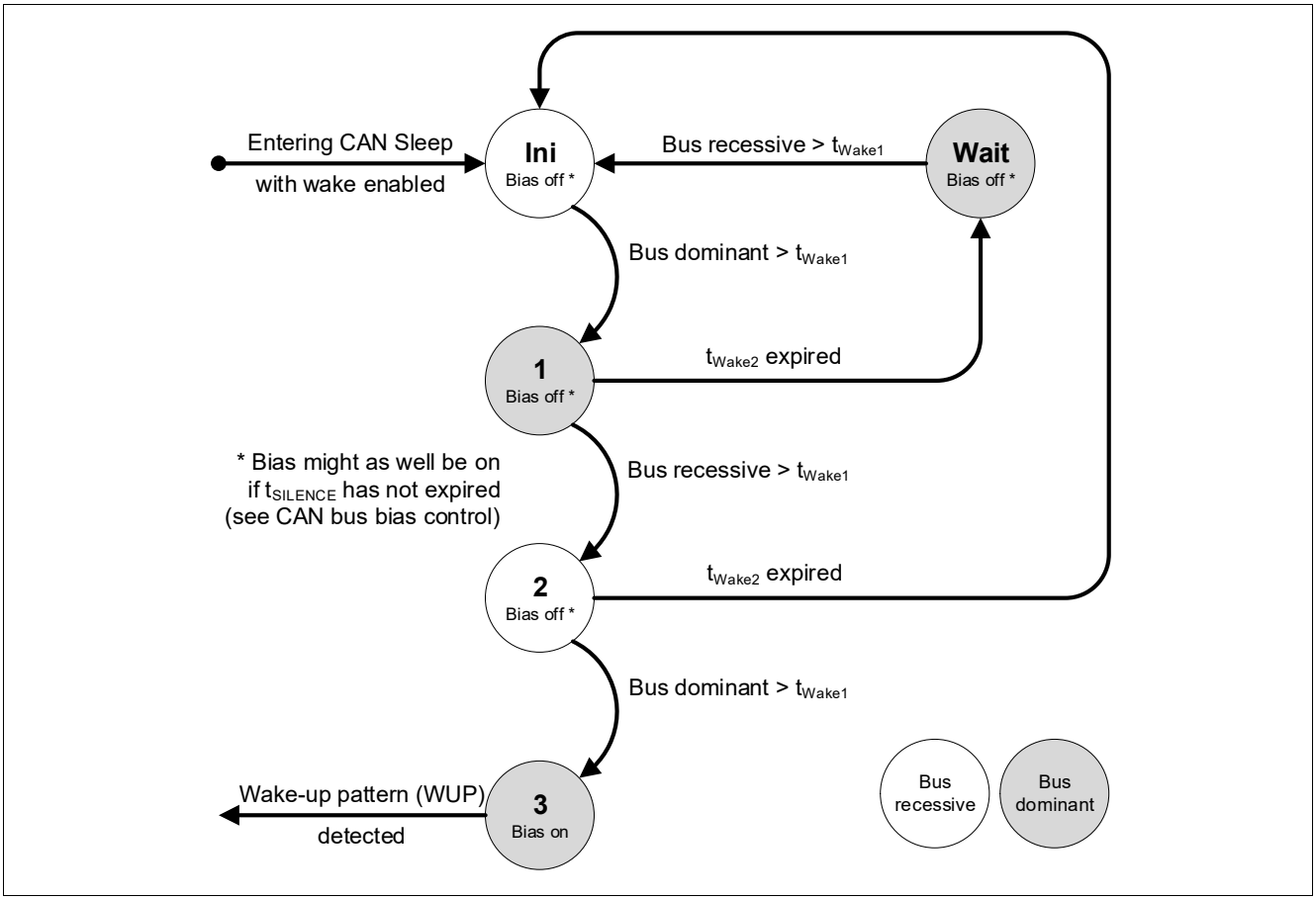


Figure 176 CAN wake-up pattern detection according to ISO 11898-2

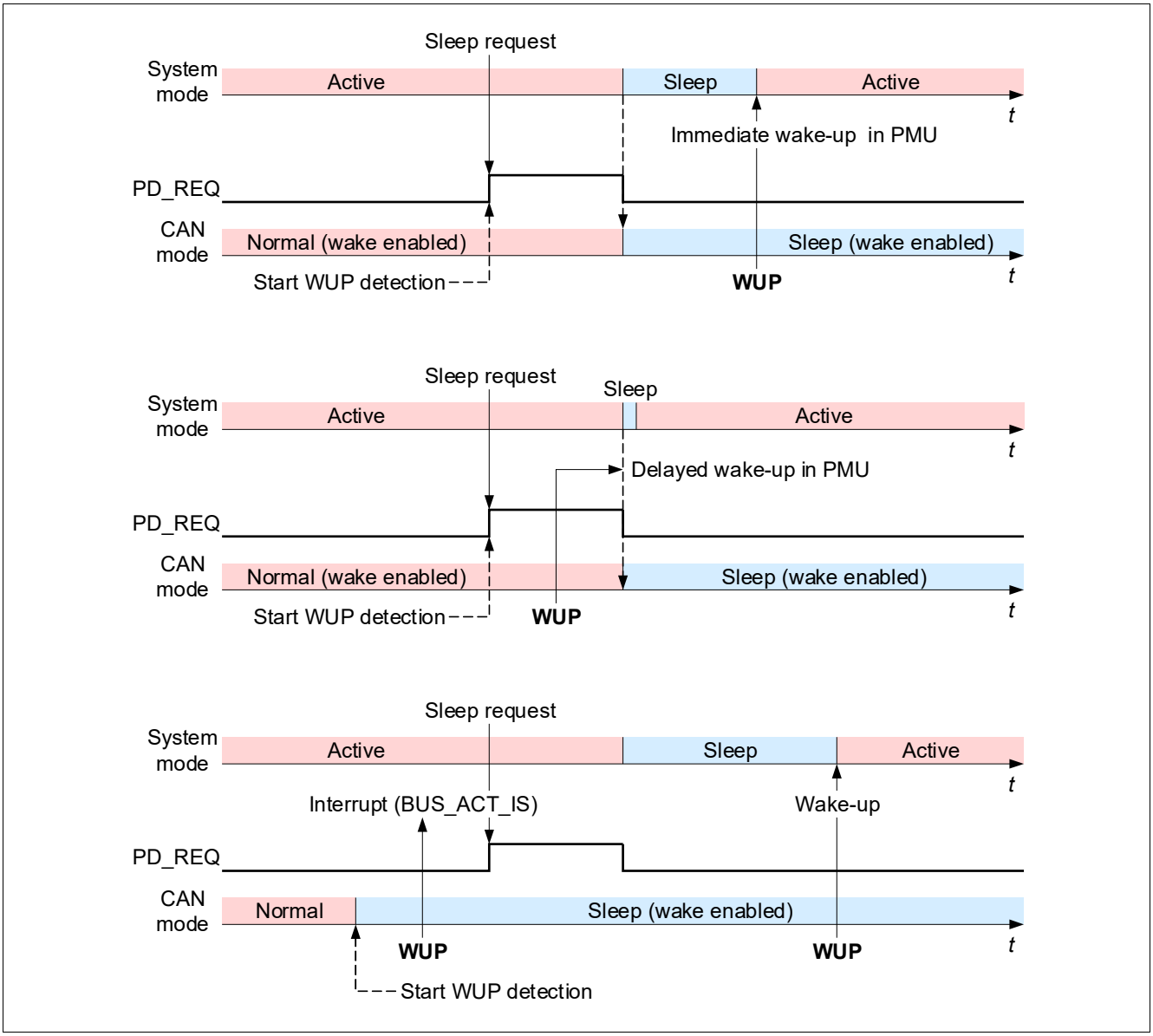


Figure 177 CAN wake-up timing related to system power modes

12.8 CAN bus bias control

The CAN bus lines CANH/L need a proper common mode voltage when the bus is in recessive state.

Therefore a voltage source is implemented that generates four different bus bias voltages for different states of the CAN transceiver, depending on the CAN mode and the bus activity. In accordance to the CAN configuration, four bus biasing types are possible.

Table 156 CAN bus bias voltages

Bias mode	Bias voltage (V_{CANL/H_NM})	Bias source	CANTRX mode	VCAN state
Bias on	VCAN / 2	VCAN and resistor divider	CAN normal, CAN receive-only	VCAN in regulation
Bias on	2.5 V typ.	Derived from VDD5V_PD and resistors	CAN normal, CAN receive-only, CAN sleep after wake-up, CAN sleep during $t_{SILENCE}$	VCAN UV or VCAN off
Bias off	0 V	GNDCAN	CAN sleep (wake-up enabled)	
No bias	Floating	None	CAN off	

*Note: When entering CAN sleep mode with wake enabled, the biasing is switched off (i.e. connected to GNDCAN) after $t_{SILENCE}$ has expired and no bus activity. For more details, please refer to **Figure 178**.*

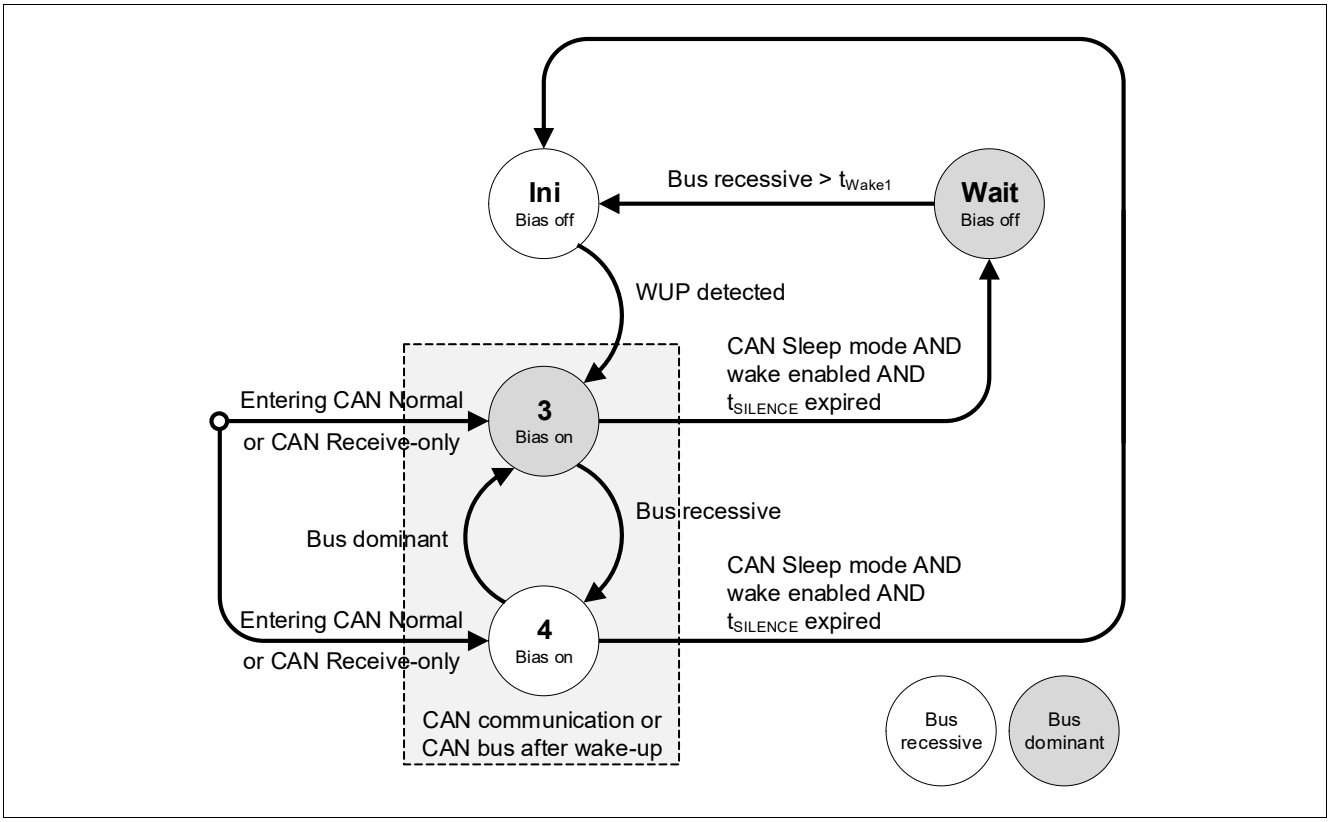


Figure 178 CAN bus biasing

12.9 TxD dominant timeout detection and protection mechanisms

A permanent dominant level on the TxD input signal would block the CAN communication because the bus would be held in dominant state.

In order to prevent this scenario the transmitter disconnects the TxD input from the bus driver after a certain time ($t > t_{TXDCAN\_TO}$) and keeps the transmitter recessive. The other CAN nodes on the bus would still be able to communicate, and the receiver could continue to listen to the bus traffic.

The transmitter stage is activated again after the dominant timeout condition is removed and the IRQS.TXD_TO_STS status flag is cleared.

This functionality can be enabled/disabled via the CTRL.EN_TXD_TO bit. Status information can be gathered via the IRQS.TXD_TO_STS and IRQS.TXD_TO_IS.

12.10 Bus dominant timeout detection

The CAN communication is blocked when the bus is always dominant. A timeout detection is implemented to recognize this case, to store it in a register and/or to generate an interrupt. When the receiver output is dominant for longer than $t_{BUS\_CAN\_TO}$ in CAN normal mode or CAN receive-only mode, the CAN transceiver sets the status bits IRQS.BUS_TO_STS and IRQS.BUS_TO_IS.

Note: The CAN bus dominant timeout detection has no effect on the function of the CAN transceiver, it just indicates the occurrence of the event.

12.11 CAN transceiver HW monitors

The ADC2 supervises the VCAN voltage and informs the CAN transceiver about the status. In case VCAN is too low, the CAN transceiver needs to take actions to keep the CAN communication uncompromised.

In case of an undervoltage event:

- The transceiver disables the transmitter
- The transceiver switches the CAN bus bias source. V_{CANL/H_NM} is derived from the internal power-down supply VDD5V_PD (instead of VCAN)
- The transceiver indicates the VCAN undervoltage event in the IRQS.UV_STS read-only register bit

Note: Should the VCAN supply recover, the ADC2 CMPSTAT.CMP_LO_STS.3 status bit should be cleared and the transmitter stage automatically enables again. To re-activate TxD it is necessary to ensure a recessive state after the failure is cleared. The TxD signal can only be set to dominant after the enabling time ($t_{CAN,EN}$) expired.

The temperature of the CAN transceiver is monitored locally. In case of an overtemperature event:

- The status bits IRQS.OT_STS and IRQS.OT_IS are set
- The transmitter keeps the bus in recessive state

Note: Should the overtemperature condition disappear, the IRQS.OT_STS can be cleared and the transmitter automatically resumes operation. To re-activate TxD it is necessary to ensure a recessive state after the failure is cleared.

12.12 Programmer's guide

12.12.1 CANTRX module initialization

The following steps should be considered to initialize properly (including diagnostics) the CAN transceiver.

Table 157 Basic configuration steps to active the CAN interface

Step 1	CTRL register	• Enable CANTRX: CTRL.EN bit = 1 • Configure CANTRX operating mode: CTRL.MODE bit = 11 (normal mode) • CANTRX transmitter de-activated due to TxD dominant timeout: CTRL.EN_TXD_TO bit = 1 (optional) • Configure CANTRX TxD source: CTRL.TXD_IN_SEL = 00 (MultiCAN+ protocol controller)

12.13 Register description CANTRX

12.13.1 CANTRX Address Maps

Table 158 Register Address Space - CANTRX

Module	Base Address	End Address	Note
CANTRX	$4800C000_H$	$4800FFFF_H$	

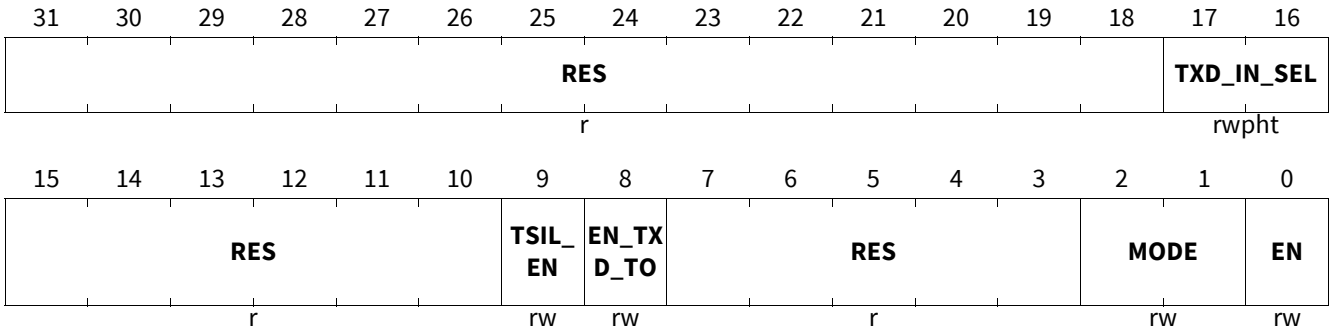
Table 159 Register Overview - CANTRX (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CTRL	CAN transceiver control	0000_H	610
IRQS	CAN transceiver interrupt status	0004_H	611
IRQCLR	CAN transceiver interrupt status register clear	0008_H	612
IRQSET	CAN transceiver interrupt status register set	$000C_H$	613
IRQEN	CAN transceiver interrupt enable	0010_H	614

12.13.2 CANTRX Registers

CAN transceiver control

CTRL
CAN transceiver control **(0000_H)** **RESET_TYPE_5 Value: $0000\ 0300_H$**



Field	Bits	Type	Description
EN	0	rw	**CAN transceiver enable** 0_B **DISABLE**, CAN transceiver disable 1_B **ENABLE**, CAN transceiver enable
MODE	2:1	rw	**CAN mode control** 00_B **OFF**, Sleep mode 01_B **REC_ONLY**, Receive-only mode 10_B **NU**, Not used (selects sleep mode) 11_B **NORMAL**, Normal mode, transmit and receive
RES	7:3, 15:10, 31:18	r	**Reserved** Always read as 0
EN_TXD_TO	8	rw	**Enable transmitter deactivation due to TXD dominant timeout** 0_B **DISABLE**, TXD dominant timeout disable 1_B **ENABLE**, TXD dominant timeout enable
TSIL_EN	9	rw	**Enable tsilence counter** 0_B **DISABLE**, Tsilence is omitted 1_B **ENABLE**, Tsilence is used for bus bias control
TXD_IN_SEL	17:16	rwpht	**TXD input selector** *Note: These bits can only be changed if EN = 0 or MODE[1] = 0, i.e. the transceiver is not in CAN normal mode.* 00_B **TXDA**, CAN protocol controller TXD 01_B **TXDB**, UART0 TXD 10_B **TXDC**, GPT12 T3OUT 11_B **TXDD**, P0.3

Register description CANTRX

CAN transceiver interrupt status

IRQS
CAN transceiver interrupt status **(0004_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES											UV_STS	RES	OT_STS	TXD_TO_STS	BUS_TO_STS
r											r	r	rhxr	rhxr	rhxr

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES												BUS_ACT_IS	OT_IS	TXD_TO_IS	BUS_TO_IS
r												rhxre	rhxre	rhxre	rhxre

Field	Bits	Type	Description
BUS_TO_IS	0	rhxre	**Bus dominant timeout interrupt status** 0_B **BUS_OK**, No bus dominant timeout interrupt occurred 1_B **BUS_TO**, Bus dominant timeout nterrupt occurred
TXD_TO_IS	1	rhxre	**TXD dominant timeout interrupt status** 0_B **TXD_OK**, No TXD dominant timeout interrupt occurred 1_B **TXD_TO**, TXD dominant timeout interrupt occurred
OT_IS	2	rhxre	**CAN overtemperature interrupt status** 0_B **NO_OT**, No OT interrupt occurred 1_B **OT**, OT interrupt occurred
BUS_ACT_IS	3	rhxre	**Bus active during CAN sleep interrupt status** This interrupt only occurs when the system is in active mode, before a sleep/stop request 0_B **NO_WUP**, No wake-up interrupt occurred 1_B **WUP**, Wake-up interrupt occurred
RES	15:4, 19, 31:21	r	**Reserved** Always read as 0
BUS_TO_STS	16	rhxr	**Bus dominant timeout status** BUS_TO_STS can only be cleared when the receiver output becomes recessive or the CAN mode is switched to sleep mode. 0_B **BUS_OK**, No bus dominant timeout occurred 1_B **BUS_TO**, Bus dominant timeout occurred
TXD_TO_STS	17	rhxr	**TXD dominant timeout status** TXD_TO_STS can only be cleared when the TXD input becomes recessive or the CAN normal mode is left. 0_B **TXD_OK**, No TXD dominant timeout occurred 1_B **TXD_TO**, TXD dominant timeout occurred

Register description CANTRX

Field	Bits	Type	Description
OT_STS	18	rhxr	**CAN overtemperature status** OT_STS can only be cleared when the overtemperature condition is not present anymore or the transmitter is disabled. 0_B **NO_OT**, No overtemperature occurred 1_B **OT**, Overtemperature occurred
UV_STS	20	r	**CAN supply undervoltage status** This bit only replicates the ADC2 status signal VCAN_UV, which is a latched status and not a real-time value. 0_B **VCAN_OK**, CAN supply voltage nominal 1_B **VCAN_UV**, CAN supply voltage below threshold

CAN transceiver interrupt status register clear

IRQCLR
CAN transceiver interrupt status register clear (0008_H) **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES													OT_SC	TXD_TO_SC	BUS_TO_SC
r													w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES												BUS_ACT_ISC	OT_ISC	TXD_TO_ISC	BUS_TO_ISC
r												w	w	w	w

Field	Bits	Type	Description
BUS_TO_ISC	0	w	**Bus dominant timeout interrupt status clear** 0_B **KEEP**, Keep interrupt status 1_B **CLEAR**, Clear interrupt status
TXD_TO_ISC	1	w	**TXD dominant timeout interrupt status clear** 0_B **KEEP**, Keep interrupt status 1_B **CLEAR**, Clear interrupt status
OT_ISC	2	w	**CAN overtemperature interrupt status clear** 0_B **KEEP**, Keep interrupt status 1_B **CLEAR**, Clear interrupt status
BUS_ACT_ISC	3	w	**Bus active during CAN sleep interrupt status clear** 0_B **KEEP**, Keep interrupt status 1_B **CLEAR**, Clear interrupt status
RES	15:4, 31:19	r	**Reserved** Always read as 0
BUS_TO_SC	16	w	**Bus dominant timeout status clear** 0_B **KEEP**, Keep status 1_B **CLEAR**, Clear status

Register description CANTRX

Field	Bits	Type	Description
TXD_TO_SC	17	w	**TXD dominant timeout status clear** 0_B **KEEP**, Keep status 1_B **CLEAR**, Clear status
OT_SC	18	w	**CAN overtemperature status clear** 0_B **KEEP**, Keep status 1_B **CLEAR**, Clear status

CAN transceiver interrupt status register set

IRQSET
CAN transceiver interrupt status register set (000C$_H$) **RESET_TYPE_5 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES													OT_SS	TXD_TO_SS	BUS_TO_SS
r													w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES												BUS_ACT_ISS	OT_ISS	TXD_TO_ISS	BUS_TO_ISS
r												w	w	w	w

Field	Bits	Type	Description
BUS_TO_ISS	0	w	**Bus dominant timeout interrupt status set** 0_B **KEEP**, Keep interrupt status 1_B **SET**, Set interrupt status
TXD_TO_ISS	1	w	**TXD dominant timeout interrupt status set** 0_B **KEEP**, Keep interrupt status 1_B **SET**, Set interrupt status
OT_ISS	2	w	**CAN overtemperature interrupt status set** 0_B **KEEP**, Keep interrupt status 1_B **SET**, Set interrupt status
BUS_ACT_ISS	3	w	**Bus active during CAN sleep interrupt status set** 0_B **KEEP**, Keep interrupt status 1_B **SET**, Set interrupt status
RES	15:4, 31:19	r	**Reserved** Always read as 0
BUS_TO_SS	16	w	**Bus dominant timeout status set** 0_B **KEEP**, Keep status 1_B **SET**, Set status
TXD_TO_SS	17	w	**TXD dominant timeout status set** 0_B **KEEP**, Keep status 1_B **SET**, Set status

Register description CANTRX

Field	Bits	Type	Description
OT_SS	18	w	**CAN overtemperature status set** 0_B **KEEP**, Keep status 1_B **SET**, Set status

CAN transceiver interrupt enable

IRQEN
CAN transceiver interrupt enable **(0010_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES												BUS_ACT_IEN	OT_IEN	TXD_TO_IEN	BUS_TO_IEN
r												rw	rw	rw	rw

Field	Bits	Type	Description
BUS_TO_IEN	0	rw	**Bus dominant timeout interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
TXD_TO_IEN	1	rw	**TXD dominant timeout interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
OT_IEN	2	rw	**CAN overtemperature interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
BUS_ACT_IEN	3	rw	**Bus active during CAN sleep interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
RES	31:4	r	**Reserved** Always read as 0

13 General Purpose Ports (GPIO)

13.1 Features overview

The TLE989x/TLE988x has many digital port pins, which can be used as General Purpose I/Os (GPIO) and are connected to the on-chip peripheral units.

The TLE989x/TLE988x has port pins organized into three parallel ports: Port 0 (P0), Port 1 (P1) and Port 2 (P2). Each port pin has a pair of internal pull-up and pull-down devices that can be individually enabled or disabled. P0 and P1 are bidirectional and can be used as general purpose input/output (GPIO) or to perform alternate input/output functions for the on-chip peripherals. When configured as an output, the open drain mode can be selected. On Port 2 (P2) analog inputs are shared with general purpose digital inputs.

The GPIOs provide a generic and flexible software and hardware interface for all standard digital I/Os. Each port has the same software interfaces for the operation as General Purpose I/O and it further provides the connectivity to the on-chip peripherals and the control for the pad characteristics. :

The GPIO provides following features:

- Bidirectional port features (P0, P1)
  - P0/P1: Configurable pin direction
  - P0/P1: Configurable pull-up/pull-down devices
  - P0/P1: Configurable open drain mode
  - P0/P1: Configurable drive strength
  - P0/P1: Configurable slew rate
  - P0/P1: Transfer of data through digital inputs and outputs (general purpose I/O)
  - P0/P1: Possible readback of pin status when GPIO is configured as output (short detection)
  - P0/P1: Alternate input/output for on-chip peripherals
  - P0/P1: up to seven alternate output connections from peripherals selectable. The three configuration bits per GPIO are located in the same register
  - P0/P1: separate input and output registers, which allows to evaluate the input while the output is active (plausibility check)
  - P0/P1: dedicated output modification registers (enabling set, clear, toggle functionality) to avoid read-modify-write operations
  - P0/P1: default configuration during bootup is input and floating (no pull-up/pull-down)
- Analog port features (P2)
  - P2: Configurable pull-up/pull-down devices
  - P2: Transfer of data through digital inputs
  - P2: Alternate inputs for on-chip peripherals
  - P2: Disabling of digital input stage on shared analog input ports
- Wake-up feature
  - Configurable wake-up from stop mode via GPIO (rising edge only, falling edge only, both edges), e.g. for wake-up on a sensor signal
  - In total 6 port pins can be configured for wake-up, freely selectable from P0/P1/P2
  - No lost wake-up, independent from the timing relationship between the wake-up event and the stop-entry command

13.2 Block diagram

Port 0 and Port 1

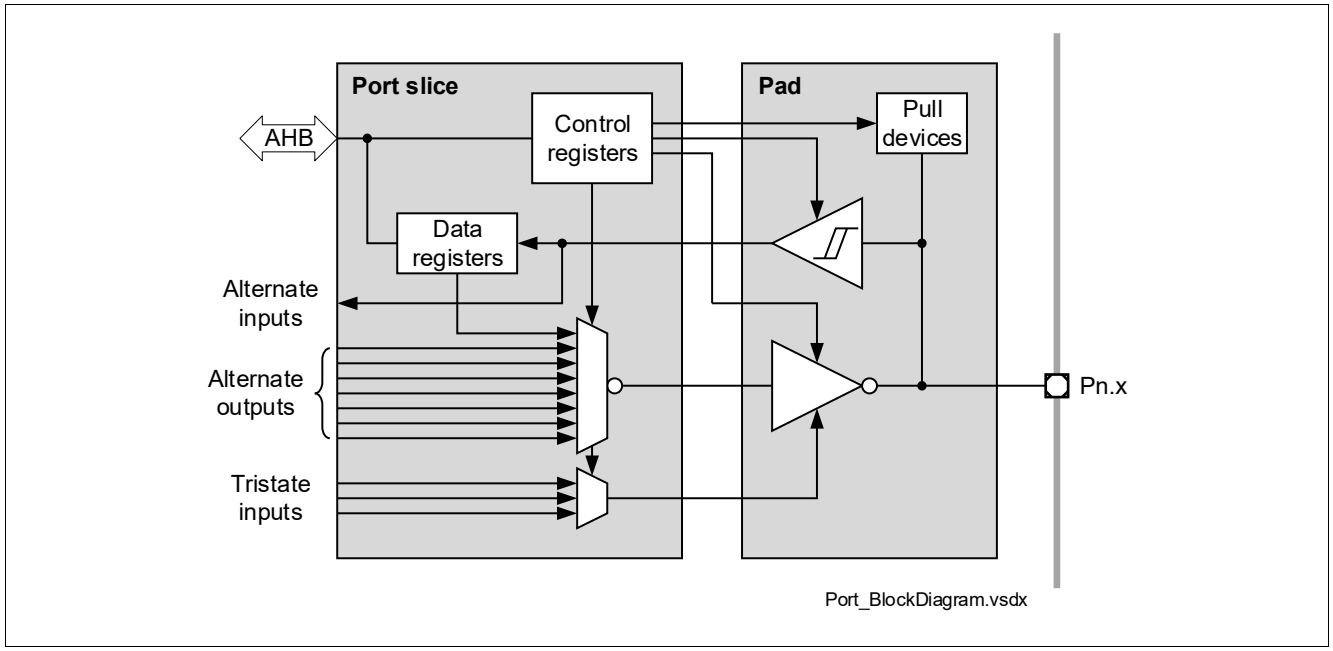


Figure 179 General structure of bidirectional port

Port 2

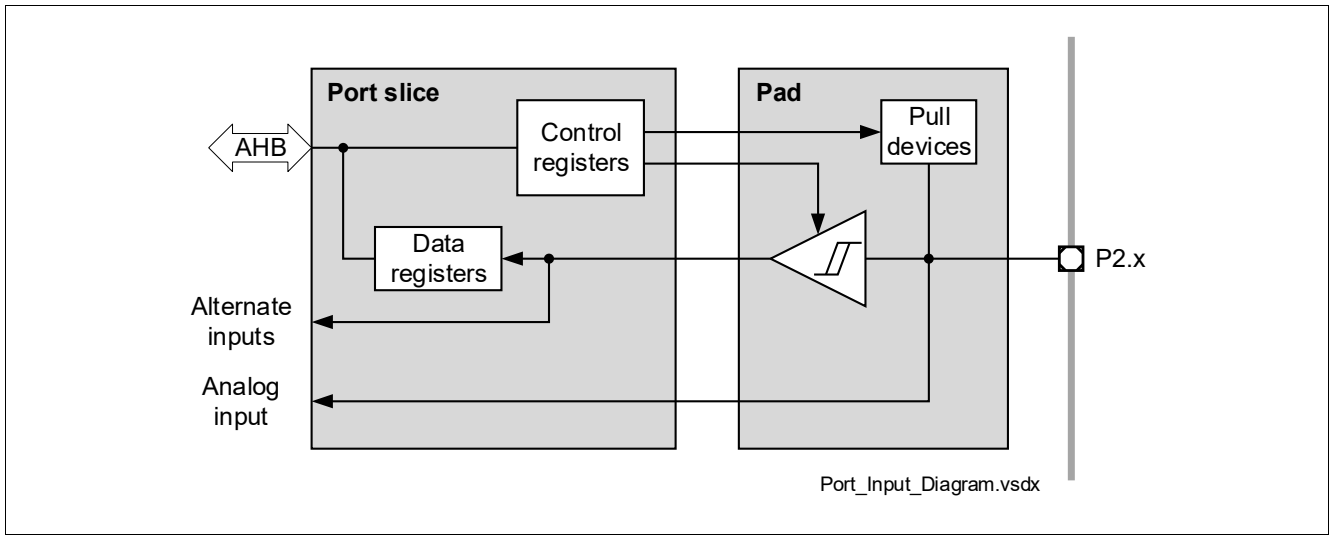


Figure 180 General structure of input port

13.3 Toplevel signals

The connectivity of the port pins is summarized in chapter **Alternate functions at GPIOs**.

13.4 Interrupts

The port pins do not generate interrupts themselves.

*Note: For external interrupts (EXINT) via GPIO, please refer to the SCU chapter **Interrupt and exception control**.*

13.5 Operation mode behavior

The GPIOs follow the operation modes according to **Table 160**.

Table 160 Operation mode behavior GPIO

Reset	• RESET_TYPE_3 resets the GPIO configuration settings
Power-up / Power-down	• After power-up the GPIO are set as input by default • At power-down the GPIO are reset by RESET_TYPE_3
Active mode	• The GPIO are set as input by default • The GPIO can be configured by setting the AHB registers
Stop mode	• The GPIO keep the configurations when entering stop mode • The GPIO can be used as wake-up source (to be configured in PMU) • The GPIO configurations are kept after wake-up
Sleep mode / Fail-sleep-mode	• The GPIO are not supplied in sleep mode • The configurations of the GPIO are reset by RESET_TYPE_3 • The GPIO start as input after wake-up, like after power-up
Fail-safe state	• The GPIO behave like in the selected mode, no difference to normal operation

13.6 GPIO data and control registers

The main function of the GPIOs is to provide multiple digital I/Os (Port0.x and Port1.x) and multiple analog and digital Inputs (Port2.x). Each port function can be configured by control and data registers.

An overview of the GPIO registers is shown in **Table 161**.

Table 161 GPIO register overview

Register name	Type	Function	Available in P0	Available in P1	Available in P2
Pn_OUT	Data	Output data	P0	P1	–
Pn_OMR	Data	Output modification	P0	P1	–
Pn_IN	Data	Input data	P0	P1	P2
Pn_DIR	Control	Direction	P0	P1	–
Pn_INDIS	Control	Input disable	–	–	P2
Pn_OD	Control	Open drain	P0	P1	–
Pn_PUD	Control	Pull-up/pull-down device	P0	P1	P2
Pn_ALTSELx	Control	Alternate output select	P0	P1	–
Pn_POCON	Control	Output driver control (port control)	P0	P1	–

More details about the GPIO registers:

- Output data registers **P0_OUT**, **P1_OUT**:
  If a port pin is used as a general purpose output, the output data is written to register Pn_OUT.POx (pin x of port n). When directly writing to Pn_OUT, all data bits are written at the same time.
- Output modification registers **P0_OMR**, **P1_OMR**:
  When only certain output data bits inside Pn_OUT should be set to ‘1’ or reset to ‘0’, register Pn_OMR.PSx or Pn_OMR.PRx can be used for this. With this register only those output data bits are changed that have their bits set in Pn_OMR. In case both Pn_OMR.PSx and Pn_OMR.PRx of one pin are set, the corresponding Pn_OUT.POx is toggled, i.e. inverted from its previous state.
  With this approach unintended bit changes can be avoided, which might occur in a read-modify-write-sequence, when a bit changes between read and write-back (e.g. in an interrupt service routine).
- Input data registers **P0_IN**, **P1_IN**, **P2_IN**:
  The digital state of the port pin is shown in Pn_IN.PIx.
  For P0 and P1, this register can also be used to read back the port data of a pin that is configured as an output (e.g. as check for possible shorts outside of the device). For P0 and P1 the input driver is always enabled.
- Direction control registers **P0_DIR**, **P1_DIR**:
  The direction of a bidirectional port pin can be controlled as input or output.
  Default setting after reset is input.
- Input disable register **P2_INDIS**:
  This register disables the digital input path of the port 2 pin. This setting is recommended for analog measurements to avoid disturbances.
- Open drain control registers **P0_OD**, **P1_OD**:
  Each port pin in output mode can be configured as an open drain driver instead of a push-pull driver. In this case only the pull-down driver is active when driving a ‘0’. For driving a ‘1’ the port pin relies on the setting of the internal weak pull-up or some external pull-up circuit.

- Pull-up/pull-down device control registers **P0_PUD**, **P1_PUD**, **P2_PUD**:
  Internal weak pull-up or pull-down devices can be applied to each port pin. They can be used to support a default input state when an input pin is not driven from the outside or with an open drain output.
  Default setting after reset is floating. In case a defined state of the GPIO is not ensured by external circuitry, it is recommended to enable the internal weak pull-up or pull-down.
- Alternate output select control registers **P0_ALTSEL0**, **P0_ALTSEL1**, **P1_ALTSEL0**:
  An output port pin is not limited to direct register control, but can also be used by a module as an output channel. The selection is done with Pn_ALTSELx. 3 bits allow to select between the output register or 1 out of 7 alternate functions. The connection overview can be found in chapter “Alternate functions summary”.
- Output driver control registers **P0_POCON**, **P1_POCON**:
  This register controls the output driver strength of a port pin to adapt to different external loads. 4 driver strengths are available.

13.7 Port pins as wake-up source

Up to 6 port pins can be configured as a wake-up source to wake-up the system from the stop mode (e.g. wake-up on a sensor signal).

For each of the intended wake-up sources PMU.**WAKE_CTRL**.GPIO_WAKE_EN has to be set accordingly.

The selection of the corresponding port pin is done in PMU.**WAKE_GPIO_CTRLx (x=0-5)**.INP.

For each of the wake-up sources, rising and/or falling edge detection can be configured individually in PMU.**WAKE_GPIO_CTRLx (x=0-5)**.RI or FA, respectively.

The wake-up feature can be enabled continuously (default) or with “cyclic sense” (wake-up detection only enabled during cyclic-sense-on-time) by setting PMU.**WAKE_GPIO_CTRLx (x=0-5)**.CYC.

A filter time can be configured in PMU.**WAKE_FILT_CTRL**.GPIO_FT.

The wake-up status is handled in PMU.**WAKE_STS**.GPIO, PMU.**WAKE_STS_SET**.GPIO_SET and PMU.**WAKE_STS_CLR**.GPIO_CLR.

More details about the wake-up feature are described in the PMU chapter “**Wake control**“.

13.8 Register description GPIO

13.8.1 GPIO Address Maps

Table 162 Register Address Space - GPIO

Module	Base Address	End Address	Note
GPIO	48030000_H	$48033FFF_H$	AHB bus interface for general purpose I/O ports

Table 163 Register Overview - GPIO (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
P0_OUT	Port 0 output register	0000_H	621
P0_OMR	Port 0 output modification register	0004_H	621
P0_IN	Port 0 input register	0008_H	622
P0_DIR	Port 0 direction control	$000C_H$	622
P0_OD	Port 0 open drain control	0010_H	623
P0_PUD	Port 0 pull-up/pull-down control	0014_H	623
P0_ALTSEL0	Port 0 alternate output select 0	0018_H	624
P0_ALTSEL1	Port 0 alternate output select 1	$001C_H$	625
P0_POCON	Port 0 output driver control	0020_H	625
P1_OUT	Port 1 output register	0024_H	626
P1_OMR	Port 1 output modification register	0028_H	626
P1_IN	Port 1 input register	$002C_H$	627
P1_DIR	Port 1 direction control	0030_H	627
P1_OD	Port 1 open drain control	0034_H	628
P1_PUD	Port 1 pull-up/pull-down control	0038_H	628
P1_ALTSEL0	Port 1 alternate output select 0	$003C_H$	629
P1_POCON	Port 1 output driver control	0040_H	629
P2_IN	Port 2 input register	0044_H	630
P2_INDIS	Port 2 input disable register	0048_H	630
P2_PUD	Port 2 pull-up/pull-down control	$004C_H$	631

13.8.2 GPIO Registers

13.8.2.1 AHB bus interface for general purpose I/O ports

Port 0 output register

P0_OUT
Port 0 output register **(0000_H)** **RESET_TYPE_3 Value: 0000 0000_H**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					PO10	PO9	PO8	PO7	PO6	PO5	PO4	PO3	PO2	PO1	PO0
r					rwx	rwx	rwx	rwx	rwx	rwx	rwx	rwx	rwx	rwx	rwx

Field	Bits	Type	Description
POx (x=0-10)	x	rwx	**Output bit** 0_B **0**, Output value = 0 1_B **1**, Output value = 1
RES	31:11	r	**Reserved** Always read as 0

Port 0 output modification register

P0_OMR
Port 0 output modification register **(0004_H)** **RESET_TYPE_3 Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES					PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
r					w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					PS10	PS9	PS8	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
r					w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-10)	x	w	**Output set bit** The bits PSx and PRx set, reset or toggle the corresponding bit in the port output register POx. PRx = 0 / PSx = 0 : Bit POx is not changed PRx = 0 / PSx = 1 : Bit POx is set PRx = 1 / PSx = 0 : Bit POx is reset PRx = 1 / PSx = 1 : Bit POx is toggled

Register description GPIO

Field	Bits	Type	Description
RES	15:11, 31:27	r	**Reserved** Always read as 0
PRx (x=0-10)	x+16	w	**Output reset bit**

Port 0 input register

P0_IN
Port 0 input register **(0008_H)** **RESET_TYPE_3 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					**PI10**	**PI9**	**PI8**	**PI7**	**PI6**	**PI5**	**PI4**	**PI3**	**PI2**	**PI1**	**PI0**
r					r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
PIx (x=0-10)	x	r	**Input bit** 0_B **0**, Input value = 0 1_B **1**, Input value = 1
RES	31:11	r	**Reserved** Always read as 0

Port 0 direction control

P0_DIR
Port 0 direction control **($000C_H$)** **RESET_TYPE_3 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					**DIR10**	**DIR9**	**DIR8**	**DIR7**	**DIR6**	**DIR5**	**DIR4**	**DIR3**	**DIR2**	**DIR1**	**DIR0**
r					rw	rwpht	rwpht	rwpht	rwpht	rwpht	rwpht	rw	rw	rw	rw

Field	Bits	Type	Description
DIRx (x=0-3)	x	rw	**Direction control bit** 0_B **IN**, Pin direction is input 1_B **OUT**, Pin direction is output
DIRx (x=4-9)	x	rwpht	**Direction control bit** 0_B **IN**, Pin direction is input 1_B **OUT**, Pin direction is output

Field	Bits	Type	Description
DIR10	10	rw	**Direction control bit** 0_B **IN**, Pin direction is input 1_B **OUT**, Pin direction is output
RES	31:11	r	**Reserved** Always read as 0

Port 0 open drain control

P0_OD

Port 0 open drain control **(0010_H)** **RESET_TYPE_3 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					**OD10**	**OD9**	**OD8**	**OD7**	**OD6**	**OD5**	**OD4**	**OD3**	**OD2**	**OD1**	**OD0**
r					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ODx (x=0-10)	x	rw	**Open drain control bit** 0_B **PP**, Push-pull driver for active 0 and 1 state 1_B **OD**, Open drain driver for active 0 state only
RES	31:11	r	**Reserved** Always read as 0

Port 0 pull-up/pull-down control

P0_PUD

Port 0 pull-up/pull-down control **(0014_H)** **RESET_TYPE_3 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES					**PUDSEL10**	**PUDSEL9**	**PUDSEL8**	**PUDSEL7**	**PUDSEL6**	**PUDSEL5**	**PUDSEL4**	**PUDSEL3**	**PUDSEL2**	**PUDSEL1**	**PUDSEL0**
r					rw	rwpht	rwpht	rwpht	rwpht	rwpht	rwpht	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					**PUDEN10**	**PUDEN9**	**PUDEN8**	**PUDEN7**	**PUDEN6**	**PUDEN5**	**PUDEN4**	**PUDEN3**	**PUDEN2**	**PUDEN1**	**PUDEN0**
r					rw	rwpht	rwpht	rwpht	rwpht	rwpht	rwpht	rw	rw	rw	rw

Field	Bits	Type	Description
PUDENx (x=0-3)	x	rw	**Pull-up/pull-down enable bit** 0_B **DIS**, Pull-up/pull-down devices are disabled 1_B **EN**, Pull-up or pull-down device is enabled

Register description GPIO

Field	Bits	Type	Description
PUDENx (x=4-9)	x	rwpht	**Pull-up/pull-down enable bit** 0_B **DIS**, Pull-up/pull-down devices are disabled 1_B **EN**, Pull-up or pull-down device is enabled
PUDEN10	10	rw	**Pull-up/pull-down enable bit** 0_B **DIS**, Pull-up/pull-down devices are disabled 1_B **EN**, Pull-up or pull-down device is enabled
RES	15:11, 31:27	r	**Reserved** Always read as 0
PUDSELx (x=0-3)	x+16	rw	**Pull-up/pull-down select bit** 0_B **PD**, Pull-down device is selected 1_B **PU**, Pull-up device is selected
PUDSELx (x=4-9)	x+16	rwpht	**Pull-up/pull-down select bit** 0_B **PD**, Pull-down device is selected 1_B **PU**, Pull-up device is selected
PUDSEL10	26	rw	**Pull-up/pull-down select bit** 0_B **PD**, Pull-down device is selected 1_B **PU**, Pull-up device is selected

Port 0 alternate output select 0

P0_ALTSEL0
Port 0 alternate output select 0 **(0018_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

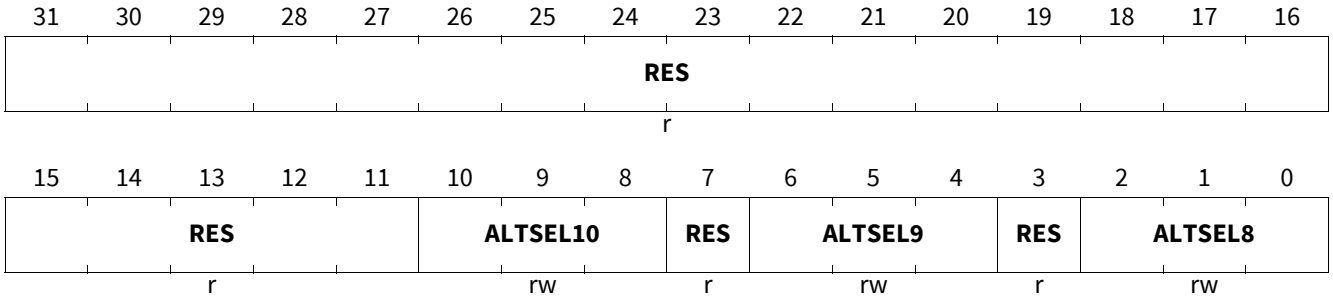
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES	**ALTSEL7**			**RES**	**ALTSEL6**			**RES**	**ALTSEL5**			**RES**	**ALTSEL4**		
r	rw			r	rw			r	rw			r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	**ALTSEL3**			**RES**	**ALTSEL2**			**RES**	**ALTSEL1**			**RES**	**ALTSEL0**		
r	rw			r	rw			r	rw			r	rw		

Field	Bits	Type	Description
ALTSELx (x=0-7)	4*x+2:4*x	rw	**Alternate output select** 000_B **GPIO**, Register-controlled GPIO selected 001_B **ALT1**, Alternate output selected … 111_B **ALT7**, Alternate output selected
RES	3, 7, 11, 15, 19, 23, 27, 31	r	**Reserved** Always read as 0

Port 0 alternate output select 1

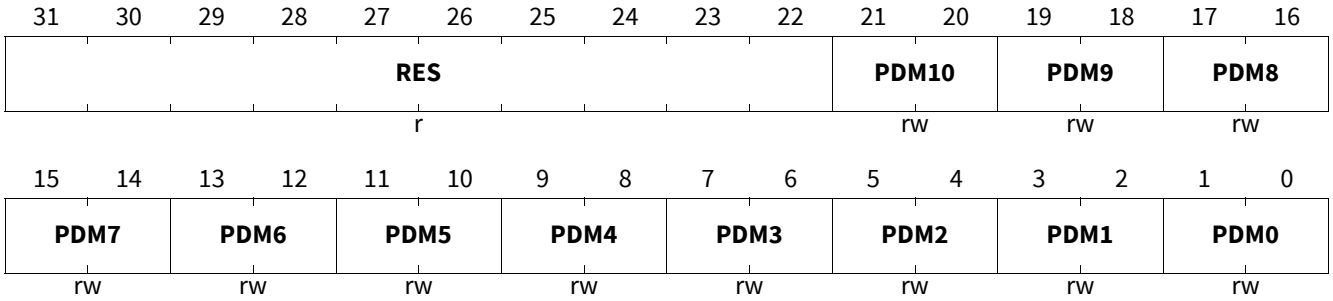
P0_ALTSEL1
Port 0 alternate output select 1 **($001C_H$)** **RESET_TYPE_3 Value: 0000 0000_H**



Field	Bits	Type	Description
ALTSELx (x=8-10)	4*x-30:4*x-32	rw	**Alternate output select** 000_B **GPIO**, Register-controlled GPIO selected 001_B **ALT1**, Alternate output selected … 111_B **ALT7**, Alternate output selected
RES	3, 7, 31:11	r	**Reserved** Always read as 0

Port 0 output driver control

P0_POCON
Port 0 output driver control **(0020_H)** **RESET_TYPE_3 Value: 0000 0000_H**



Field	Bits	Type	Description
PDMx (x=0-10)	2*x+1:2*x	rw	**Output driver mode control** 00_B **S**, Strong driver and sharp edge 01_B **SM**, Strong driver and medium edge 10_B **M**, Medium driver 11_B **W**, Weak driver
RES	31:22	r	**Reserved** Always read as 0

Register description GPIO

Port 1 output register

P1_OUT
Port 1 output register **(0024_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES											PO4	PO3	PO2	PO1	PO0
r											rwx	rwx	rwx	rwx	rwx

Field	Bits	Type	Description
POx (x=0-4)	x	rwx	**Output bit** 0_B **0**, Output value = 0 1_B **1**, Output value = 1
RES	31:5	r	**Reserved** Always read as 0

Port 1 output modification register

P1_OMR
Port 1 output modification register **(0028_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES											PR4	PR3	PR2	PR1	PR0
r											w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES											PS4	PS3	PS2	PS1	PS0
r											w	w	w	w	w

Field	Bits	Type	Description
PSx (x=0-4)	x	w	**Output set bit** The bits PSx and PRx set, reset or toggle the corresponding bit in the port data register POx. PRx = 0 / PSx = 0 : Bit POx is not changed PRx = 0 / PSx = 1 : Bit POx is set PRx = 1 / PSx = 0 : Bit POx is reset PRx = 1 / PSx = 1 : Bit POx is toggled
RES	15:5, 31:21	r	**Reserved** Always read as 0
PRx (x=0-4)	x+16	w	**Output reset bit**

Register description GPIO

Port 1 input register

P1_IN
Port 1 input register **($002C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES											PI4	PI3	PI2	PI1	PI0
r											r	r	r	r	r

Field	Bits	Type	Description
PIx (x=0-4)	x	r	**Input bit** 0_B **0**, Input value = 0 1_B **1**, Input value = 1
RES	31:5	r	**Reserved** Always read as 0

Port 1 direction control

P1_DIR
Port 1 direction control **(0030_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES											DIR4	DIR3	DIR2	DIR1	DIR0
r											rwpht	rwpht	rw	rw	rw

Field	Bits	Type	Description
DIRx (x=0-2)	x	rw	**Direction control bit** 0_B **IN**, Pin direction is input 1_B **OUT**, Pin direction is output
DIRx (x=3-4)	x	rwpht	**Direction control bit** 0_B **IN**, Pin direction is input 1_B **OUT**, Pin direction is output
RES	31:5	r	**Reserved** Always read as 0

Port 1 open drain control

P1_OD
Port 1 open drain control **(0034_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES											OD4	OD3	OD2	OD1	OD0
r											rw	rw	rw	rw	rw

Field	Bits	Type	Description
ODx (x=0-4)	x	rw	**Open drain control bit** 0_B **PP**, Push-pull driver for active 0 and 1 state 1_B **OD**, Open drain driver for active 0 state only
RES	31:5	r	**Reserved** Always read as 0

Port 1 pull-up/pull-down control

P1_PUD
Port 1 pull-up/pull-down control **(0038_H)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES											PUDSEL4	PUDSEL3	PUDSEL2	PUDSEL1	PUDSEL0
r											rwpht	rwpht	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES											PUDEN4	PUDEN3	PUDEN2	PUDEN1	PUDEN0
r											rwpht	rwpht	rw	rw	rw

Field	Bits	Type	Description
PUDENx (x=0-2)	x	rw	**Pull-up/pull-down enable bit** 0_B **DIS**, Pull-up/pull-down devices are disabled 1_B **EN**, Pull-up or pull-down device is enabled
PUDENx (x=3-4)	x	rwpht	**Pull-up/pull-down enable bit** 0_B **DIS**, Pull-up/pull-down devices are disabled 1_B **EN**, Pull-up or pull-down device is enabled
RES	15:5, 31:21	r	**Reserved** Always read as 0

Register description GPIO

Field	Bits	Type	Description
PUDSELx (x=0-2)	x+16	rw	**Pull-up/pull-down select bit** 0_B **PD**, Pull-down device is selected 1_B **PU**, Pull-up device is selected
PUDSELx (x=3-4)	x+16	rwpht	**Pull-up/pull-down select bit** 0_B **PD**, Pull-down device is selected 1_B **PU**, Pull-up device is selected

Port 1 alternate output select 0

P1_ALTSEL0

Port 1 alternate output select 0 **(003C$_H$)** **RESET_TYPE_3 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES													ALTSEL4		
r													rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	ALTSEL3			RES	ALTSEL2			RES	ALTSEL1			RES	ALTSEL0		
r	rw			r	rw			r	rw			r	rw		

Field	Bits	Type	Description
ALTSELx (x=0-4)	4*x+2:4*x	rw	**Alternate output select** 000_B **GPIO**, Register-controlled GPIO selected 001_B **ALT1**, Alternate output selected … 111_B **ALT7**, Alternate output selected
RES	3, 7, 11, 15, 31:19	r	**Reserved** Always read as 0

Port 1 output driver control

P1_POCON

Port 1 output driver control **(0040$_H$)** **RESET_TYPE_3 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						PDM4		PDM3		PDM2		PDM1		PDM0	
r						rw		rw		rw		rw		rw	

Field	Bits	Type	Description
PDMx (x=0-4)	2*x+1:2*x	rw	**Output driver mode control** 00_B **S**, Strong driver and sharp edge 01_B **SM**, Strong driver and medium edge 10_B **M**, Medium driver 11_B **W**, Weak driver
RES	31:10	r	**Reserved** Always read as 0

Port 2 input register

P2_IN
Port 2 input register **(0044_H)** **RESET_TYPE_3 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						**PI9**	**PI8**	**PI7**	**PI6**	**PI5**	**PI4**	**PI3**	**PI2**	**PI1**	**PI0**
r						r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
PIx (x=0-9)	x	r	**Input bit** 0_B **0**, Input value = 0 1_B **1**, Input value = 1
RES	31:10	r	**Reserved** Always read as 0

Port 2 input disable register

P2_INDIS
Port 2 input disable register **(0048_H)** **RESET_TYPE_3 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						**INDIS 9**	**INDIS 8**	**INDIS 7**	**INDIS 6**	**INDIS 5**	**INDIS 4**	**INDIS 3**	**INDIS 2**	**INDIS 1**	**INDIS 0**
r						rwpht	rwpht	rwpht	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
INDISx (x=0-6)	x	rw	**Input disable control bit** 0_B **INEN**, Digital input receiver is enabled 1_B **INDIS**, Digital input receiver is disabled, no disturbance of analog input
INDISx (x=7-9)	x	rwpht	**Input disable control bit** 0_B **INEN**, Digital input receiver is enabled 1_B **INDIS**, Digital input receiver is disabled, no disturbance of analog input
RES	31:10	r	**Reserved** Always read as 0

Port 2 pull-up/pull-down control

P2_PUD
Port 2 pull-up/pull-down control **($004C_H$)** **RESET_TYPE_3 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						PUDSEL9	PUDSEL8	PUDSEL7	PUDSEL6	PUDSEL5	PUDSEL4	PUDSEL3	PUDSEL2	PUDSEL1	PUDSEL0
r						rwpht	rwpht	rwpht	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						PUDEN9	PUDEN8	PUDEN7	PUDEN6	PUDEN5	PUDEN4	PUDEN3	PUDEN2	PUDEN1	PUDEN0
r						rwpht	rwpht	rwpht	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PUDENx (x=0-6)	x	rw	**Pull-up/pull-down enable bit** 0_B **DIS**, Pull-up/pull-down devices are disabled 1_B **EN**, Pull-up or pull-down device is enabled
PUDENx (x=7-9)	x	rwpht	**Pull-up/pull-down enable bit** 0_B **DIS**, Pull-up/pull-down devices are disabled 1_B **EN**, Pull-up or pull-down device is enabled
RES	15:10, 31:26	r	**Reserved** Always read as 0
PUDSELx (x=0-6)	x+16	rw	**Pull-up/pull-down select bit** 0_B **PD**, Pull-down device is selected 1_B **PU**, Pull-up device is selected
PUDSELx (x=7-9)	x+16	rwpht	**Pull-up/pull-down select bit** 0_B **PD**, Pull-down device is selected 1_B **PU**, Pull-up device is selected

14 High-Voltage Monitor Input (MON)

14.1 Features overview

The High-Voltage Monitor Input (MON) is dedicated to monitor external voltage levels above or below the threshold V_{MONth}. Each MONx input can further be used to create a wake-up event by detecting a level change while crossing the threshold. This applies to any system operation mode, especially for the power-down modes. Furthermore each MONx input can be sampled by the ADC1 as an analog input. It can also be used as a high-voltage PWM input signal for Timer21.

The MON provides following features:

- High-voltage inputs with threshold voltage 3 V (typ.)
- Wake-up capability for system Stop mode and system Sleep mode
- Edge sensitive wake-up feature configurable for transitions from low to high, high to low or both directions
- MON input level status (high/low) can be read in Active mode
- MON inputs can also be evaluated with ADC1 in Active mode to sense high-voltage signals, using adjustable threshold values for interrupt generation
- Selectable pull-up and pull-down current sources available
- MON inputs can be selected as inputs for Timer21 to evaluate high-voltage PWM input signals
- MON inputs can be used for edge detection (interrupt generation for transitions from low to high, high to low or both directions)

14.2 Block diagram

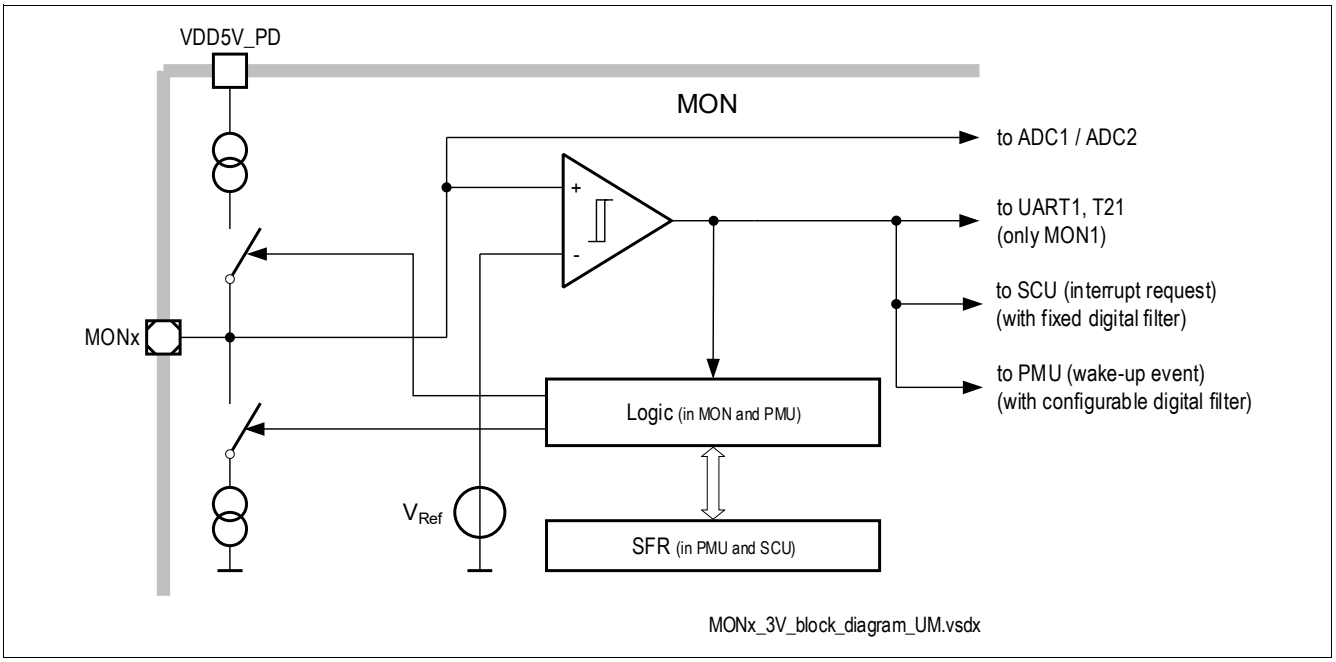


Figure 181 Block diagram MON

14.3 Toplevel signals

The MONx modules have power supplies as specified in **Table 164** and connections as specified in **Table 165**.

Table 164 MONx power supplies

Signal	Direction	Description
VDD5V_PD	Power	5 V always-on supply
GND_A	Power	Supply ground

Table 165 MONx connections

Signal	Direction	Description	From/To
MON1	Input	High-voltage monitor input voltage	MON1 pin
MON2 (only in 64-pin-device)	Input	High-voltage monitor input voltage	MON2 pin
MON3 (only in 64-pin-device)	Input	High-voltage monitor input voltage	MON3 pin
MON1O	Output	to UART1 as HV UART	UART1
MON1O	Output	to T21 as capture input	T21
MON1A	Output	to ADC1, ATT_TYP0, uncalibrated	ADC1
MON1A	Output	to ADC1, ATT_TYP1	ADC1
MON2A (only in 64-pin-device)	Output	to ADC1, ATT_TYP0, uncalibrated	ADC1
MON2A (only in 64-pin-device)	Output	to ADC1, ATT_TYP1	ADC1
MON3A (only in 64-pin-device)	Output	to ADC1, ATT_TYP0, uncalibrated	ADC1
MON3A (only in 64-pin-device)	Output	to ADC1, ATT_TYP1	ADC1
MON1A	Output	to ADC2	ADC2
MON2A (only in 64-pin-device)	Output	to ADC2	ADC2
MON3A (only in 64-pin-device)	Output	to ADC2	ADC2
MON1O	Output	to interrupt request	SCU
MON2O (only in 64-pin-device)	Output	to interrupt request	SCU
MON3O (only in 64-pin-device)	Output	to interrupt request	SCU
MON1O	Output	to PMU as wake-up event	PMU
MON2O (only in 64-pin-device)	Output	to PMU as wake-up event	PMU
MON3O (only in 64-pin-device)	Output	to PMU as wake-up event	PMU

14.4 Interrupts

Interrupts of the MONx modules are handled in the SCU.

Figure 182 shows the interrupt structure.

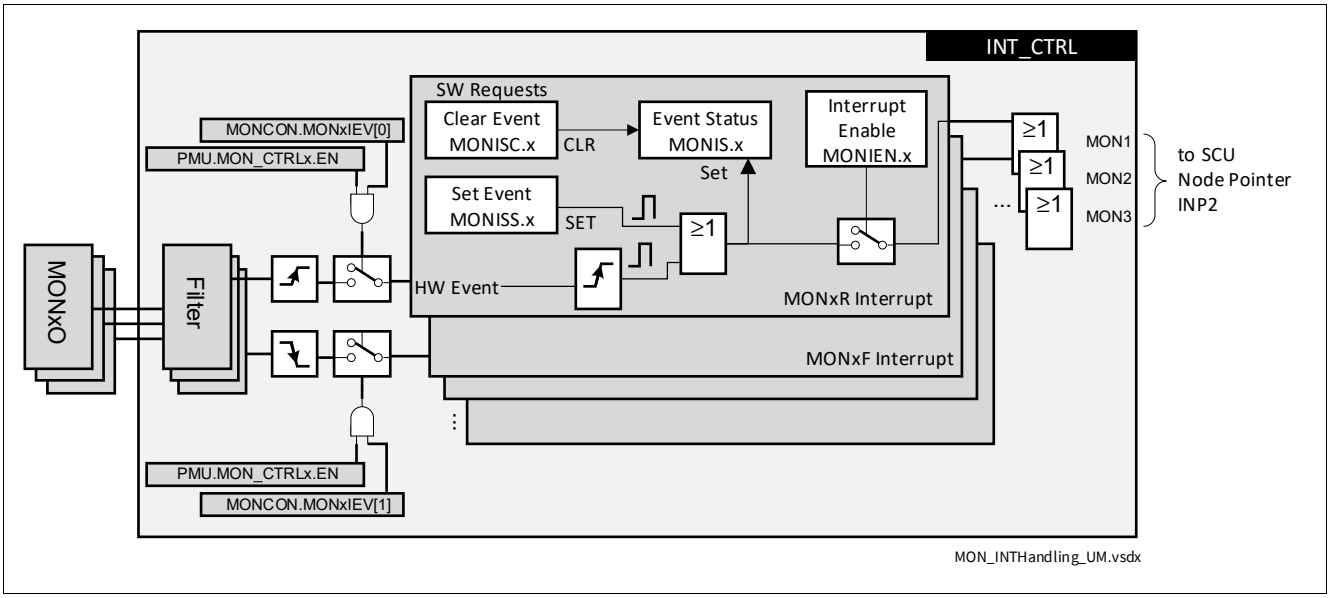


Figure 182 MONx interrupt handling in the SCU

Interrupts can be generated for changes in MON status: rising edge only, falling edge only, both edges

The SCU registers used for MONx interrupt configuration are SCU.**MONIEN**, SCU.**MONIS**, SCU.**MONISS**, SCU.**MONISC**, SCU.**MONCON** and SCU.**INP2**.

Note: Interrupts can also be generated for MON level supervision in ADC (value above upper threshold, value below lower threshold).

14.5 Operation mode behavior

The high-voltage monitor inputs follow the operation modes according to **Table 166**.

Table 166 Operation mode behavior MON

Reset	• RESET_TYPE_2 resets the MONx configuration settings in the PMU • RESET_TYPE_3 resets the MONx interrupt settings in the SCU
Power-up / Power-down	• The supplies and bias current of the HV monitor inputs are activated • At power-up all HV monitor inputs are enabled by default • The HV monitor inputs operate as long as they are not gated by RESET_TYPE_2
Active mode	• The HV monitor inputs are enabled by default • The HV monitor input wake-up capabilities are enabled by default
Stop mode	• The HV monitor inputs configuration is kept unchanged • The HV monitor input wake-up capabilities are kept unchanged
Sleep mode Fail-sleep mode	• The HV monitor inputs configuration is kept unchanged • The HV monitor input wake-up capabilities are kept unchanged

14.6 MON configuration registers

Registers for MONx configuration are located in the PMU:

- PMU.MON_CTRL0 for MON1
- PMU.MON_CTRL1 for MON2 (only in 64-pin-device)
- PMU.MON_CTRL2 for MON3 (only in 64-pin-device)

These registers are referred to as "**MON_CTRLx (x=0-2)**" in the following description.

14.6.1 MON modes and status

Table 167 shows an overview of the available modes of a MONx, dependent from the enable-bit **MON_CTRLx (x=0-2)**.EN.

Table 167 MONx_EN MON mode definition

PMU.MON_CTRLx (x=0-2).EN	MON mode	Description
0	disabled	Monitoring input is disabled (no wake-up possible, no PU/PD current sources)
1	normal mode / power saving mode	Monitoring input is active during system ACTIVE mode Monitoring input automatically enters power saving mode in system SLEEP mode and STOP mode

Note: In case a monitoring input is deactivated by setting bit MONx_EN to zero, it can neither be used as a wake-up source nor can it be used to detect logic levels!

In Active mode, the status (high/low) can be read in PMU.**MON_STS**.STS.

MON status is read as 0, when MON is not enabled.

The switching behavior of the HV monitor input includes a hysteresis $V_{MONth,hys}$.

14.6.2 Pull-up, pull-down currents

The MONx includes an input circuit with pull-up (can be activated by PMU.**MON_CTRLx (x=0-2)**.PU bit) and pull-down (can be activated by PMU.**MON_CTRLx (x=0-2)**.PD bit) current sources to define a certain voltage level with open inputs. The currents are $I_{PU, MON}$ and $I_{PD, MON}$.

Automatic current source selection is enabled by setting both PU and PD at the same time to 1. The behavior is shown in **Figure 183**.

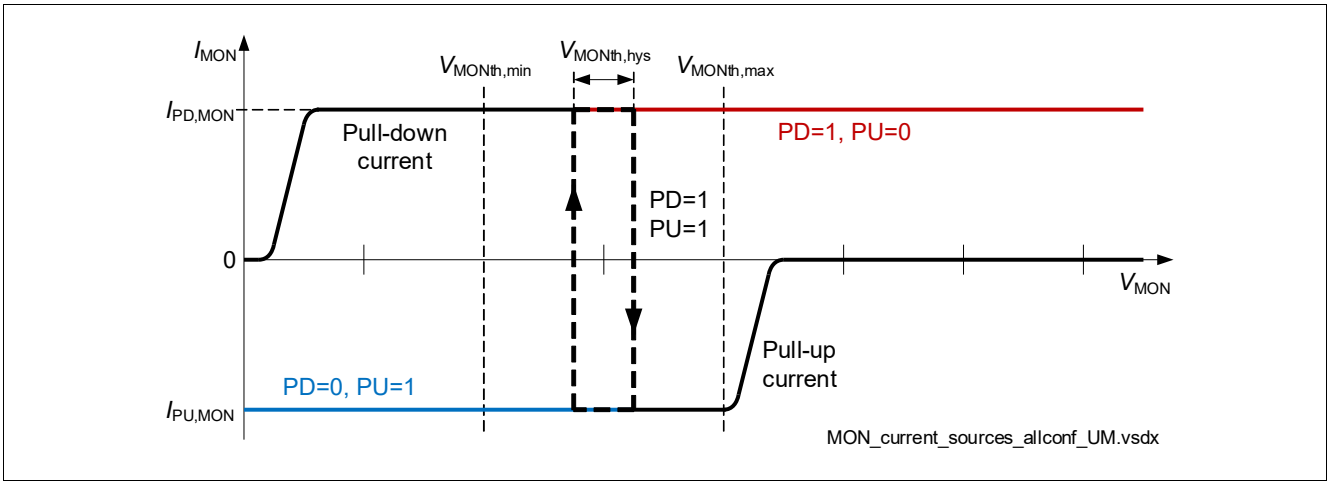


Figure 183 HV MON input characteristics for pull-up and pull-down currents (positive current flowing into the pin)

Table 168 includes all pull-up and pull-down setup scenarios which can be chosen for one MONx.

Table 168 Pull-up / pull-down input current

PU	PD	Output current	Description
0	0	leakage current	Pull-up/pull-down current source disabled
0	1	pull-down	Pull-down current source enabled (for low active switches)
1	0	pull-up	Pull-up current source enabled (for high active switches)
1	1	automatically switched	Support the current status: Pull-up current for MONx voltage above V_{MONth} Pull-down current for MONx voltage below V_{MONth}

14.6.3 Filter time

The filter time t_{MONx_FT} can be configured with PMU.**WAKE_FILT_CTRL**.MON_FT (16 µs or 32 µs typ.). This filter time applies to wake detection (in STOP or SLEEP mode) to avoid detection of unwanted voltage transients at the module input.

14.6.4 MON as wake-up source

The MONx module can be configured as a wake-up source to wake-up the system from the Stop mode or the Sleep mode.

First, PMU.**WAKE_CTRL**.MON_WAKE_EN has to be set accordingly.

For a wake-up on a positive voltage transition, the PMU.**MON_CTRLx (x=0-2)**.WAKE_RISE bit has to be configured. For a wake-up on a negative voltage transition, the PMU.**MON_CTRLx (x=0-2)**.WAKE_FALL bit has to be configured.

The wake-up feature can be enabled continuously (default) or with “cyclic sense” (wake-up detection only enabled during cyclic-sense-on-time) by setting PMU.**MON_CTRLx (x=0-2)**.CYC_SENSE_EN.

The wake-up status is handled in PMU.**WAKE_STS**.MON, PMU.**WAKE_STS_SET**.MON_SET and PMU.**WAKE_STS_CLR**.MON_CLR.

More details about the wake-up feature are described in the PMU chapter “**Wake control**“.

15 Analog Reference Voltage Generation (ARVG)

15.1 Features overview

The Analog Reference Voltage Generation module (ARVG) is responsible for the generation of reference voltages that can be used by the various analog peripherals in the device.

The ARVG provides following features:

- VREF1V2 internal reference voltage generation (V_{REF1V2}, 1.211 V typ.) for ADC2 and the NVM
- VREF1V2 is monitored by ADC1
- VAREF reference voltage generation (V_{REF5V}, 5 V typ.) for ADC1, CSA, CSC and SDADC; VAREF has a pin and needs a buffer cap to VAGND (see C_{VAREF})
- VAREF has an overcurrent (undervoltage) monitor (VAREF_OC)
- VAREF has an overvoltage monitor within the PMU (VAREF_OV), for more details please refer to the **Power Management Unit (PMU)** chapter

15.2 Block diagram

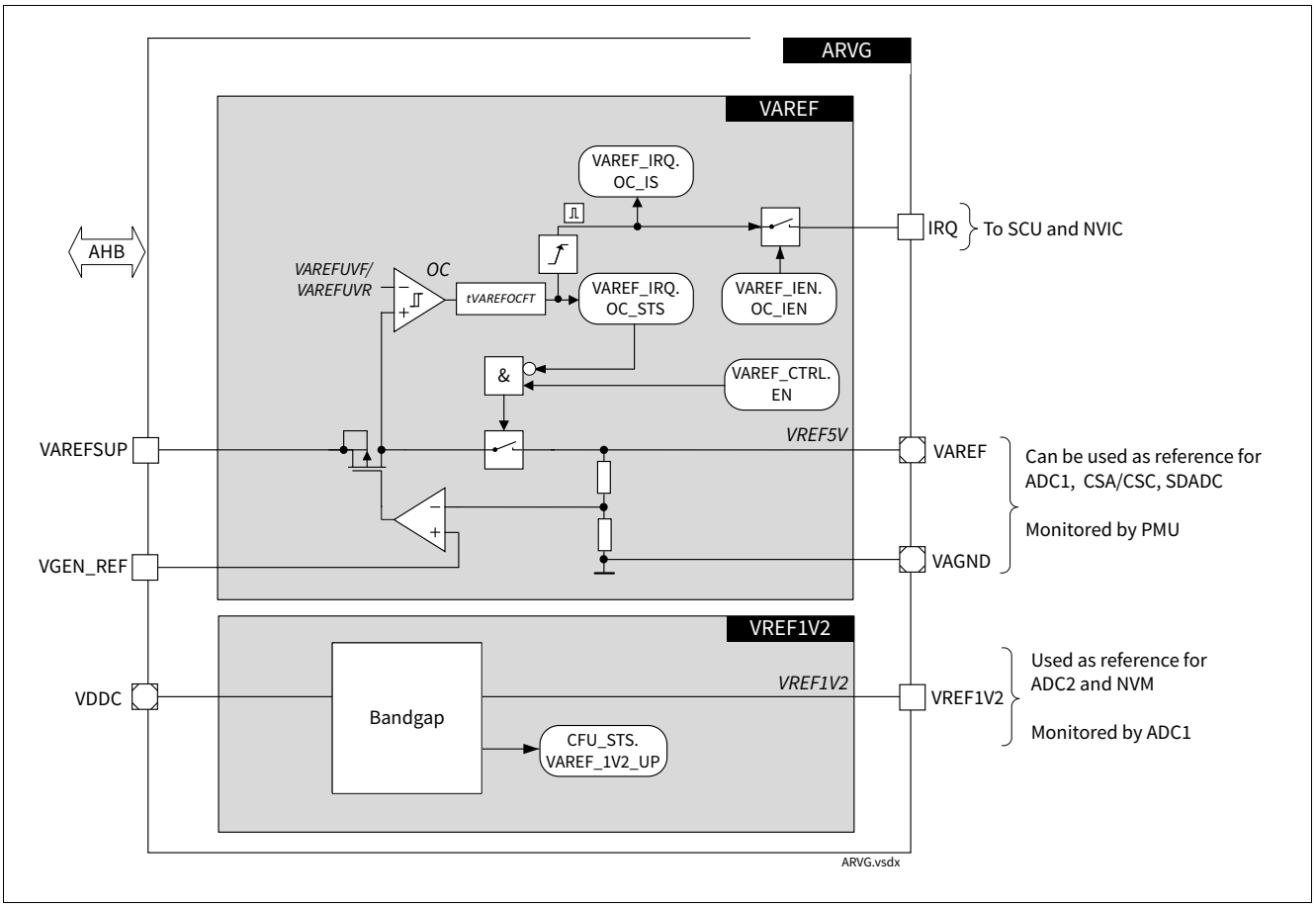


Figure 184 Block diagram ARVG

15.3 Toplevel signals

The ARVG module interconnects as specified in **Table 169** and **Table 170**.

Table 169 ARVG power domain

Signal	Direction	Description	From/To
VAREFSUP	Power	Input supply	PMU
VGEN_REF	Input	Voltage reference	PMU
VDDC	Power	Input supply	PMU, pin
VREF1V2	Power	Reference output voltage (V_{REF1V2})	ADC2 NVM
VAREF	Power	Reference output voltage (V_{REF5V})	ADC1, SDADC, CSA/CSC, pin
VAGND	Power	Analog supply ground	Pin

Table 170 ARVG control signals

Signal	Direction	Description	From/To
IRQ	Output	Interrupt service request signal	SCU
AHB slave bus I/F	Input/Output	Slave AHB-lite bus interface	AHB

15.4 Interrupts

Events

The ARVG module provides following events:

- VAREF overcurrent (undervoltage) interrupt status bit VAREF_IRQ.OC_IS
- VAREF overcurrent (undervoltage) life status bit VAREF_IRQ.OC_STS with protection mechanism (VAREF generation switched off)
- VAREF1V2 status bit CFU_STS.VAREF_1V2_UP

Interrupts

To register the occurrence of an overcurrent event (OC), the ARVG sets the interrupt status bit OC_IS. The interrupt request can be enabled via bit OC_IEN. The interrupt request is propagated to the SCU for further processing and distribution. An interrupt is requested regardless of the status of bit OC_IS (edge triggered). The interrupt request can be masked by disabling bit OC_IEN.

To clear the interrupt status bit (OC_IS), bit OC_IS_CLR has to be set.

For debug and test purposes, the overcurrent event (OC) can be emulated via setting bit OC_IS_SET.

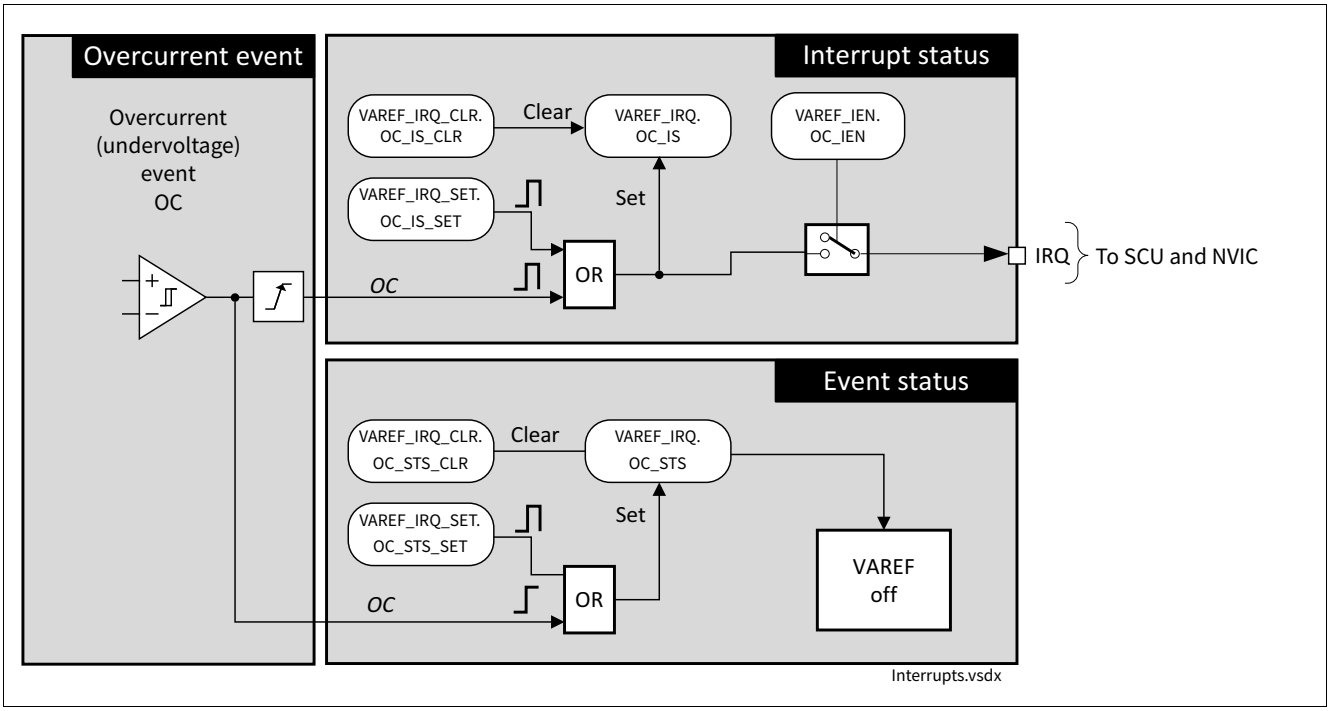


Figure 185 Interrupt and status registers

For more details, refer to the **Register description ARVG**; following SFRs are available:

- CFU_STS
- VAREF_IRQ
- VAREF_IRQ_SET
- VAREF_IRQ_CLR
- VAREF_IEN

15.5 Operation mode behavior

Table 171 ARVG state-of-operation vs. SoC system power modes

Reset	The ARVG module is controlled by a RESET_TYPE_5. An activated reset has following effects: • All SFRs are reset • VREF1V2 and VAREF reference voltages are disabled
Power-up	• The ARVG module is in reset state as long as the supply voltage is not sufficient
Power-down	• The ARVG module enters reset state as the supply voltage is no longer sufficient
Active mode	• VREF1V2 reference voltage is active • VAREF reference voltage is active via setting bit VAREF_CTRL.EN = 1
Stop mode	• VREF1V2 and VAREF reference voltages are disabled
Sleep mode	• VREF1V2 and VAREF reference voltages are disabled
Fail-sleep mode	• VREF1V2 and VAREF reference voltages are disabled

15.6 ARVG state-of-operation

The Analog Reference Voltage Generation module (ARVG) generates the reference voltages for analog peripherals (ADC1, ADC2, CSA, CSC, SDADC, NVM).

15.6.1 VREF1V2 reference voltage

The VREF1V2 is an internal reference voltage (V_{REF1V2}) used for the ADC2 and the NVM. The VREF1V2 rail turns on automatically during the SoC power-up phase.

The status of this reference voltage (i.e. turned on/off) can be monitored via the CFU_STS.VREF1V2_UP bit. In case of a reference system start-up failure, the majority of the analog peripheral functionalities (including monitoring and diagnostics) will be lost.

15.6.2 VAREF reference voltage

The VAREF regulator provides a reference voltage (V_{REF5V}) to the ADC1, SDADC and CSA/CSC modules. The VAREF regulator can be enabled/disabled by software via the VAREF_CTRL.EN control bit.

The VAREF has an overcurrent protection mechanism. The overcurrent detection reacts if the VAREF voltage is lower than the $V_{AREFUVF}$ threshold for a duration longer than the VAREF OC filter time ($t_{VAREFOCFT}$, 4 µs typ).

The VAREF reference is automatically switched off upon an overcurrent (i.e. undervoltage) event and bits VAREF_IRQ.OC_STS and VAREF_IRQ.OC_IS are set.

To restart the VAREF reference generator after an overcurrent (i.e. undervoltage) event, the VAREF_IRQ.OC_STS status flag has to be cleared by software (via VAREF_IRQ_CLR.OC_STS_CLR) and the VAREF voltage has to be higher than the $V_{AREFUVR}$ threshold for a duration longer than the VAREF OC filter time ($t_{VAREFOCFT}$, 4 µs typ).

Note: The VAREF reference voltage is a relatively high-impedance source. Do not connect any external circuitry to the VAREF pin.

The VAREF reference voltage must be stable for accurate measurements. An external ceramic capacitor (C_{VAREF}) is used to buffer the reference pin and filter out mid- and high- frequency noise. Good reference supply decoupling is important to achieve optimum performance. The C_{VAREF} capacitor needs to be placed as close as possible to the VAREF and VAGND pins of the device.

Notes

1. *Regarding dimensioning the buffer capacitor at VAREF, please refer to C_{VAREF} electrical characteristics.*
2. *Maintaining a low noise reference system in a digital environment is challenging. At a board level, the VAGND pin should not be connected with any other reference potential (e.g. GNDVSSC or GNDVSSP). The ADCs input signal reference potential and the SoC analog reference should join (star connection) at the VAGND pin.*

15.6.3 External VAREF reference voltage

The SoC analog peripherals can also use an external reference voltage. In this case, the internal VAREF reference voltage must be disabled (VAREF_CTRL.EN bit = 0) and an external voltage source must be applied between the VAREF and VAGND pins. For more details on the required external reference sourcing capability, please refer to the analog peripherals specification.

Notes

1. *The external VAREF voltage applied to the device shall be set in accordance to the analog peripherals specification (e.g. ADC1, CSA, CSC, SDADC).*
2. *The VAREF overcurrent (i.e. undervoltage) monitor is disabled when the internal VAREF reference voltage is disabled (i.e. VAREF_CTRL.EN bit = 0).*

15.6.4 VAREF, VREF1V2 reference voltages supervision

The SoC architecture supports cross reference voltage supervision. Such a functionality allows continuous system-monitoring by reading back the reference voltage of a companion analog peripheral. The ADC1 can be used to monitor the ADC2 reference voltage (VREF1V2). Conversely, the ADC2 can be used to monitor the ADC1 reference voltage (VAREF).

15.7 Programmer's guide

15.7.1 ARVG module initialization

The following steps should be considered to initialize properly (including diagnostics) the analog reference voltage generation module.

Table 172 Basic configuration steps

Step 1	CFU_STS register VAREF_IRQ register FS_STS register	Check for possible hardware failure trigger source(s): • Central reference system start-up failure: CFU_STS.VREF1V2_UP bit = 0 [1)] • VAREF overcurrent (i.e. undervoltage): VAREF_IRQ.OC_STS bit = 1 [2)] • VAREF overvoltage: FS_STS.VAREF_OV_STS bit = 1 [3)]
Step 2	VAREF_CTRL register	Enable VAREF voltage reference (optional)[4)]: • Internal VAREF reference voltage: VAREF_CTRL.EN bit = 1 • External VAREF reference voltage: VAREF_CTRL.EN bit = 0
Step 3	VAREF_IEN register	Enable monitoring interrupt generation (optional): • VAREF overcurrent (undervoltage): VAREF_IEN.OC_IEN bit = 1

1) In case of a reference system start-up failure, the majority of the analog peripheral functionalities (including monitoring and diagnostics) is lost.
2) The VAREF reference voltage is automatically switched off in case of an overcurrent (i.e. undervoltage) event.
3) The PMU safe-shutdown supervision logic brings the power bridge into a safe off-state. For more details, please check the PMU safe-shutdown chapter.
4) Some analog peripherals (e.g. ADC1, CSA/CSC, SDADC) require a reference voltage.

15.8 Register description ARVG

15.8.1 ARVG Address Maps

Table 173 Register Address Space - ARVG

Module	Base Address	End Address	Note
AHB	48038000_H	$4803BFFF_H$	

Table 174 Register Overview - ARVG (ascending Offset Address)

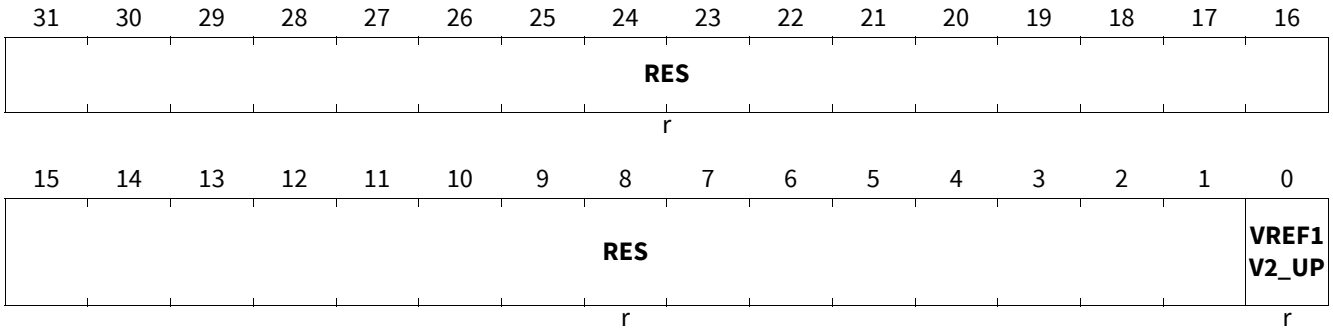
Short Name	Long Name	Offset Address	Page Number
CFU_STS	CFU status register	0000_H	644
VAREF_IRQ	VAREF Interrupt Status Register	0004_H	644
VAREF_IRQ_SET	VAREF Interrupt Set Register	0008_H	645
VAREF_IRQ_CLR	VAREF Interrupt Clear Register	$000C_H$	645
VAREF_IEN	VAREF Interrupt Enable Register	0010_H	646
VAREF_CTRL	VAREF control register	0018_H	646

15.8.2 ARVG Registers

CFU status register

CFU_STS

CFU status register **(0000_H)** **RESET_TYPE_5 Value: 0000 0000$_H$**

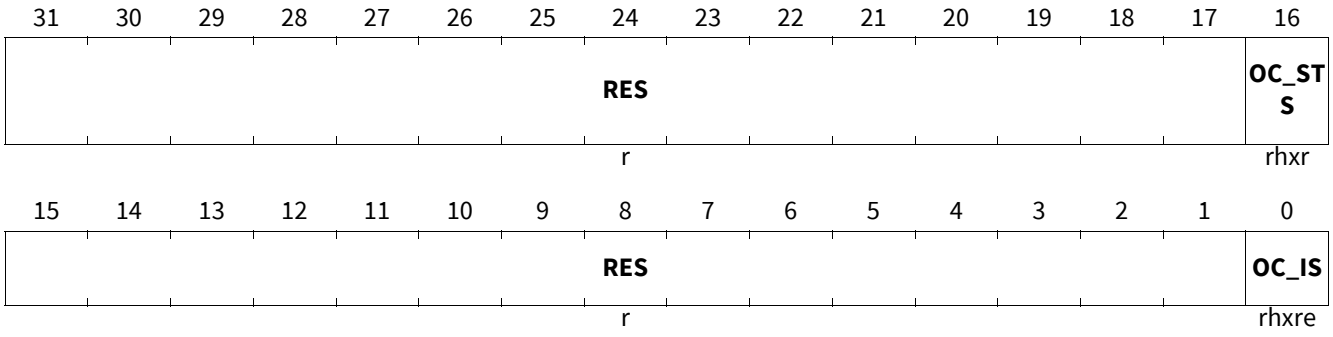


Field	Bits	Type	Description
VREF1V2_UP	0	r	**Reference voltage status** 0_B **UP_0**, CFU bandgap not started 1_B **UP_1**, CFU bandgap started
RES	31:1	r	**Reserved**

VAREF Interrupt Status Register

VAREF_IRQ

VAREF Interrupt Status Register **(0004_H)** **RESET_TYPE_5 Value: 0000 0000$_H$**



Field	Bits	Type	Description
OC_IS	0	rhxre	**VAREF overcurrent (undervoltage) interrupt.** 0_B **OC_0**, VAREF overcurrent (undervoltage) interrupt not triggered 1_B **OC_1**, VAREF overcurrent (undervoltage) interrupt triggered
RES	15:1, 31:17	r	**Reserved**

Register description ARVG

Field	Bits	Type	Description
OC_STS	16	rhxr	**VAREF overcurrent (undervoltage) status flag. The detection circuit monitors VAREF voltage.** 0_B **OC_0**, No overcurrent (undervoltage) event on VAREF 1_B **OC_1**, An overcurrent (undervoltage) event occurred on VAREF. VAREF automatically switches off when this bit is set.

VAREF Interrupt Set Register

VAREF_IRQ_SET
VAREF Interrupt Set Register **(0008_H)** **RESET_TYPE_5 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															OC_STS_SET
r															w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															OC_IS_SET
r															w

Field	Bits	Type	Description
OC_IS_SET	0	w	**VAREF overcurrent (undervoltage) interrupt set** 0_B **SET_0**, do nothing 1_B **SET_1**, Set VAREF overcurrent (undervoltage) interrupt flag
RES	15:1, 31:17	r	**Reserved**
OC_STS_SET	16	w	**VAREF overcurrent (undervoltage) status flag set. Setting this bit will trigger a VAREF switch off.** 0_B **SET_0**, do nothing 1_B **SET_1**, Set VAREF overcurrent (undervoltage) status flag

VAREF Interrupt Clear Register

VAREF_IRQ_CLR
VAREF Interrupt Clear Register **($000C_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															OC_STS_CLR
r															w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															OC_IS_CLR
r															w

Register description ARVG

Field	Bits	Type	Description
OC_IS_CLR	0	w	**VAREF overcurrent (undervoltage) interrupt clear** 0_B **CLR_0**, do nothing 1_B **CLR_1**, Clear VAREF overcurrent (undervoltage) interrupt flag
RES	15:1, 31:17	r	**Reserved**
OC_STS_CLR	16	w	**VAREF overcurrent (undervoltage) status flag clear** 0_B **CLR_0**, do nothing 1_B **CLR_1**, Clear VAREF overcurrent (undervoltage) status flag

VAREF Interrupt Enable Register

VAREF_IEN
VAREF Interrupt Enable Register **(0010_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
OC_IEN	0	rw	**VAREF Overcurrent (undervoltage) Interrupt Enable** 0_B **IEN_0**, VAREF overcurrent (undervoltage) interrupt disabled 1_B **IEN_1**, VAREF overcurrent (undervoltage) interrupt enabled
RES	31:1	r	**Reserved**

VAREF control register

VAREF_CTRL
VAREF control register **(0018_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Register description ARVG

Field	Bits	Type	Description
EN	0	rw	**VAREF enable bit** 0_B **EN_0**, VAREF disabled 1_B **EN_1**, VAREF enabled
RES	31:1	r	**Reserved**

16 Analog Digital Converter 1 (ADC1)

16.1 Features overview

The ADC1 is a successive approximation analog to digital converter which can be used for analog signal measurement especially optimized for BLDC motor control. It has a deterministic behavior regarding to the sample event and conversion timing, even for a sequence of conversions. The ADC1 operates greatly autonomous in background avoiding real-time critical interaction by the CPU or DMA.

The ADC1 provides following features:

- A/D kernel performance:
  - 12-bit resolution for all analog inputs
  - High accuracy of typ. 0.5% of the input range (MV_{ACC}, HV_{ACC1})
  - Fast sampling time ($tsamp_{MV}$, $tsamp_{HV}$, $tsamp_{SHx}$)
  - Fast total conversion time (typ. 800 ns for MV/HV inputs and 1600 ns for SHx inputs)
- Analog inputs ANx:
  - Up to 11 middle voltage inputs (range MVRNG)
  - 8 factory calibrated high voltage inputs (range HVRNG1)
  - Referenced to VAREF/VAGND via internal VAREF or VDDEXT voltage regulators
- Digital channels with channel control and result generation:
  - Each analog input can be freely assigned to one or more out of 20 possible digital channels
  - Each digital channel has its own result register with a result event (IRQ capable)
  - One out of 4 conversion classes can be assigned to a digital channel
  - Up to 4 digital comparators with 8-bit upper and lower thresholds can monitor a channel result and generate a compare event (IRQ and/or interconnect signal)
  - Up to 4 first order IIR filter s with programmable characteristics can be assigned to a channel result
  - Programmable repeat feature for each channel
- Trigger and gating control:
  - For deterministic control of complex conversion sequences with respect to time-accuracy and time-equidistancy
  - 36 trigger inputs can be selected for hardware or software-based start event of a conversion sequence
  - 16 gating inputs can be selected for hardware or software-based gating of a trigger event
- Sequencer:
  - Allows to build complex and variable conversion schemes
  - Up to 4 independent sequences with up to 4 digital channels can be freely assembled
  - Hardware or software-based trigger of a sequence with deterministic end of sequence event (IRQ capable)
  - Possible trigger features: self-trigger or next-sequence-trigger (round robin capable)
  - Shadow mechanism for data coherency when updating the sequencer configuration on-the-fly
- Conversion class control:
  - Programmable sample time adjustment to adapt to the analog input characteristic
  - Programmable noise reduction feature (oversampling, averaging, sample point adjust)
  - Programmable broken wire detection feature for external sensors

  - Programmable calibration feature to compensate drift and temperature effects
- Interrupt and DMA:
  - The ADC1 events can be routed to 4 interrupt node pointers (with 4 IRQ lines)
  - ADC1 events can be mapped to 8 DMA channels

16.2 Block diagram

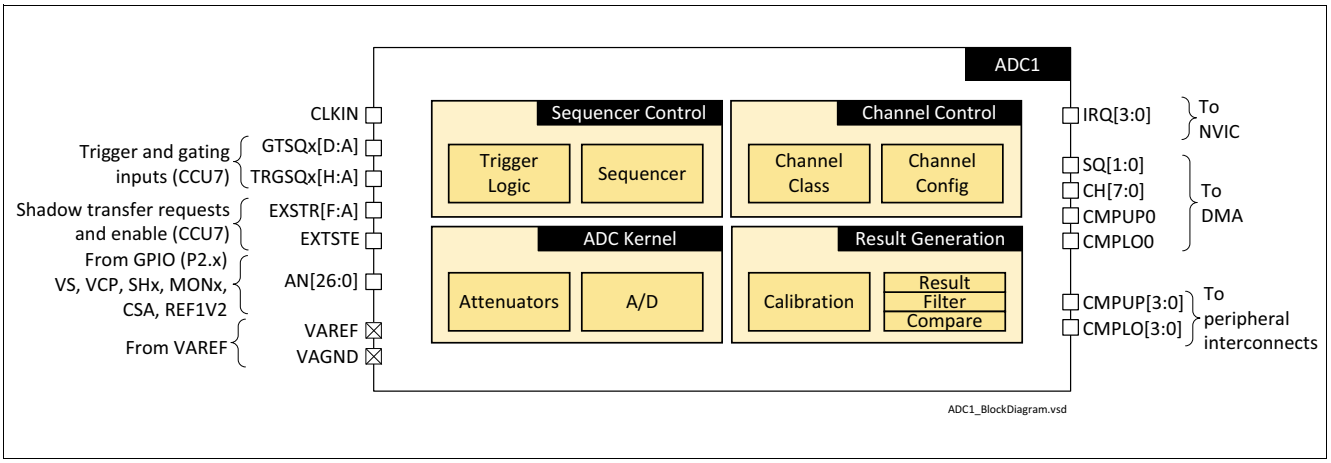


Figure 186 Block diagram ADC1

16.3 Toplevel signals

Table 175 ADC1 clock domain

Name	Direction	Description	From
CLKIN	Input	ADC1 module clock input	See **Product definitions**, **SCU interconnections**

Table 176 ADC1 reference voltage

Name	Direction	Description	From/To
VAREF	Input	Analog reference voltage	VAREF pin
VAGND	Input	Analog reference ground	VAGND pin

Table 177 ADC1 analog inputs

Name	Direction	Description	From/To
AN[15:0]	Input	HV input	See **Product definitions**, **ADC1 interconnections**
AN[26:16]	Input	MV input	

Table 178 ADC1 trigger inputs

Name	Direction	Description	From
TRGSQ0[H:A]	Input	Trigger input for sequencer SQ0	See **Product definitions**, **ADC1 interconnections**
TRGSQ1[H:A]	Input	Trigger input for sequencer SQ1	
TRGSQ2[H:A]	Input	Trigger input for sequencer SQ2	
TRGSQ3[H:A]	Input	Trigger input for sequencer SQ3	

Table 179 ADC1 gating inputs

Name	Direction	Description	From/To
GTSQ0[D:A]	Input	Gating input for sequencer SQ0	See **Product definitions**, **ADC1 interconnections**
GTSQ1[D:A]	Input	Gating input for sequencer SQ1	
GTSQ2[D:A]	Input	Gating input for sequencer SQ2	
GTSQ3[D:A]	Input	Gating input for sequencer SQ3	

Table 180 ADC1 synchronization inputs

Name	Direction	Description	From/To
EXTSTE	Input	External shadow transfer enable signal	See **Product definitions**, **ADC1 interconnections**
EXTST_[F:A]	Input	External shadow transfer events	

Table 181 ADC1 outputs

Name	Direction	Description	From/To
CMPLO[3:0]	Output	Lower threshold compare output of result generation	See **Product definitions**, **ADC1 interconnections**
CMPUP[3:0]	Output	Upper threshold compare output of result generation	
CH[7:0]	Output	Channel finished event	
SQ[1:0]	Output	Sequencer finished event	

Table 182 ADC1 interrupt requests

Name	Direction	Description	From/To
IRQ[3:0]	Output	Interrupt service request	See **Product definitions**, **ADC1 interconnections**

Table 183 ADC1 DMA requests

Name	Direction	Description	From/To
CH[7:0]	Output	Channel result event	See **Product definitions**, **ADC1 interconnections**
SQ[1:0]	Output	Sequencer finished event	
CMPLO0	Output	Lower threshold compare output of result generation	
CMPUP0	Output	Upper threshold compare output of result generation	

16.4 Interrupts

Events

The ADC1 provides following events:

- Channel “conversion finished” events (see CHSTAT.CHn)
- Sequencer “finished” events (see SQSTAT.SQx, SQSTAT.COLLx, SQSTAT.WFRx)
- Compare “match” events (see CMPSTAT.CMP_UP_IS, CMPSTAT_CMP_LO_IS). The compare events are interconnected to other peripherals (see **Product definitions**, **ADC1 interconnections**)

Interrupt requests

The events can request an interrupt. Therefore the events can be assigned via the node pointer scheme to an interrupt request at the NVIC. The ADC1 has four node pointers with four interrupt request lines (ADC1.IRQ[3:0]).

The node pointer scheme is described in **Figure 187**.

The node pointer to NVIC assignment is described in **Product definitions**, **Interrupt request mapping**.

DMA service request

The events can be assigned to a DMA service request. The DMA assignment is described in **Product definitions**, **DMA request mapping**.

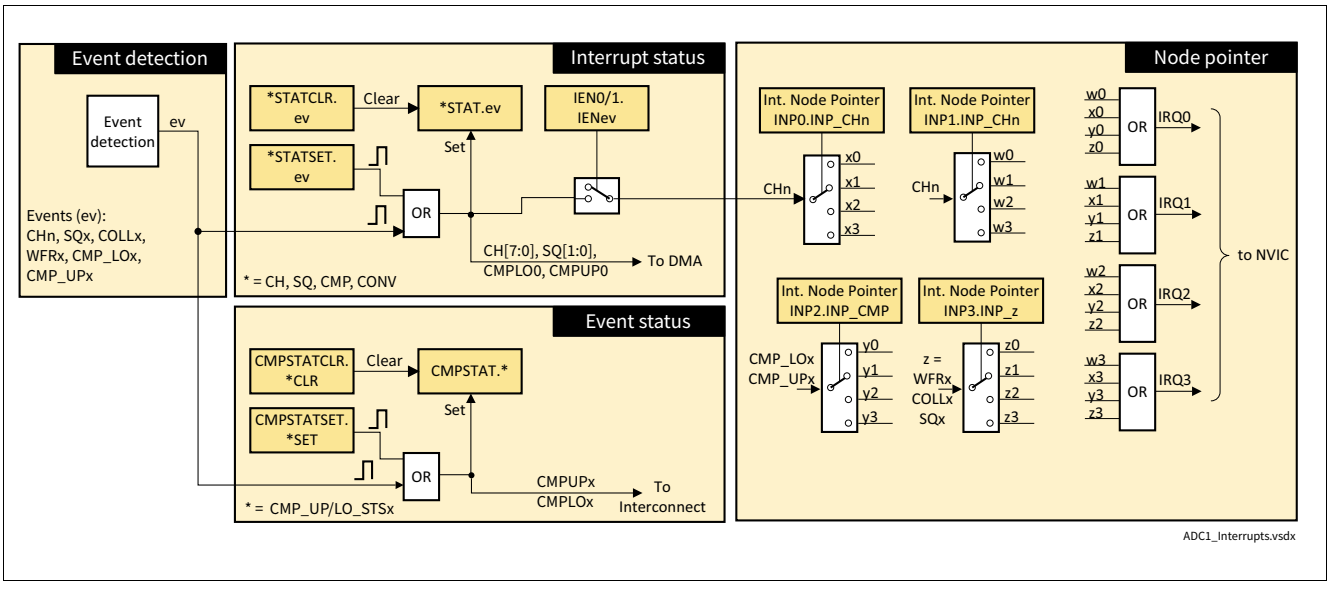


Figure 187 Event handling

16.5 Operation mode behavior

Table 184 Operation mode behavior ADC1

Reset	The ADC module is reset via RESET_TYPE_4 and RESET_TYPE_5. An activated RESET has following effects: • All SFRs of the ADC are reset according to its reset type • The state machines of the ADC kernel and sequencer are reset • The generated events show reset value
Power-up / Power-down	• The ADC is kept in reset state as long as supply is not in specified operating range • The ADC is released from reset state when supply is in specified operating range
Active mode	• Once the ADC module is enabled (EN = 1), the ADC kernel startup phase starts (tup, tsucal) • When disabled (EN = 0) the SFR content is kept
Stop mode	• The ADC is unpowered, the SFR content and all states are lost • The Stop mode entry is triggered by setting PMCON0.STOP, the entry sequence is done automatically • The ADC wakes from Stop mode with a power-up
Sleep mode	• The ADC is unpowered, the SFR content and all states are lost • The Stop mode entry is triggered by setting PMCON0.SLEEP, the entry sequence is done automatically • The ADC wakes from Sleep mode with a power-up
Fail-sleep mode	• None

16.6 Working principle

The ADC working principle can be taken from **Table 185** and **Figure 188**.

Table 185 ADC working principle

Step	Description	Condition	Repeat
Step 1:	A trigger event (TRGSQx) requests a sequence. The sequencer checks the WFR condition (VALID flags of its slots) and discards the trigger or continues. The sequencer checks the COLL condition and sets the trigger request to pending or to active. In case of an active trigger request it continues with Step 2.	WFRCFG COLLCFG	
Step 2:	The sequencer reads from the requested sequence the first (second, third, forth) slot channel number (CHSELx) and provides it to the channel control.		
Step 3:	The channel control provides the channel specific configuration to the ADC1 kernel (INSEL, CONVCFG) and to the result generation (INSEL, CHSEL, FILSEL, CMPSEL, CHREP).		
Step 4:	The sequencer triggers the start of conversion (SoC) to the ADC1 kernel.		
Step 5:	The ADC1 kernel performs the conversion according to the channel configuration and signalizes the end of conversion (EoC) to the sequencer control.		
Step 6:	The ADC1 kernel provides the digital result to the result generation. The result generation post-processes it and generates the result events (VALIDn, CHn, FILx, CMPx) accordingly.		
Step 7:	The sequencer repeats with Step 4 in case the channel has to be repeated (CHREPn), otherwise it continues with Step 8. Note: a channel event CHn is signalized after every EoC.	CHREPn	Step 4
Step 8:	The sequencer repeats with Step 2 with its next defined slot, otherwise it continues with Step 9.	SLOTS	Step 2
Step 9:	The sequencer signalizes the end of a sequence via its SQx event.		
Step 10:	The sequencer repeats with Step 2 in case the sequence has to be repeated (SQREPx). Otherwise it continues with Step 11.	SQREPx	Step 2
Step 11:	The sequencer is set to idle.		

Note: An ongoing sequence cannot be interrupted or aborted.

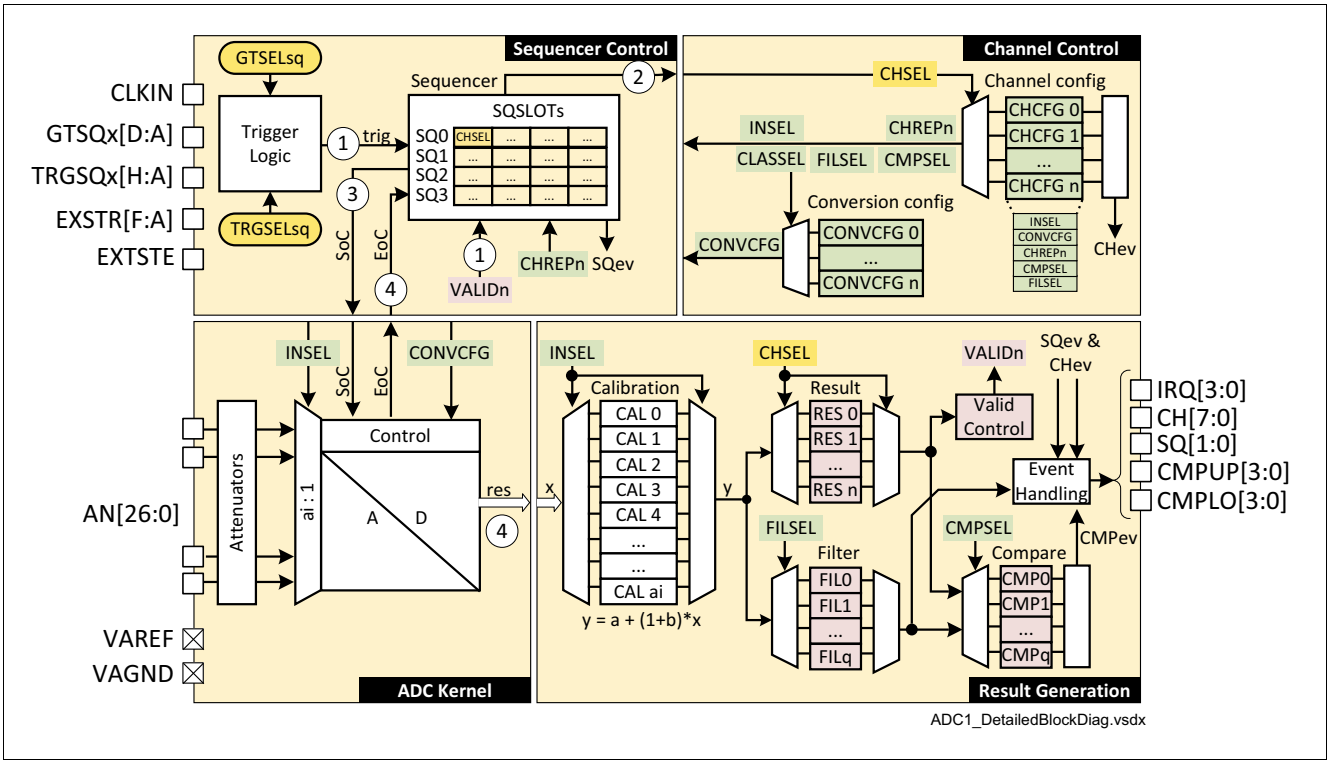


Figure 188 ADC1 detailed and functional block diagram

16.7 ADC1 global control

The global control allows specific settings which apply globally to the ADC1 module.

16.7.1 Enabling the ADC1

The ADC1 can be enabled via setting bit GLOBCONF.EN. After enabling the ADC1, the startup phases starts (see **Chapter 16.7.3**).

The ADC1 can be used after the startup phase has been finished. This is indicated by SUSSTAT.READY, see **Chapter 16.7.3**.

Notes

1. *When disabling the ADC1 via EN=0, the ADC kernel must be idle, i.e. no conversion must be ongoing and the trigger sources shall be disabled. The startup phases must be finished.*
2. *It has to be ensured that a reference voltage at VAREF is present when enabling the ADC1. In case internal VREF5V is used, this can be done in ARVG, VAREF_CTRL.EN=1.*
3. *The reference voltage VAREF can be measured and supervised by ADC2, see **Product definitions**, **ADC2 interconnections***

16.7.2 Clock configuration

The ADC1 module has an input clock CLKIN which has to be adapted to the internal module clock ADC_CLK, (see **Figure 189**).

- ADC_CLK must fulfill the limits specified in *fadc*
- ADC_CLK is used for all timing calculation
- ADC_CLK can be programmed via CLKCON.CLKDIV

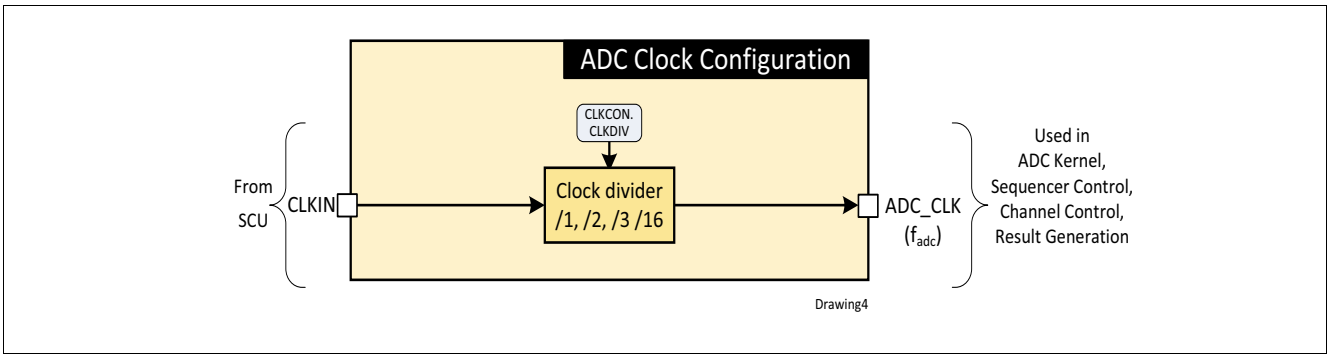


Figure 189 Clock configuration

16.7.3 Startup phases

The ADC kernel has to run two startup phases before it is ready to convert.

The end of the startup phase including the calibration phase is indicated in SUSSTAT.READY.

Notes

1. *The ADC (sequencer control, channel control, result generation) can be configured while kernel is in the startup phase.*
2. *A start of conversion request (SoC) is ignored while the ADC kernel is in the startup phase.*

16.7.3.1 Startup phase

- The startup phase is mandatory once the ADC is enabled via GLOBCONF.EN=0 to 1
- Within this phase the analog kernel has to settle
- The startup phase lasts for:
  - tup = 266 * ADC_CLK
  - while this time a conversion request is ignored

16.7.3.2 Startup calibration phase

- The startup calibration phase is needed after initial power-up, wake-up from Stop or Sleep mode
- Within this phase an initial offset calibration is done within the kernel to compensate device individual drift and temperature dependent effects. The startup calibration starts automatically after the startup phase
- The startup calibration phase lasts for:
  - tsucal = 290 * ADC_CLK
  - while this time a conversion request is ignored
- Additionally an on-the-fly post calibration feature is available, see **Chapter 16.8.7**

16.7.4 Suspend mode

For debugging purpose the module can be suspended if enabled. In suspend mode the ADC_CLK is halted but the SFR access via the debugger is possible.

The suspend mode can be enabled in SUSCTR.SUSEN.

There are two suspend modes, configurable in SUSCTR.SUSMOD:

- Hard suspend, i.e. the module clock is halted immediately
- Soft suspend, i.e. the module clock is halted when the sequencer is idle

The suspend status can be read form SUSSTAT.STAT.

16.8 ADC1 kernel features

The ADC1 kernel converts a selected analog signal AN[26:0] into a digital word (result).

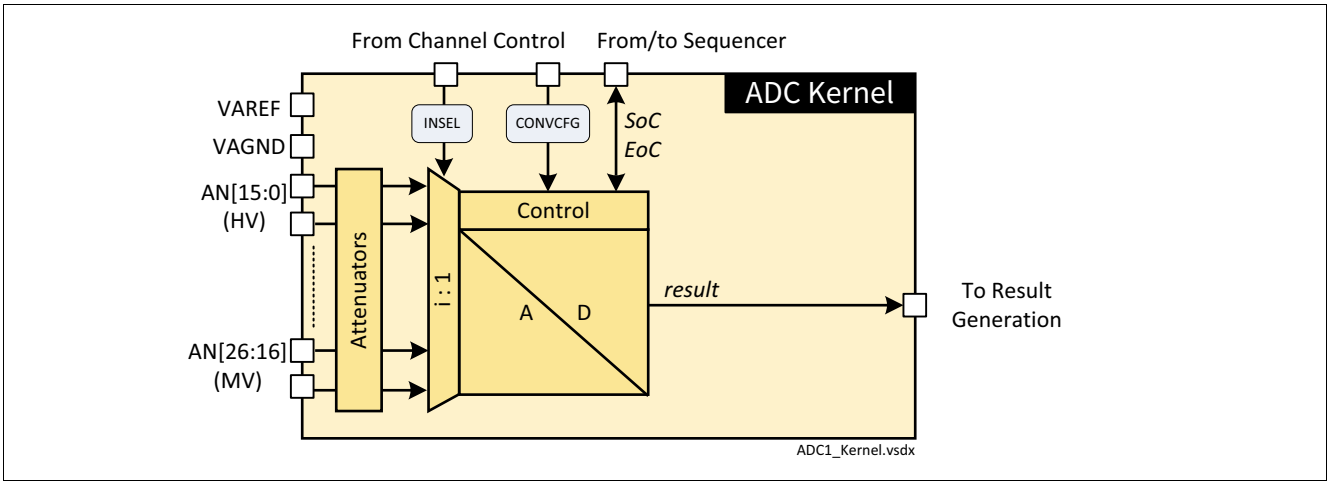


Figure 190 Kernel block diagram with data flow

16.8.1 Kernel type, reference and resolution

The ADC1 is analog to digital converter which uses the successive approximation (SAR) principle, see **Figure 191**

The ADC1 uses VAREF and VAGND pins as reference voltage.

The reference voltage range has to be respected, see VAREF.

The reference source can be:

- VREF5V (via internal reference, see chapter ARVG)
- VDDEXT (via external connection to VAREF, note: VREF5V must be disabled)
- External source connected to VAREF/VAGND

The ADC1 resolution is 12 bit.

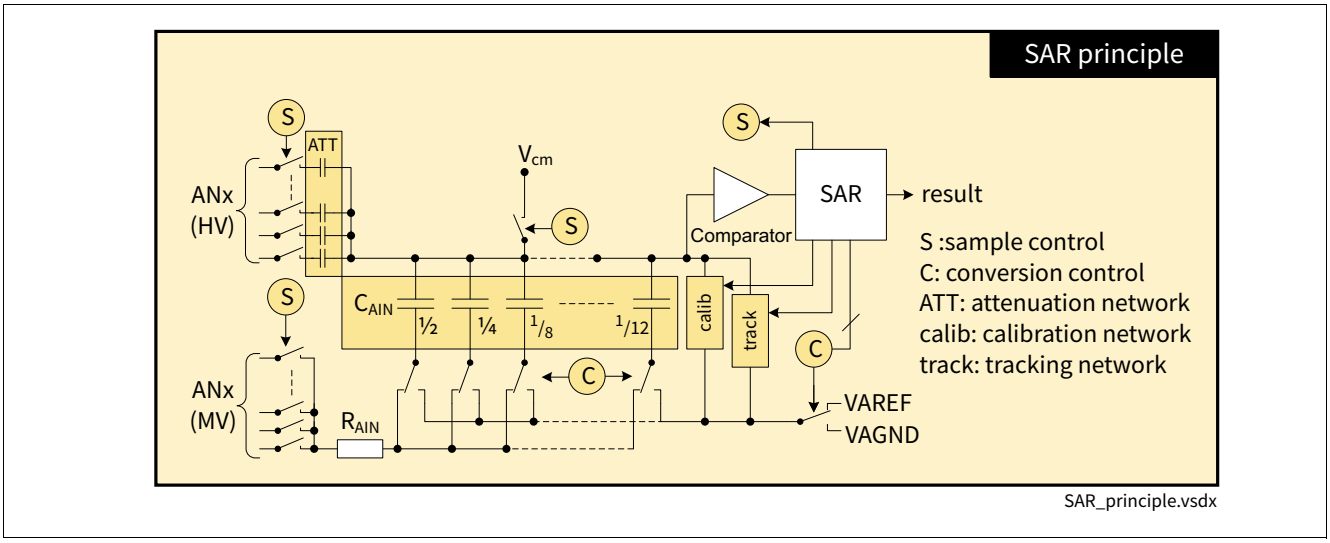


Figure 191 SAR principle, S (sample), C (conversion)

Note: It has to be ensured that a reference voltage at VAREF is present when enabling the ADC1. In case internal VREF5V is used, this can be done in ARFG, VAREF_CTRL.EN=1. The VAREF can be supervised by ADC2.

16.8.2 Kernel analog inputs

For adaption to the different voltage classes different attenuation factors are set. These attenuators are calibrated at device manufacturing and provided to calibration unit in the result generation (see **Chapter 16.11.1**).

The ADC1 has following analog inputs, see also **Table 186**:

- High voltage (HV) inputs with two attenuators for each input (ATT_TYP0, ATT_TYP1) to adapt to the input voltage range (HV_RNG0, HV_RANG1) to the ADC1 voltage range (VAREF):
  - ATT_TYP0 = 102/512; not calibrated, therefore not recommended to use these settings
  - ATT_TYP1 = 72/512; calibrated
- Middle voltage (MV) inputs with one attenuator for each input (ATT_TYP2) to adapt to the input voltage range (MV_RNG) to the ADC1 voltage range (VAREF):
  - ATT_TYP2 = 1; calibrated
- The ADC1 analog full scale (FS) is reached at an input voltage according to formula:

(16.1)

$$FS = HV_{RNGx} \times ATT\_TYPx,\ x=0..1$$

$$FS = MV_{RNG} \times ATT\_TYP2$$

The analog input to be measured is selected by the channel control bitfield CHCFGn.INSEL.

Note: The number of the analog inputs is different from the number of the digital channels.

Table 186 ADC1 analog inputs with attenuators and calibration

Name	ATT_TYPx	Calibration
AN0	ATT_TYP0	no
AN1	ATT_TYP1	yes
AN2	ATT_TYP0	no
AN3	ATT_TYP1	yes
AN4	ATT_TYP1	yes
AN5	ATT_TYP1	yes
AN6	ATT_TYP1	yes
AN7	ATT_TYP1	yes
AN8	ATT_TYP1	yes
AN9	ATT_TYP1	yes
AN10	ATT_TYP0	no
AN11	ATT_TYP1	yes
AN12	ATT_TYP0	no

Table 186 ADC1 analog inputs with attenuators and calibration (cont'd)

Name	ATT_TYPx	Calibration
AN13	ATT_TYP1	yes
AN14	ATT_TYP0	no
AN15	ATT_TYP1	yes
AN[26:16]	ATT_TYP2	yes

16.8.3 Kernel conversion timing scheme

The ADC1 has a programmable conversion scheme. Each analog input (ANi, i = 0 to 26)can be assigned to one or more digital channels (via CHCFGn.INSEL, n = 0 to 19).

There are four conversion classes available (via CONVCFGx, x = 0 to 3), which can be configured for the digital channel. The configuration options for one complete conversion can be seen in **Figure 192**.

Note: An ongoing conversion cannot be interrupted or aborted.

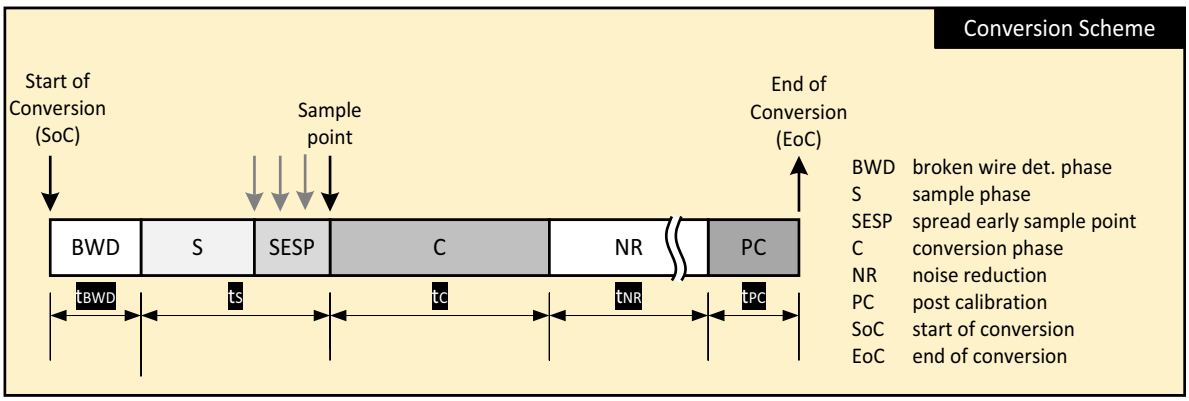


Figure 192 Conversion timing scheme

The ADC1 conversion scheme has the following sequence:

- A conversion starts with a Start of Conversion (SoC) trigger from the sequencer control
- An optional broken wire detection phase (BWD) can be configured
- This is followed by the sampling phase (S) where the selected analog input is connected to the ADC1's capacitor field
- This can be extended by the spread early sample point (SESP) which randomly spreads the sample point when using oversampling
- This is followed by the conversion phase (C) where the successive approximation determines the digital result
- This is followed by an optional noise reduction phase (NR) where oversampling or sub-conversions can improve the result accuracy
- This is followed by an optional post calibration phase (PC) which calibrates the ADC1 kernel's capacitor network
- This is followed by the End of Conversion (EoC) signal to the sequencer control
- The kernel result (12 bit wide) is forwarded to the result generation

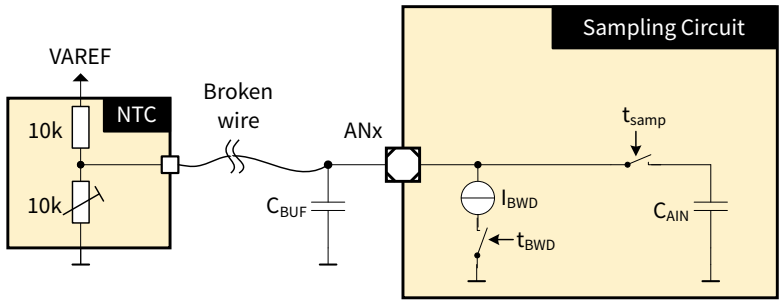
16.8.4 Kernel sample time features

16.8.4.1 Broken wire detection (BWD)

An optional current source can be configured to sink the input voltage with a defined current (see electrical characteristics: I_{BWD} and I_{BWDH}) within a defined time (t_{BWD}). This feature can manipulate the sensor signal in a deterministic manner and can be used to detect a broken wire. The effect of BWD current source can be calculated according to example in **Figure 193**. Therefore plausibility measurements with and without BWD have to be done to determine if the sensor wire is broken or not.

The broken wire detection can be enabled and its time t_{BWD} can be configured in CONVCFGx.BWD (x = 0 to 3).

The broken wire current strength (I_{BWD}, I_{BWDH}) can be selected via CONVCFGx.BWD_HI_CUR (x = 0 to 3).



Example for usage of BWD

1	ADC_CLK	30 MHz
	Conversion rate	1 ms
	Buffer cap	10 nF
	Source resistance	10 k
	Source charge time (8.4 * 10 k * 10 nF)	840 us
2	BWD off	
	Conversion result	2.500 V
3	BWD on	
	I_{BWD}	640 uA
	t_{BWD} (BWD=3)	533 ns
	BWD effect (dV) (I_{BWDH} * t_{BWD} / C_{BUF})	34 mV
	Conversion result after first conv.	2.466 mV

4 Repeat 2 and 3 with the defined conversion rate.
In case of a broken wire the source will not re-charge the buffer cap which is sunk by the BWD conversion. The conversion result will decrease by dV with every BWD conversion.
Without BWD, the buffer cap will keep its "wrong" voltage much longer.

BWD_Example.vsdx

Figure 193 Example for usage of broken wire detection feature

16.8.4.2 Sample time (STC)

Within the sample time the external signal on ANx charges the internal hold capacitor CAIN, i.e. the time is physically determined by the source resistance RANx and the internal input resistance RAIN, see **Figure 195**.

The minimum sampling time must be as long as it takes to charge a capacitor via a resistor until it reaches the voltage with an error smaller than the ADC1 resolution (here 12 bit), otherwise the ADC1 performance is worse than specified (see **Figure 195**):

- The minimum sampling time (assuming RANx = 0) for HV inputs must be fulfilled, see t_{sampHV}

- The minimum sampling time (assuming RANx = 0) for MV inputs must be fulfilled, see t_{sampMV}

The sample time is configured in CONVCFGx.STC according to following formula, see also **Figure 194**:

(16.2)

$$t_{samp} = STCval \times ADC\_CLK$$

CLKIN [MHz]	CLKDIV	Resulting ADC_CLK [MHz] (condition: < 50 MHz)	STC setting	STC resulting multiplier	Minimum possible tsampLV [ns] (condition: >200 ns)	Resulting tconv [ns] (option: 17*ADC_CLK)	Resulting total conversion time [ns]	
60	1	60,0	not allowed					
60	**2**	**30,0**	**5**	**7**	**233,3**	**566,7**	**800,0**	**default setting**
60	3	20,0	2	4	200,0	850,0	1050,0	
60	16	3,8	0	2	533,3	4533,3	5066,7	
40	1	40,0	6	8	200,0	425,0	625,0	
40	2	20,0	2	4	200,0	850,0	1050,0	
40	3	13,3	1	3	225,0	1275,0	1500,0	
40	16	2,5	0	2	800,0	6800,0	7600,0	
20	1	20,0	2	4	200,0	850,0	1050,0	
20	2	10,0	0	2	200,0	1700,0	1900,0	
20	3	6,7	0	2	300,0	2550,0	2850,0	
20	16	1,3	0	2	1600,0	13600,0	15200,0	

Figure 194 Minimal possible sampling time at different module clocks (CLKIN)

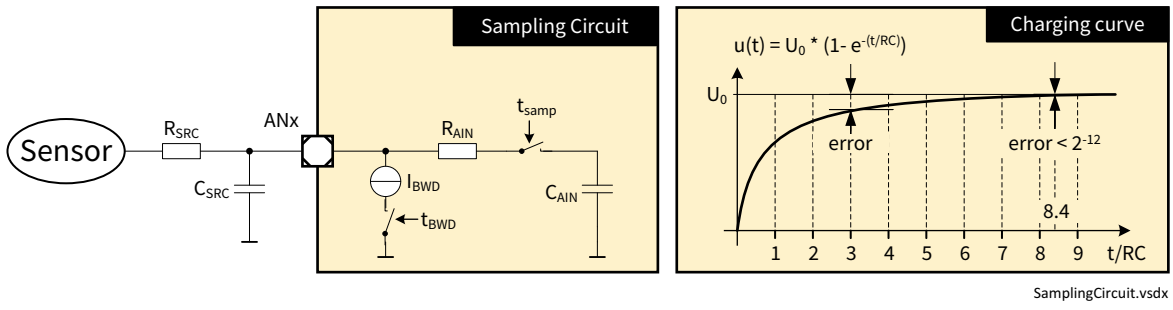


Figure 195 Equivalent sampling circuit and charging curve

16.8.4.3 Spread early sample point (SESP)

The ADC1 supports a spread early sampling point mode (SESP) to suppress correlated noise effects, i.e. system clock related noise which cannot be suppressed with oversampling. In this mode the sampling point is spread randomly within a range from 0 to 16 ADC_CLKs, see **Figure 196**.

The SESP mode can be enabled in bitfield CONVCFGx.SESP.

Note: The SESP mode should not be used for AC signals since it decreases the SNR performance. It is targeted for DC measurements.

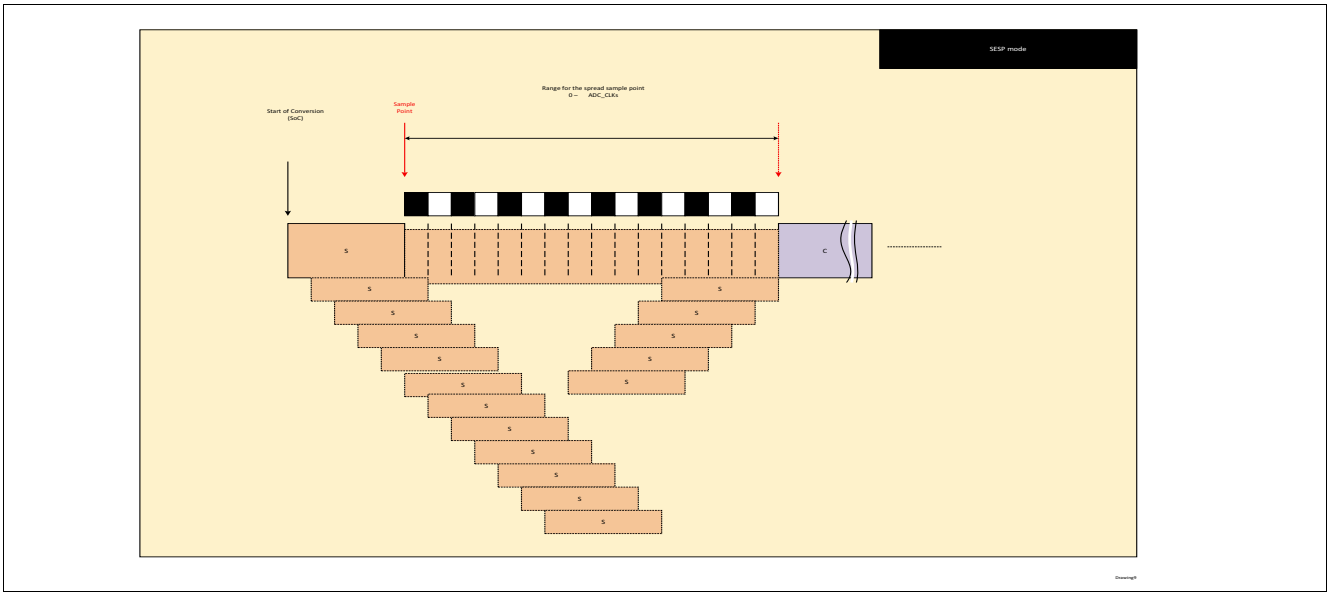


Figure 196 Spread early sample point

16.8.5 Kernel conversion time features

Within the conversion time the charge stored in the binary weighted capacitor array CAIN is redistributed via a successive approximation (SAR) algorithm. This algorithm switches the weighted capacitors accordingly between VAREF and VAGND, see **Figure 191**. At the end of conversion phase the digital result is determined. The EoC event is generated after the total conversion phase has been finished (optional noise reduction and post-calibration). The result can be post-processed via several methods.

The conversion time is subdivided into steps due to the SAR algorithm. Within the first step the charge of the capacitor representing the most significant bit (MSB, equal to half of CAIN, see CAREF) has to be redistributed. Therefore the reference source connected to VAREF has to charge via its resistance RAREF the capacitance 1/2 * CAIN. Within the second step only 1/4 * CAIN has to be redistributed by the reference source and so on.

The ADC1 has an option to double the time for the most significant bit (MSB) conversion. The MSB doubling feature is configured in CONVCFGx.MSBD.

The conversion time t_{conv} is calculated according to following formula:

(16.3)

$$t_{conv} = (17 + MSBD) \times ADC\_CLK$$

16.8.6 Kernel noise reduction features

The ADC1 supports two modes for noise reduction by averaging, (see **Figure 197**):

- A tracking conversion mode
- An oversampling conversion mode

Note: The tracking conversion and oversampling conversion modes cannot be combined.

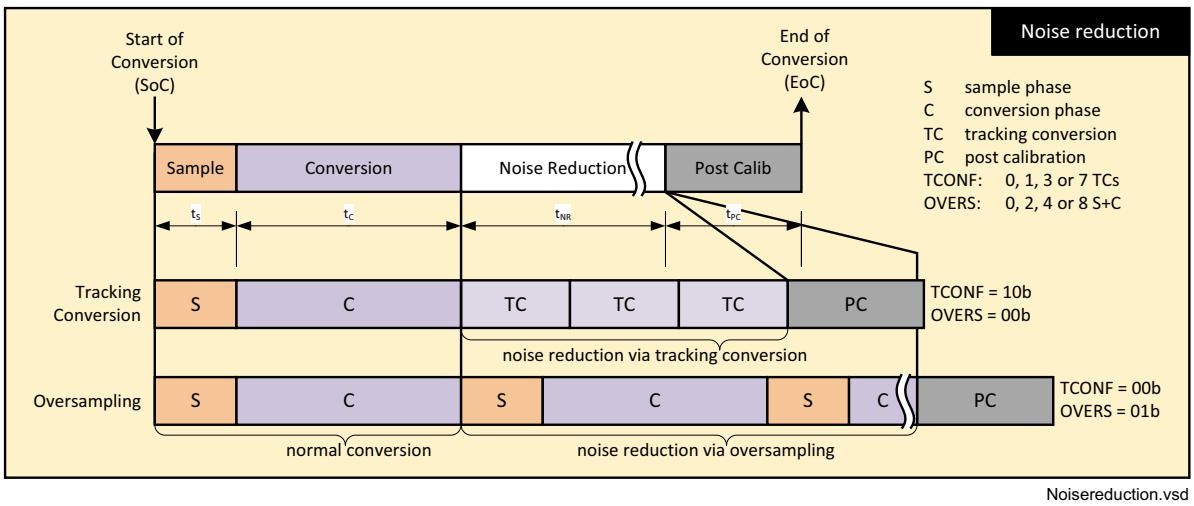


Figure 197 Noise reduction options

Tracking conversion scheme (TCONF)

- TCONF reduces the internal converter noise by averaging
- A programmable number of sub-conversion follow a normal conversion phase automatically. The number of TCONFs are configured in bitfield CONVCFGx.TCONF
- The TCONFs run the successive approximation on the five lowest significant bits (LSB), the higher significant bits are taken from the previous normal conversion
- The averaged kernel result (12 bit wide) is forwarded to the result generation
- The time for one tracking conversion t_{TC} is calculated according to following formula:

(16.4)

$$t_{TC} = 8 \times ADC\_CLK$$

Oversampling conversion scheme (OVERS)

- OVERS reduces the internal converter noise and the sampling noise by averaging
- A programmable number of OVERS follow a normal conversion phase. The number of additional OVERS are configured in bitfield CONVCFGx.OVERS
- The oversample timing is the same as for the conversions with OVERS = 0
- OVERS run the normal successive approximation
- The averaged kernel result (12 bit wide) is forwarded to the result generation

16.8.7 Post calibration

- The post calibration is an on-the-fly calibration to compensate device individual drift and temperature dependent effects.
- The post calibration follows the conversion phase.
- The post calibration is optional and can be programmed by channel control's bitfield CONVCFGx.PCAL.
- The time for the post calibration is calculated according to following formula:

(16.5)

$$t_{PC} = 10 \times ADC\_CLK$$

16.9 Sequencer control features

The sequencer allows automatic conversion sequences without software interference. The sequencer allows to trigger single conversions and conversion sequences with a deterministic timing.

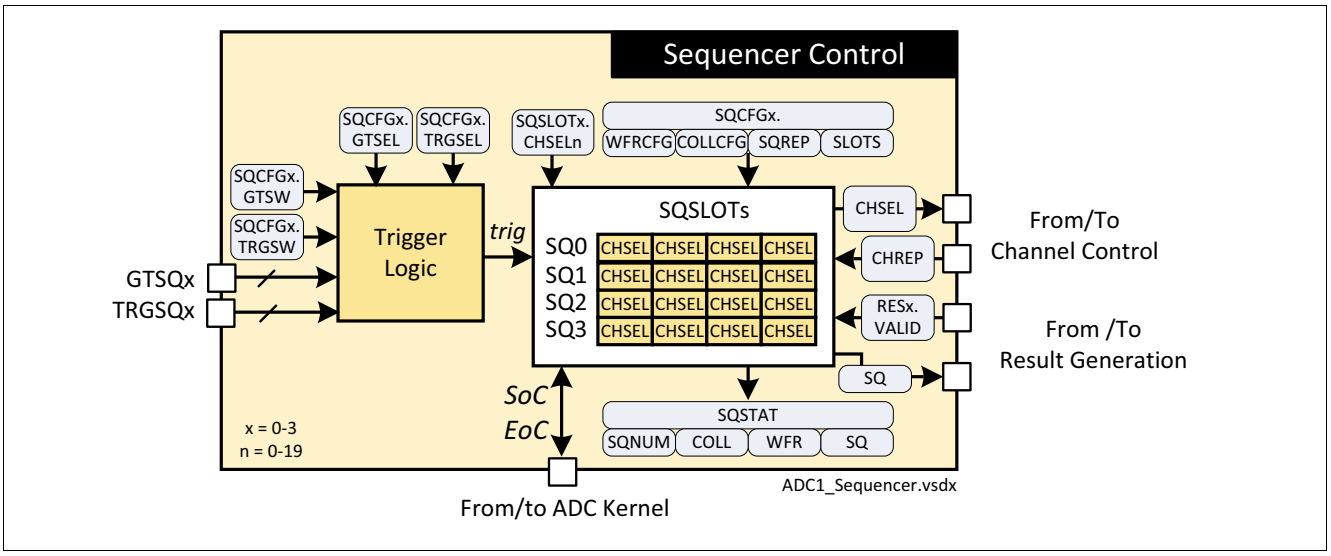


Figure 198 ADC1 block diagram with data flow

16.9.1 Sequence definition

The sequencer can generate four programmable sequences, configurable in SQCFGx (x = 0 to 3).

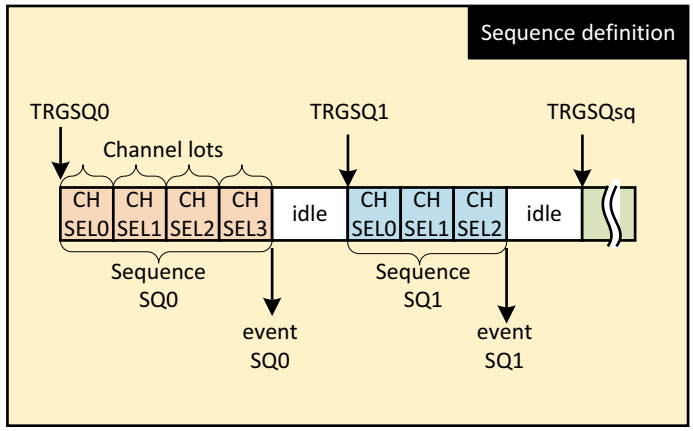


Figure 199 Sequence definition

Sequence trigger and gating logic

Each sequence can be triggered by hardware or software request events selected via SQCFGx.TRGSEL.

Each trigger request for a sequence can be gated by level signals selected via SQCFGx.GTSEL.

A sequence can be triggered by itself (endless once started) or by the previous sequence (round robin).

The external trigger and gating connections can be found in **Product definitions**, **ADC1 interconnections**.

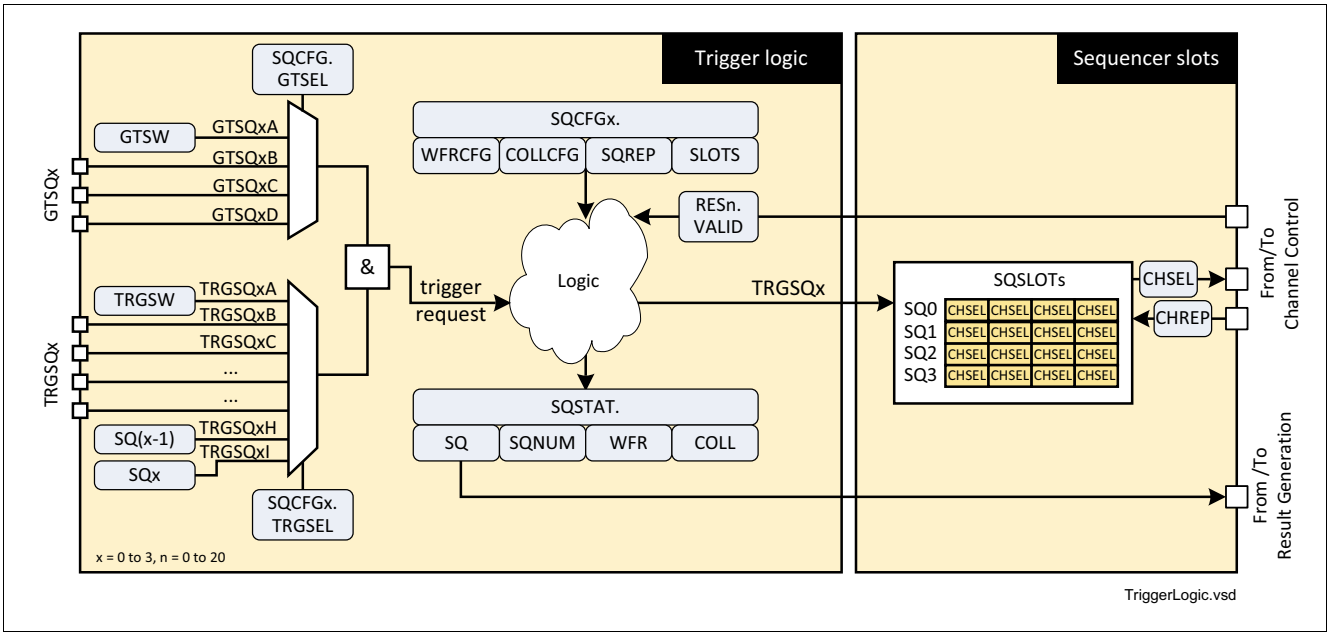


Figure 200 Sequencer trigger and gating logic

Sequence slot (SLOTS)

Each sequence has four channel slots, see **Figure 199**. The number of active channel slots is programmable in bitfield SQCFQx.SLOTS.

Each channel slot can be assigned to a channel via bitfield SQSLOTx.CHSELn (x = 0 to 3, n = 0 to 19).

Each channel has a repeat option, configurable in CHCFGn.CHREP. The channel associated to a slot is repeated according to its CHREP setting.

Sequencer status

The sequencer is either idle or active:

- It is set to active (SQx is started) in case the sequence start conditions (see COLL, WFR) are fulfilled
- It is set to idle after the SQx is finished

The status of a started sequence is displayed in SQSTAT.SQx.

The “sequence finished” event can request an interrupt if enabled in IEN0.IEN_SQ.

The number of the active sequence is indicated in SQSTAT.SQNUM.

Sequence repetition (SQREP)

Each sequence can be repeated automatically; therefore the sequence repetition counter in SQCFGx.SQREP can be programmed accordingly.

Sequence arbitration and collision (COLL)

Following two situations are called sequencer collision:

- In case of concurrent trigger requests (at the same time), the sequence with the lower number has priority over the higher number
- In case a trigger request comes in while a sequence is active, the ongoing sequence is always finished

In both cases a collision event is set in the sequence with the lower number SQSTAT.COLL which can request an interrupt if enabled IEN1.IEN_COLLx.

In both cases the trigger request which loses the arbitration can be handled in following manner:

- The trigger request is discarded in case the collision configuration in SQCFGx.COLLCFG=0
- The trigger request is set to pending in case the collision configuration in SQCFGx.COLLCFG=1; in this case it takes part in the next arbitration cycle once the ongoing sequence is idle

Sequence wait for read (WFR)

The wait for read mechanism protects unread results from being overwritten. This can be useful in case a set of result values must be coherent, i.e. belong to the same “set of measurements”.

Therefore each channel result register has a flag which indicates if the result has been read by the CPU/DMA or not, see RESn.VALID (n = 0 to 19). Valid means the result has not been read (VALID = 1).

Before the sequencer starts a sequence it checks all result valid flags (RESn.VALID) which are assigned to its channel slots. In case one or more valid flags are set the sequencer can handle this situation as following:

- The trigger request is discarded (the sequence is not started) if SQCFGx.WFRCFG = 1; a WFR event is indicated in SQSTAT.WFR and can request an interrupt if enabled in IEN1.IEN_WFRx. This way an unread result is not overwritten
- The trigger request is serviced (the sequence is started) if SQCFGx.WFRCFG = 0, i.e. a valid result will be overwritten with a new conversion result

Note: The WFR condition is not checked once a sequence is started, i.e. in case the same channel is assigned to different slots or a channel is repeated (CHREP). In these cases the result will be overwritten regardless of its VALID flag. The WFR condition is also not checked in case a sequence is repeated (SQREP).

16.9.2 Sequencer state machine

A conversion sequence acts according **Table 185** and **Figure 188**.

Note: An ongoing sequence cannot be interrupted or aborted.

16.9.3 Sequencer synchronization

For dynamic reconfiguration there is a shadow mechanism implemented for the sequencer SFRs SQCFG and SQSLOT. This allows software to write asynchronous to the running sequencer and ensures data coherency.

Note: Coherency means: a set of data in different SFRs belongs together.

A locking mechanism protects the settings for an active sequence. This protects the sequence setting itself but does not ensure data coherency for more than one sequence, i.e. in case settings for several sequences have to be updated at the same time.

The shadow mechanism allows to update several sequences with data coherency.

For motor control applications, the ADC sampling is triggered by external events generated by the PWM timing. The PWM registers need to be updated on-the-fly and have a similar shadow mechanism. In case the PWM registers and the ADC registers need coherency, the update mechanism of both the ADC and the PWM must be synchronized. This is possible via the external synchronization inputs EXTSTE and EXTSTR[F:A].

In case the synchronization feature is not used, a bypass is offered. This mechanism makes the shadow mechanism transparent in case the sequencer is idle. Therefore GLOBCONF.ISTE has to be set, see **Figure 201**. This is also a recommended method for initializing the ADC1, see **Table 188**.

The shadow mechanism works as following (see **Figure 201**):

- The bitfields SQCFGx. SLOTS, SQREP,WFRCFG, COLLCFG, TRGSEL, GTSEL and SQSLOTx.CHSEL are implemented as shadow SFR
- The shadow transfer mechanism is triggered via the “shadow transfer request (ST)” in case it is enabled via “shadow transfer enable (STE)”
- There are several options for ST:
  - A software trigger event via ST_xSW or
  - An external event EXTSTR[F:A]
  - The selection is done via ST_xSEL
- There are several options for STE:
  - A software trigger event via STE_xSW or
  - An external event EXTSTE
  - The selection is done via STE_xSEL
- The bitfields are grouped as following:
  - x = SQ: CHSEL, SLOTS, SQREP, WFRCFG, COLLCFG
  - x = TRG: TRGSEL
  - x = GTG: GTSEL

The shadow mechanism is configurable in SHDCTR like following:

- Select the source for the enable gate via:
  - SHDCTR.STE_SQSEL
  - SHDCTR.STE_TRGSEL
  - SHDCTR.STE_GTGSEL

- Select the source for the request event via:
  - SHDCTR.ST_SQSEL
  - SHDCTR.ST_TRGSEL
  - SHDCTR.ST_GTGSEL
- Enable the shadow transfer via:
  - STESHDCTR.STE_SQ = 1
  - SHDCTR.STE_TRG = 1
  - SHDCTR.STE_GTG = 1
- In case of software enable and request is selected, request shadow transfer via:
  - SHDCTR.ST_SQ = 1
  - SHDCTR.STE_TRG = 1
  - SHDCTR.STE_GTG = 1

Note: For initial initialization a shadow transfer can be forced via SHDCTR=0x00770000.

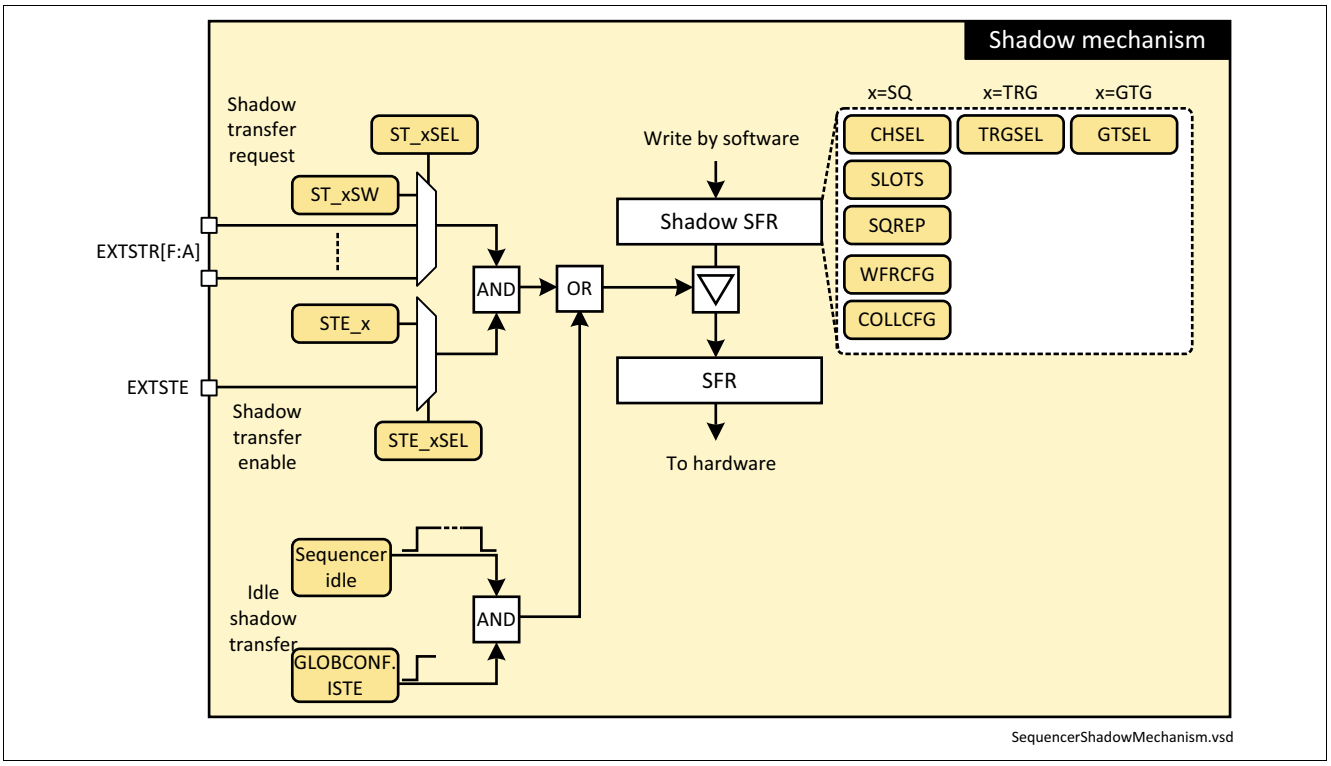


Figure 201 Hardware-software synchronization via shadow mechanism

Interrupt request

The sequencer has channel has 8 events (CMPUPx and CMPLOx) which can request an interrupt, see **Figure 187**.

These events are registered in CMPSTAT.CMP_LO_ISx, CMP_UP_ISx.

The events can be set by software in CMPSTATSET.CMP_UP_ISSET/CMP_LO_ISSET.

The events can be cleared by software in CMPSTATCLR.CMP_UP_ISCLR/CMP_LO_ISCLR.

The request can be enabled in IEN0.IEN_UPx/LOx.

The interrupt node can be assigned to the NVIC via INP2.INP_CMP_LOx/INP_CMP_UPx.

DMA request

The compare channel has 8 events (CMPUPx and CMPLOx) which can request a DMA, see **Figure 187**.

Therefore the DMA channel has to be assigned in SCU.DMAP_ADC.ADC1_CMPHI0/ADC1_CMPLO0 and the DMA has to be configure accordingly, see chapter MCU/DMA.

16.10 Channel control features

The channel control provides the individual channel settings which are used by the sequencer, by the ADC1 kernel and by the result generation. A high flexibility for application specific use cases can be achieved.

There are 20 channel configuration registers CHCFGn (n = 0 to 19).

Each channel control register has one corresponding result register RESn (n = 0 to 19).

Note: The channel control has no shadow mechanism (unlike the sequencer). Therefore dynamic reconfiguration has to be done while sequencer is idle.

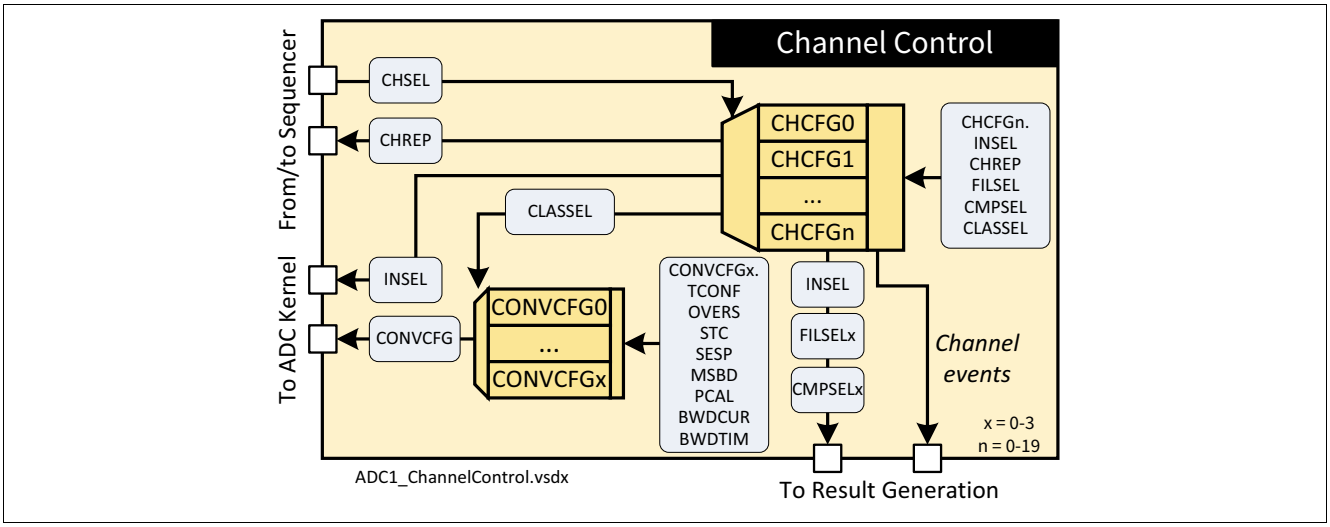


Figure 202 ADC1 block diagram with data flow

Analog input selection (INSEL)

The analog input selection is done in CHCFGn.INSEL. An analog input AN[i] can be freely assigned to one or more channels CHn. It is possible to assign the same analog input to multiple channels. Each channel has one dedicated result register (CHn --> RESn). This allows to measure the same analog input multiple times within one or different sequences but with different result registers. This relaxes the real-time condition for reading the results, see **Chapter 16.12.6**.

The sequencer selects via SQSLOT.CHSEL the CHCFGn. The channel control provides the INSEL to the ADC1 kernel and to the result generation.

Channel repeat (CHREP)

A channel can be repeated automatically. Therefore the numbers of repetitions are configured in CHCFGn.CHREP (CHREP = 0 means no repetitions, i.e. one conversion), see **Chapter 16.12.3**.

The sequencer selects via SQSLOT.CHSEL the CHCFGn. The channel control provides the CHREP to the sequencer which starts the conversion (SoC) and repeats the channel according to the value in CHREP.

Filter channel selection (FILSEL)

For result postprocessing a digital channel CHn can be assigned to one of four digital filter channels. The selection of the filter channel is done via CHCFGn.FILSEL. With FILSEL=4, no filter channel is assigned.

The sequencer addresses the channel control via SQSLOT.CHSEL the CHCFGn. The channel control provides the FILSEL to the result generation. The result generation assigns the corresponding filter channel to the result register for CHn.

The filter configuration itself is done in FILTCFG, see **Chapter 16.11.3**.

Compare channel selection (CMPSEL)

For result postprocessing a digital channel CHn can be assigned to one of four compare channels. The selection of the compare channel is done via CHCFGn.CMPSEL. With CMPSEL=4, no compare channel is assigned.

The sequencer addresses the channel control via SQSLOT.CHSEL which provides the channel individual CMPSEL setting to the result generation. The result generation compares the RESn of CHn with the selected compare channel. The compare events and IRQs can be generated accordingly.

The compare configuration itself is done in COMPCFGx, see **Chapter 16.11.4**.

Conversion class selection (CLASSEL)

The ADC1 kernel can be configured for every of its conversions (CONFCFG). A channel can be assigned to one our of four different conversion classes. The selection is done via CHCFGn.CLASSEL. This selects the corresponding CONVCFG setting.

The sequencer selects via SQSLOT.CHSEL the CHCFGn. The channel control selects via CLASSEL the corresponding CONVCFG settings, which are provided to the ADC1 kernel.

Conversion configuration (CONVCFG)

The ADC1 kernel configuration for every of its conversions is taken from CONFCFGx.

The sequencer selects via SQSLOT.CHSEL the CHCFGn. The channel control selects via CLASSSEL the corresponding CONVCFG settings, which are provided to the ADC1 kernel. Following settings are possible, see **Chapter 16.8**:

- Tracking conversion configuration TCONF
- Oversampling OVERS
- Sample time control STC
- Spread early sample point SESP
- Most significant bit doubling MSBD
- Post calibration PCAL
- Broken wire detection time BWDTIM
- Broken wire detection current BWDCUR

Interrupt request

The sequencer control has 12 events (SQx, COLLx, WFRx, x= 3 to 0) which can request an interrupt, see **Figure 187**.

These events are registered in SQSTAT.SQx/COLLx/WFRx.

The events can be set by software in SQSTATSET.SQSETx/COLLSETx/WFRSETx.

The events can be cleared by software in SQSTATCLR.SQCLRx/COLLCLRx/WFRCLRx.

The interrupt request can be enabled in IEN0.SQx and in IEN1.COLLx/WFRx.

The interrupt node can be assigned to the NVIC via INP3.INP_SQx/COLLx/WFRx.

DMA request

The sequencer control has 2 events (SQ0 and SQ1) which can request a DMA, see **Figure 187**.

Therefore the DMA channel has to be assigned in SCU.DMAP_ADC.ADC1_SQx (x = 1, 0) and the DMA has to be configure accordingly.

16.11 Result generation features

The result generation is according to following digital signal path:

- The analog input calibration
- The result filters with configuration options
- The result register with a valid control
- The compare registers with configuration options
- The event handling

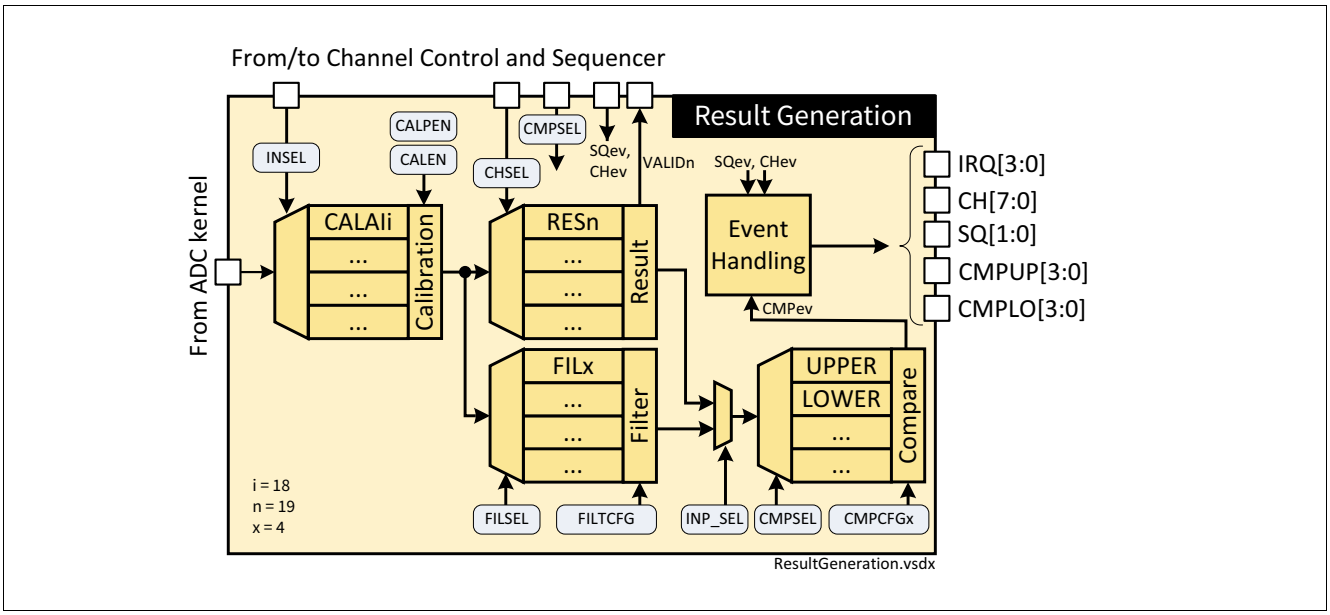


Figure 203 ADC1 block diagram with data flow

16.11.1 Analog input calibration

There is a calibration mechanism which compensates offset and gain errors of the signal chain. These are:

- Offset and gain errors of the ADC1
- Offset and gain errors of the attenuators
- Offset and gain errors of the reference voltage VREF5V

All these errors are summed up in the overall offset (CALOFFS, factor a) and the overall gain (CALGAIN, factor b).

The calibration is device specific and individual for each analog input. The calibration values are determined at the device production site and stored in the 100 TP page of the flash. After a RESET_TYPE_4 they are loaded by firmware into the calibration registers CALAIi (i = 1, 3, 5, 6, 7, 9, 11, 13, 15 to 26).

The calibration is enabled per channel in CALEN.CALENi. In case the calibration is disabled the factors are fixed to one (a = b = 1).

The output of the calibration is always a 14 bit wide result, regardless of CALENi.

The calibration values are protected to avoid unintentional programming. The protection can be disabled for user access via CALPEN.CALPENi = 0.

The calibration unit receives the input selection from the channel configuration via CHCFGn.INSEL and selects the calibration value accordingly. This value is used for calculating the calibrated result (14 bit) from the raw result (12bit) which is taken from the ADC1 kernel. The calibrated result is forwarded to the result register selected via CHCFGn.CHSEL and to the result filters selected via CHCFGn.FILSEL.

The calibrated result is calculated according to following formula:

(16.6)

$$y = a + (1 + b) \times x$$

With:

- a: CALOFFS
- b: CALCGAIN
- x: raw value from ADC1 kernel (12 bit)
- y: calibrated value (14 bit) Offset and gain errors of the reference voltage VREF5V

16.11.2 Result register and valid control

There are 20 result registers RESn (n = 0 to 19) which are one-to-one assigned to the digital channels CHn.

The result register contain the 14-bit result value (RESn.RESULT) and the valid information (RESn.VALID).

The digital signal path is like following:

- The result register RESn is selected by the sequencer via CHCFGn.CHSEL
- The result input is taken from the calibrated result

The valid control is available for each result register and is used in the sequencer control for the wait for read feature (SQCFG.WFRCFG). The flag RESn.VALID is set according to following definition:

- VALID is set if the result is updated with a new value
- VALID is cleared if the result is read by CPU/DMA

The result events are available as interrupt request and DMA request according to the scheme in **Figure 187**.

16.11.3 Result filter

There is an optional and configurable first order IIR filter which can be used to post-process the digital result. This is especially useful together with the channel repeat functionality for oversampling the input signal.

There are four independent filter channels available. The filter channel can be selected in CHCFGn.FILSEL, see **Chapter 16.10**.

The filter is configured in FILTCFG.COEFAx (x = 0 to 3).

The filter result is calculated according to following formula, see also **Figure 204**:

(16.7)

$$y[n] = a \times x[n] + (1 - a) \times y[n - 1]$$

With:

- a = COEF_Ax, selectable from 1/2, 1/4, 1/8, 1/16
- x: result from input calibration (12 to 14 bit)
- y: filter output

This filter is a first order IIR-type with one pole:

(16.8)

$$y[n] = b0 \times x[n] + b1 \times x[n - 1] + a1 \times y[n - 1]$$

With:

- b0 = COEF_Ax, selectable from 1/2, 1/4, 1/8, 1/16

- b1 = 0
- a1 = 1 - b0

The 14-bit wide filter result is available in FILx.FILRESULT. The FILRESULT can be forwarded to the compare unit via CMPCFGx.INP_SEL = 1.

The filter status flag in FILSTAT.FILx indicate that a new filter result is available.

The events can be set by software in FILSTATSET.FILSETx.

The filter status can be cleared in FILSTATCLR.FILCLRx.

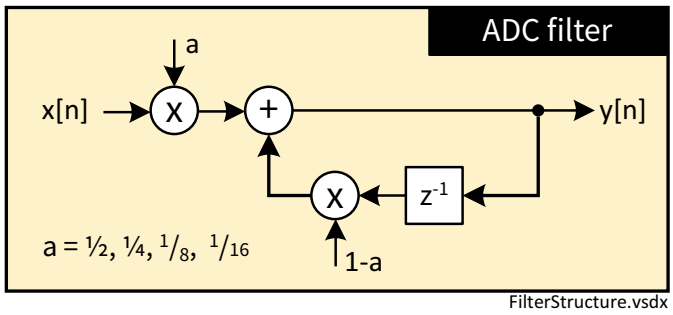


Figure 204 Filter structure

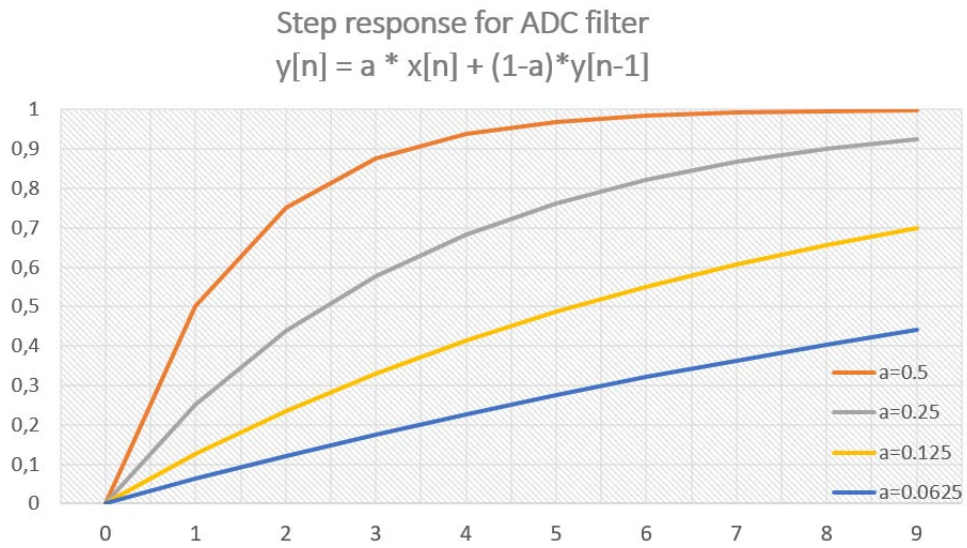


Figure 205 Filter step response

16.11.4 Compare channels

There are four digital compare channels with programmable thresholds for background signal supervision with limit checking, see **Chapter 206**.

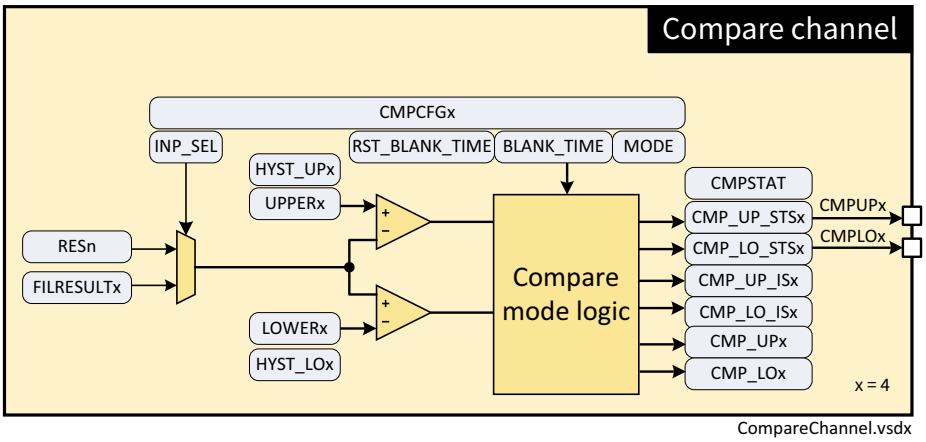


Figure 206 Compare channel

The digital signal path is like following, see **Figure 206**:

- The compare channel is selected by the channel control via CHCFGn.CMPSEL
- The input is taken from the result register RESn.RESULT or from the filter result FILx.FILRESULT. The selection is done via CMPCFGx.INP_SEL
- There are four programmable 8-bit threshold values (left-aligned) for an upper and lower boundary in CMPCFGx.UPPER, CMPCFGx.LOWER together with a selectable hysteresis in CMPCFGx.HYST_UP, CMPCFGx.HYST_LO
- In case a boundary is crossed, the corresponding status flag is set according to the selected compare mode in CMPCFGx.MODE. The compare rule can be taken from **Table 187**. There are following status flags indicating the compare event:
  - latched status bit (CMPSTAT.CMP_LO_STSx, CMPSTAT.CMP_UP_STS), must be cleared by software
  - life status bit (CMPSTAT.CMP_LOx, CMPSTAT.CMP_UPx), is cleared by hardware
  - interrupt status (CMPSTAT.CMP_LO_ISx, CMPSTAT.CMP_UP_IS), must be cleared by software
- The latched status flags are interconnected to other peripherals, see **Product definitions**, **ADC1 interconnections**

There are following configuration options:

Compare mode (MODE)

There are three compare modes (via CMPCFGx.MODE) with dedicated compare logic, see **Table 187**:

- Range mode for detection if a signal is leaving a voltage range between an UPPER and a LOWER boundary
- OV mode for detection if a signal is above a lower and a higher boundary (two thresholds)
- UV mode for detection if a signal is below a higher and a lower boundary (two thresholds)

Table 187 Compare mode logic

Status	Status	RESx	Range mode	OV mode	UV mode
UPx	set	RESx	> UPPER	> UPPER	< UPPER
UPx	clear	RESx	≤ UPPER - HYST_UP	≤ UPPER - HYST_UP	≥ UPPER + HYST_UP
LOx	set	RESx	< LOWER	> LOWER	< LOWER
LOx	clear	RESx	≥ LOWER + HYST_LO	≤ LOWER - HYST_LO	≥ LOWER + HYST_LO

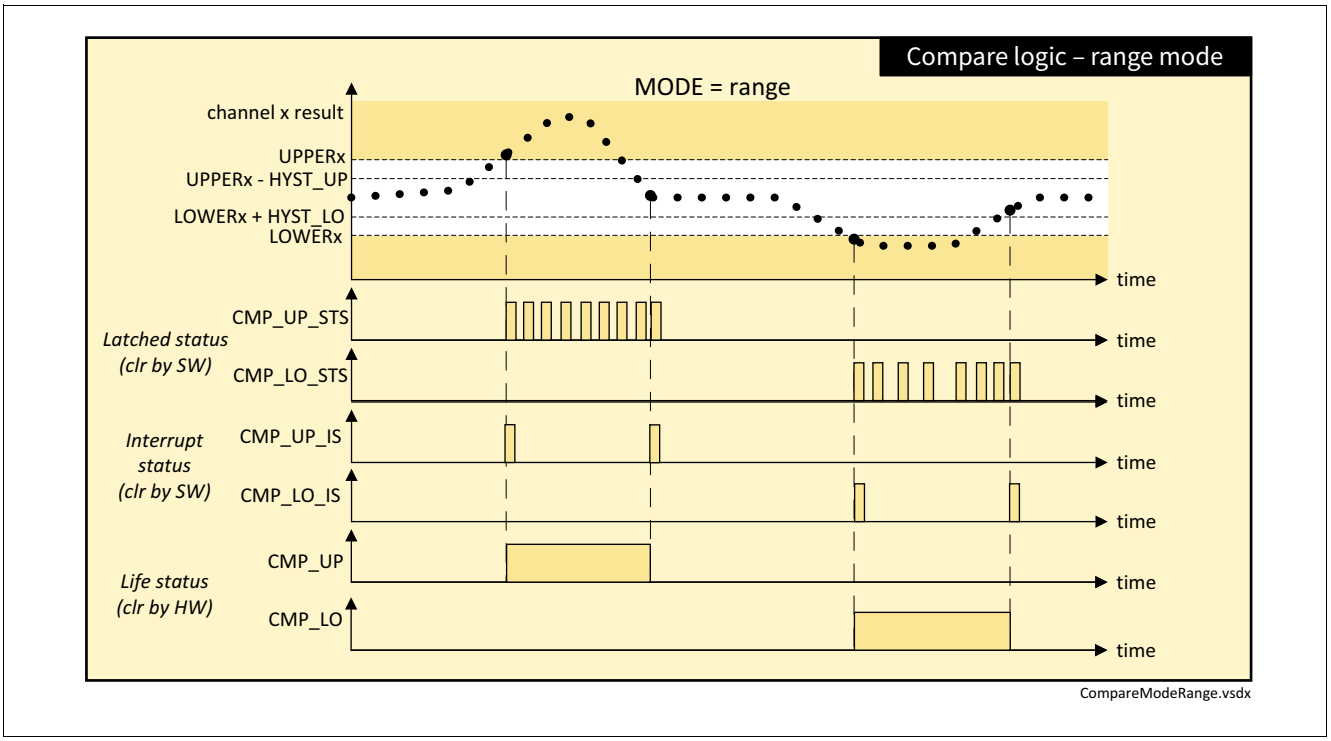


Figure 207 Compare mode logic range mode

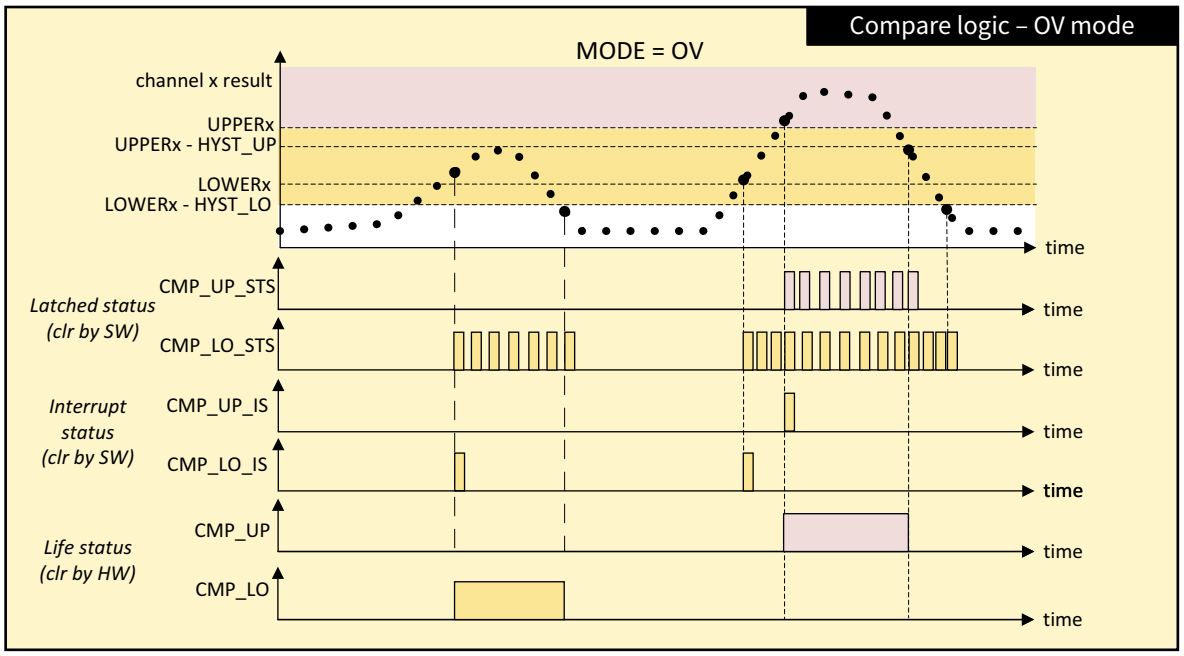


Figure 208 Compare mode logic OV mode

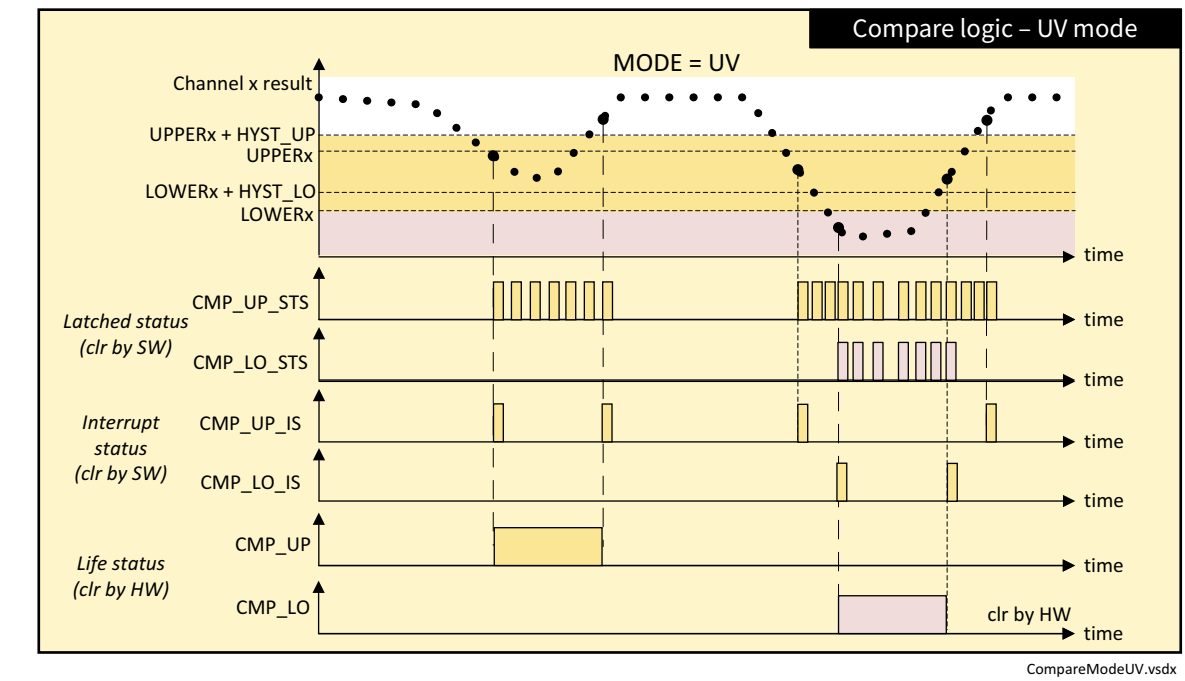


Figure 209 Compare mode logic UV mode

Blank time (BLANK_TIME)

There is a blank timer for each compare channel, which counts the number of programmed conversions. The counter starts with the first conversion and ends when the programmed number is reached. While it is counting, the compare logic is gated.

- The blank time is configured in CMPCFGx.BLANK_TIME
- The blank timer can be restarted via CMPCFGx.RST_BLANK_TIME

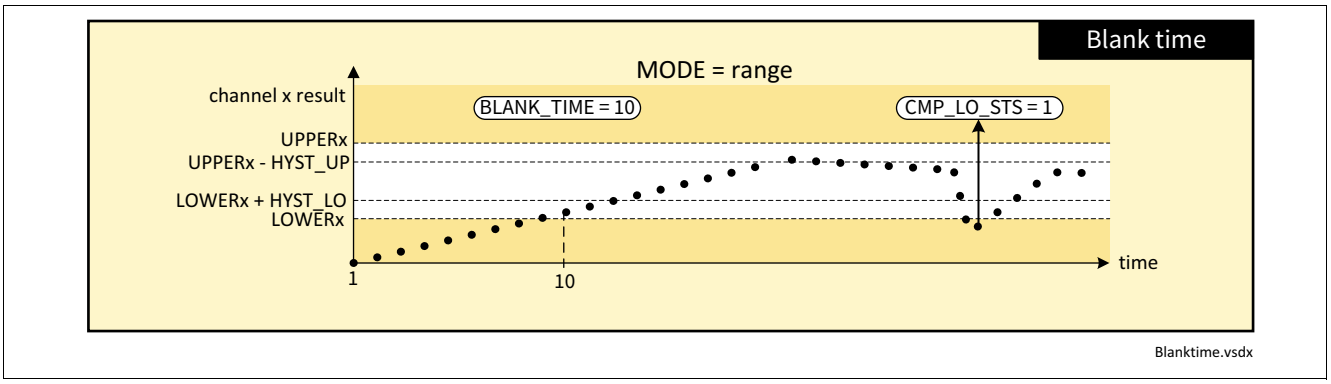


Figure 210 Blank time example

Interrupt request

The compare channel has 8 events (CMPUPx and CMPLOx) which can request an interrupt, see **Figure 187**.

These events are registered in CMPSTAT.CMP_LO_ISx, CMP_UP_ISx.

The events can be set by software in CMPSTATSET.CMP_UP_ISSET/CMP_LO_ISSET.

The events can be cleared by software in CMPSTATCLR.CMP_UP_ISCLR/CMP_LO_ISCLR.

The request can be enabled in IEN0.IEN_UPx/LOx.

The interrupt node can be assigned to the NVIC via INP2.INP_CMP_LOx/INP_CMP_UPx.

DMA request

The compare channel has 8 events (CMPUPx and CMPLOx) which can request a DMA, see **Figure 187**.

Therefore the DMA channel has to be assigned in SCU.DMAP_ADC.ADC1_CMPHI0/ADC1_CMPLO0 and the DMA has to be configure accordingly, see chapter MCU/DMA.

Interconnects

The compare channel has 8 events (CMPUPx and CMPLOx) which are interconnected to other peripherals, see **Product definitions**, **ADC1 interconnections**.

16.12 Programmer's guide

16.12.1 Initial initialization sequence (example)

In the following an initialization example for ADC1 is described in **Table 188**. This initializes a sequence according to **Figure 211**.

- The sequence trigger sources are done by software
- Gating is disabled by choosing SQCFGx.GTSEL = SWQGTxA and setting bit SQCFGx.GTSW = 1
- The number of active slots per sequence is four, SQCFGx.SLOTS =4
- The collision is configured to "set to pending", SQCFX.COLLCFG = 1
- As the software trigger for all four sequences happens at the same time, a collision happens immediately with the trigger event. The sequencer starts with the highest prior (SQ0) and sets the other to pending until the SQ0 is finished. Then it starts SQ1 etc.
- The SQ3 event is configured as IRQ source. Within this ISR the RESn can be read

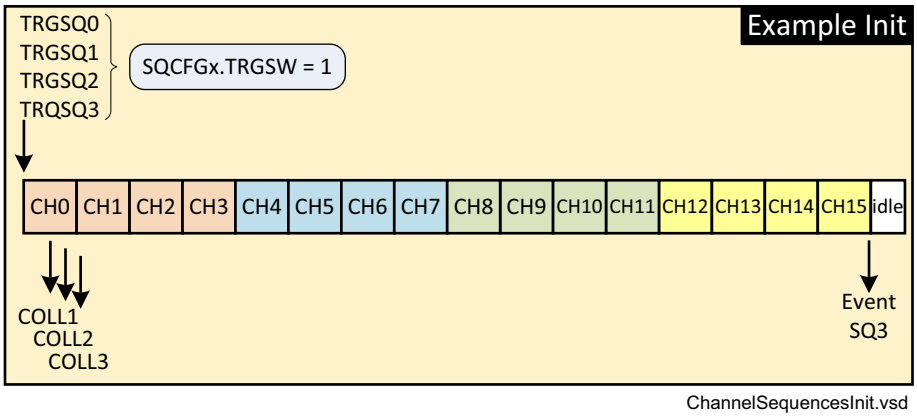


Figure 211 Initialization example

Table 188 Initialization sequence

Step	Description	Option	Selection/criteria	Configuration example (pseudo code)
1	Ensure reference	manda-tory	A reference voltage must be present at VAREF	VAREF_CTRL.EN=1 /* e.g. VREF5V is enabled */
2	Configure clock	manda-tory	Define ADC_CLK depending on CLK input	CLKCON.CLKDIV = 1; /* fadc=60 MHz, if fsys0=60 MHz */
3	Enable ADC	manda-tory	After enable wait for READY bit	GLOBCONF.EN = 1; /* enable ADC kernel */ wait for SUSSTAT.READY with timeout
4	Shadow transfer	manda-tory	Enable shadow transfer for initialization	GLOBCONF.ISTE =1 /* idle shadow transfer enable */
5	Debug behavior	optional	Debug mode and enable	SUSCTR.SUSMOD = 1; /* soft suspend */ SUSCTR.SUSEN = 0; /* disable suspend */

Table 188 Initialization sequence (cont'd)

Step	Description	Option	Selection/criteria	Configuration example (pseudo code)
6	Sample time configuration	manda-tory	Assign sample time to conversion classes	CONVCFG0.STC = 4; /* e.g. MV channels tsamp = 6*ADC_CLK ≥ 200 ns @fsys0 =60 MHz */ CONVCFG1.STC = 13; /* e.g. HV channels tsamp = 32*ADC_CLK ≥ 1000 ns @fsys0 =60 MHz */
7	Doubling the MSB time	optional	Define MSBD for each conversion class	CONVCFG0.MSBD = 1; /* e.g. MV: MSBD enabled */ CONVCFGx.MSBD = 0; /* e.g. HV: MSBD disabled */
8	Broken wire detection	optional	Define broken wire timing and strength for each conversion class	CONVCFGx.BWDTIM = 0; /* e.g. BWD disabled */ CONVCFGx.BWDCUR = 0; /* e.g. IBWD selected */
9	Post-calibration	optional	Define post calibration for each conversion class	CONVCFGx = 1; /* e.g. PCAL enabled */
10	Noise reduction	optional	Define noise reduction feature per conversion class	CONVCFGx.SESP = 0; /* SESP disabled */ CONVCFGx.TCONV = 0; /* TCONV disabled */ CONVCFGx.OVERS = 0; /* OVERS disabled */
11	Filter configuration	optional	Select filter coefficient for filter channel	FILCFG.COEFF_Ax = 0; /* Ax = 1/2 */
12	Compare source selection	optional	Select source for compare channel, either from RESn or FILx	CMPCFGx.INP_SEL = 0; /* 0 = RESn, 1 = FILx */
13	Compare mode selection	optional	Select the mode for the compare channel(s)	CMPCFGx.MODE = 0; /*e.g. 0 = range mode */
14	Compare configuration	optional	Configure thresholds and hysteresis for the compare channel(s)	CMPCFGx.UPPER = xx; /* upper threshold */ CMPCFGx.HYST_UP = xy; /* select upper hysteresis */ CMPCFGx.LOWER = xx; /* lower threshold */ CMPCFGx.HYST_LO = xy; /* select lower hysteresis */
15	Compare blank time	optional	Configure blank time for the compare channel(s)	CMPCFGx.BLANK_TIME = 0; /* e.g. 0 = blank time is off */ CMPCFGx.RST_BLANK_TIME =1; /* restart blank timer */

Table 188 Initialization sequence (cont'd)

Step	Description	Option	Selection/criteria	Configuration example (pseudo code)
16	Assign analog inputs to digital channels	manda-tory	Assign all AN[i] to CH[n]	CHCFGn.INSEL = AN[i]; e.g. CHCFG0.INSEL = 0x01; /* AN1 --> CH0 */ CHCFG1.INSEL = 0x03; /* AN3 --> CH1 */ CHCFG2.INSEL = 0x04; /* AN4 --> CH2 */ ... CHCFG14.INSEL = 0x11; /* AN18 --> CH15 */ CHCFG15.INSEL = 0x11; /* AN18 --> CH15 */
17	Channel repetition	optional	Define the channel repetition	CHCFGn.CHREP = 0; /* x=0 for zero repetitions */
18	Filter selection	optional	Assign a 4 filter channel to CH[n]	CHCFGn.FILSEL = 4; /* x = 4 for no filter assigned */ CHCFGn.FILSEL = x; /* x = 0 to 3: FIL0 to FIL3 */
19	Compare channel selection	optional	Assign a compare channel to CH[n]	CHCFGn.CMPSEL = 4; /* x = 4 for no compare assigned */ CHCFGn.CMPSEL = 4; /* x = 4 for no compare assigned */
20	Configure kernel conversion options	manda-tory	Assign a kernel conversion class per CH[n]	CHCFGn.CLASSEL = 0; /* x=0 for CONVCFG0 */
21	Calibration configuration	optional	Define calibration options per analog input	CALPEN.CALPENi = 1; /* calibration protection enabled */ CALEN.CALENi = 1; /* calibration enabled */
22	Slots per sequence	manda-tory	Define the number of slots per sequence	SQCFGx.SLOTS = 4; /* i = 0 for no slot, i = 4 for 4 slots */
23	Sequence repetitions	optional	Define repetitions per sequence	SQCFGx.SQREP = 0; /* i = 0 for 0 repetitions */
24	Collision configuration	optional	define the collision handling per sequence	SQCFGx.COLLCFG = 1; /* 1: collision is set to pending */
25	Wait for read configuration	optional	Define the wait for read handling per sequence	SQCFGx.WFRCFG = 0; /* 0: WFR is disabled */
27	Trigger selection	manda-tory	Assign a trigger event per sequence	SQCFGx.TRGSEL = 0; /* 0: SW trigger */
28	Gating selection	manda-tory	Assign a gating condition per sequence	SQCFGx.GTSEL = 0; /* 0: SW gating */
29	Gating by software	manda-tory	Define gating for the trigger per sequence	SQCFGx.GTSW = 1; /* 0: no gating */

Table 188 Initialization sequence (cont'd)

Step	Description	Option	Selection/criteria	Configuration example (pseudo code)
30	Sequence slot	manda-tory	Assign channels to slots; SLOT[i] of SQ[x] with CHn via CHSEL	SQSLOTx.CHSEL[i] = n; /* n = 0 for CH0, n =19 for CH19 */ e.g. SQSLOT0 = 0x03020100; /* CH3 - CH2 - CH1 - CH0 */ SQSLOT1 = 0x07060504; /* CH7 - CH6 - CH5 - CH4 */ SQSLOT2 = 0x0B0A0908; /* CH11 - CH10 - CH9 - CH8 */ SQSLOT3 = 0x0F0E0D0C; /* CH15 - CH14 - CH13 - CH12 */
31	Interrupt node pointer	optional	Configure node pointer	INP0 = 0x0; /* all CHn events to ADC.IRQ0 */ INP1= 0x0; /* all CHn events to ADC.IRQ0 */ INP2= 0x00000055; /* all CMPx events to ADC.IRQ1 */ INP3= 0x00FFFFAA; /*all SQ events to IRQ2, all COLLx and WFRx to ADC.IRQ3*/
32	Interrupt enable	optional	Enable events for interrupt request	IEN0 = 0x00800000; /* SQ3 enabled, all CHn and CMPx events disabled */ IEN1 = 0; /* all COLLx and WRFx events disabled */
33	Trigger event	optional	Create trigger event by SW	SQCFGx.TRGSW = 1 /* trigger event */
34	Shadow transfer	optional	Disable shadow transfer in idle	GLOBCONF.ISTE = 0 /* idle shadow transfer disable */
35	Shadow transfer	optional	Define shadow transfer for updating SQCFGx and SQSLOTx	SHDCTR = 0x00770000; /* all selection to SW, and force shadow transfer, STE_SQSEL = STE_TRGSEL = STE_GTSEL = ST_SQSEL = ST_TRGSEL = ST_GTSEL = 0; STE_SQ = STE_TRG = STE_GTG = ST_SQ = ST_TRG = ST_GTG = 1; all SQx start by SW */

36	ISR	optional	Handle IS flags accordingly	if (SQSTAT.SQ3 == 1) { /* SQ3 event ? */
				var_ch0 = RES0.RESULT; /* read result ch0 */
				var_ch1 = RES1.RESULT;
				var_ch2 = RES2.RESULT;
				...
				var_chn = RESn.RESULT;
				SQSTATCLR.SQx = 0; } /* clear SQ3 event if set */

-	Dynamic reconfiguration	yes	Note: e.g. blank time has to be restarted when filter is used with compare and dynamically reconfigured	CMPCFGx.RST_BLANK_TIME = 1; /* restart blank timer */
-	DMA request	yes	Configure DMA accordingly if assigned	See DMA

16.12.2 Sequencer example 1 (all possible sequences are active))

In this example all four possible sequences are active, a result filter is used, a DMA transfer is requested and a collision occurs, see **Figure 212**.

- The sequence trigger sources come from external, e.g. by CCU7 events
- Gating is disabled by choosing SQCFGx.GTSEL = SWQGTxA and setting bit SQCFGx.GTSW = 1
- The number of active slots per sequence is different
- For SQ0, a result filter is assigned to CH1 because two consecutive CH1 conversions are triggered and CPU has probably no time to read the RES1 before it is overwritten with the second conversion. Instead the filtered result can be read with the SQ0 event
- For SQ1, two consecutive CH4 conversions are configured. A DMA request is configured in order to relax the real-time condition for reading the RES4 twice via CPU
- The TRGSQ2 comes while the SQ1 is still active, hence a COLL2 event is generated. COLL2 can be used to indicate that the CH9 and CH8 conversions did not happen at the appropriate time. In case the exact sampling point for the associated analog inputs is importance (e.g. because it is an AC signal) the values can be disregarded for further post-processing
- All SQx events can be used to trigger a separate interrupt. Alternatively the SQ3 event can be used as an “end-of-SQ0-SQ1-SQ2-SQ3-interrupt” for reading all results within one interrupt service routine

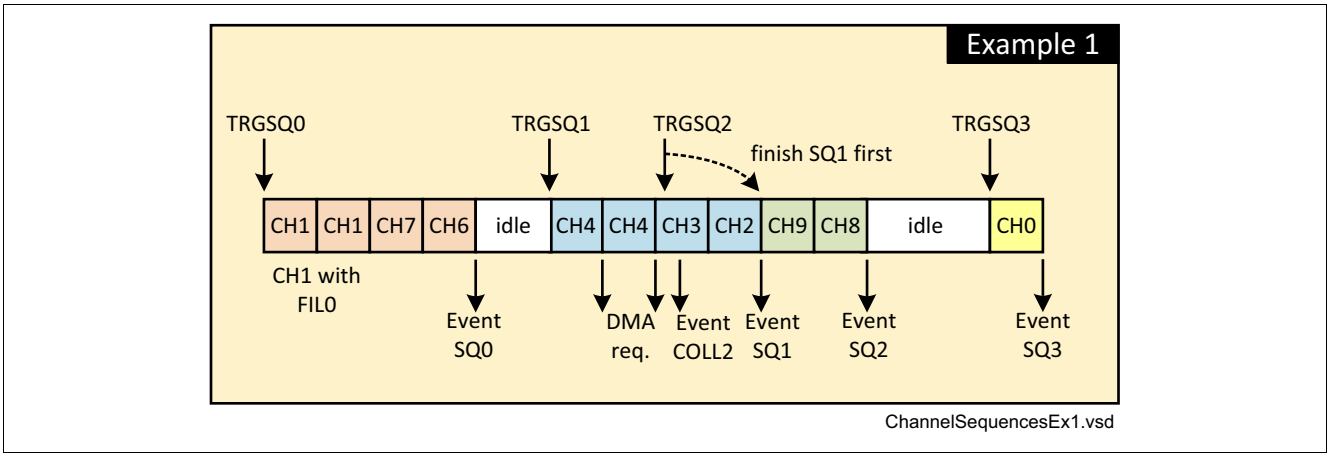


Figure 212 Sequence example 1

16.12.3 Sequencer example 2 (SQ3 is used)

In this example SQ3 is used with a multiple conversion of CH0 and a result filter, see **Figure 213**.

- The trigger event is generated by setting SQCFG3.TRGSW = 1
- The gating is disabled by choosing bitfield SQCFG3.GTSEL = SWQGT3A and setting bit SQCFG3.GTSW = 1
- There is only one slot for SQ3 (slot 0) used, which is assigned to channel CH0
- The channel CH0 is repeated 7 times (total 8 conversions). Therefore the channel repeat counter in CHCFG0.CHREP =7
- The wait for read feature is disabled (CHCFG0.WFR=0) because of the result filter for CH0 which allows to read the filtered result in FIL0.FILRESULT

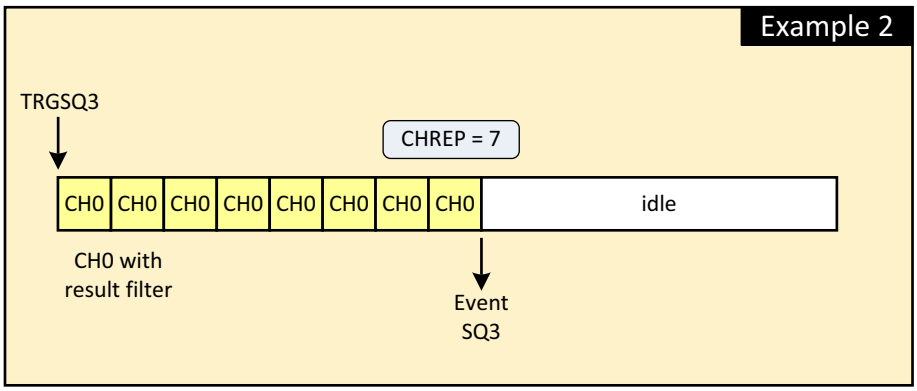


Figure 213 Sequence example 2

16.12.4 Sequencer example 3 (wait-for-read)

In this example the effect of a wait-for-read is shown, see **Figure 214**.

- The WFR is enabled for both sequences
- The SQ1 event requests an interrupt
- The TRGSQ0 comes before the CPU has read RES1, RES2, RES3. Therefore CH1, CH2, CH3 are not converted in the second SQ1, the conversions are skipped. The WFR event is set and can be used as indicator that CH1, CH2, CH3 results are valid but old, i.e. from the previous SQ0

Note: This is useful if coherency for a set of data is important. In this case, CH1 to CH8 results belong together. With the WFR mechanism it can be avoided that the second SQ0 sequence would overwrite the unread RES1-3.

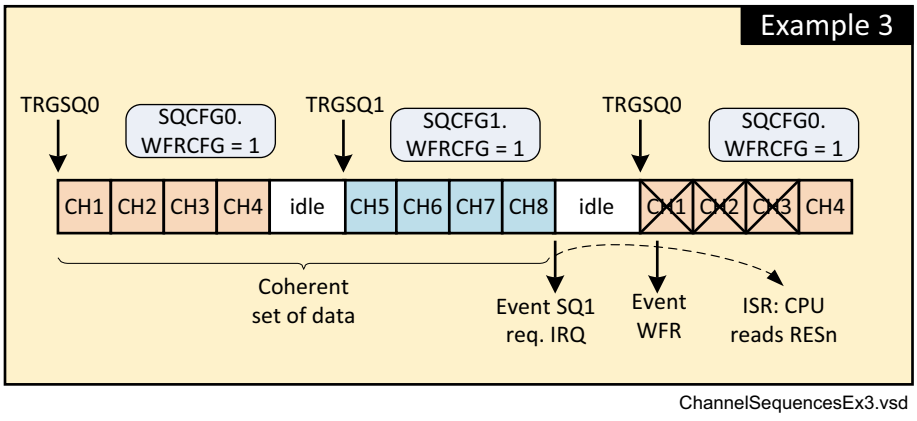


Figure 214 Sequence example 3

16.12.5 Sequencer example 4 (sequences trigger themselves)

In this example the sequences trigger themselves, see **Figure 215**.

- The trigger for SQ0 comes from an external event (e.g. a timer)
- The trigger for SQ1 comes from the SQ0 event
- The trigger for SQ2 comes from SQ1 event

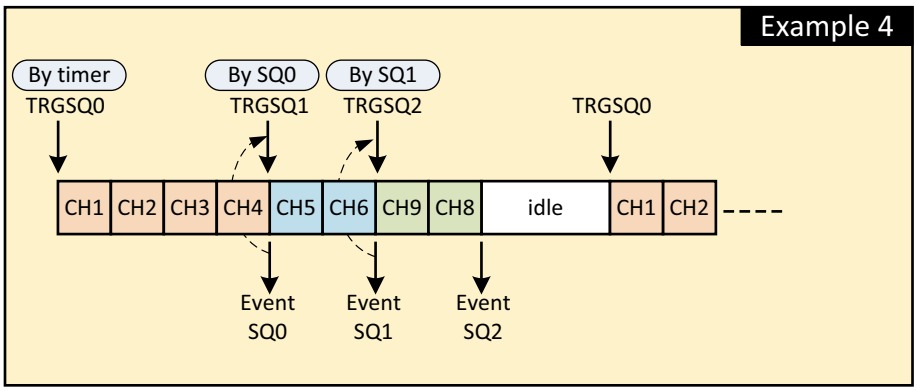


Figure 215 Sequence example 4

16.12.6 Sequencer example for single shunt BLDC application

The following example shows how to measure the relevant signals in a single shunt topology using a space vector modulation scheme (SVM), see **Figure 216**.

- The B6-Bridge is controlled by CCU7 signals (CC70/CC71/CC72) via the BDRV, generating the SVM scheme
- The DC link voltage (VDC) is measured by the ADC1 at VDH input pin
- The DC link current (IDC) is measured by the ADC1 over the CSA output. From IDC the appropriate phase currents (Iu, Iv, Iw) can be reconstructed according to the known SVM scheme
- Several other signals, e.g. supply voltage (VS), temperature (via P2.x), analog rotary sensor (via P2.x) have to be measured additionally (not shown here)

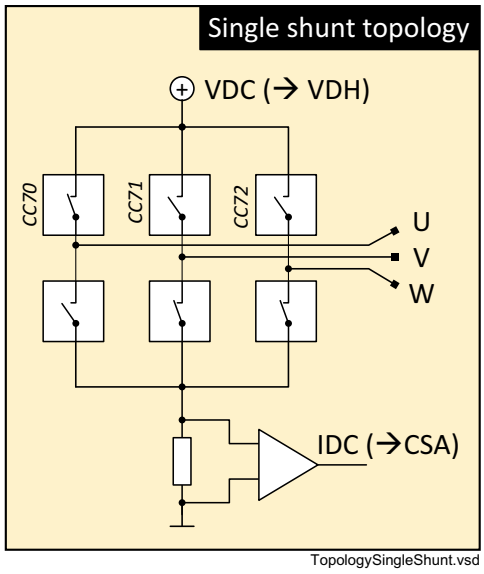


Figure 216 Single shunt topology

As it can be seen from **Figure 217**, both the VDC and the IDC signals change its shape within the PWM period:

- VDC shows a periodic voltage ripple; the VDC can be measured either at a “quiet” time within the PWM scheme, e.g. at the T12ZM and/or at the T12PM event (both are equidistant trigger points). Alternatively it can be measured together with the IDC. This has the advantage that a coherent set of data IDC-VDC is

available. It has the disadvantage that the VDC measurement might be at the switching event, where noise effects can influence the measurement.

- IDC is a pulsed and spiky waveform. The exact sampling time is important. It should be in the middle of the current pulse or at least a sampling time (tsamp) before the switching event. These exact trigger points can be generated by T13 (T13CM for -Iu, T13PM for Iv).

In the SVM example four sequences are configured with VDH, CSA assigned to different channels, see **Figure 217**.

- The trigger for SQ1 comes from T12ZM and measures VDH in CH0 at a “quiet” time
- The trigger for SQ3 comes from T15CM and measures CSA in CH6 (here phase current -Iu) and VDH in CH1
- The trigger for SQ0 comes from T12PM and measures VDH in CH2, P2.4 in CH4, P2.5 in CH5 at a “quiet” time
- The trigger for SQ2 comes from T15PM and measures CSA in CH7 (here phase current Iv) and VDH in CH3

Due to the flexible assignment between analog inputs and channels the conversion results are in different registers, i.e there is no hard real-time condition for reading results. The software can read the results with one control interrupt. The same interrupt can be used to re-configure the sequence if necessary, e.g. T12PM.

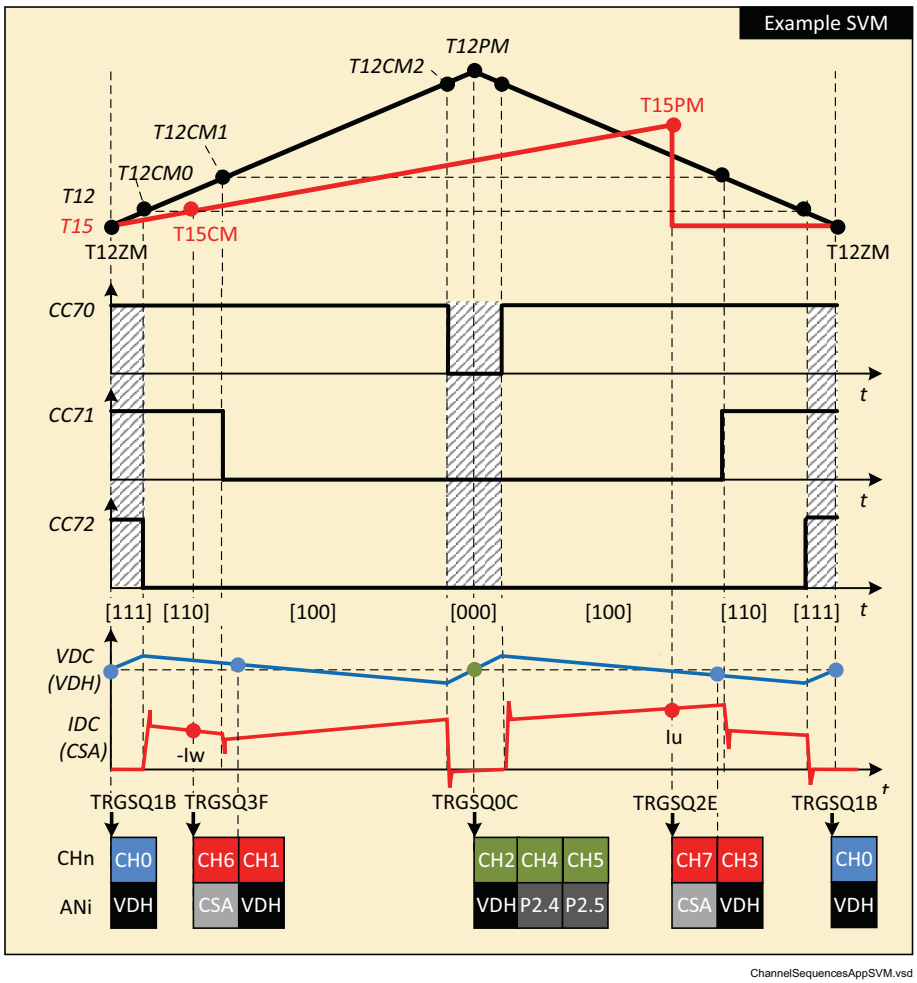


Figure 217 Sequence example SVM (7-segment)

16.13 Register description ADC1

16.13.1 ADC1 Address Maps

Table 189 Register Address Space - ADC1

Module	Base Address	End Address	Note
ADC1	40000000_H	$40003FFF_H$	

Table 190 Register Overview - ADC1 (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
GLOBCONF	Global Configuration Register	0000_H	690
CLKCON	Clock Control Register	0004_H	690
SUSCTR	Suspend Control Register	0008_H	691
SUSSTAT	Suspend Status Register	$000C_H$	691
SQCFGx	Sequence Configuration Register	0010_H+x*8	692
SQSLOTx	SQ Channel Slot Register	0014_H+x*8	693
SQSTAT	Sequence Status Register	0030_H	694
SQSTATCLR	Sequence Status Clear Register	0034_H	695
SQSTATSET	Sequence Status Clear Register	0038_H	695
CHCFGx	Channel Configuration Register	$003C_H$+x*4	696
CHSTAT	Channel Status Register	$008C_H$	697
CHSTATCLR	Channel Status Register	0090_H	698
CHSTATSET	Channel Status Set Register	0094_H	698
CONVCFGx	Conversion Configuration Register	0098_H+x*4	699
CALEN	Calibration Enable	$00A8_H$	700
CALPEN	Calibration Protection Enable	$00AC_H$	700
FILTCFG	Filter Configuration	$00B0_H$	701
FILx	Filter Result Register	$00B4_H$+x*4	701
FILSTAT	Filter Status Register	$00C4_H$	702
FILSTATCLR	Filter Status Clear Register	$00C8_H$	702
FILSTATSET	Filter Status Set Register	$00CC_H$	703
RESx	Result Register	$00D0_H$+x*4	703
CMPCFGx	Compare Channel x Control Register	0120_H+x*4	704
CMPSTAT	Compare Status Register	0130_H	705
CMPSTATCLR	Compare Status Clear Register	0134_H	706
CMPSTATSET	Compare Status Set Register	0138_H	707
IEN0	Interrupt Enable Register 0	$013C_H$	707

Table 190 Register Overview - ADC1 (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Page Number
IEN1	Interrupt Enable Register 1	0140_H	708
INP0	Interrupt Node Pointer Register 0	0144_H	709
INP1	Interrupt Node Pointer Register 1	0148_H	709
INP2	Interrupt Node Pointer Register 2	$014C_H$	710
INP3	Interrupt Node Pointer Register 3	0150_H	710
SHDCTR	Shadow Transfer Control Register	0164_H	711
CALAI1	Calibration Setting for Analog Input 1	0178_H	713
CALAI3	Calibration Setting for Analog Input 3	$017C_H$	713
CALAI5	Calibration Setting for Analog Input 5	0180_H	714
CALAI7	Calibration Setting for Analog Input 7	0184_H	715
CALAI9	Calibration Setting for Analog Input 9	0188_H	715
CALAI11	Calibration Setting for Analog Input 11	$018C_H$	716
CALAI13	Calibration Setting for Analog Input 13	0190_H	717
CALAI15	Calibration Setting for Analog Input 15	0194_H	717
CALAI16	Calibration Setting for Analog Input 16	0198_H	718
CALAI17	Calibration Setting for Analog Input 17	$019C_H$	719
CALAI18	Calibration Setting for Analog Input 18	$01A0_H$	719
CALAI19	Calibration Setting for Analog Input 19	$01A4_H$	720
CALAI20	Calibration Setting for Analog Input 20	$01A8_H$	721
CALAI21	Calibration Setting for Analog Input 21	$01AC_H$	721
CALAI22	Calibration Setting for Analog Input 22	$01B0_H$	722
CALAI23	Calibration Setting for Analog Input 23	$01B4_H$	723
CALAI24	Calibration Setting for Analog Input 24	$01B8_H$	723
CALAI25	Calibration Setting for Analog Input 25	$01BC_H$	724
CALAI26	Calibration Setting for Analog Input 26	$01C0_H$	725

16.13.2 ADC1 Registers

Global Configuration Register

GLOBCONF
Global Configuration Register **(0000_H)** **RESET_TYPE_5 Value: 0000 0000_H**

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

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

Field	Bits	Type	Description
EN	0	rw	**Module Enable** 0_B **Disabled**, ADC1 Module is disabled 1_B **Enabled**, ADC1 Module is enabled
RES	4:1, 31:6	r	**Reserved**
ISTE	5	rw	**Idle shadow transfer enable** 0_B **Disabled**, Shadow transfer done based on selected shadow transfer trigger source 1_B **Enabled**, Shadow transfer is automatically enabled when no SQ is running

Clock Control Register

CLKCON
Clock Control Register **(0004_H)** **RESET_TYPE_5 Value: 0000 0001_H**

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

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

Field	Bits	Type	Description
CLKDIV	3:0	rw	**Clock Divider Settings** 0_H **DIV_1**, Divide by 1 … F_H **DIV_16**, Divide by 16
RES	31:4	r	**Reserved**

Register description ADC1

Suspend Control Register

SUSCTR

Suspend Control Register **(0008_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
SUSEN	0	rw	**ADC1 Suspend Enable** 0_B **Disable**, ADC1 Suspend Mode is disabled 1_B **Enable**, ADC1 Suspend Mode is enabled
SUSMOD	1	rw	**Suspend Mode** 0_B **Hard**, Hard Suspend Mode 1_B **Soft**, Soft Suspend Mode
RES	31:2	r	**Reserved**

Suspend Status Register

SUSSTAT

Suspend Status Register **($000C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														READY	STAT
r														r	r

Field	Bits	Type	Description
STAT	0	r	**Suspend Mode Status** 0_B **No_suspend**, Suspend Mode is not entered 1_B **Suspend**, Suspend mode is entered
READY	1	r	**Module Ready** 0_B **not_ready**, ADC is not ready 1_B **Ready**, ADC is ready for operation
RES	31:2	r	**Reserved**

Register description ADC1

Sequence Configuration Register

SQCFGx (x=0-3)
Sequence Configuration Register **(0010_H+x*8)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GTSW	TRGSW	GTSEL		TRGSEL				WFRCFG	COLLCFG	SQREP		RES	SLOTS		
rw	w	rw		rw				rw	rw	rw		r	rw		

Field	Bits	Type	Description
SLOTS	2:0	rw	**Number of used Slots in Sequence** 000_B **0**, No Slot used - SQ disabled 001_B **1**, Slots used … 100_B **4**, Slots used 101_B **5**, No Slot used - SQ disabled … 111_B **7**, No Slot used - SQ disabled
RES	3, 31:16	r	**Reserved**
SQREP	5:4	rw	**Sequence repetition** 00_B **1_REP**, The Sequence is executed 1 time 01_B **2_REP**, The Sequence is executed 2 times 10_B **4_REP**, The Sequence is executed 4 times 11_B **8_REP**, The Sequence is executed 8 times
COLLCFG	6	rw	**Collision Config** 0_B **Discard**, The trigger request for a sequence is discarded in case of a collision 1_B **Pend**, The trigger request for a sequence is set to pending in case of a collision
WFRCFG	7	rw	**Wait for Read Enable** 0_B **Disable**, Wait for Read disabled 1_B **Enable**, Wait for Read enabled

Register description ADC1

Field	Bits	Type	Description
TRGSEL	11:8	rw	**Trigger Select** 0_H **TRGSQxA**, Software Trigger (TRGSW) 1_H **TRGSQxB**, External Trigger source B 2_H **TRGSQxC**, External Trigger source C 3_H **TRGSQxD**, External Trigger source D 4_H **TRGSQxE**, External Trigger source E 5_H **TRGSQxF**, External Trigger source F 6_H **TRGSQxG**, External Trigger source G 7_H **TRGSQxH**, Sequence End Event SQ(x) 8_H **TRGSQxI**, Sequence End Event SQ(x-1) 9_H **not_used**, TRGSQxA (TRGSW) selected … F_H **not_used**, TRGSQxA (TRGSW) selected
GTSEL	13:12	rw	**Gating Select** 00_B **GTSQxA**, Software Gating (GTSW) 01_B **GTSQxB**, External Gating source B 10_B **GTSQxC**, External Gating source C 11_B **GTSQxD**, External Gating source D
TRGSW	14	w	**Software Trigger Bit** 0_B **Not_Triggered**, SQ not triggered 1_B **Triggered**, SQ is triggered
GTSW	15	rw	**Trigger Software Gating** 0_B **Gated**, The selected trigger source is gated off 1_B **Ungated**, The selected trigger source is not gated

SQ Channel Slot Register

SQSLOTx (x=0-3)
SQ Channel Slot Register **(0014_H+x*8)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES3			**CHSEL3**					**RES2**			**CHSEL2**				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES1			**CHSEL1**					**RES0**			**CHSEL0**				
r			rw					r			rw				

Field	Bits	Type	Description
CHSELi (i=0-3)	8*i+4:8*i	rw	**Channel Select** 00_H **CH0**, Channel 0 selected … 13_H **CH19**, Channel 19 selected 14_H **not_used**, Channel 0 selected … $1F_H$ **not_used**, Channel 0 selected

Register description ADC1

Field	Bits	Type	Description
RESi (i=0-3)	8*i+7:8*i+5	r	**Reserved**

Sequence Status Register

SQSTAT
Sequence Status Register **(0030_H)** **RESET_TYPE_5 Value: $0007\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES													SQNUM		
r													r		

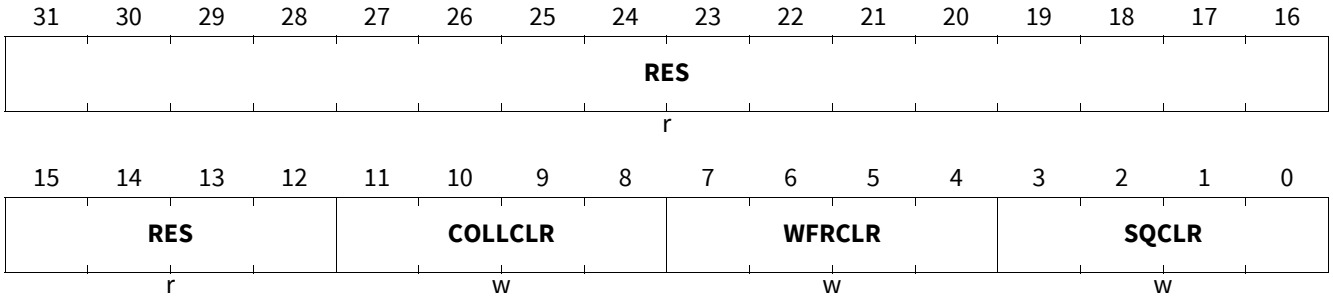
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				COLL				WFR				SQ			
r				rhxre				rhxre				rhxre			

Field	Bits	Type	Description
SQ	3:0	rhxre	**SQ Interrupt Status** Each bit represents the interrupt status of a sequence 0: Not finished - Sequence has not finished or has not started 1: finished - Sequence has finished
WFR	7:4	rhxre	**Wait for Read Status** Each bit represents the wait for read interrupt status of a sequence 0: No Error - No wait for read event occured 1: Error -Wait for read Error occured
COLL	11:8	rhxre	**Collision Status** Each bit represents the collision interrupt Status of a sequence 0: No Error - No Sequence Collision occured 1: Error - Sequence Collision occured
RES	15:12, 31:19	r	**Reserved**
SQNUM	18:16	r	**Actual Sequence processed** 000_B **SQ0**, SQ 0 is running 001_B **SQ1**, SQ1 is running 010_B **SQ2**, SQ2 is running 011_B **SQ3**, SQ3 is running 111_B **NoSQ**, No SQ is running

Register description ADC1

Sequence Status Clear Register

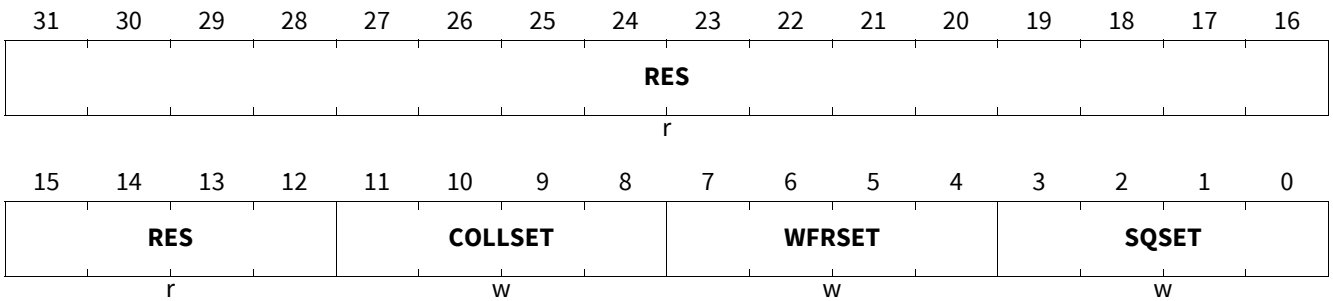
SQSTATCLR
Sequence Status Clear Register **(0034_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
SQCLR	3:0	w	**SQ Interrupt Status Clear** Each bit represents the corresponding interrupt Status clear 0: Not Cleared - SQ Status not cleared 1: Cleared - SQ Status cleared
WFRCLR	7:4	w	**Wait for Read Status Clear** Each bit represents the corresponding Wait for Read Status Clear 0: Not Cleared - Wait for read Status not cleared 1: Cleared - Wait for read Status cleared
COLLCLR	11:8	w	**Collision Status Clear** Each bit represents the corresponding Collision Status Clear 0: Not Cleared - Sequence Collision Status not cleared 1: Cleared - Sequence Collision Status cleared
RES	31:12	r	**Reserved**

Sequence Status Clear Register

SQSTATSET
Sequence Status Clear Register **(0038_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
SQSET	3:0	w	**SQ Interrupt Status Set** Each bit represents the corresponding SQ Interrupt Status Set 0: Not Set - SQ Status not set 1: Set - SQ Status set

Register description ADC1

Field	Bits	Type	Description
WFRSET	7:4	w	**Wait for Read Status Set** Each bit represents the corresponding wait for Read Status Set 0: Not_Set - Wait for read Status 1: Set - Wait for Read Status set
COLLSET	11:8	w	**Collision Status Set** Each bit represents the corresponding Collision Status Set 0: Not Set - Sequence Collision Status not set 1: Set - Sequence Collision Status set
RES	31:12	r	**Reserved**

Channel Configuration Register

CHCFGx (x=0-19)
Channel Configuration Register **($003C_H$+x*4)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES								CLASSEL		CMPSEL			FILSEL		
r								rw		rw			rw		

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

Field	Bits	Type	Description
INSEL	4:0	rw	**ADC Input Selection** 00_H **AN0**, selected … $1A_H$ **AN26**, selected $1B_H$ **not_used**, Input 0 selected … **not_used**, Input 0 selected
RES	7:5, 15:12, 31:24	r	**Reserved**
CHREP	11:8	rw	**Channel Repetition** 0_H**CHREP0**: 0 repetition, 1 conversion 1_H**CHREP1**: 1 repetition, 2 conversions 2_H**CHREP2**: 2 repetitions, 3 conversions ... F_H**CHREP15**: 15 repetitions, 16 conversions

Register description ADC1

Field	Bits	Type	Description
FILSEL	18:16	rw	**Filter Selection** 000_B **FIL0**, Filter Channel 0 001_B **FIL1**, Filter Channel 1 010_B **FIL2**, Filter Channel 2 011_B **FIL3**, Filter Channel 3 100_B **not_assigned**, Filter channel not assigned … 111_B **not_assigned**, Filter channel not assigned
CMPSEL	21:19	rw	**Compare Selection** 000_B **CMP0**, Compare Channel 0 001_B **CMP1**, Compare Channel 1 010_B **CMP2**, Compare Channel 2 011_B **CMP3**, Compare Channel 3 100_B **not_assigned**, Compare channel not assigned … 111_B **not_assigned**, Compare channel not assigned
CLASSEL	23:22	rw	**Conversion Class Selection** 00_B **CONVCFG0**, select settings from CONVCFG0 01_B **CONVCFG1**, select settings from CONVCFG1 10_B **CONVCFG2**, select settings from CONVCFG2 11_B **CONVCFG3**, select settings from CONVCFG3

Channel Status Register

CHSTAT
Channel Status Register **($008C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES			**CHNUM**					**RES**				**CH**			
r			r					r				rhxre			

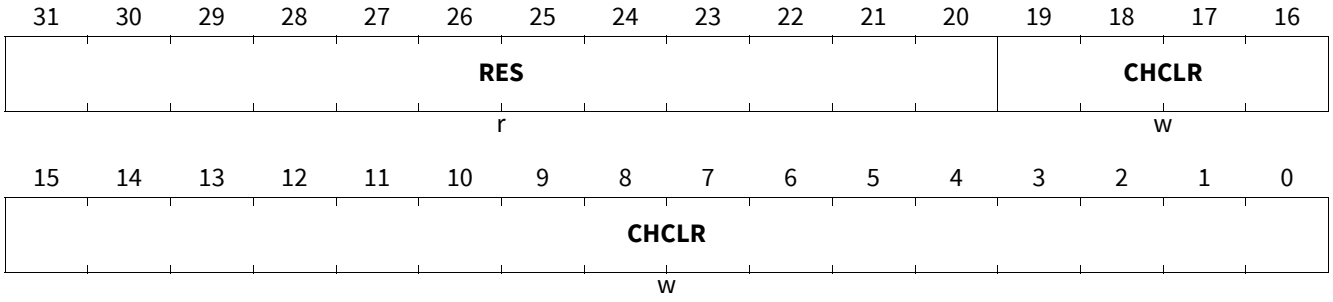
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH															
rhxre															

Field	Bits	Type	Description
CH	19:0	rhxre	**Channel Status** Each bit represents the corresponding Channel Status 0: Not finished - Channel conversion not finished 1: finished - Channel conversion finished
RES	23:20, 31:29	r	**Reserved**
CHNUM	28:24	r	**Current Channel under conversion**

Register description ADC1

Channel Status Register

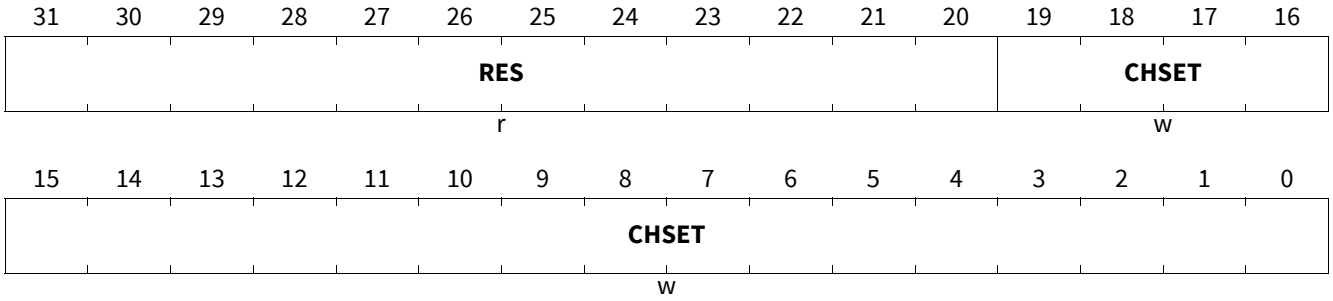
CHSTATCLR
Channel Status Register **(0090_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHCLR	19:0	w	**Channel Status clear flag** Each bit represents the corresponding Channel Status clear flag 0: Not Cleared - Channel Status flag not cleared 1: Cleared - Channel Status flag cleared
RES	31:20	r	**Reserved**

Channel Status Set Register

CHSTATSET
Channel Status Set Register **(0094_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHSET	19:0	w	**Channel Status set flag** Each bit represents the corresponding Channel Status set flag 0: Not Set - Channel Status flag not set 1: Set - Channel Status flag set 00000_H**Not_Set**, Corresponding Channel Status flag not set $FFFFF_H$**Set**, Corresponding Channel Status flag set
RES	31:20	r	**Reserved**

Register description ADC1

Conversion Configuration Register

CONVCFGx (x=0-3)
Conversion Configuration Register **(0098_H+x*4)** **RESET_TYPE_5 Value: 0000 0040_H**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	BWD_HI_CUR	BWD		PCAL	MSBD	RES	SESP	STC				OVERS		TCONF	
r	rw	rw		rw	rw	r	rw	rw				rw		rw	

Field	Bits	Type	Description
TCONF	1:0	rw	**Tracking Conversion config**
OVERS	3:2	rw	**Oversampling config**
STC	7:4	rw	**Sample Time config** 0_H **STC2**, tsamp = 2*ADC_CLK 1_H **STC3**, tsamp = 3*ADC_CLK 2_H **STC4**, tsamp = 4*ADC_CLK 3_H **STC5**, tsamp = 5*ADC_CLK 4_H **STC6**, tsamp = 6*ADC_CLK 5_H **STC7**, tsamp = 7*ADC_CLK 6_H **STC8**, tsamp = 8*ADC_CLK 7_H **STC9**, tsamp = 9*ADC_CLK 8_H **STC10**, tsamp = 10*ADC_CLK 9_H **STC11**, tsamp = 11*ADC_CLK A_H **STC12**, tsamp = 12*ADC_CLK B_H **STC16**, tsamp = 16*ADC_CLK C_H **STC20**, tsamp = 20*ADC_CLK D_H **STC32**, tsamp = 32*ADC_CLK E_H **STC64**, tsamp = 64*ADC_CLK F_H **STC128**, tsamp = 128*ADC_CLK Configures the sampling time
SESP	8	rw	**Spread early sample point config**
RES	9, 31:15	r	**Reserved**
MSBD	10	rw	**MSB doubling enable**
PCAL	11	rw	**Post Calibration enable**
BWD	13:12	rw	**BWD timing config** 00_B **0_ADC_CLK**, tBWD = 0 (BWD is disabled) 01_B **4_ADC_CLK**, tBWD = 4*ADC_CLK 10_B **8_ADC_CLK**, tBWD = 8*ADC_CLK 11_B **16_ADC_CLK**, tBWD = 16*ADC_CLK

Register description ADC1

Field	Bits	Type	Description
BWD_HI_CUR	14	rw	**BWD current configuration** 0_B **IBWD**, selected 1_B **IBWDH**, selected (High current Mode)

Calibration Enable

CALEN
Calibration Enable **($00A8_H$)** **RESET_TYPE_5 Value: 07FF AAAA$_H$**

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

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

Field	Bits	Type	Description
CALEN	26:0	rw	**Channel Calibration Enable** Each bit represents the corresponding Channel Calibration Enable 0: Channel calibration disabled 1: Channel calibration enabled
RES	31:27	r	**Reserved**

Calibration Protection Enable

CALPEN
Calibration Protection Enable **($00AC_H$)** **RESET_TYPE_5 Value: 07FF FFFF$_H$**

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

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

Field	Bits	Type	Description
CALPEN	26:0	rw	**Calibration Protection** Each bit represents the corresponding Calibration Protection 0: Channel calibration setting is unprotected 1: Channel calibration setting protected
RES	31:27	r	**Reserved**

Register description ADC1

Filter Configuration

FILTCFG
Filter Configuration **($00B0_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES3		COEF_A3		RES2		COEF_A2		RES1		COEF_A1		RES0		COEF_A0	
r		rw		r		rw		r		rw		r		rw	

Field	Bits	Type	Description
COEF_Ax (x=0-3)	4*x+1:4*x	rw	**Filter Coefficient** 00_B **1_2**, Filter coefficient 1/2 01_B **1_4**, Filter coefficient 1/4 10_B **1_8**, Filter coefficient 1/8 11_B **1_16**, Filter coefficient 1/16
RESx (x=0-3)	4*x+3:4*x+2	r	**Reserved**
RES	31:16	r	**Reserved**

Filter Result Register

FILx (x=0-3)
Filter Result Register **($00B4_H$+x*4)** **RESET_TYPE_5 Value: $XXXX\ XXXX_H$**

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

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

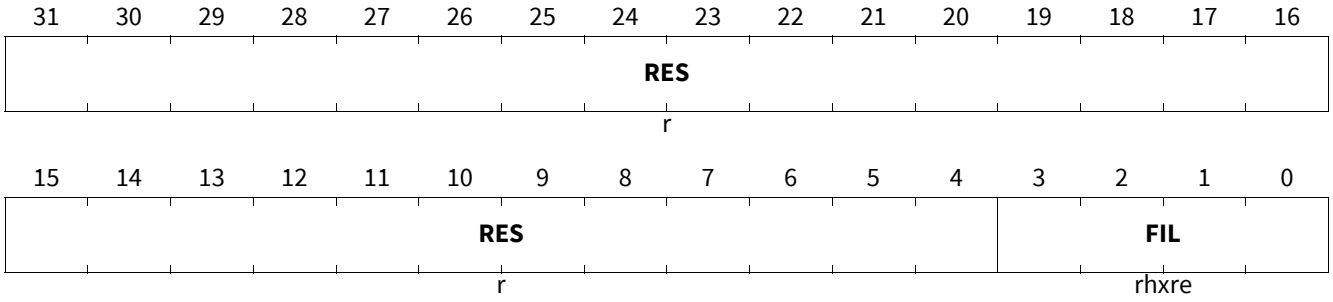
Field	Bits	Type	Description
FILRESULT	15:0	r	**Filter Result Value**
RES	31:16	r	**Reserved**

Register description ADC1

Filter Status Register

FILSTAT

Filter Status Register **($00C4_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

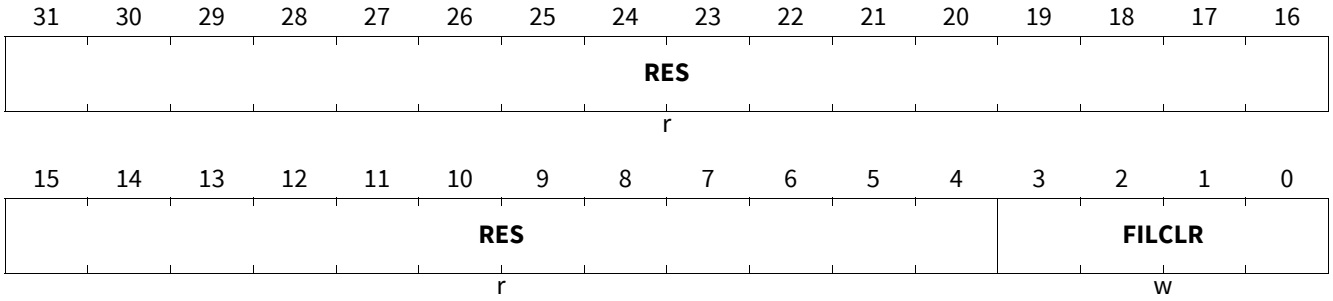


Field	Bits	Type	Description
FIL	3:0	rhxre	**Filter Event flag** Each bit represents the corresponding Filter Event flag 0: No result - No new result available 1: New Result - New Result available
RES	31:4	r	**Reserved**

Filter Status Clear Register

FILSTATCLR

Filter Status Clear Register **($00C8_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

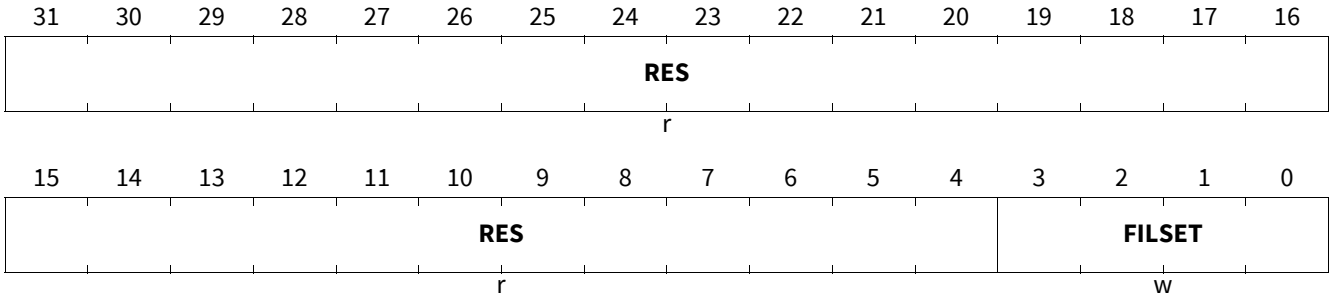


Field	Bits	Type	Description
FILCLR	3:0	w	**Filter Event flag** Each bit represents the corresponding Filter Event clear 0: Not Cleared - Corresponding Filter Status flag not cleared 1: Cleared - Corresponding Filter Status flag cleared
RES	31:4	r	**Reserved**

Register description ADC1

Filter Status Set Register

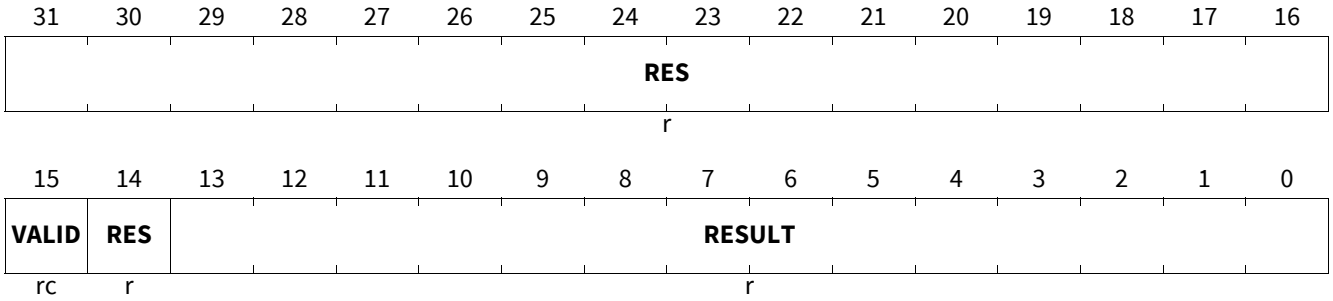
FILSTATSET
Filter Status Set Register **($00CC_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
FILSET	3:0	w	**Filter Event flag** Each bit represents the corresponding Filter Event set 0: Not Set - Corresponding Filter Status flag not set 1: Set - Corresponding Filter Status flag set
RES	31:4	r	**Reserved**

Result Register

RESx (x=0-19)
Result Register **($00D0_H$+x*4)** **RESET_TYPE_5 Value: $XXXX\ XXXX_H$**



Field	Bits	Type	Description
RESULT	13:0	r	**Result Value**
RES	14, 31:16	r	**Reserved**
VALID	15	rc	**Valid flag** 0_B **Not_Valid**, No new result available 1_B **Valid**, New result available

Register description ADC1

Compare Channel x Control Register

CMPCFGx (x=0-3)
Compare Channel x Control Register **(0120_H+x*4)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE		HYST_UP		RST_BLANK_TIME	BLANK_TIME			UPPER							
rw		rw		w	rw			rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HYST_LO		RES			INP_SEL	LOWER							
r		rw		r			rw	rw							

Field	Bits	Type	Description
LOWER	7:0	rw	**Lower Compare Value**
INP_SEL	8	rw	**Input selection for the comparator unit** 0_B **AHB**, Selects the ahb result 1_B **FILTER**, Selects the filter result
RES	11:9, 15:14	r	**Reserved - 0** Read as 0; should be written with 0.
HYST_LO	13:12	rw	**Hysteresis set for lower compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
UPPER	23:16	rw	**Upper Compare Value**
BLANK_TIME	26:24	rw	**Blank Time configuration** 000_B **OFF**, 001_B **BLANK_5**, 5 conversions blank time 010_B **BLANK_10**, 10 conversions blank time 011_B **BLANK_20**, 20 conversions blank time 100_B **BLANK_30**, 30 conversions blank time 101_B **BLANK_40**, 40 conversions blank time 110_B **BLANK_50**, 50 conversions blank time 111_B **BLANK_60**, 60 conversions blank time
RST_BLANK_TIME	27	w	**Restart Blank time** 0_B **NO_RESTART**, No Restart of blank time 1_B **RESTART**, Restart blank time
HYST_UP	29:28	rw	**Hysteresis setting for upper compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16

Register description ADC1

Field	Bits	Type	Description
MODE	31:30	rw	**Compare Mode** 00_B **RANGE**, Upper and Lower voltage/limit measurement 01_B **UV**, undervoltage/-limit measurement 10_B **OV**, overvoltage/-limit measurement 11_B **RES**, Reserved (RANGE)

Compare Status Register

CMPSTAT
Compare Status Register **(0130_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES				**CMP_UP**				**CMP_UP_IS**				**CMP_UP_STS**			
r				r				rhxre				rhxr			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				**CMP_LO**				**CMP_LO_IS**				**CMP_LO_STS**			
r				r				rhxre				rhxr			

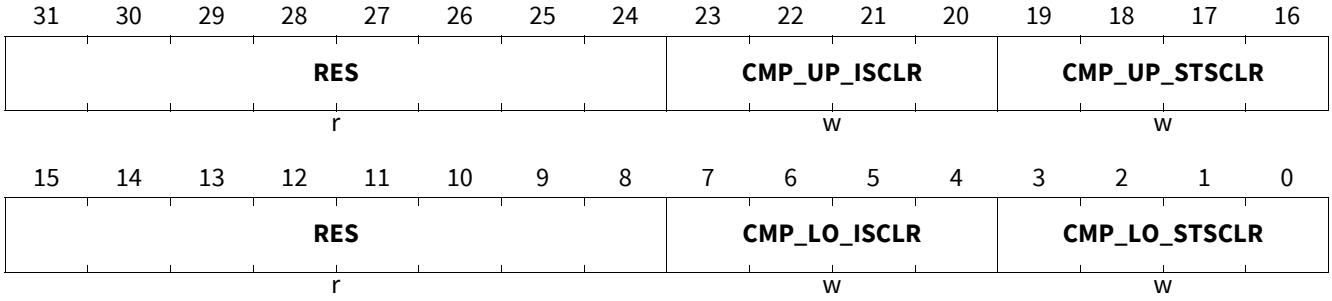
Field	Bits	Type	Description
CMP_LO_STS	3:0	rhxr	**Latched Compare low Status** Each bit represents the corresponding Compare low Status 0: INACTIVE - No Lower Compare Event occured 1: ACTIVE - Lower Compare Event occured
CMP_LO_IS	7:4	rhxre	**Compare low Interrupt Status** Each bit represents the corresponding Compare low Interrupt Status 0: INACTIVE - No Lower Compare Event occured 1: ACTIVE - Lower Compare Event occured
CMP_LO	11:8	r	**Compare low Status** Each bit represents the corresponding Compare low Status 0: INACTIVE - No Lower Compare Event occured 1: ACTIVE - Lower Compare Event occured
RES	15:12, 31:28	r	**Reserved**
CMP_UP_STS	19:16	rhxr	**Latched Compare up Status** Each bit represents the corresponding Compare up Status 0: INACTIVE - No Upper Compare Event occured 1: ACTIVE - Upper Compare Event occured
CMP_UP_IS	23:20	rhxre	**Compare up Interrupt Status** Each bit represents the corresponding Compare up Interrupt Status 0: INACTIVE - No Upper Compare Event occured 1: ACTIVE - Upper Compare Event occured

Register description ADC1

Field	Bits	Type	Description
CMP_UP	27:24	r	**Compare up Status** Each bit represents the corresponding Compare up Status 0: INACTIVE - No Upper Compare Event occured 1: ACTIVE - Upper Compare Event occured

Compare Status Clear Register

CMPSTATCLR
Compare Status Clear Register **(0134_H)** **RESET_TYPE_5 Value: 0000 0000_H**



Field	Bits	Type	Description
CMP_LO_STSCLR	3:0	w	**Compare low Status clear** Each bit represents the corresponding Compare low Status clear 0: Not Cleared - Corresponding Compare Status flag not cleared 1: Cleared - Corresponding Compare Status flag cleared
CMP_LO_ISCLR	7:4	w	**Compare low Interrupt clear** Each bit represents the corresponding Compare low interrupt clear 0: Not Cleared - Corresponding Compare Status flag not cleared 1: Cleared - Corresponding Compare Status flag cleared
RES	15:8, 31:24	r	**Reserved**
CMP_UP_STSCLR	19:16	w	**Compare up Status clear** Each bit represents the corresponding Compare up Status clear 0: Not Cleared - Corresponding Compare Status flag not cleared 1: Cleared - Corresponding Compare Status flag cleared
CMP_UP_ISCLR	23:20	w	**Compare up Interrupt clear** Each bit represents the corresponding Compare up Interrupt clear 0: Not Cleared - Corresponding Compare Status flag not cleared 1: Cleared - Corresponding Compare Status flag cleared

Register description ADC1

Compare Status Set Register

CMPSTATSET
Compare Status Set Register **(0138_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES								CMP_UP_ISSET				CMP_UP_STSSET			
r								w				w			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								CMP_LO_ISSET				CMP_LO_STSSET			
r								w				w			

Field	Bits	Type	Description
CMP_LO_STSSET	3:0	w	**Compare low Status set** Each bit represents the corresponding Compare low Status set 0: Not Set - Corresponding Compare Status flag not set 1: Set - Corresponding Compare Status flag set
CMP_LO_ISSET	7:4	w	**Compare low Interrupt set** Each bit represents the corresponding Compare low Interrupt set 0: Not Set - Corresponding Compare Status flag not set 1: Set - Corresponding Compare Status flag set
RES	15:8, 31:24	r	**Reserved**
CMP_UP_STSSET	19:16	w	**Compare up Status set** Each bit represents the corresponding Compare up Status set 0: Not Set - Corresponding Compare Status flag not set 1: Set - Corresponding Compare Status flag set
CMP_UP_ISSET	23:20	w	**Compare up Interrupt set** Each bit represents the corresponding Compare up Interrupt set 0: Not Set - Corresponding Compare Status flag not set 1: Set - Corresponding Compare Status flag set

Interrupt Enable Register 0

IEN0
Interrupt Enable Register 0 **($013C_H$)** **RESET_TYPE_5 Value: $03FF\ FFFF_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IEN_UP				IEN_LO				IEN_SQ				IEN_CH			
rw				rw				rw				rw			

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

Register description ADC1

Field	Bits	Type	Description
IEN_CH	19:0	rw	**Channel Interrupt Enable** Each bit represents the corresponding Channel Interrupt Enable 0:Disable - Interrupt disabled 1:Enable - Interrupt enabled
IEN_SQ	23:20	rw	**Sequence Interrupt Enable** Each bit represents the corresponding Sequence Interrupt Enable 0: Disable - Interrupt disabled 1: Enable - Interrupt Enabled
IEN_LO	27:24	rw	**Compare LO Interrupt Enable** Each bit represents the corresponding Compare LO Interrupt Enable 0: Disable - Interrupt disabled 1: Enable - Interrupt Enabled
IEN_UP	31:28	rw	**Compare UP Interrupt Enable** Each bit represents the corresponding Compare UP Interrupt Enable 0: Disable - Interrupt disabled 1: Enable - Interrupt enabled

Interrupt Enable Register 1

IEN1
Interrupt Enable Register 1 **(0140_H)** **RESET_TYPE_5 Value: 03FF FFFF$_H$**

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

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

Field	Bits	Type	Description
IEN_COLL	3:0	rw	**Collision Interrupt Enable** Each bit represents the corresponding Collision Interrupt Enable 0: Disable - Interrupt disabled 1: Enable - Interrupt enabled
IEN_WFR	7:4	rw	**WFR Interrupt Enable** Each bit represents the corresponding WFR Interrupt Enable 0: Disable - Interrupt disabled 1: Enable - Interrupt Enabled
RES	8, 31:9	r	**Reserved - always write 0**

Register description ADC1

Interrupt Node Pointer Register 0

INP0
Interrupt Node Pointer Register 0 **(0144_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INP_CH15		INP_CH14		INP_CH13		INP_CH12		INP_CH11		INP_CH10		INP_CH9		INP_CH8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INP_CH7		INP_CH6		INP_CH5		INP_CH4		INP_CH3		INP_CH2		INP_CH1		INP_CH0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
INP_CHx (x=0-15)	2*x+1:2*x	rw	**Channel Interrupt Node Pointer** 00_B **IRQ0**, mapped to IRQ0 01_B **IRQ1**, mapped to IRQ1 10_B **IRQ2**, mapped to IRQ2 11_B **IRQ3**, mapped to IRQ3

Interrupt Node Pointer Register 1

INP1
Interrupt Node Pointer Register 1 **(0148_H)** **RESET_TYPE_5 Value: $03FF\ FFFF_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						RES		INP_CH19		INP_CH18		INP_CH17		INP_CH16	
r						r		rw		rw		rw		rw	

Field	Bits	Type	Description
INP_CHx (x=16-19)	2*x-31:2*x-32	rw	**Channel Interrupt Node Pointer** 00_B **IRQ0**, mapped to IRQ0 01_B **IRQ1**, mapped to IRQ1 10_B **IRQ2**, mapped to IRQ2 11_B **IRQ3**, mapped to IRQ3
RES	9:8, 31:10	r	**Reserved**

Register description ADC1

Interrupt Node Pointer Register 2

INP2
Interrupt Node Pointer Register 2 **($014C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INP_CMP_UP3		INP_CMP_UP2		INP_CMP_UP1		INP_CMP_UP0		INP_CMP_LO3		INP_CMP_LO2		INP_CMP_LO1		INP_CMP_LO0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
INP_CMP_LOx (x=0-3)	2*x+1:2*x	rw	**Compare Lo Interrupt Node Pointer** 00_B **IRQ0**, mapped to IRQ0 01_B **IRQ1**, mapped to IRQ1 10_B **IRQ2**, mapped to IRQ2 11_B **IRQ3**, mapped to IRQ3
INP_CMP_UPx (x=0-3)	2*x+9:2*x+8	rw	**Compare Up Interrupt Node Pointer** 00_B **IRQ0**, mapped to IRQ0 01_B **IRQ1**, mapped to IRQ1 10_B **IRQ2**, mapped to IRQ2 11_B **IRQ3**, mapped to IRQ3
RES	31:16	r	**Reserved**

Interrupt Node Pointer Register 3

INP3
Interrupt Node Pointer Register 3 **(0150_H)** **RESET_TYPE_5 Value: $03FF\ FFFF_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES								INP_WFR3		INP_WFR2		INP_WFR1		INP_WFR0	
r								rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INP_COLL3		INP_COLL2		INP_COLL1		INP_COLL0		INP_SQ3		INP_SQ2		INP_SQ1		INP_SQ0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
INP_SQx (x=0-3)	2*x+1:2*x	rw	**Sequence Interrupt Node Pointer** 00_B **IRQ0**, mapped to IRQ0 01_B **IRQ1**, mapped to IRQ1 10_B **IRQ2**, mapped to IRQ2 11_B **IRQ3**, mapped to IRQ3

Register description ADC1

Field	Bits	Type	Description
INP_COLLx (x=0-3)	2*x+9:2*x+8	rw	**Collision Interrupt Node Pointer** 00_B **IRQ0**, mapped to IRQ0 01_B **IRQ1**, mapped to IRQ1 10_B **IRQ2**, mapped to IRQ2 11_B **IRQ3**, mapped to IRQ3
INP_WFRx (x=0-3)	2*x+17:2*x+16	rw	**Wait for read Interrupt Node Pointer** 00_B **IRQ0**, mapped to IRQ0 01_B **IRQ1**, mapped to IRQ1 10_B **IRQ2**, mapped to IRQ2 11_B **IRQ3**, mapped to IRQ3
RES	31:24	r	**Reserved**

Shadow Transfer Control Register

SHDCTR
Shadow Transfer Control Register **(0164_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES									STE_GTG	STE_TRG	STE_SQ	RES	ST_GTGSW	ST_TRGSW	ST_SQSW
r									rw	rw	rw	r	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	STE_GTGSEL	STE_TRGSEL	STE_SQSEL	RES	ST_GTGSEL			RES	ST_TRGSEL			RES	ST_SQSEL		
r	rw	rw	rw	r	rw			r	rw			r	rw		

Field	Bits	Type	Description
ST_SQSEL	2:0	rw	**Sequence Shadow Transfer Selection** 000_B **ST_SQSW**, Software Shadow Transfer 001_B **EXTSTRA**, CCU7 T12 zero match 010_B **EXTSTRB**, CCU7 T12 period match 011_B **EXTSTRC**, CCU7 T13 period match 100_B **EXTSTRD**, CCU7 T14 period match 101_B **EXTSTRE**, CCU7 T15 period match 110_B **EXTSTRF**, CCU7 T16 period match 111_B **not_used**, ST_SQSW selected
RES	3, 7, 11, 15, 19, 31:23	r	**Reserved**

Register description ADC1

Field	Bits	Type	Description
ST_TRGSEL	6:4	rw	**Trigger Shadow Transfer Selection** 000_B **ST_TRGSW**, Software Shadow Transfer 001_B **EXTSTRA**, CCU7 T12 zero match 010_B **EXTSTRB**, CCU7 T12 period match 011_B **EXTSTRC**, CCU7 T13 period match 100_B **EXTSTRD**, CCU7 T14 period match 101_B **EXTSTRE**, CCU7 T15 period match 110_B **EXTSTRF**, CCU7 T16 period match 111_B **not_used**, ST_TRGSW selected
ST_GTGSEL	10:8	rw	**Gating Shadow Transfer Selection** 000_B **ST_GTGSW**, Software Shadow Transfer 001_B **EXTSTRA**, CCU7 T12 zero match 010_B **EXTSTRB**, CCU7 T12 period match 011_B **EXTSTRC**, CCU7 T13 period match 100_B **EXTSTRD**, CCU7 T14 period match 101_B **EXTSTRE**, CCU7 T15 period match 110_B **EXTSTRF**, CCU7 T16 period match 111_B **not_used**, ST_GTGSW selected
STE_SQSEL	12	rw	**Sequence Shadow Transfer Enable Selection** 0_B **STE_SQ**, Software Shadow transfer enable 1_B **EXTSTEN**, Shadow transfer with external CCU7 Shadow transfer
STE_TRGSEL	13	rw	**Trigger Shadow Transfer Enable Selection** 0_B **STE_TRG**, Software Shadow transfer enable 1_B **EXTSTEN**, Shadow transfer with external CCU7 Shadow transfer
STE_GTGSEL	14	rw	**Gating Shadow Transfer Enable Selection** 0_B **STE_GTG**, Software Shadow transfer enable 1_B **EXTSTEN**, Shadow transfer with external CCU7 Shadow transfer
ST_SQSW	16	w	**Sequence Software Shadow Transfer** 0_B **Not_Triggered**, Shadow transfer not triggered 1_B **Triggered**, Shadow transfer is triggered
ST_TRGSW	17	w	**Trigger Software Shadow Transfer** 0_B **Not_Triggered**, Shadow transfer not triggered 1_B **Triggered**, Shadow transfer is triggered
ST_GTGSW	18	w	**Gating Software Shadow Transfer** 0_B **Not_Triggered**, Shadow transfer not triggered 1_B **Triggered**, Shadow transfer is triggered
STE_SQ	20	rw	**Sequence Shadow Transfer Enable** 0_B **Disable**, Shadow transfer disabled 1_B **Enable**, Shadow transfer enabled
STE_TRG	21	rw	**Trigger Shadow Transfer Enable** 0_B **Disable**, Shadow transfer disabled 1_B **Enable**, Shadow transfer enabled
STE_GTG	22	rw	**Gating Shadow Transfer Enable** 0_B **Disable**, Shadow transfer disabled 1_B **Enable**, Shadow transfer enabled

Calibration Setting for Analog Input 1

CALAI1
Calibration Setting for Analog Input 1 **(0178_H)** **Reset Value: Table 191**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CALGAIN									
r						rwpht									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CALOFFS					
r										rwpht					

Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 191 Reset Values of CALAI1

Reset Type	Reset Value	Note
RESET_TYPE_4	$0000\ 0000_H$	RESET_TYPE_4
TRIM_2	$0000\ 0000_H$	TRIM

Calibration Setting for Analog Input 3

CALAI3
Calibration Setting for Analog Input 3 **($017C_H$)** **Reset Value: Table 192**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CALGAIN									
r						rwpht									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CALOFFS					
r										rwpht					

Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set

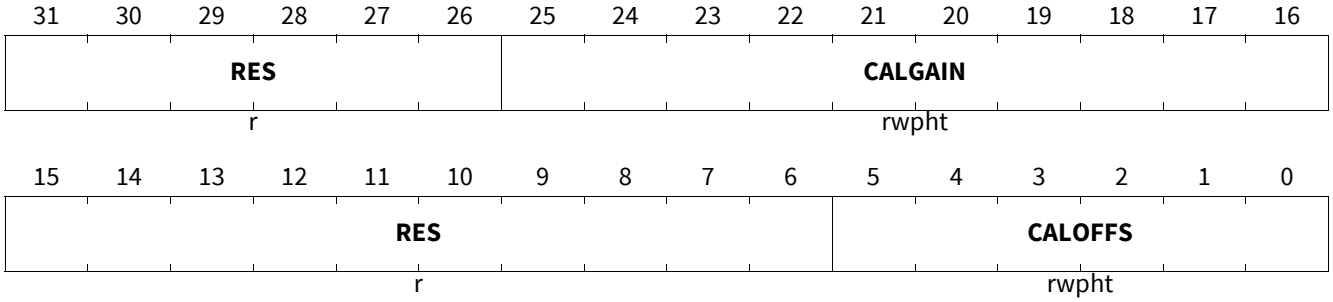
Field	Bits	Type	Description
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 192 Reset Values of CALAI3

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0000$_H$	RESET_TYPE_4
TRIM_2	0000 0000$_H$	TRIM

Calibration Setting for Analog Input 5

CALAI5
Calibration Setting for Analog Input 5 (0180$_H$) Reset Value: Table 193



Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 193 Reset Values of CALAI5

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0000$_H$	RESET_TYPE_4
TRIM_2	0000 0000$_H$	TRIM

Calibration Setting for Analog Input 7

CALAI7
Calibration Setting for Analog Input 7 **(0184_H)** **Reset Value: Table 194**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CALGAIN									
r						rwpht									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CALOFFS					
r										rwpht					

Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 194 Reset Values of CALAI7

Reset Type	Reset Value	Note
RESET_TYPE_4	$0000\ 0000_H$	RESET_TYPE_4
TRIM_2	$0000\ 0000_H$	TRIM

Calibration Setting for Analog Input 9

CALAI9
Calibration Setting for Analog Input 9 **(0188_H)** **Reset Value: Table 195**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CALGAIN									
r						rwpht									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CALOFFS					
r										rwpht					

Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set

Register description ADC1

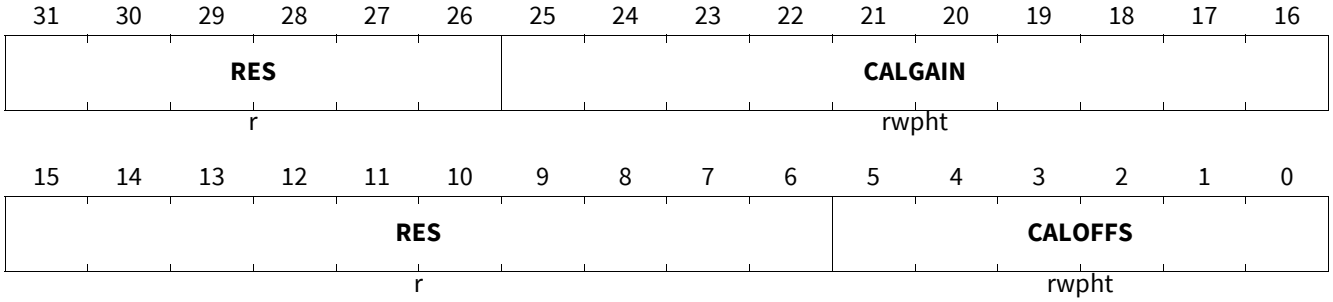
Field	Bits	Type	Description
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 195 Reset Values of CALAI9

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0000$_H$	RESET_TYPE_4
TRIM_2	0000 0000$_H$	TRIM

Calibration Setting for Analog Input 11

CALAI11
Calibration Setting for Analog Input 11 (018C$_H$) Reset Value: Table 196



Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 196 Reset Values of CALAI11

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0000$_H$	RESET_TYPE_4
TRIM_2	0000 0000$_H$	TRIM

Calibration Setting for Analog Input 13

CALAI13
Calibration Setting for Analog Input 13 **(0190_H)** **Reset Value: Table 197**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CALGAIN									
r						rwpht									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CALOFFS					
r										rwpht					

Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 197 Reset Values of CALAI13

Reset Type	Reset Value	Note
RESET_TYPE_4	$0000\ 0000_H$	RESET_TYPE_4
TRIM_2	$0000\ 0000_H$	TRIM

Calibration Setting for Analog Input 15

CALAI15
Calibration Setting for Analog Input 15 **(0194_H)** **Reset Value: Table 198**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CALGAIN									
r						rwpht									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CALOFFS					
r										rwpht					

Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set

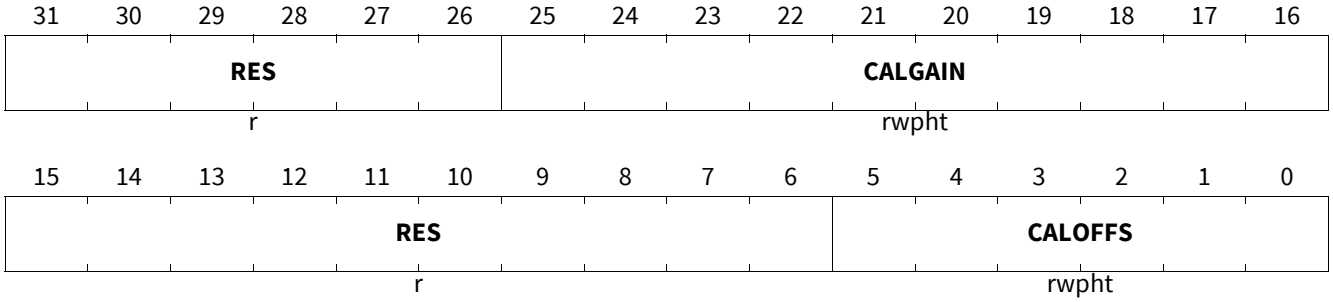
Field	Bits	Type	Description
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 198 Reset Values of CALAI15

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0000$_H$	RESET_TYPE_4
TRIM_2	0000 0000$_H$	TRIM

Calibration Setting for Analog Input 16

CALAI16
Calibration Setting for Analog Input 16 (0198$_H$) Reset Value: Table 199



Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 199 Reset Values of CALAI16

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0000$_H$	RESET_TYPE_4
TRIM_2	0000 0000$_H$	TRIM

Calibration Setting for Analog Input 17

CALAI17
Calibration Setting for Analog Input 17 **($019C_H$)** **Reset Value: Table 200**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CALGAIN									
r						rwpht									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CALOFFS					
r										rwpht					

Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 200 Reset Values of CALAI17

Reset Type	Reset Value	Note
RESET_TYPE_4	$0000\ 0000_H$	RESET_TYPE_4
TRIM_2	$0000\ 0000_H$	TRIM

Calibration Setting for Analog Input 18

CALAI18
Calibration Setting for Analog Input 18 **($01A0_H$)** **Reset Value: Table 201**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CALGAIN									
r						rwpht									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CALOFFS					
r										rwpht					

Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set

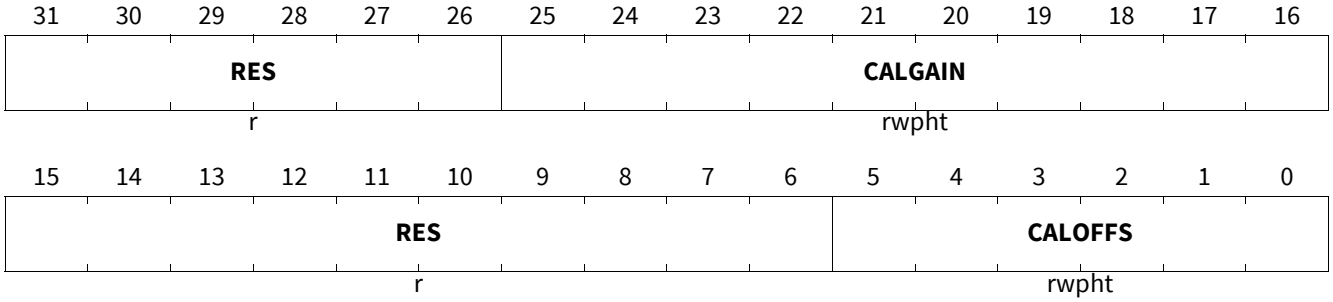
Field	Bits	Type	Description
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 201 Reset Values of CALAI18

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0000$_H$	RESET_TYPE_4
TRIM_2	0000 0000$_H$	TRIM

Calibration Setting for Analog Input 19

CALAI19
Calibration Setting for Analog Input 19 (01A4$_H$) Reset Value: Table 202



Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 202 Reset Values of CALAI19

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0000$_H$	RESET_TYPE_4
TRIM_2	0000 0000$_H$	TRIM

Calibration Setting for Analog Input 20

CALAI20
Calibration Setting for Analog Input 20 **($01A8_H$)** **Reset Value: Table 203**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CALGAIN									
r						rwpht									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CALOFFS					
r										rwpht					

Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 203 Reset Values of CALAI20

Reset Type	Reset Value	Note
RESET_TYPE_4	$0000\ 0000_H$	RESET_TYPE_4
TRIM_2	$0000\ 0000_H$	TRIM

Calibration Setting for Analog Input 21

CALAI21
Calibration Setting for Analog Input 21 **($01AC_H$)** **Reset Value: Table 204**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CALGAIN									
r						rwpht									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CALOFFS					
r										rwpht					

Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set

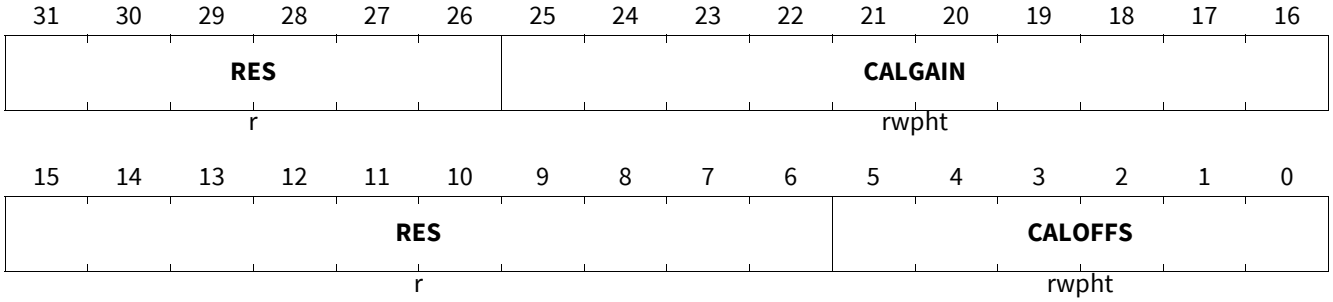
Field	Bits	Type	Description
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 204 Reset Values of CALAI21

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0000$_H$	RESET_TYPE_4
TRIM_2	0000 0000$_H$	TRIM

Calibration Setting for Analog Input 22

CALAI22
Calibration Setting for Analog Input 22 (01B0$_H$) Reset Value: Table 205



Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 205 Reset Values of CALAI22

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0000$_H$	RESET_TYPE_4
TRIM_2	0000 0000$_H$	TRIM

Calibration Setting for Analog Input 23

CALAI23
Calibration Setting for Analog Input 23 **($01B4_H$)** **Reset Value: Table 206**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CALGAIN									
r						rwpht									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CALOFFS					
r										rwpht					

Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 206 Reset Values of CALAI23

Reset Type	Reset Value	Note
RESET_TYPE_4	$0000\ 0000_H$	RESET_TYPE_4
TRIM_2	$0000\ 0000_H$	TRIM

Calibration Setting for Analog Input 24

CALAI24
Calibration Setting for Analog Input 24 **($01B8_H$)** **Reset Value: Table 207**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CALGAIN									
r						rwpht									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										CALOFFS					
r										rwpht					

Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set

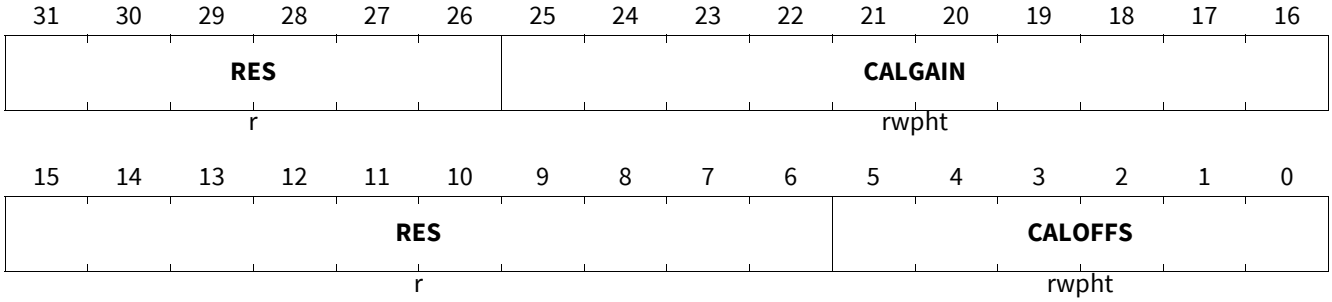
Field	Bits	Type	Description
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 207 Reset Values of CALAI24

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0000$_H$	RESET_TYPE_4
TRIM_2	0000 0000$_H$	TRIM

Calibration Setting for Analog Input 25

CALAI25
Calibration Setting for Analog Input 25 (01BC$_H$) Reset Value: Table 208



Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

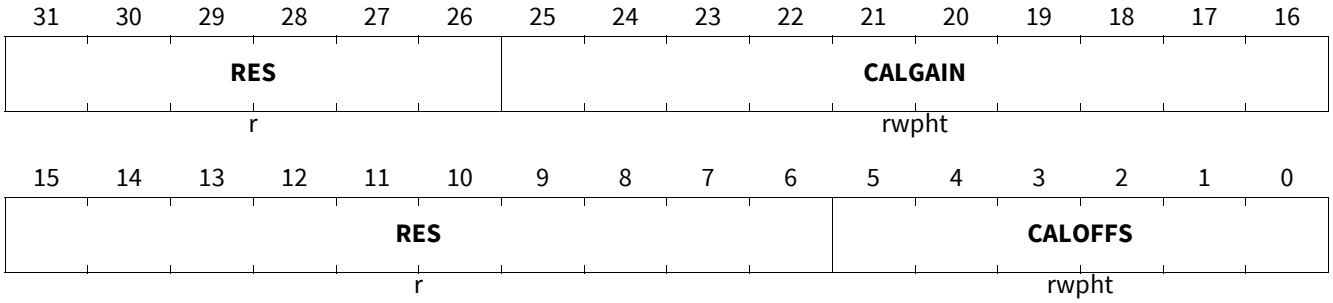
Table 208 Reset Values of CALAI25

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0000$_H$	RESET_TYPE_4
TRIM_2	0000 0000$_H$	TRIM

Register description ADC1

Calibration Setting for Analog Input 26

CALAI26
Calibration Setting for Analog Input 26 **($01C0_H$)** **Reset Value: Table 209**



Field	Bits	Type	Description
CALOFFS	5:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:6, 31:26	r	**Reserved**
CALGAIN	25:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 209 Reset Values of CALAI26

Reset Type	Reset Value	Note
RESET_TYPE_4	$0000\ 0000_H$	RESET_TYPE_4
TRIM_2	$0000\ 0000_H$	TRIM

17 Monitoring Analog Digital Converter 2 (ADC2)

17.1 Features overview

The ADC2 is a successive approximation analog to digital converter which is used for diagnosis of internal system voltages. The ADC2 has a pre-configured sequence of conversions with a deterministic timing. It runs fully autonomous in background, provides the digital results and generates events for interrupts and interconnects.

The ADC2 has following features:

- A/D kernel performance:
  - 10-bit resolution for all analog inputs
  - Wide input range for middle and high voltage inputs from typ. 5 V to 50 V (MV_{RNG}, $HV_{RNG0/1/2/3}$)
  - High accuracy of typ. 1% of the input range (MV_{ACC}, $HV_{ACC0/1/2/3}$)
  - Fast sampling time ($tsamp_{MV}$, $tsamp_{HV}$)
  - Fast total conversion time (typ. 1 µs for MV and typ. 2 µs for HV inputs)
- Analog inputs ANx:
  - 8 factory calibrated middle voltage inputs (range MV_{RNG}), see **ADC2 interconnections**
  - Up to 7 factory calibrated high voltage inputs (ranges $HV_{RNG0/1/2/3}$), see **ADC2 interconnections**
  - Referenced to internally generated VREF1V2 reference voltage (see the **Analog Reference Voltage Generation (ARVG)** chapter)
- Digital channels with channel control and result generation:
  - Each analog input is assigned to one digital channel and has a separate result register
  - 6 digital channels are pre-set for monitoring and protection function for BDRV and CANTRX (NMI capable)
  - 2 freely selectable digital comparators with programmable upper and lower thresholds (8-bit) for user defined monitoring (IRQ capable)
  - 2 freely selectable first order IIR filters with programmable characteristics for result post-processing
  - Results can be read at any time by user software
- Sequencer:
  - One fixed conversion sequence is pre-programmed and runs in a round-robin scheme autonomously in background
- Interrupt and DMA:
  - The ADC2 events can generate a NMI
  - The ADC2 events can be mapped to 2 interrupt node pointers (with 2 IRQ lines)

17.2 Block diagram

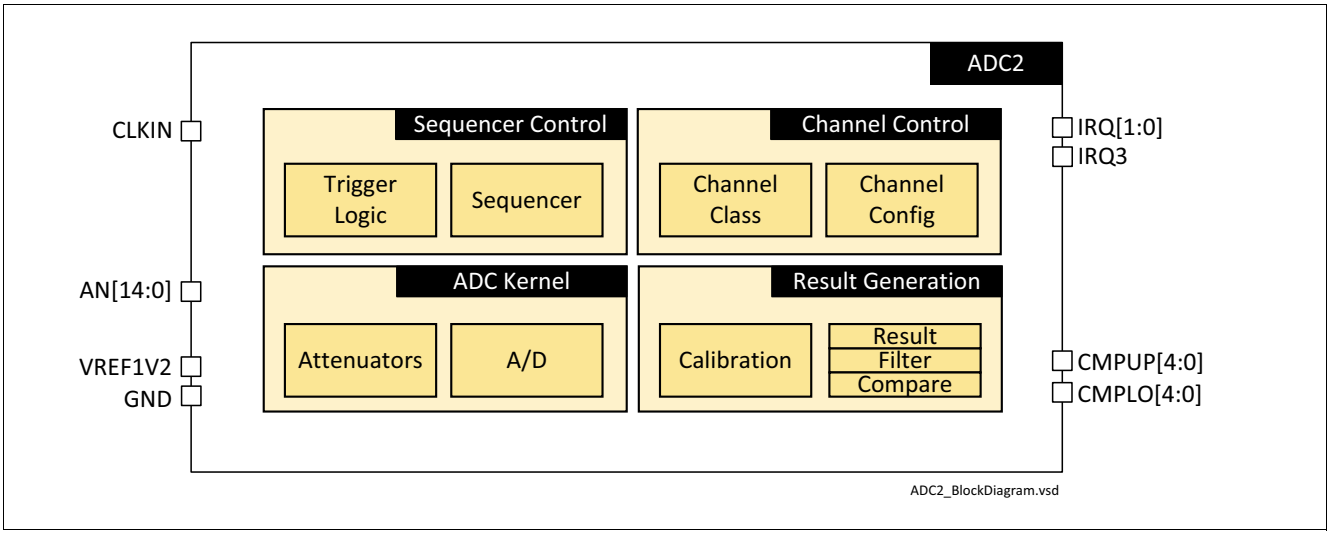


Figure 218 Block diagram ADC2

17.3 Toplevel signals

Table 210 ADC2 clock domain

Signal	Direction	Description	From/To
CLKIN	Input	Module clock ADC_CLK or f_{ADC2} in this chapter	See **Product definitions**, **SCU interconnections**

Table 211 ADC2 reference voltage

Name	Direction	Description	From/To
VREF1V2	Input	Analog reference voltage	ARVG
VAGND	Input	Analog reference ground	VAGND pin

Table 212 ADC2 analog inputs

Name	Direction	Description	From/To
AN[6:0]	Input	HV input	See **Product definitions**, **ADC2 interconnections**
AN[14:7]	Input	MV input	

Table 213 ADC2 outputs

Name	Direction	Description	From/To
CMPLO[3:0]	Output	Lower threshold compare output of result generation	see **Product definitions**, **ADC2 interconnections**
CMPUP[3:0]	Output	Upper threshold compare output of result generation	

Table 214 ADC2 interrupt requests

Name	Direction	Description	From/To
IRQ[1:0]	Output	Interrupt service request	see **Product definitions**, **Interrupt request mapping**
IRQ3	Output	Exception service request	see **Product definitions**, **Exception request mapping**

17.4 Interrupts

Events

The ADC2 provides following events:

- Channel “conversion finished” events (see CHSTAT.CH[n])
- Sequencer “finished” events (see SQSTAT.SQ[x], SQSTAT.COLL[x])
- Compare “match” events (see CMPSTAT.CMP_UP_IS[x], CMPSTAT.CMP_LO_IS[x]). The compare events are interconnected to other peripherals (see **Product definitions**, **ADC2 interconnections**)

Interrupt requests

The events can request an interrupt. Therefore the events can be assigned via the node pointer scheme to an interrupt or an exception request at the NVIC. The ADC2 has three node pointers with three interrupt request lines (ADC2.IRQ3, ADC2.IRQ[1:0]).

The node pointer scheme is described in **Figure 219**.

The node pointer to NVIC assignment is described in **Product definitions**, **Interrupt request mapping** and **Product definitions**, **Exception request mapping**.

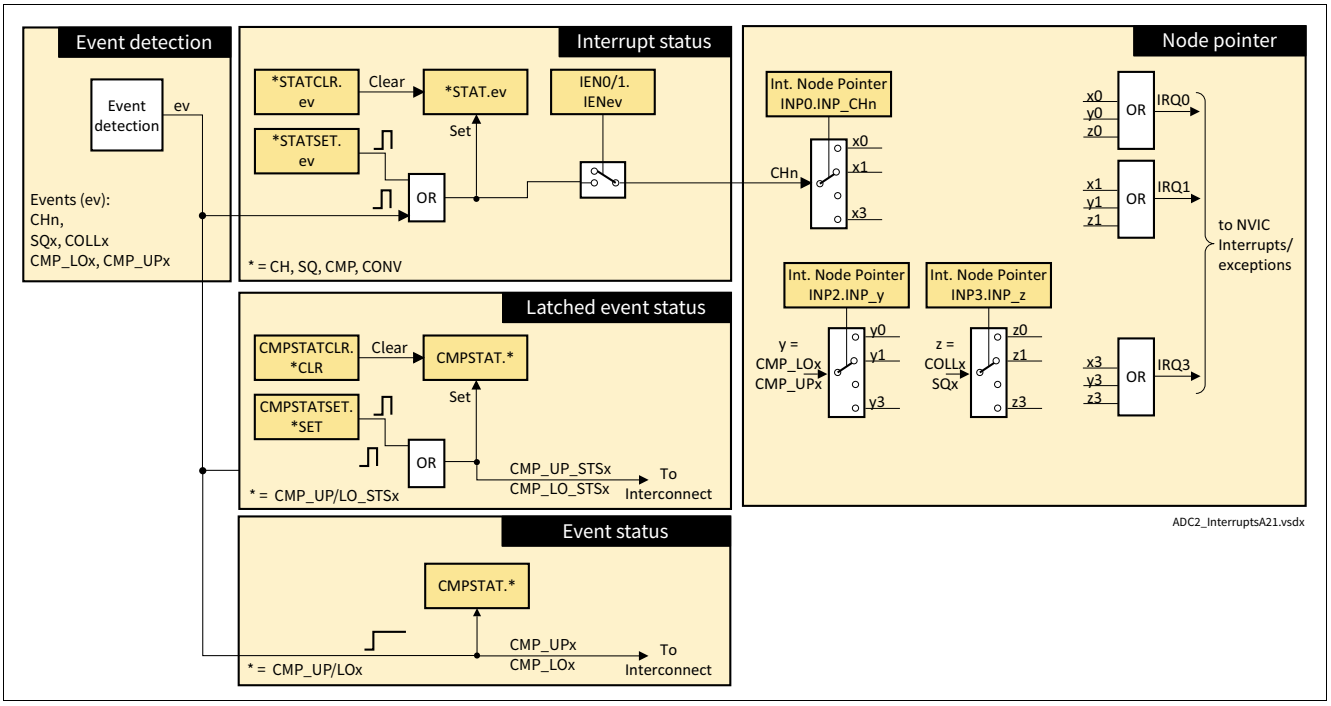


Figure 219 Event handling

17.5 Operation mode behavior

Table 215 Operation mode behavior ADC2

Reset	The ADC module is reset via RESET_TYPE_4 and RESET_TYPE_5. An activated RESET has following effects: • All SFRs of the ADC are reset according to its reset type • The state machines of the ADC kernel and sequencer are reset • The generated events show reset value
Power-up/ Power-down	• The ADC is kept in reset state as long as supply is not in specified operating range • The ADC is released from reset state when supply is in specified operating range
Active mode	• The ADC module is enabled automatically
Stop mode	• The ADC is unpowered, the SFR content and all states are lost • The Stop mode entry is triggered by setting PMCON0.STOP, the entry sequence is done automatically • The ADC wakes from Stop mode with a power-up
Sleep mode	• The ADC is unpowered, the SFR content and all states are lost • The Stop mode entry is triggered by setting PMCON0.SLEEP, the entry sequence is done automatically • The ADC wakes from Sleep mode with a power-up
Fail sleep mode	• None

17.6 Working principle

The ADC2 working principle is similar to the ADC1. The ADC2 has a pre-programmed conversion scheme which cannot be influenced by the user. The user interface is described in the following chapters.

The working principle can be taken from **Table 216** and **Figure 220**.

Table 216 ADC2 working principle

Step	Description	Condition	Repeat
Step 1:	A software trigger event requests the initial sequence SQ0. (TRGSW). SQ1 trigger event is EVSQ2, SQ2 trigger event is EVSQ2, SQ3 trigger event is EVSQ2. Once started SQ0 trigger source is re-programmed to SQ3 event (EVSQ3). This allows a continues round-robing sequence.		
Step 2:	The sequencer reads from the requested sequence the first (second, third, forth) slot channel number (CHSELx) and provides it to the channel control.		
Step 3:	The channel control provides the channel specific configuration to the ADC kernel (INSEL, CONVCFG) and to the result generation (INSEL, CHSEL, FILSEL, CMPSEL, CHREP).		
Step 4:	The sequencer triggers the start of conversion (SoC) to the ADC kernel.		
Step 5:	The ADC kernel performs the conversion according to the channel configuration and signalizes the end of conversion (EoC) to the sequencer control.		
Step 6:	The ADC kernel provides the digital result to the result generation. The result generation post-processes it and generates the result events (VALIDn, CHn, FILx, CMPx) accordingly.		
Step 7:	The sequencer repeats with Step 4 in case the channel has to be repeated (CHREPn), otherwise it continues with Step 8. Note: a channel event CHn is signalized after every EoC.	CHREPn	Step 4
Step 8:	The sequencer repeats with Step 2 with its next defined slot, otherwise it continues with Step 9.	SLOTS	Step 2
Step 9:	The sequencer signalizes the end of a sequence via its SQx event.		
Step 10:	The sequencer repeats with Step 2 in case the sequence has to be repeated (SQREPx). Otherwise it continues with Step 11.	SQREPx	Step 2
Step 11:	The sequencer is set to idle.		

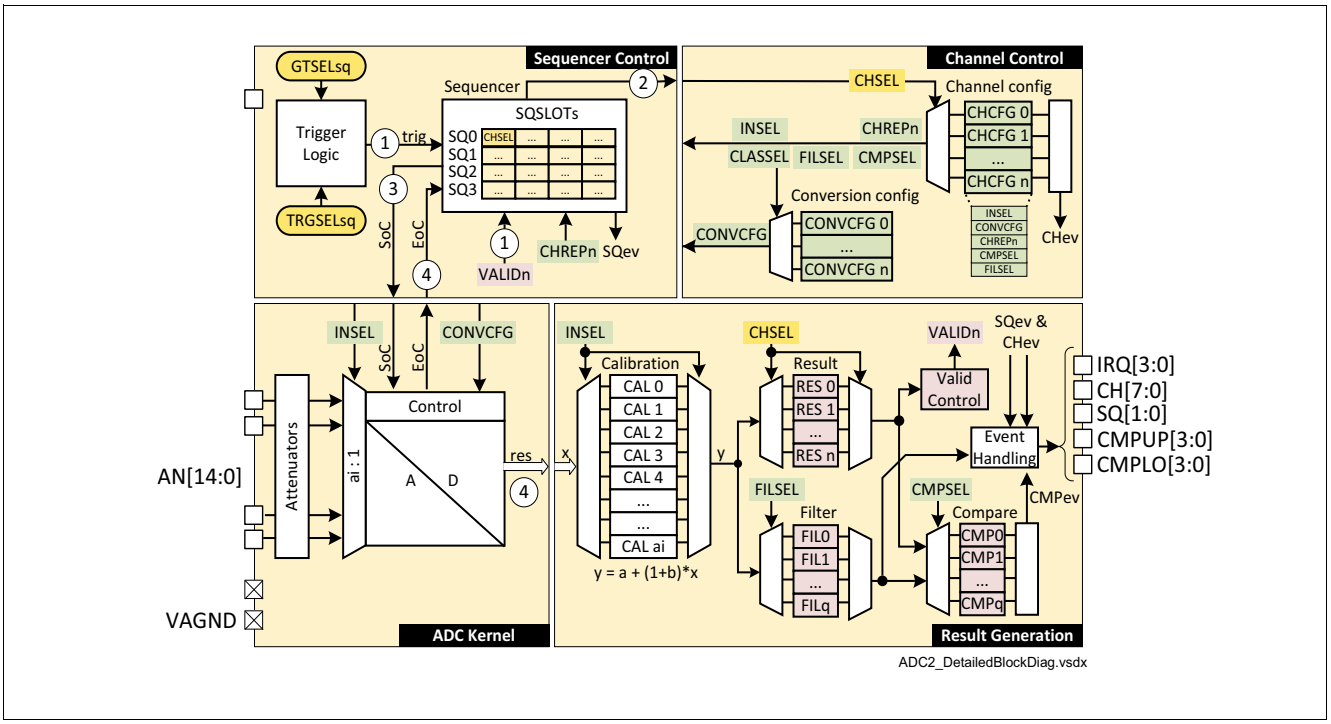


Figure 220 ADC2 detailed and functional block diagram

The following sections describe the pre-configuration of the ADC2, the possibilities of user configuration, and how the ADC2 measurement results can be accessed. The sequencer control, channel control, and result generation of the ADC2 use the same building blocks as the ADC1 and are not described here (for details see the **Analog Digital Converter 1 (ADC1)** chapter).

17.7 Pre-configuration of ADC2

The ADC2 SFRs are pre-configured and can be taken from **Table 217**.

The ADC2 conversion sequence is pre-configured and can be taken from **Figure 221**.

- The analog inputs ANi (i = 0 to 14) have a fixed assignment to the digital channels CHn, (n = 0 to 14)
- Each digital channel CHn has a fixed result register RESn, (n = 0 to 14)
- Some digital channels CHn have a fixed compare channel assignment via CHCFGn.CMPSEL (n = 0 to 14). Six out of eight compare channels are pre-configured via CMPCFGq (q = 0 to 7)
- Some digital channels CHn have a fixed filter channel assignment via CHCFGn.FILSEL (n = 0 to 14). Six out of eight filter channels are pre-configured via FILTCFG.COEF_Aq (q = 0 to 7)

The fixed assignment ensures that the diagnosis and supervisory functions of the ADC2 are not changed. This is necessary as there are hardware protection mechanism associated.

User configurable options

The user can configure compare and filter channels according to his application specific needs. The user configurable options are marked as “user cfg” in **Table 217**.

- Some digital channels CHn have user configurable compare channels. Two out of eight compare channels can be configured via CMPCFGq (q = 0 to 7)
- Some digital channels CHn have user configurable filter channels. Two out of eight filter channels can be configured via FILTCFG.COEF_Aq (q = 0 to 7)
- Compare and filter events can be configured as interrupt requests (IRQs) via IEN0/1, INP0/2/3. The event status is available in CMPSTAT and FILSTAT and can be serviced via the corresponding “SET” and “CLR” bits

Note: Writing to pre-configured registers (type “rwpt”) raises exceptions (bus faults/hard faults).

Table 217 ADC2 pre-configuration

Analog Input Signal	CHn/ RESn	INSEL [1)]	Assigned SQx	Assigned SLOTx	Assigned CMPSEL	Assigned FILSEL	Assigned CLASSEL	Fixed attenuation
VS	0	00_H	0	0	user cfg	user cfg	user cfg	ATT_TYP0 (12/256)
VSD	1	03_H	0	3	0	0	1	ATT_TYP1 (10/256)
VSD	2	01_H	0	1	1	1	1	ATT_TYP3 (38/256)
VCP	3	02_H	0	2	2	2	1	ATT_TYP2 (6/256)
MON1	4	04_H	1	1	user cfg	user cfg	user cfg	ATT_TYP1 (10/256)
MON2	5	05_H	1	2	user cfg	user cfg	user cfg	ATT_TYP1 (10/256)
MON3	6	06_H	1	3	user cfg	user cfg	user cfg	ATT_TYP1 (10/256)
VDDEXT	7	07_H	2	0	user cfg	user cfg	user cfg	ATT_TYP4 (58/256)

Table 217 ADC2 pre-configuration (cont'd)

Analog Input Signal	CHn/ RESn	INSEL [1]	Assigned SQx	Assigned SLOTx	Assigned CMPSEL	Assigned FILSEL	Assigned CLASSEL	Fixed attenuation
VCAN	8	08_H	2	1	3	3	0	ATT_TYP4 (58/256)
VDDP	9	09_H	2	2	user cfg	user cfg	user cfg	ATT_TYP4 (58/256)
VDDC	10	$0A_H$	2	3	user cfg	user cfg	user cfg	ATT_TYP4 (58/256)
VAREF	11	$0B_H$	3	0	user cfg	user cfg	user cgf	ATT_TYP4 (58/256)
VGEN_ADC	12	$0C_H$	3	1	user cfg	user cfg	user cfg	ATT_TYP4 (58/256)
TEMP0 (BRDRV)	13	$0D_H$ [2]	1	0	4	4	0	ATT_TYP4 (58/256)
TEMP1 (SYS_TEMP)	14	$0D_H$ [2]	3	2	5	5	0	ATT_TYP4 (58/256)

1) The values for INSEL are hard-coded in the ADC2 and therefore are not visible in register bit fields.
2) The two temperature sensors share the same ADC2 analog input.

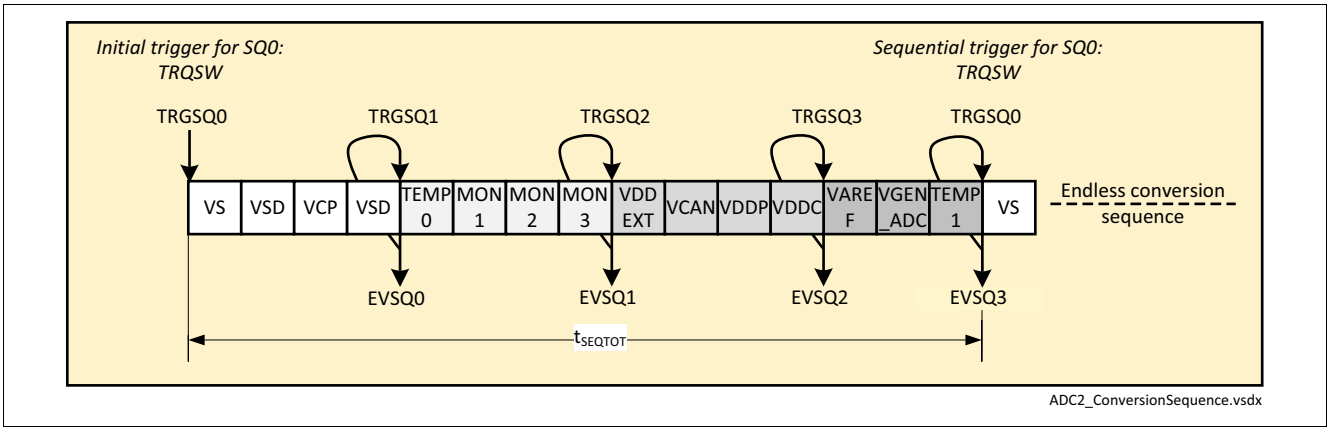


Figure 221 Pre-programmed conversion sequence

Note: The sequence is valid for both package variants (48 pins and 64 pins). For 48-pin package MON2 and MON3 are not available, therefore the associated results in RES5 (MON2) and RES6 (MON3) have an undefined value.

17.8 Analog and digital values

The nominal transfer curve of the ADC2 is given by the formulas:

(17.1)

$$\text{ADCout} = \frac{V_{AN} * \text{ATT\_TYP}x}{V_{LSB}}$$

where ADCout is the digital output value, V_{AN} is the analog input voltage, ATT_TYPx the fixed attenuation factor of the channel (see **Table 217**), and

(17.2)

$$V_{LSB} = \frac{V_{REF1V2}}{2^n - 1}$$

where V_{REF1V2} is the reference voltage provided by the ARVG module (typ. 1.211 V) and n is the bit width of the corresponding register bit field (see **Chapter 17.11**):

- n = 8 for compare values (CMPCFGx.UPPER and CMPCFGx.LOWER, x = 0 to 7)
- n = 12 for calibrated result values (RESn.RESULT, n = 0 to 14)
- n = 14 for IIR filtered values (FILq.FILRESULT, q = 0 to 7)

17.9 Clock configuration

The ADC2 module has an input clock CLKIN which has to be adapted to the internal module clock ADC_CLK (see **Figure 222**).

- ADC_CLK must fulfill the limits specified in f_{ADC2}
- ADC_CLK is used for all timing calculation
- ADC_CLK can be programmed via CLKCON.CLKDIV

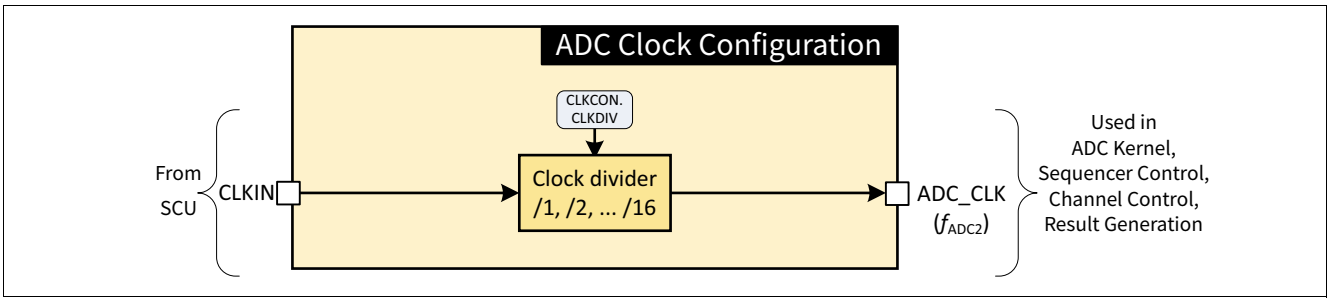


Figure 222 Clock configuration

*Note: The sample times are pre-configured to fulfill $tsamp_{HV}$ and $tsamp_{MV}$ at f_{ADC2} = 30 MHz, i.e. depending on the configuration of CLKIN (see the **System Control Unit (SCU)** chapter) , CLKCON.CLKDIV must be configured to get $f_{ADC2} \leq$ 30 MHz.*

17.10 Conversion sequence timing

The conversion sequence consists of 15 analog input channels as shown in **Figure 221**. The total time of one sequence and hence its repetition rate can be calculated as follows:

- 7 HV channels (VS, 2x VSD, VCP, MON1, MON2, MON3; pre-assigned to CONVCFG1):
  - Pre-configured sample time (CONVCFG1.STC): 44 ADC_CLK cycles
  - Pre-configured conversion time: 18 ADC_CLK cycles
  - Total per HV channel: 62 ADC_CLK cycles
- 8 MV channels (VDDEXT, VCAN, VDDP, VDDC, VAREF, VGEN_ADC, TEMP0, TEMP1; pre-assigned to CONVCFG0):
  - Pre-configured sample time (CONVCFG0.STC): 12 ADC_CLK cycles
  - Pre-configured conversion time: 18 ADC_CLK cycles
  - Total per MV channel: 30 ADC_CLK cycles

As a result, the total conversion sequence takes 674 ADC_CLK cycles.

Depending on f_{ADC2} this results to, e.g.:

- t_{SEQTOT} = 22.47 µs @ f_{ADC2} = 30 MHz
- t_{SEQTOT} = 33.7 µs @ f_{ADC2} = 20 MHz

Note: These numbers are only valid for the pre-configured conversion sequence. Any changes on user-configurable channels will influence t_{SEQTOT} accordingly.

17.11 Results generation features

The result generation is according to following digital signal path:

- Analog input calibration
- Result register with a valid control
- Result filters with configuration options
- Compare registers with configuration options
- Event handling

17.11.1 Analog input calibration

The analog input calibration can be enabled by CALEN.CALEN[i] (i = 0 to 14) and protected by CALPEN.CALPEN[i] (i = 0 to 14).

17.11.2 Result register and valid control

After passing through the calibration unit the results are 12 bits wide and stored in the RESn.RESULT (n = 0 to 14) bit fields. If calibration is disabled the raw 10-bit ADC result is stored at RESn.RESULT[11:2] while RESn.RESULT[1:0] = 00_B.

The flag RESn.VALID is set according to following definition:

- VALID is set if the result is updated with a new value
- VALID is cleared if the result is read by CPU

17.11.3 Result filter

There is an optional and configurable first order IIR filter which can be used to post-process the digital result. There are eight independent filter channels available. The filter channel can be selected in CHCFGn.FILSEL

(n = 0 to 14) or is already pre-configured according to **Table 217**. The filter is configured in FILTCFG.COEF_Aq (q = 0 to 7) and the resulting 14-bit values are stored in FILq.FILRESULT (q = 0 to 7).

17.11.4 Compare channels

There are eight digital compare channels with programmable thresholds for background signal supervision with limit checking. The compare channel can be selected in CHCFGn.CMPSEL (n = 0 to 14) or is already pre-configured according to **Table 217**. The compare channel is configured in CMPCFGq (q = 0 to 7). For a summary of the settings of the pre-configured compare channels see **Table 218**:

Table 218 Pre-configured compare channels

q	Analog input CHn	Fixed attenuation	MODE[1]	UPPER[1] / LOWER[1]	[V][2]	HYST_UP[1] / HYST_LO[1]	BLANK_TIME[1]
0	VSD 03_H	ATT_TYP1 (10/256)	OV	EF_H	28.94 V	HYST8	BLANK_5
				95_H	18.04 V	HYST8	
1	VSD 01_H	ATT_TYP3 (38/256)	UV	$A6_H$	5.29 V	HYST8	BLANK_5
				$A6_H$	5.29 V	HYST8	
2	VCP 02_H	ATT_TYP2 (6/256)	RANGE	DA_H	44.00 V	HYST8	BLANK_5
				46_H	14.13 V	HYST8	
3	VCAN 08_H	ATT_TYP4 (58/256)	RANGE	FE_H	5.30 V	HYST4	BLANK_10
				DC_H	4.59 V	HYST4	
4	TEMP0 $0D_H$	ATT_TYP4 (58/256)	OV	37_H	1.15 V	HYST4	BLANK_5
				$2D_H$	0.94 V	HYST4	
5	TEMP1 $0E_H$	ATT_TYP4 (58/256)	OV	37_H	1.15 V	HYST4	BLANK_5
				$2D_H$	0.94 V	HYST4	

1) See corresponding bitfield description in CMPCFGq (q = 0 to 7).
2) Resulting voltage values.

17.11.5 Event handling

The result and compare events are available as interrupt request according to the scheme in **Figure 219**.

17.12 Suspend mode feature

For debugging purpose the module can be suspended if enabled. In suspend mode the ADC_CLK is halted but the SFR access via the debugger is possible.

The suspend mode can be enabled in SUSCTR.SUSEN.

There are two suspend modes, configurable in SUSCTR.SUSMODE:

- Hard suspend, i.e. the module clock is halted immediately
- Soft suspend, i.e. the module clock is halted after the ongoing sequence is finished and its results are written

The suspend status can be read from SUSSTAT.STAT.

17.13 Register description ADC2

17.13.1 ADC2 Address Maps

Table 219 Register Address Space - ADC2

Module	Base Address	End Address	Note
ADC2	$4804C000_H$	$4804FFFF_H$	

Table 220 Register Overview - ADC2 (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
GLOBCONF	Global Configuration Register	0000_H	740
CLKCON	Clock Control Register	0004_H	740
SUSCTR	Suspend Control Register	0008_H	741
SUSSTAT	Suspend Status Register	$000C_H$	741
SQSTAT	Sequence Status Register	0010_H	742
SQSTATCLR	Sequence Status Clear Register	0014_H	742
SQSTATSET	Sequence Status Clear Register	0018_H	743
CHSTAT	Channel Status Register	$001C_H$	744
CHSTATCLR	Channel Status Register	0020_H	744
CHSTATSET	Channel Status Register	0024_H	745
CALEN	Calibration Enable	0028_H	745
CALPEN	Calibration Protection Enable	$002C_H$	746
FILx	Filter Result Register	0030_H+x*4	746
FILSTAT	Filter Status Register	0050_H	747
FILSTATCLR	Filter Status Clear Register	0054_H	747
FILSTATSET	Filter Status Clear Register	0058_H	748
RESx	Result Register	$005C_H$+x*4	748
CMPSTAT	Compare Status Register	0098_H	749
CMPSTATCLR	Compare Status Clear Register	$009C_H$	749
CMPSTATSET	Compare Status Clear Register	$00A0_H$	750
IEN0	Interrupt Enable Register 0	$00A4_H$	751
IEN1	Interrupt Enable Register 1	$00A8_H$	751
INP0	Interrupt Node Pointer Register 0	$00AC_H$	752
INP2	Interrupt Node Pointer Register 2	$00B0_H$	753
INP3	Interrupt Node Pointer Register 3	$00B4_H$	753
INTSTAT	Internal Configuration Register	$00C8_H$	754
SQCFG0	Sequence Configuration Register 0	$00D8_H$	754

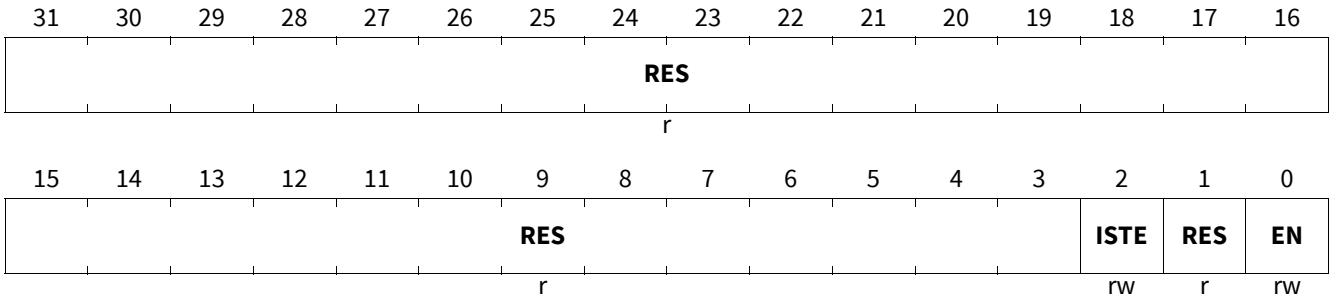
Table 220 Register Overview - ADC2 (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Page Number
SQCFG1	Sequence Configuration Register 1	$00DC_H$	755
SQCFG2	Sequence Configuration Register 2	$00E0_H$	756
SQCFG3	Sequence Configuration Register 3	$00E4_H$	757
CHCFG0	Channel Configuration Register 0	$00E8_H$	758
CHCFG1	Channel Configuration Register 1	$00EC_H$	759
CHCFG2	Channel Configuration Register 2	$00F0_H$	760
CHCFG3	Channel Configuration Register 3	$00F4_H$	761
CHCFG4	Channel Configuration Register 4	$00F8_H$	762
CHCFG5	Channel Configuration Register 5	$00FC_H$	763
CHCFG6	Channel Configuration Register 6	0100_H	764
CHCFG7	Channel Configuration Register 7	0104_H	765
CHCFG8	Channel Configuration Register 8	0108_H	766
CHCFG9	Channel Configuration Register 9	$010C_H$	767
CHCFG10	Channel Configuration Register 10	0110_H	768
CHCFG11	Channel Configuration Register 11	0114_H	769
CHCFG12	Channel Configuration Register 12	0118_H	770
CHCFG13	Channel Configuration Register 13	$011C_H$	771
CHCFG14	Channel Configuration Register 14	0120_H	772
CMPSTAT2	Compare Status Register	0124_H	773
CALAIx	Calibration Setting for Analog Inputs	0128_H+x*4	774
SQSLOT0	SQ Channel Slot Register 0	0168_H	774
SQSLOT1	SQ Channel Slot Register 1	$016C_H$	776
SQSLOT2	SQ Channel Slot Register 2	0170_H	777
SQSLOT3	SQ Channel Slot Register 3	0174_H	778
CONVCFG0	Conversion Configuration Register 0	0178_H	779
CONVCFG1	Conversion Configuration Register 1	$017C_H$	780
CONVCFG2	Conversion Configuration Register 2	0180_H	781
CONVCFG3	Conversion Configuration Register 3	0184_H	782
CMPCFG0	Compare Channel 0 Control Register	0188_H	783
CMPCFG1	Compare Channel 1 Control Register	$018C_H$	784
CMPCFG2	Compare Channel 2 Control Register	0190_H	786
CMPCFG3	Compare Channel 3 Control Register	0194_H	787
CMPCFG4	Compare Channel 4 Control Register	0198_H	788
CMPCFG5	Compare Channel 5 Control Register	$019C_H$	790
CMPCFG6	Compare Channel 6 Control Register	$01A0_H$	791
CMPCFG7	Compare Channel 7 Control Register	$01A4_H$	792
FILTCFG	Filter Configuration	$01A8_H$	793

17.13.2 ADC2 Registers

Global Configuration Register

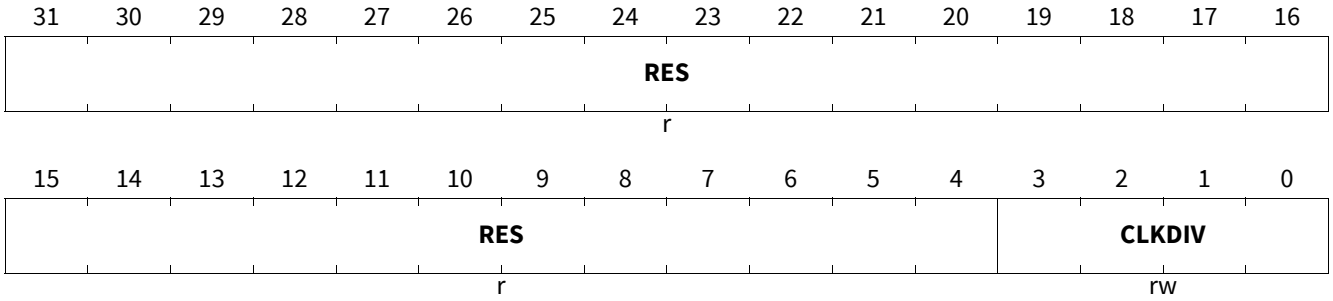
GLOBCONF
Global Configuration Register **(0000_H)** **RESET_TYPE_5 Value: 0000 0001$_H$**



Field	Bits	Type	Description
EN	0	rw	**Module Enable** 0_B **Disabled**, ADC2 Module is disabled 1_B **Enabled**, ADC2 Module is enabled
RES	1, 31:3	r	**Reserved**
ISTE	2	rw	**Idle shadow transfer enable** 0_B **Disabled**, Shadow transfer done based on selected shadow transfer trigger source 1_B **Enabled**, Shadow transfer is automatically enabled when no SQ is running

Clock Control Register

CLKCON
Clock Control Register **(0004_H)** **RESET_TYPE_5 Value: 0000 0001$_H$**



Field	Bits	Type	Description
CLKDIV	3:0	rw	**Clock Divider Settings** 0_H **DIV_1**, Divide by 1 … F_H **DIV_16**, Divide by 16
RES	31:4	r	**Reserved**

Register description ADC2

Suspend Control Register

SUSCTR

Suspend Control Register **(0008_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
SUSEN	0	rw	**ADC2 Suspend Enable** 0_B **Disable**, ADC2 Suspend Mode is disabled 1_B **Enable**, ADC2 Suspend Mode is enabled
SUSMOD	1	rw	**Suspend Mode** 0_B **Hard**, Hard Suspend Mode 1_B **Soft**, Soft Suspend Mode
RES	31:2	r	**Reserved**

Suspend Status Register

SUSSTAT

Suspend Status Register **($000C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
STAT	0	r	**Suspend Mode Status** 0_B **No_suspend**, Suspend Mode is not entered 1_B **Suspend**, Suspend mode is entered
RES	31:1	r	**Reserved**

Register description ADC2

Sequence Status Register

SQSTAT
Sequence Status Register **(0010_H)** **RESET_TYPE_5 Value: $0007\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES													SQNUM		
r													r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				COLL				RES				SQ			
r				rhxre				r				rhxre			

Field	Bits	Type	Description
SQ	3:0	rhxre	**SQ Interrupt Status** Each bit represents the interrupt status of a sequence 0: Not finished - Sequence has not finished or has not started 1: finished - Sequence has finished
RES	7:4, 15:12, 31:19	r	**Reserved**
COLL	11:8	rhxre	**Collision Status** Each bit represents the collision interrupt Status of a sequence 0: No Error - No Sequence Collision occured 1: Error - Sequence Collision occured
SQNUM	18:16	r	**Actual Sequence processed** 000_B **SQ0**, SQ 0 is running 001_B **SQ1**, SQ1 is running 010_B **SQ2**, SQ2 is running 011_B **SQ3**, SQ3 is running 111_B **NoSQ**, No SQ is running

Sequence Status Clear Register

SQSTATCLR
Sequence Status Clear Register **(0014_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				COLLCLR				RES				SQCLR			
r				w				r				w			

Register description ADC2

Field	Bits	Type	Description
SQCLR	3:0	w	**SQ Interrupt Status Clear** Each bit represents the corresponding interrupt Status clear 0: Not Cleared - SQ Status not cleared 1: Cleared - SQ Status cleared
RES	7:4, 31:12	r	**Reserved**
COLLCLR	11:8	w	**Collision Status Clear** Each bit represents the corresponding Collision Status Clear 0: Not Cleared - Sequence Collision Status not cleared 1: Cleared - Sequence Collision Status cleared

Sequence Status Clear Register

SQSTATSET
Sequence Status Clear Register **(0018_H)** **RESET_TYPE_5 Value: 0000 0000_H**

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

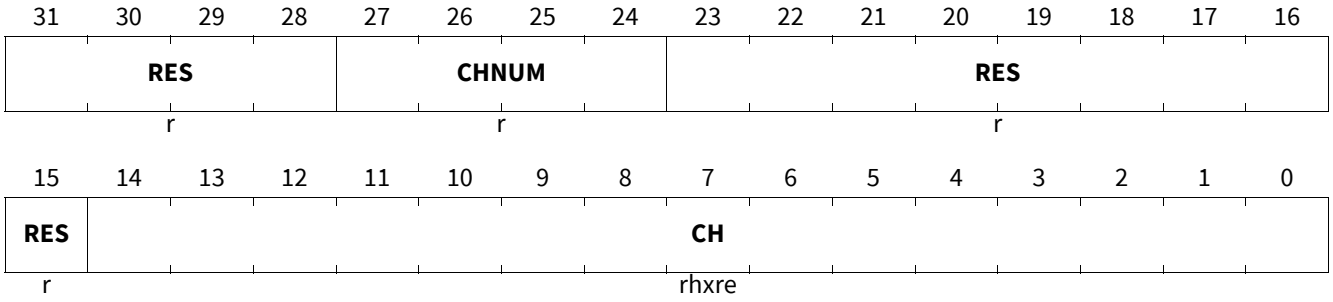
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				COLLSET				RES				SQSET			
r				w				r				w			

Field	Bits	Type	Description
SQSET	3:0	w	**SQ Interrupt Status Set** Each bit represents the corresponding SQ Interrupt Status Set 0: Not Set - SQ Status not set 1: Set - SQ Status set
RES	7:4, 31:12	r	**Reserved**
COLLSET	11:8	w	**Collision Status Set** Each bit represents the corresponding Collision Status Set 0: Not Set - Sequence Collision Status not set 1: Set - Sequence Collission Status set

Register description ADC2

Channel Status Register

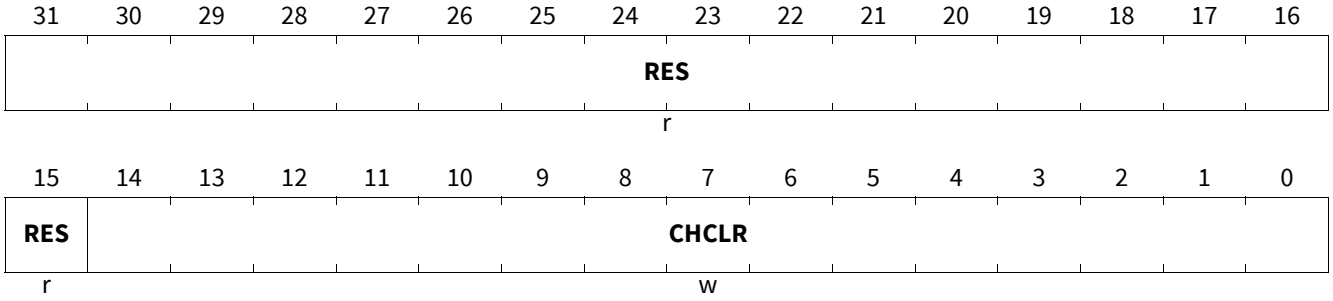
CHSTAT
Channel Status Register **($001C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CH	14:0	rhxre	**Channel Status** Each bit represents the corresponding Channel Status 0: Not finished - Channel conversion not finished 1: finished - Channel conversion finished
RES	23:15, 31:28	r	**Reserved**
CHNUM	27:24	r	**Current Channel under conversion**

Channel Status Register

CHSTATCLR
Channel Status Register **(0020_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CHCLR	14:0	w	**Channel Status clear flag** Each bit represents the corresponding Channel Status clear flag 0: Not Cleared - Channel Status flag not cleared 1: Cleared - Channel Status flag cleared
RES	31:15	r	**Reserved**

Register description ADC2

Channel Status Register

CHSTATSET
Channel Status Register **(0024_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								CHSET							
r								w							

Field	Bits	Type	Description
CHSET	14:0	w	**Channel Status set flag** Each bit represents the corresponding Channel Status set flag 0: Not Set - Channel Status flag not set 1: Set - Channel Status flag set
RES	31:15	r	**Reserved**

Calibration Enable

CALEN
Calibration Enable **(0028_H)** **RESET_TYPE_5 Value: $0000\ 7FFF_H$**

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

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

Field	Bits	Type	Description
CALEN	14:0	rw	**Channel Calibration Enable** Each bit represents the corresponding Channel Calibration Enable 0: Channel calibration disabled 1: Channel calibration enabled
RES	31:15	r	**Reserved**

Register description ADC2

Calibration Protection Enable

CALPEN
Calibration Protection Enable **($002C_H$)** **RESET_TYPE_5 Value: $0000\ 3FFF_H$**

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

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

Field	Bits	Type	Description
CALPEN	14:0	rw	**Calibration Protection** Each bit represents the corresponding Calibration Protection 0: Channel calibration setting is unprotected 1: Channel calibration setting protected
RES	31:15	r	**Reserved**

Filter Result Register

FILx (x=0-7)
Filter Result Register **(0030_H+x*4)** **RESET_TYPE_5 Value: $XXXX\ XXXX_H$**

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

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

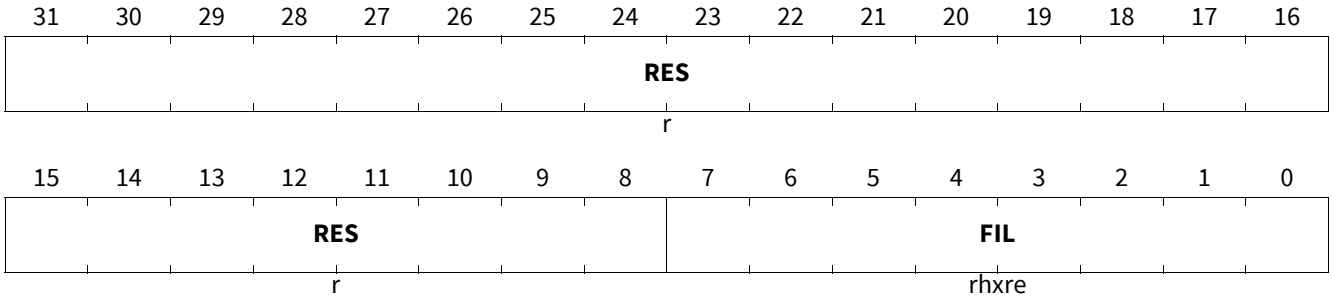
Field	Bits	Type	Description
FILRESULT	13:0	r	**Filter Result Value**
RES	31:14	r	**Reserved**

Register description ADC2

Filter Status Register

FILSTAT

Filter Status Register **(0050_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

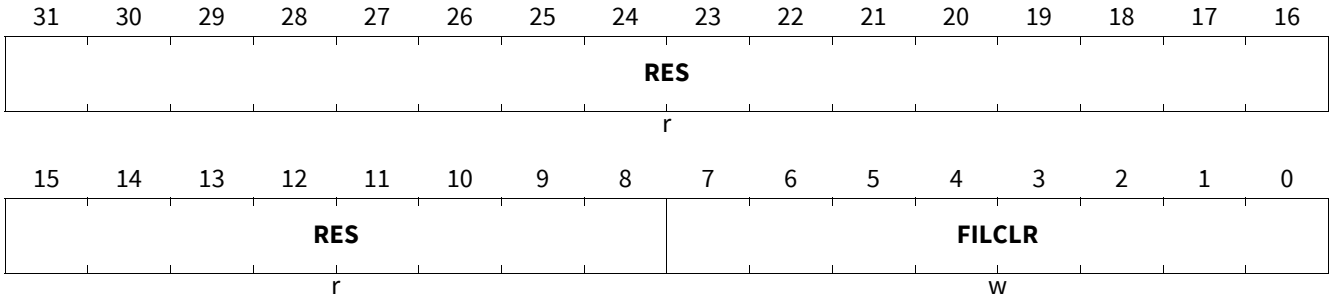


Field	Bits	Type	Description
FIL	7:0	rhxre	**Filter Event flag** Each bit represents the corresponding Filter Event flag 0: No result - No new result available 1: New Result - New Result available
RES	31:8	r	**Reserved**

Filter Status Clear Register

FILSTATCLR

Filter Status Clear Register **(0054_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

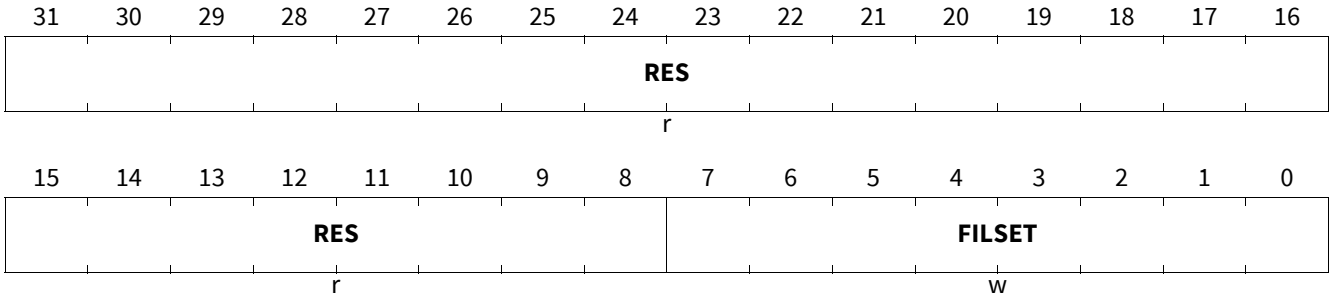


Field	Bits	Type	Description
FILCLR	7:0	w	**Filter Event flag** Each bit represents the corresponding Filter Event clear 0: Not Cleared - Corresponding Filter Status flag not cleared 1: Cleared - Corresponding Filter Status flag cleared
RES	31:8	r	**Reserved**

Register description ADC2

Filter Status Clear Register

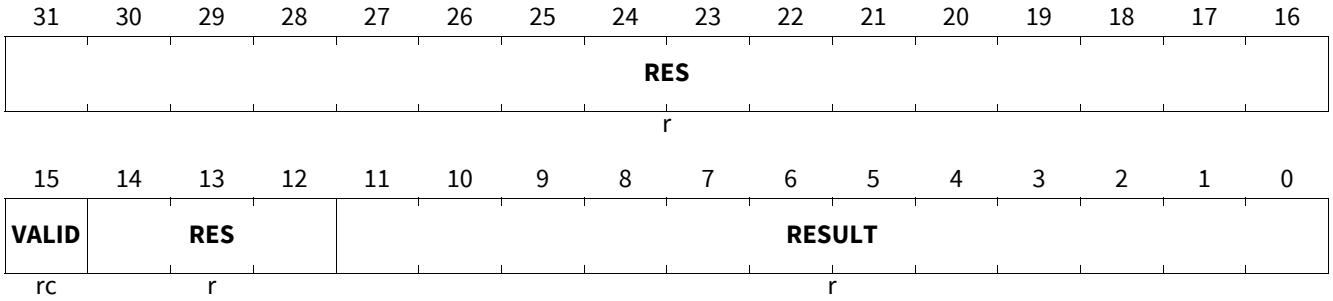
FILSTATSET
Filter Status Clear Register **(0058_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
FILSET	7:0	w	**Filter Event set flag** Each bit represents the corresponding Filter Event set 0: Not Set - Corresponding Filter Status flag not set 1: Set - Corresponding Filter Status flag set
RES	31:8	r	**Reserved**

Result Register

RESx (x=0-14)
Result Register **($005C_H$+x*4)** **RESET_TYPE_5 Value: $XXXX\ XXXX_H$**



Field	Bits	Type	Description
RESULT	11:0	r	**Result Value**
RES	14:12, 31:16	r	**Reserved**
VALID	15	rc	**Valid flag** 0_B **Not_Valid**, No new result available 1_B **Valid**, New result available

Register description ADC2

Compare Status Register

CMPSTAT
Compare Status Register **(0098_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMP_UP_IS								CMP_UP_STS							
rhxre								rhxr							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP_LO_IS								CMP_LO_STS							
rhxre								rhxr							

Field	Bits	Type	Description
CMP_LO_STS	7:0	rhxr	**Latched Compare low Status** Each bit represents the corresponding Compare low Status 0: INACTIVE - No Lower Compare Event occured 1: ACTIVE - Lower Compare Event occured
CMP_LO_IS	15:8	rhxre	**Compare low Interrupt Status** Each bit represents the corresponding Compare low Interrupt Status 0: INACTIVE - No Lower Compare Event occured 1: ACTIVE - Lower Compare Event occured
CMP_UP_STS	23:16	rhxr	**Latched Compare up Status** Each bit represents the corresponding Compare up Status 0: INACTIVE - No Upper Compare Event occured 1: ACTIVE - Upper Compare Event occured
CMP_UP_IS	31:24	rhxre	**Compare up Interrupt Status** Each bit represents the corresponding Compare up Interrupt Status 0: INACTIVE - No Upper Compare Event occured 1: ACTIVE - Upper Compare Event occured

Compare Status Clear Register

CMPSTATCLR
Compare Status Clear Register **($009C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMP_UP_ISCLR								CMP_UP_STSCLR							
w								w							

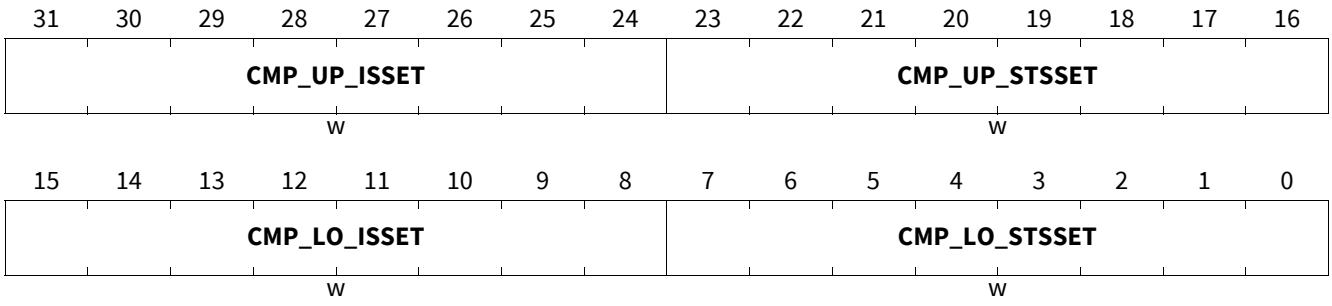
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP_LO_ISCLR								CMP_LO_STSCLR							
w								w							

Register description ADC2

Field	Bits	Type	Description
CMP_LO_STSCLR	7:0	w	**Compare low Status clear** Each bit represents the corresponding Compare low Status clear 0: Not Cleared - Corresponding Compare Status flag not cleared 1: Cleared - Corresponding Compare Status flag cleared
CMP_LO_ISCLR	15:8	w	**Compare low Interrupt clear** Each bit represents the corresponding Compare low interrupt clear 0: Not Cleared - Corresponding Compare Interrupt flag not cleared 1: Cleared - Corresponding Compare Interrupt flag cleared
CMP_UP_STSCLR	23:16	w	**Compare up Status clear** Each bit represents the corresponding Compare up status clear 0: Not Cleared - Corresponding Compare Status flag not cleared 1: Cleared - Corresponding Compare Status flag cleared
CMP_UP_ISCLR	31:24	w	**Compare up Interrupt clear** Each bit represents the corresponding Compare up Interrupt clear 0: Not Cleared - Corresponding Compare Status flag not cleared 1: Cleared - Corresponding Compare Status flag cleared

Compare Status Clear Register

CMPSTATSET
Compare Status Clear Register **($00A0_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CMP_LO_STSSET	7:0	w	**Compare low Status set** Each bit represents the corresponding Compare low Status set 0: Not Set - Corresponding Compare Status flag not set 1: Set - Corresponding Compare Status flag set
CMP_LO_ISSET	15:8	w	**Compare low Interrupt set** Each bit represents the corresponding Compare low Interrupt set 0: Not Set - Corresponding Compare Status flag not set 1: Set - Corresponding Compare Status flag set
CMP_UP_STSSET	23:16	w	**Compare up Status set** Each bit represents the corresponding Compare up Status set 0: Not Set - Corresponding Compare Status flag not set 1: Set - Corresponding Compare Status flag set

Register description ADC2

Field	Bits	Type	Description
CMP_UP_ISSET	31:24	w	**Compare up Interrupt set** Each bit represents the corresponding Compare up Interrupt set 0: Not Set - Corresponding Compare Status flag not set 1: Set - Corresponding Compare Status flag set

Interrupt Enable Register 0

IEN0
Interrupt Enable Register 0 **($00A4_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
IEN_CH	14:0	rw	**Channel Interrupt Enable** Each bit represents the corresponding Channel Interrupt Enable 0:Disable - Interrupt disabled 1:Enable - Interrupt enabled
RES	15, 31:16	r	**Reserved - always write 0**

Interrupt Enable Register 1

IEN1
Interrupt Enable Register 1 **($00A8_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
IEN_SQ	3:0	rw	**Sequence Interrupt Enable** Each bit represents the corresponding Sequence Interrupt Enable 0: Disable - Interrupt disabled 1: Enable - Interrupt Enabled

Register description ADC2

Field	Bits	Type	Description
IEN_COLL	7:4	rw	**Collision Interrupt Enable** Each bit represents the corresponding Collision Interrupt Enable 0: Disable - Interrupt disabled 1: Enable - Interrupt enabled
RES	15:8	r	**Reserved**
IEN_LO	23:16	rw	**Compare LO Interrupt Enable** Each bit represents the corresponding Compare LO Interrupt Enable 0: Disable - Interrupt disabled 1: Enable - Interrupt Enabled
IEN_UP	31:24	rw	**Compare UP Interrupt Enable** Each bit represents the corresponding Compare Up Interrupt Enable 0: Disable - Interrupt disabled 1: Enable - Interrupt enabled

Interrupt Node Pointer Register 0

INP0
Interrupt Node Pointer Register 0 **($00AC_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		**INP_CH14**		**INP_CH13**		**INP_CH12**		**INP_CH11**		**INP_CH10**		**INP_CH9**		**INP_CH8**	
r		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INP_CH7		**INP_CH6**		**INP_CH5**		**INP_CH4**		**INP_CH3**		**INP_CH2**		**INP_CH1**		**INP_CH0**	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
INP_CHx (x=0-14)	2*x+1:2*x	rw	**Channel Interrupt Node Pointer** 00_B **IRQ0**, mapped to IRQ0 01_B **IRQ1**, mapped to IRQ1 10_B **IRQ2**, not mapped 11_B **IRQ3**, mapped to IRQ3
RES	31:30	r	**Reserved**

Register description ADC2

Interrupt Node Pointer Register 2

INP2
Interrupt Node Pointer Register 2 **($00B0_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INP_CMP_UP 7		INP_CMP_UP 6		INP_CMP_UP 5		INP_CMP_UP 4		INP_CMP_UP 3		INP_CMP_UP 2		INP_CMP_UP 1		INP_CMP_UP 0	
rw		rw		rw		rw		rw		rw		rw		rw	

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

Field	Bits	Type	Description
INP_CMP_LOx (x=0-7)	2*x+1:2*x	rw	**Compare Lo Interrupt Node Pointer** 00_B **IRQ0**, mapped to IRQ0 01_B **IRQ1**, mapped to IRQ1 10_B **IRQ2**, not mapped 11_B **IRQ3**, mapped to IRQ3
INP_CMP_UPx (x=0-7)	2*x+17:2*x+16	rw	**Compare Up Interrupt Node Pointer** 00_B **IRQ0**, mapped to IRQ0 01_B **IRQ1**, mapped to IRQ1 10_B **IRQ2**, not mapped 11_B **IRQ3**, mapped to IRQ3

Interrupt Node Pointer Register 3

INP3
Interrupt Node Pointer Register 3 **($00B4_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INP_COLL3		INP_COLL2		INP_COLL1		INP_COLL0		INP_SQ3		INP_SQ2		INP_SQ1		INP_SQ0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
INP_SQx (x=0-3)	2*x+1:2*x	rw	**Sequence Interrupt Node Pointer** 00_B **IRQ0**, mapped to IRQ0 01_B **IRQ1**, mapped to IRQ1 10_B **IRQ2**, not mapped 11_B **IRQ3**, mapped to IRQ3

Register description ADC2

Field	Bits	Type	Description
INP_COLLx (x=0-3)	2*x+9:2*x+8	rw	**Collision Interrupt Node Pointer** 00_B **IRQ0**, mapped to IRQ0 01_B **IRQ1**, mapped to IRQ1 10_B **IRQ2**, not mapped 11_B **IRQ3**, mapped to IRQ3
RES	31:16	r	**Reserved**

Internal Configuration Register

INTSTAT
Internal Configuration Register **($00C8_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

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

Field	Bits	Type	Description
LOW_VOLT_STS	0	r	**ADC2 low voltage Status** 0_B **INACTIVE**, ADC2 is in normal operating condition 1_B **ACTIVE**, ADC2 is in Low Voltage operating condition
RES	31:1	r	**Reserved**

Sequence Configuration Register 0

SQCFG0
Sequence Configuration Register 0 **($00D8_H$)** **RESET_TYPE_5 Value: 0000 0204$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	TRGSW	RES				TRGSEL		RES		SQREP		RES	SLOTS		
r	w	r				rwpt		r		rwpt		r	rwpt		

Field	Bits	Type	Description
SLOTS	2:0	rwpt	**Number of used Slots in Sequence** 000_B **0**, No Slot used - SQ disabled 001_B **1**, Slots used … 100_B **4**, Slots used 101_B **5**, No Slot used - SQ disabled … 111_B **7**, No Slot used - SQ disabled
RES	3, 7:6, 13:10, 31:15	r	**Reserved**
SQREP	5:4	rwpt	**Sequence repetition** 00_B **1_REP**, The Sequence is executed 1 time 01_B **2_REP**, The Sequence is executed 2 times 10_B **4_REP**, The Sequence is executed 4 times 11_B **8_REP**, The Sequence is executed 8 times
TRGSEL	9:8	rwpt	**Trigger Select** 00_B **SWTRIG**, Software Trigger 01_B **TRIG_A**, External Trigger source A 10_B **TRIG_B**, External Trigger source B 11_B **NOT_USED**, not used - SWTRIG selected
TRGSW	14	w	**Software Trigger Bit** 0_B **Not_Triggered**, SQ not triggered 1_B **Triggered**, SQ is triggered

Sequence Configuration Register 1

SQCFG1
Sequence Configuration Register 1 **($00DC_H$)** **RESET_TYPE_5 Value: $0000\ 0204_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	TRGSW	RES				TRGSEL		RES		SQREP		RES	SLOTS		
r	w	r				rwpt		r		rwpt		r	rwpt		

Field	Bits	Type	Description
SLOTS	2:0	rwpt	**Number of used Slots in Sequence** 000_B **0**, No Slot used - SQ disabled 001_B **1**, Slots used … 100_B **4**, Slots used 101_B **5**, No Slot used - SQ disabled … 111_B **7**, No Slot used - SQ disabled
RES	3, 7:6, 13:10, 31:15	r	**Reserved**
SQREP	5:4	rwpt	**Sequence repetition** 00_B **1_REP**, The Sequence is executed 1 time 01_B **2_REP**, The Sequence is executed 2 times 10_B **4_REP**, The Sequence is executed 4 times 11_B **8_REP**, The Sequence is executed 8 times
TRGSEL	9:8	rwpt	**Trigger Select** 00_B **SWTRIG**, Software Trigger 01_B **TRIG_A**, External Trigger source A 10_B **TRIG_B**, External Trigger source B 11_B **NOT_USED**, not used - SWTRIG selected
TRGSW	14	w	**Software Trigger Bit** 0_B **Not_Triggered**, SQ not triggered 1_B **Triggered**, SQ is triggered

Sequence Configuration Register 2

SQCFG2
Sequence Configuration Register 2 **($00E0_H$)** **RESET_TYPE_5 Value: $0000\ 0204_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	TRGSW	RES				TRGSEL		RES		SQREP		RES	SLOTS		
r	w	r				rwpt		r		rwpt		r	rwpt		

Field	Bits	Type	Description
SLOTS	2:0	rwpt	**Number of used Slots in Sequence** 000_B **0**, No Slot used - SQ disabled 001_B **1**, Slots used … 100_B **4**, Slots used 101_B **5**, No Slot used - SQ disabled … 111_B **7**, No Slot used - SQ disabled
RES	3, 7:6, 13:10, 31:15	r	**Reserved**
SQREP	5:4	rwpt	**Sequence repetition** 00_B **1_REP**, The Sequence is executed 1 time 01_B **2_REP**, The Sequence is executed 2 times 10_B **4_REP**, The Sequence is executed 4 times 11_B **8_REP**, The Sequence is executed 8 times
TRGSEL	9:8	rwpt	**Trigger Select** 00_B **SWTRIG**, Software Trigger 01_B **TRIG_A**, External Trigger source A 10_B **TRIG_B**, External Trigger source B 11_B **NOT_USED**, not used - SWTRIG selected
TRGSW	14	w	**Software Trigger Bit** 0_B **Not_Triggered**, SQ not triggered 1_B **Triggered**, SQ is triggered

Sequence Configuration Register 3

SQCFG3
Sequence Configuration Register 3 **($00E4_H$)** **RESET_TYPE_5 Value: $0000\ 0203_H$**

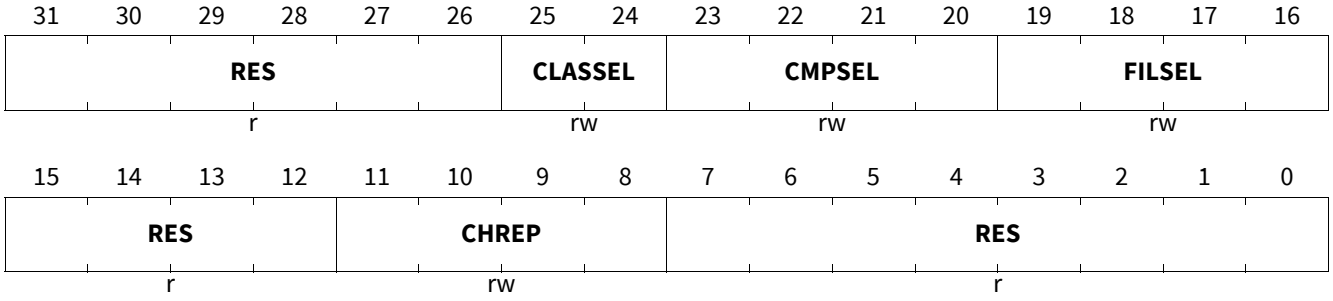
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	TRGSW	RES				TRGSEL		RES		SQREP		RES	SLOTS		
r	w	r				rwpt		r		rwpt		r	rwpt		

Field	Bits	Type	Description
SLOTS	2:0	rwpt	**Number of used Slots in Sequence** 000_B **0**, No Slot used - SQ disabled 001_B **1**, Slots used … 100_B **4**, Slots used 101_B **5**, No Slot used - SQ disabled … 111_B **7**, No Slot used - SQ disabled
RES	3, 7:6, 13:10, 31:15	r	**Reserved**
SQREP	5:4	rwpt	**Sequence repetition** 00_B **1_REP**, The Sequence is executed 1 time 01_B **2_REP**, The Sequence is executed 2 times 10_B **4_REP**, The Sequence is executed 4 times 11_B **8_REP**, The Sequence is executed 8 times
TRGSEL	9:8	rwpt	**Trigger Select** 00_B **SWTRIG**, Software Trigger 01_B **TRIG_A**, External Trigger source A 10_B **TRIG_B**, External Trigger source B 11_B **NOT_USED**, not used - SWTRIG selected
TRGSW	14	w	**Software Trigger Bit** 0_B **Not_Triggered**, SQ not triggered 1_B **Triggered**, SQ is triggered

Channel Configuration Register 0

CHCFG0
Channel Configuration Register 0 **($00E8_H$)** **RESET_TYPE_5 Value: $0188\ 0000_H$**



Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**

Register description ADC2

Field	Bits	Type	Description
CHREP	11:8	rw	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions
FILSEL	19:16	rw	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rw	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rw	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 1

CHCFG1
Channel Configuration Register 1 **($00EC_H$)** **RESET_TYPE_5 Value: $0100\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CLASSEL		CMPSEL				FILSEL			
r						rwpt		rwpt				rwpt			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				CHREP				RES							
r				rwpt				r							

Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rwpt	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rwpt	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rwpt	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rwpt	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 2

CHCFG2
Channel Configuration Register 2 **($00F0_H$)** **RESET_TYPE_5 Value: $0111\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						**CLASSEL**		**CMPSEL**				**FILSEL**			
r						rwpt		rwpt				rwpt			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				**CHREP**				**RES**							
r				rwpt				r							

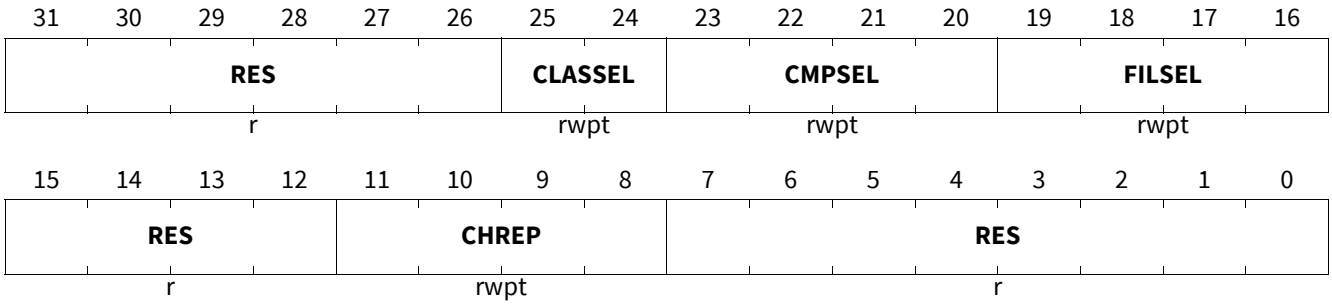
Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rwpt	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rwpt	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rwpt	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rwpt	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 3

CHCFG3
Channel Configuration Register 3 **($00F4_H$)** **RESET_TYPE_5 Value: $0122\ 0000_H$**



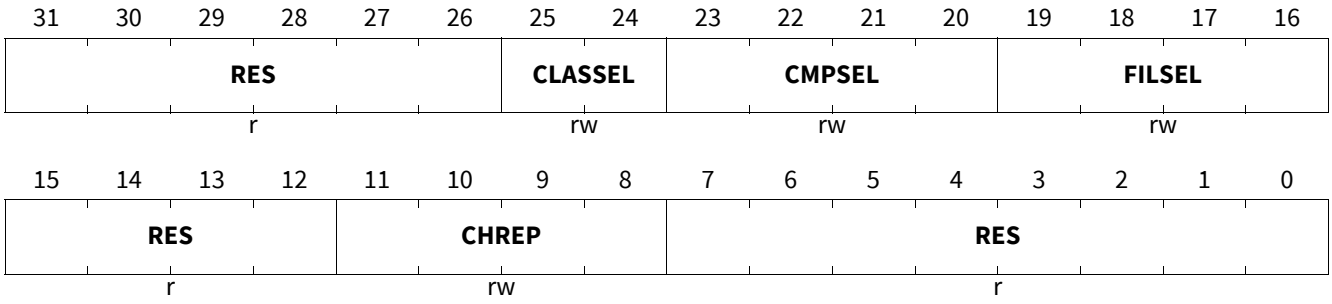
Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rwpt	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rwpt	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rwpt	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rwpt	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 4

CHCFG4
Channel Configuration Register 4 **($00F8_H$)** **RESET_TYPE_5 Value: $0188\ 0000_H$**



Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rw	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rw	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rw	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rw	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 5

CHCFG5
Channel Configuration Register 5 **(00FC$_H$)** **RESET_TYPE_5 Value: 0188 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CLASSEL		CMPSEL				FILSEL			
r						rw		rw				rw			

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

Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rw	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rw	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rw	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rw	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 6

CHCFG6
Channel Configuration Register 6 **(0100_H)** **RESET_TYPE_5 Value: $0188\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CLASSEL		CMPSEL				FILSEL			
r						rw		rw				rw			

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

Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rw	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rw	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rw	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rw	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 7

CHCFG7
Channel Configuration Register 7 **(0104_H)** **RESET_TYPE_5 Value: $0088\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CLASSEL		CMPSEL				FILSEL			
r						rw		rw				rw			

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

Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rw	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rw	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rw	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rw	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 8

CHCFG8
Channel Configuration Register 8 **(0108_H)** **RESET_TYPE_5 Value: $0033\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CLASSEL		CMPSEL				FILSEL			
r						rwpt		rwpt				rwpt			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				CHREP				RES							
r				rwpt				r							

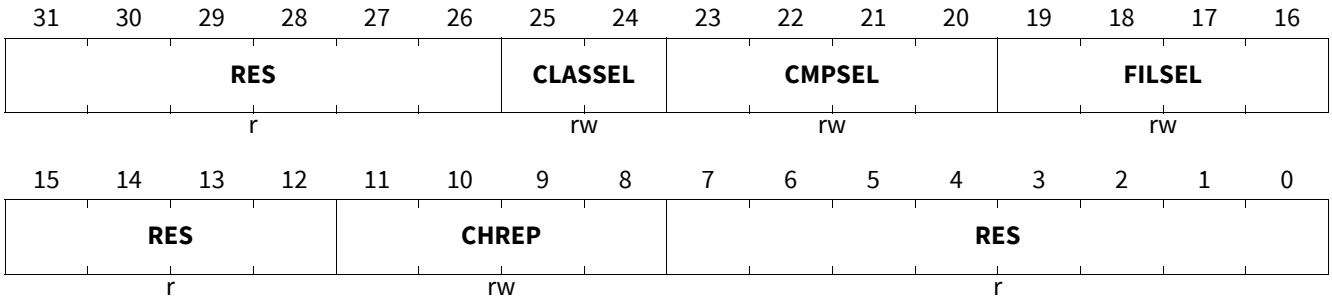
Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rwpt	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rwpt	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rwpt	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rwpt	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 9

CHCFG9
Channel Configuration Register 9 **($010C_H$)** **RESET_TYPE_5 Value: $0088\ 0000_H$**



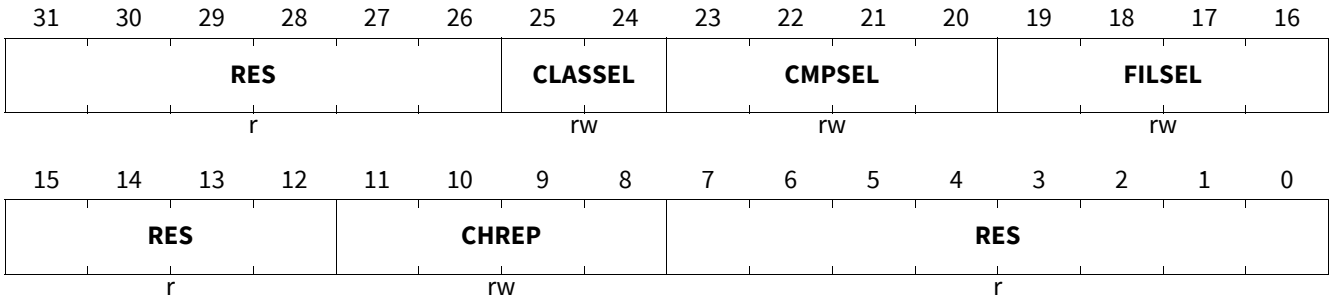
Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rw	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rw	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rw	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rw	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 10

CHCFG10
Channel Configuration Register 10 **(0110_H)** **RESET_TYPE_5 Value: $0088\ 0000_H$**



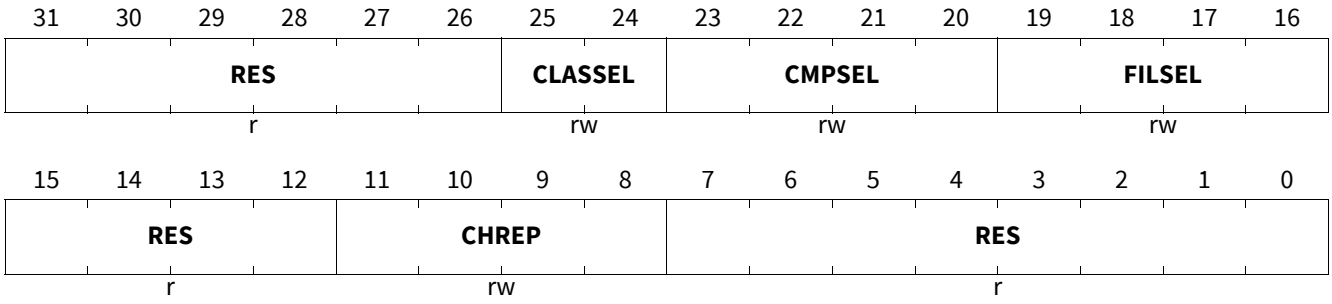
Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rw	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rw	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rw	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rw	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 11

CHCFG11
Channel Configuration Register 11 **(0114_H)** **RESET_TYPE_5 Value: $0088\ 0000_H$**



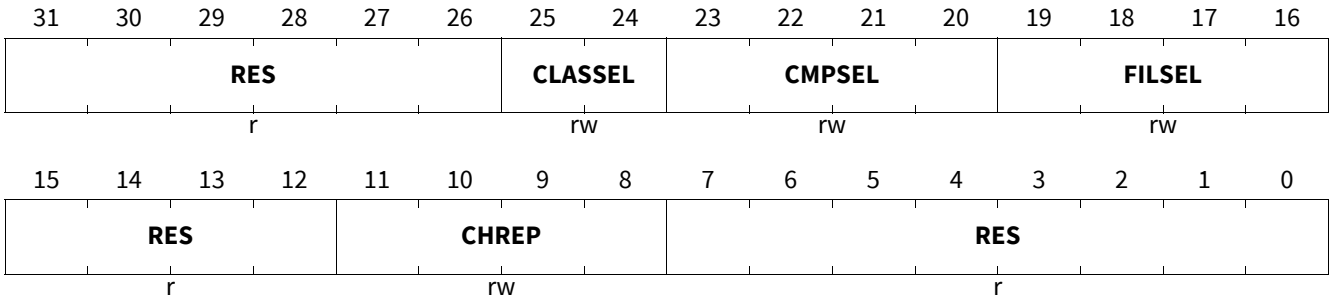
Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rw	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rw	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rw	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rw	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 12

CHCFG12
Channel Configuration Register 12 **(0118_H)** **RESET_TYPE_5 Value: $0088\ 0000_H$**



Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rw	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rw	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rw	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rw	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 13

CHCFG13
Channel Configuration Register 13 **($011C_H$)** **RESET_TYPE_5 Value: $0044\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CLASSEL		CMPSEL				FILSEL			
r						rwpt		rwpt				rwpt			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				CHREP				RES							
r				rwpt				r							

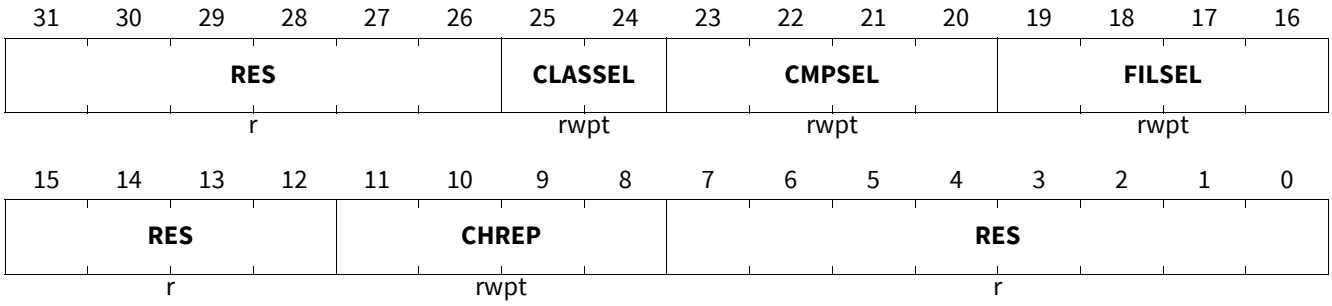
Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rwpt	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rwpt	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rwpt	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rwpt	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Channel Configuration Register 14

CHCFG14
Channel Configuration Register 14 **(0120_H)** **RESET_TYPE_5 Value: $0055\ 0000_H$**



Field	Bits	Type	Description
RES	7:0, 15:12, 31:26	r	**Reserved**
CHREP	11:8	rwpt	**Channel Repetition** 0_H **CHREP1**, 1 repetitions … F_H **CHREP16**, 16 repetitions

Register description ADC2

Field	Bits	Type	Description
FILSEL	19:16	rwpt	**Filter Selection** 0_H **FILTCH0**, Filter Channel 0 … 7_H **FILTCH7**, Filter Channel 7 8_H **not_assigned**, Filter channel not assigned … F_H **not_assigned**, Filter channel not assigned
CMPSEL	23:20	rwpt	**Compare Selection** 0_H **CMPCH0**, Compare Channel 0 … 7_H **CMPCH7**, Compare Channel 7 8_H **not_assigned**, Compare Channel not assigned … F_H **not_assigned**, Compare Channel not assigned
CLASSEL	25:24	rwpt	**Conversion Class Selection** 00_B **CLASS0**, Conversion Class 0 01_B **CLASS1**, Conversion Class 1 10_B **CLASS2**, Conversion Class 2 11_B **CLASS3**, Conversion Class 3

Compare Status Register

CMPSTAT2
Compare Status Register **(0124_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES								**CMP_UP**							
r								r							

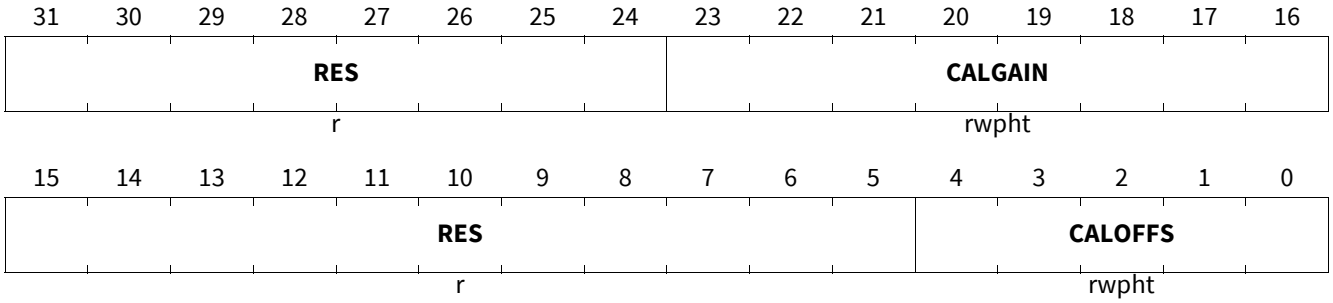
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								**CMP_LO**							
r								r							

Field	Bits	Type	Description
CMP_LO	7:0	r	**Compare low Status** Each bit represents the corresponding Compare low Status 0: INACTIVE - No Lower Compare Event occured 1: ACTIVE - Lower Compare Event occured
RES	15:8, 31:24	r	**Reserved**
CMP_UP	23:16	r	**Compare up Status** Each bit represents the corresponding Compare up Status 0: INACTIVE - No Upper Compare Event occured 1: ACTIVE - Upper Compare Event occured

Register description ADC2

Calibration Setting for Analog Inputs

CALAIx (x=0-14)
Calibration Setting for Analog Inputs **(0128_H+x*4)** **Reset Value: Table 221**



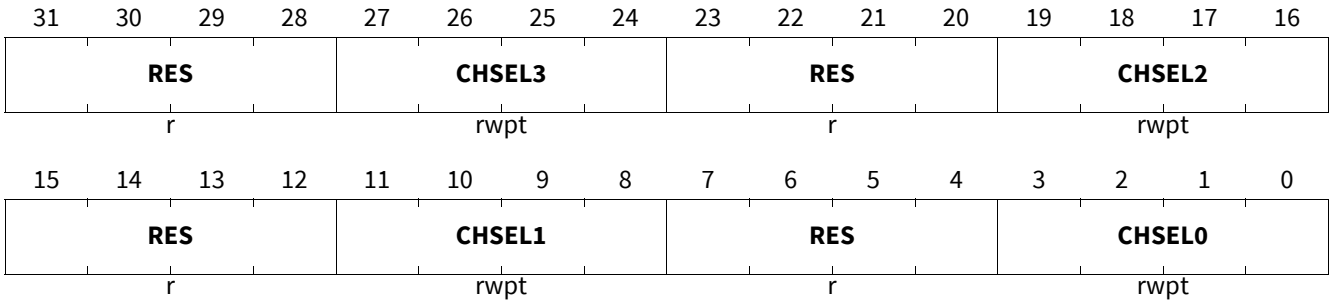
Field	Bits	Type	Description
CALOFFS	4:0	rwpht	**Calibration Offset** This setting can only be changed when the corresponding CALPEN bit is not set
RES	15:5, 31:24	r	**Reserved**
CALGAIN	23:16	rwpht	**Calibration Gain** This setting can only be changed when the corresponding CALPEN bit is not set

Table 221 Reset Values of CALAIx (x=0-14)

Reset Type	Reset Value	Note
RESET_TYPE_4	$0000\ 0000_H$	RESET_TYPE_4
TRIM_1	$0000\ 0000_H$	TRIM

SQ Channel Slot Register 0

SQSLOT0
SQ Channel Slot Register 0 **(0168_H)** **Reset Value: Table 222**



Field	Bits	Type	Description
CHSEL0	3:0	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected
RES	7:4, 15:12, 23:20, 31:28	r	**Reserved**
CHSEL1	11:8	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected
CHSEL2	19:16	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected
CHSEL3	27:24	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected

Table 222 Reset Values of SQSLOT0

Reset Type	Reset Value	Note
RESET_TYPE_4	0302 0100_H	RESET_TYPE_4
TRIM_1	0302 0100_H	RESET

Register description ADC2

SQ Channel Slot Register 1

SQSLOT1
SQ Channel Slot Register 1 **($016C_H$)** **Reset Value: Table 223**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES				CHSEL3				RES				CHSEL2			
r				rwpt				r				rwpt			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				CHSEL1				RES				CHSEL0			
r				rwpt				r				rwpt			

Field	Bits	Type	Description
CHSEL0	3:0	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected
RES	7:4, 15:12, 23:20, 31:28	r	**Reserved**
CHSEL1	11:8	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected
CHSEL2	19:16	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected
CHSEL3	27:24	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected

Table 223 Reset Values of SQSLOT1

Reset Type	Reset Value	Note
RESET_TYPE_4	0605 040D$_H$	RESET_TYPE_4
TRIM_1	0605 040D$_H$	RESET

SQ Channel Slot Register 2

SQSLOT2
SQ Channel Slot Register 2 **(0170$_H$)** **Reset Value: Table 224**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES				CHSEL3				RES				CHSEL2			
r				rwpt				r				rwpt			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				CHSEL1				RES				CHSEL0			
r				rwpt				r				rwpt			

Field	Bits	Type	Description
CHSEL0	3:0	rwpt	**Channel Select** 0$_H$ **CH0**, Channel 0 is selected … E$_H$ **CH14**, Channel 14 is selected F$_H$ **not_used**, Channel 0 selected … **not_used**, Channel 0 selected
RES	7:4, 15:12, 23:20, 31:28	r	**Reserved**
CHSEL1	11:8	rwpt	**Channel Select** 0$_H$ **CH0**, Channel 0 is selected … E$_H$ **CH14**, Channel 14 is selected F$_H$ **not_used**, Channel 0 selected … **not_used**, Channel 0 selected
CHSEL2	19:16	rwpt	**Channel Select** 0$_H$ **CH0**, Channel 0 is selected … E$_H$ **CH14**, Channel 14 is selected F$_H$ **not_used**, Channel 0 selected … **not_used**, Channel 0 selected

Register description ADC2

Field	Bits	Type	Description
CHSEL3	27:24	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected

Table 224 Reset Values of SQSLOT2

Reset Type	Reset Value	Note
RESET_TYPE_4	0A09 0807$_H$	RESET_TYPE_4
TRIM_1	0A09 0807$_H$	RESET

SQ Channel Slot Register 3

SQSLOT3
SQ Channel Slot Register 3 (0174_H) Reset Value: Table 225

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES				**CHSEL3**				**RES**				**CHSEL2**			
r				rwpt				r				rwpt			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				**CHSEL1**				**RES**				**CHSEL0**			
r				rwpt				r				rwpt			

Field	Bits	Type	Description
CHSEL0	3:0	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected
RES	7:4, 15:12, 23:20, 31:28	r	**Reserved**
CHSEL1	11:8	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected

Register description ADC2

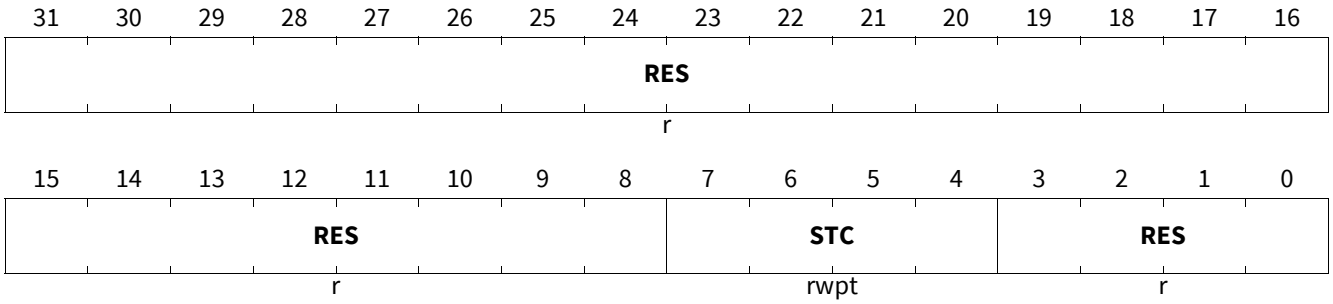
Field	Bits	Type	Description
CHSEL2	19:16	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected
CHSEL3	27:24	rwpt	**Channel Select** 0_H **CH0**, Channel 0 is selected … E_H **CH14**, Channel 14 is selected F_H **not_used**, Channel 0 selected … **not_used**, Channel 0 selected

Table 225 Reset Values of SQSLOT3

Reset Type	Reset Value	Note
RESET_TYPE_4	000E 0C0B$_H$	RESET_TYPE_4
TRIM_1	000E 0C0B$_H$	RESET

Conversion Configuration Register 0

CONVCFG0
Conversion Configuration Register 0 (0178_H) Reset Value: Table 226



Field	Bits	Type	Description
RES	3:0, 31:8	r	**Reserved**

Register description ADC2

Field	Bits	Type	Description
STC	7:4	rwpt	**Sample Time config** 0_H **not_used**, 4 ADC_CLK counts 1_H **ADCCLK4**, 4 ADC_CLK counts 2_H **ADCCLK8**, 8 ADC_CLK counts 3_H **ADCCLK12**, 12 ADC_CLK counts 4_H **ADCCLK16**, 16 ADC_CLK counts 5_H **ADCCLK20**, 20 ADC_CLK counts 6_H **ADCCLK24**, 24 ADC_CLK counts 7_H **ADCCLK28**, 28 ADC_CLK counts 8_H **ADCCLK32**, 32 ADC_CLK counts 9_H **ADCCLK36**, 36 ADC_CLK counts A_H **ADCCLK40**, 40 ADC_CLK counts B_H **ADCCLK44**, 44 ADC_CLK counts C_H **ADCCLK48**, 48 ADC_CLK counts D_H **ADCCLK52**, 52 ADC_CLK counts E_H **ADCCLK56**, 56 ADC_CLK counts F_H **ADCCLK60**, 60 ADC_CLK counts

Table 226 Reset Values of CONVCFG0

Reset Type	Reset Value	Note
RESET_TYPE_4	$0000\ 0030_H$	RESET_TYPE_4
TRIM_1	$0000\ 0030_H$	RESET

Conversion Configuration Register 1

CONVCFG1
Conversion Configuration Register 1 **($017C_H$)** **Reset Value: Table 227**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								STC				RES			
r								rwpt				r			

Field	Bits	Type	Description
RES	3:0, 31:8	r	**Reserved**

Register description ADC2

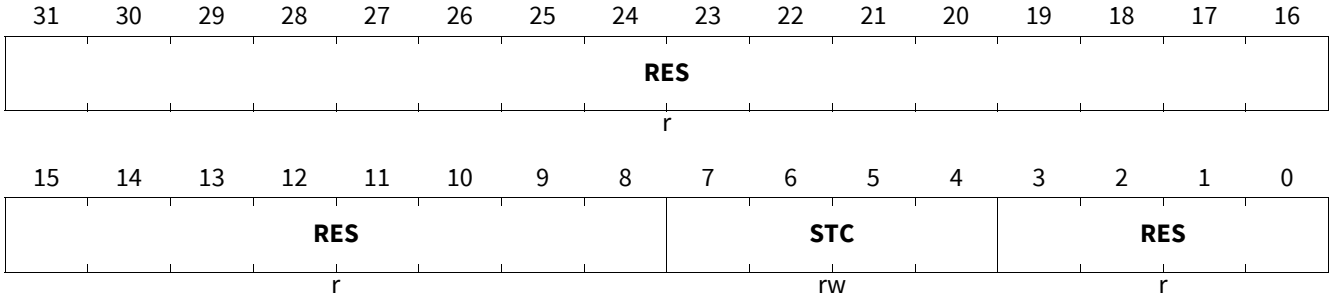
Field	Bits	Type	Description
STC	7:4	rwpt	**Sample Time config** 0_H **not_used**, 4 ADC_CLK counts 1_H **ADCCLK4**, 4 ADC_CLK counts 2_H **ADCCLK8**, 8 ADC_CLK counts 3_H **ADCCLK12**, 12 ADC_CLK counts 4_H **ADCCLK16**, 16 ADC_CLK counts 5_H **ADCCLK20**, 20 ADC_CLK counts 6_H **ADCCLK24**, 24 ADC_CLK counts 7_H **ADCCLK28**, 28 ADC_CLK counts 8_H **ADCCLK32**, 32 ADC_CLK counts 9_H **ADCCLK36**, 36 ADC_CLK counts A_H **ADCCLK40**, 40 ADC_CLK counts B_H **ADCCLK44**, 44 ADC_CLK counts C_H **ADCCLK48**, 48 ADC_CLK counts D_H **ADCCLK52**, 52 ADC_CLK counts E_H **ADCCLK56**, 56 ADC_CLK counts F_H **ADCCLK60**, 60 ADC_CLK counts

Table 227 Reset Values of CONVCFG1

Reset Type	Reset Value	Note
RESET_TYPE_4	0000 0050_H	RESET_TYPE_4
TRIM_1	0000 $00B0_H$	RESET

Conversion Configuration Register 2

CONVCFG2
Conversion Configuration Register 2 (0180_H) RESET_TYPE_4 Value: 0000 0050_H



Field	Bits	Type	Description
RES	3:0, 31:8	r	**Reserved**

Register description ADC2

Field	Bits	Type	Description
STC	7:4	rw	**Sample Time config** 0_H **not_used**, 4 ADC_CLK counts 1_H **ADCCLK4**, 4 ADC_CLK counts 2_H **ADCCLK8**, 8 ADC_CLK counts 3_H **ADCCLK12**, 12 ADC_CLK counts 4_H **ADCCLK16**, 16 ADC_CLK counts 5_H **ADCCLK20**, 20 ADC_CLK counts 6_H **ADCCLK24**, 24 ADC_CLK counts 7_H **ADCCLK28**, 28 ADC_CLK counts 8_H **ADCCLK32**, 32 ADC_CLK counts 9_H **ADCCLK36**, 36 ADC_CLK counts A_H **ADCCLK40**, 40 ADC_CLK counts B_H **ADCCLK44**, 44 ADC_CLK counts C_H **ADCCLK48**, 48 ADC_CLK counts D_H **ADCCLK52**, 52 ADC_CLK counts E_H **ADCCLK56**, 56 ADC_CLK counts F_H **ADCCLK60**, 60 ADC_CLK counts

Conversion Configuration Register 3

CONVCFG3
Conversion Configuration Register 3 **(0184_H)** **RESET_TYPE_4 Value: 0000 0050_H**

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

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

Field	Bits	Type	Description
RES	3:0, 31:8	r	**Reserved**

Register description ADC2

Field	Bits	Type	Description
STC	7:4	rw	**Sample Time config** 0_H **not_used**, 4 ADC_CLK counts 1_H **ADCCLK4**, 4 ADC_CLK counts 2_H **ADCCLK8**, 8 ADC_CLK counts 3_H **ADCCLK12**, 12 ADC_CLK counts 4_H **ADCCLK16**, 16 ADC_CLK counts 5_H **ADCCLK20**, 20 ADC_CLK counts 6_H **ADCCLK24**, 24 ADC_CLK counts 7_H **ADCCLK28**, 28 ADC_CLK counts 8_H **ADCCLK32**, 32 ADC_CLK counts 9_H **ADCCLK36**, 36 ADC_CLK counts A_H **ADCCLK40**, 40 ADC_CLK counts B_H **ADCCLK44**, 44 ADC_CLK counts C_H **ADCCLK48**, 48 ADC_CLK counts D_H **ADCCLK52**, 52 ADC_CLK counts E_H **ADCCLK56**, 56 ADC_CLK counts F_H **ADCCLK60**, 60 ADC_CLK counts

Compare Channel 0 Control Register

CMPCFG0
Compare Channel 0 Control Register **(0188_H)** **Reset Value: Table 228**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE		HYST_UP		RST_BLANK_TIME	BLANK_TIME			UPPER							
rwpt		rwpt		w	rwpt			rwpt							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HYST_LO		RES			INP_SEL	LOWER							
r		rwpt		r			rwpt	rwpt							

Field	Bits	Type	Description
LOWER	7:0	rwpt	**Lower Compare Value**
INP_SEL	8	rwpt	**Input selection for the comparator unit** 0_B **AHB**, Selects the ahb result 1_B **FILTER**, Selects the filter result
RES	11:9, 15:14	r	**Reserved - 0** Read as 0; should be written with 0.
HYST_LO	13:12	rwpt	**Hysteresis setting for lower compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16

Register description ADC2

Field	Bits	Type	Description
UPPER	23:16	rwpt	**Upper Compare Value**
BLANK_TIME	26:24	rwpt	**Blank Time configuration** 000_B **OFF**, 001_B **BLANK_5**, 5 conversions blank time 010_B **BLANK_10**, 10 conversions blank time 011_B **BLANK_20**, 20 conversions blank time 100_B **BLANK_30**, 30 conversions blank time 101_B **BLANK_40**, 40 conversions blank time 110_B **BLANK_50**, 50 conversions blank time 111_B **BLANK_60**, 60 conversions blank time
RST_BLANK_TIME	27	w	**Restart Blank time** 0_B **NO_RESTART**, No Restart of blank time 1_B **RESTART**, Restart blank time
HYST_UP	29:28	rwpt	**Hysteresis setting for upper compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
MODE	31:30	rwpt	**Compare Mode** 00_B **RANGE**, Upper and Lower voltage/limit measurement 01_B **UV**, undervoltage/-limit measurement 10_B **OV**, overvoltage/-limit measurement 11_B **RES**, Reserved (RANGE)

Table 228 Reset Values of CMPCFG0

Reset Type	Reset Value	Note
RESET_TYPE_4	A1EF 2195$_H$	RESET_TYPE_4
TRIM_1	A1EF 2195$_H$	RESET

Compare Channel 1 Control Register

CMPCFG1
Compare Channel 1 Control Register **(018C$_H$)** **Reset Value: Table 229**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE		HYST_UP		RST_BLANK_TIME	BLANK_TIME			UPPER							
rwpt		rwpt		w	rwpt			rwpt							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HYST_LO		RES			INP_SEL	LOWER							
r		rwpt		r			rwpt	rwpt							

Field	Bits	Type	Description
LOWER	7:0	rwpt	**Lower Compare Value**
INP_SEL	8	rwpt	**Input selection for the comparator unit** 0_B **AHB**, Selects the ahb result 1_B **FILTER**, Selects the filter result
RES	11:9, 15:14	r	**Reserved - 0** Read as 0; should be written with 0.
HYST_LO	13:12	rwpt	**Hysteresis setting for lower compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
UPPER	23:16	rwpt	**Upper Compare Value**
BLANK_TIME	26:24	rwpt	**Blank Time configuration** 000_B **OFF**, 001_B **BLANK_5**, 5 conversions blank time 010_B **BLANK_10**, 10 conversions blank time 011_B **BLANK_20**, 20 conversions blank time 100_B **BLANK_30**, 30 conversions blank time 101_B **BLANK_40**, 40 conversions blank time 110_B **BLANK_50**, 50 conversions blank time 111_B **BLANK_60**, 60 conversions blank time
RST_BLANK_TIME	27	w	**Restart Blank time** 0_B **NO_RESTART**, No Restart of blank time 1_B **RESTART**, Restart blank time
HYST_UP	29:28	rwpt	**Hysteresis setting for upper compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
MODE	31:30	rwpt	**Compare Mode** 00_B **RANGE**, Upper and Lower voltage/limit measurement 01_B **UV**, undervoltage/-limit measurement 10_B **OV**, overvoltage/-limit measurement 11_B **RES**, Reserved (RANGE)

Table 229 Reset Values of CMPCFG1

Reset Type	Reset Value	Note
RESET_TYPE_4	61A6 21A6$_H$	RESET_TYPE_4
TRIM_1	61A6 21A6$_H$	RESET

Register description ADC2

Compare Channel 2 Control Register

CMPCFG2
Compare Channel 2 Control Register **(0190_H)** **Reset Value: Table 230**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE		HYST_UP		RST_BLANK_TIME	BLANK_TIME			UPPER							
rwpt		rwpt		w	rwpt			rwpt							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HYST_LO		RES			INP_SEL	LOWER							
r		rwpt		r			rwpt	rwpt							

Field	Bits	Type	Description
LOWER	7:0	rwpt	**Lower Compare Value**
INP_SEL	8	rwpt	**Input selection for the comparator unit** 0_B **AHB**, Selects the ahb result 1_B **FILTER**, Selects the filter result
RES	11:9, 15:14	r	**Reserved - 0** Read as 0; should be written with 0.
HYST_LO	13:12	rwpt	**Hysteresis setting for lower compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
UPPER	23:16	rwpt	**Upper Compare Value**
BLANK_TIME	26:24	rwpt	**Blank Time configuration** 000_B **OFF**, 001_B **BLANK_5**, 5 conversions blank time 010_B **BLANK_10**, 10 conversions blank time 011_B **BLANK_20**, 20 conversions blank time 100_B **BLANK_30**, 30 conversions blank time 101_B **BLANK_40**, 40 conversions blank time 110_B **BLANK_50**, 50 conversions blank time 111_B **BLANK_60**, 60 conversions blank time
RST_BLANK_TIME	27	w	**Restart Blank time** 0_B **NO_RESTART**, No Restart of blank time 1_B **RESTART**, Restart blank time
HYST_UP	29:28	rwpt	**Hysteresis setting for upper compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16

Register description ADC2

Field	Bits	Type	Description
MODE	31:30	rwpt	**Compare Mode** 00_B **RANGE**, Upper and Lower voltage/limit measurement 01_B **UV**, undervoltage/-limit measurement 10_B **OV**, overvoltage/-limit measurement 11_B **RES**, Reserved (RANGE)

Table 230 Reset Values of CMPCFG2

Reset Type	Reset Value	Note
RESET_TYPE_4	21DA 2146$_H$	RESET_TYPE_4
TRIM_1	21DA 2146$_H$	RESET

Compare Channel 3 Control Register

CMPCFG3
Compare Channel 3 Control Register (0194_H) Reset Value: Table 231

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE		HYST_UP		RST_BLANK_TIME	BLANK_TIME			UPPER							
rwpt		rwpt		w	rwpt			rwpt							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HYST_LO		RES			INP_SEL	LOWER							
r		rwpt		r			rwpt	rwpt							

Field	Bits	Type	Description
LOWER	7:0	rwpt	**Lower Compare Value**
INP_SEL	8	rwpt	**Input selection for the comparator unit** 0_B **AHB**, Selects the ahb result 1_B **FILTER**, Selects the filter result
RES	11:9, 15:14	r	**Reserved - 0** Read as 0; should be written with 0.
HYST_LO	13:12	rwpt	**Hysteresis setting for lower compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
UPPER	23:16	rwpt	**Upper Compare Value**

Register description ADC2

Field	Bits	Type	Description
BLANK_TIME	26:24	rwpt	**Blank Time configuration** 000_B **OFF**, 001_B **BLANK_5**, 5 conversions blank time 010_B **BLANK_10**, 10 conversions blank time 011_B **BLANK_20**, 20 conversions blank time 100_B **BLANK_30**, 30 conversions blank time 101_B **BLANK_40**, 40 conversions blank time 110_B **BLANK_50**, 50 conversions blank time 111_B **BLANK_60**, 60 conversions blank time
RST_BLANK_TIME	27	w	**Restart Blank time** 0_B **NO_RESTART**, No Restart of blank time 1_B **RESTART**, Restart blank time
HYST_UP	29:28	rwpt	**Hysteresis setting for upper compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
MODE	31:30	rwpt	**Compare Mode** 00_B **RANGE**, Upper and Lower voltage/limit measurement 01_B **UV**, undervoltage/-limit measurement 10_B **OV**, overvoltage/-limit measurement 11_B **RES**, Reserved (RANGE)

Table 231 Reset Values of CMPCFG3

Reset Type	Reset Value	Note
RESET_TYPE_4	12FE 11DC$_H$	RESET_TYPE_4
TRIM_1	12FE 11DC$_H$	RESET

Compare Channel 4 Control Register

CMPCFG4

Compare Channel 4 Control Register (0198_H) Reset Value: Table 232

31 30	29 28	27	26 25 24	23 22 21 20 19 18 17 16
MODE	HYST_UP	RST_BLANK_TIME	BLANK_TIME	UPPER
rwpt	rwpt	w	rwpt	rwpt

15 14	13 12	11 10 9	8	7 6 5 4 3 2 1 0
RES	HYST_LO	RES	INP_SEL	LOWER
r	rwpt	r	rwpt	rwpt

Field	Bits	Type	Description
LOWER	7:0	rwpt	**Lower Compare Value**
INP_SEL	8	rwpt	**Input selection for the comparator unit** 0_B **AHB**, Selects the ahb result 1_B **FILTER**, Selects the filter result
RES	11:9, 15:14	r	**Reserved - 0** Read as 0; should be written with 0.
HYST_LO	13:12	rwpt	**Hysteresis setting for lower compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
UPPER	23:16	rwpt	**Upper Compare Value**
BLANK_TIME	26:24	rwpt	**Blank Time configuration** 000_B **OFF**, 001_B **BLANK_5**, 5 conversions blank time 010_B **BLANK_10**, 10 conversions blank time 011_B **BLANK_20**, 20 conversions blank time 100_B **BLANK_30**, 30 conversions blank time 101_B **BLANK_40**, 40 conversions blank time 110_B **BLANK_50**, 50 conversions blank time 111_B **BLANK_60**, 60 conversions blank time
RST_BLANK_TIME	27	w	**Restart Blank time** 0_B **NO_RESTART**, No Restart of blank time 1_B **RESTART**, Restart blank time
HYST_UP	29:28	rwpt	**Hysteresis setting for upper compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
MODE	31:30	rwpt	**Compare Mode** 00_B **RANGE**, Upper and Lower voltage/limit measurement 01_B **UV**, undervoltage/-limit measurement 10_B **OV**, overvoltage/-limit measurement 11_B **RES**, Reserved (RANGE)

Table 232 Reset Values of CMPCFG4

Reset Type	Reset Value	Note
RESET_TYPE_4	9137 112D$_H$	RESET_TYPE_4
TRIM_1	9137 112D$_H$	RESET

Register description ADC2

Compare Channel 5 Control Register

CMPCFG5
Compare Channel 5 Control Register **($019C_H$)** **Reset Value: Table 233**

31 30	29 28	27	26 25 24	23 22 21 20 19 18 17 16
MODE	HYST_UP	RST_BLANK_TIME	BLANK_TIME	UPPER
rwpt	rwpt	w	rwpt	rwpt

15 14	13 12	11 10 9	8	7 6 5 4 3 2 1 0
RES	HYST_LO	RES	INP_SEL	LOWER
r	rwpt	r	rwpt	rwpt

Field	Bits	Type	Description
LOWER	7:0	rwpt	**Lower Compare Value**
INP_SEL	8	rwpt	**Input selection for the comparator unit** 0_B **AHB**, Selects the ahb result 1_B **FILTER**, Selects the filter result
RES	11:9, 15:14	r	**Reserved - 0** Read as 0; should be written with 0.
HYST_LO	13:12	rwpt	**Hysteresis setting for lower compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
UPPER	23:16	rwpt	**Upper Compare Value**
BLANK_TIME	26:24	rwpt	**Blank Time configuration** 000_B **OFF**, 001_B **BLANK_5**, 5 conversions blank time 010_B **BLANK_10**, 10 conversions blank time 011_B **BLANK_20**, 20 conversions blank time 100_B **BLANK_30**, 30 conversions blank time 101_B **BLANK_40**, 40 conversions blank time 110_B **BLANK_50**, 50 conversions blank time 111_B **BLANK_60**, 60 conversions blank time
RST_BLANK_TIME	27	w	**Restart Blank time** 0_B **NO_RESTART**, No Restart of blank time 1_B **RESTART**, Restart blank time
HYST_UP	29:28	rwpt	**Hysteresis setting for upper compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16

Register description ADC2

Field	Bits	Type	Description
MODE	31:30	rwpt	**Compare Mode** 00_B **RANGE**, Upper and Lower voltage/limit measurement 01_B **UV**, undervoltage/-limit measurement 10_B **OV**, overvoltage/-limit measurement 11_B **RES**, Reserved (RANGE)

Table 233 Reset Values of CMPCFG5

Reset Type	Reset Value	Note
RESET_TYPE_4	9137 112D$_H$	RESET_TYPE_4
TRIM_1	9137 112D$_H$	RESET

Compare Channel 6 Control Register

CMPCFG6
Compare Channel 6 Control Register (01A0$_H$) RESET_TYPE_4 Value: 0000 0000$_H$

31 30	29 28	27	26 25 24	23 22 21 20 19 18 17 16
MODE	HYST_UP	RST_BLANK_TIME	BLANK_TIME	UPPER
rw	rw	w	rw	rw

15 14	13 12	11 10 9	8	7 6 5 4 3 2 1 0
RES	HYST_LO	RES	INP_SEL	LOWER
r	rw	r	rw	rw

Field	Bits	Type	Description
LOWER	7:0	rw	**Lower Compare Value**
INP_SEL	8	rw	**Input selection for the comparator unit** 0_B **AHB**, Selects the ahb result 1_B **FILTER**, Selects the filter result
RES	11:9, 15:14	r	**Reserved - 0** Read as 0; should be written with 0.
HYST_LO	13:12	rw	**Hysteresis setting for lower compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
UPPER	23:16	rw	**Upper Compare Value**

Register description ADC2

Field	Bits	Type	Description
BLANK_TIME	26:24	rw	**Blank Time configuration** 000_B **OFF**, 001_B **BLANK_5**, 5 conversions blank time 010_B **BLANK_10**, 10 conversions blank time 011_B **BLANK_20**, 20 conversions blank time 100_B **BLANK_30**, 30 conversions blank time 101_B **BLANK_40**, 40 conversions blank time 110_B **BLANK_50**, 50 conversions blank time 111_B **BLANK_60**, 60 conversions blank time
RST_BLANK_TIME	27	w	**Restart Blank time** 0_B **NO_RESTART**, No Restart of blank time 1_B **RESTART**, Restart blank time
HYST_UP	29:28	rw	**Hysteresis setting for upper compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
MODE	31:30	rw	**Compare Mode** 00_B **RANGE**, Upper and Lower voltage/limit measurement 01_B **UV**, undervoltage/-limit measurement 10_B **OV**, overvoltage/-limit measurement 11_B **RES**, Reserved (RANGE)

Compare Channel 7 Control Register

CMPCFG7
Compare Channel 7 Control Register **($01A4_H$)** **RESET_TYPE_4 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE		HYST_UP		RST_BLANK_TIME	BLANK_TIME			UPPER							
rw		rw		w	rw			rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HYST_LO		RES			INP_SEL	LOWER							
r		rw		r			rw	rw							

Field	Bits	Type	Description
LOWER	7:0	rw	**Lower Compare Value**
INP_SEL	8	rw	**Input selection for the comparator unit** 0_B **AHB**, Selects the ahb result 1_B **FILTER**, Selects the filter result
RES	11:9, 15:14	r	**Reserved - 0** Read as 0; should be written with 0.

Register description ADC2

Field	Bits	Type	Description
HYST_LO	13:12	rw	**Hysteresis setting for lower compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
UPPER	23:16	rw	**Upper Compare Value**
BLANK_TIME	26:24	rw	**Blank Time configuration** 000_B **OFF**, 001_B **BLANK_5**, 5 conversions blank time 010_B **BLANK_10**, 10 conversions blank time 011_B **BLANK_20**, 20 conversions blank time 100_B **BLANK_30**, 30 conversions blank time 101_B **BLANK_40**, 40 conversions blank time 110_B **BLANK_50**, 50 conversions blank time 111_B **BLANK_60**, 60 conversions blank time
RST_BLANK_TIME	27	w	**Restart Blank time** 0_B **NO_RESTART**, No Restart of blank time 1_B **RESTART**, Restart blank time
HYST_UP	29:28	rw	**Hysteresis setting for upper compare threshold** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
MODE	31:30	rw	**Compare Mode** 00_B **RANGE**, Upper and Lower voltage/limit measurement 01_B **UV**, undervoltage/-limit measurement 10_B **OV**, overvoltage/-limit measurement 11_B **RES**, Reserved (RANGE)

Filter Configuration

FILTCFG
Filter Configuration **($01A8_H$)** **Reset Value:** Table 234

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16
RES	**COEF_A7**	**RES**	**COEF_A6**	**RES**	**COEF_A5**	**RES**	**COEF_A4**
r	rw	r	rw	r	rwpt	r	rwpt

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
RES	**COEF_A3**	**RES**	**COEF_A2**	**RES**	**COEF_A1**	**RES**	**COEF_A0**
r	rwpt	r	rwpt	r	rwpt	r	rwpt

Field	Bits	Type	Description
COEF_A0	1:0	rwpt	**Filter Coefficient 0** 00_B **1_2**, Filter coefficient 1/2 01_B **1_4**, Filter coefficient 1/4 10_B **1_8**, Filter coefficient 1/8 11_B **1_16**, Filter coefficient 1/16
RES	3:2, 7:6, 11:10, 15:14, 19:18, 23:22, 27:26, 31:30	r	**Reserved**
COEF_A1	5:4	rwpt	**Filter Coefficient 0** 00_B **1_2**, Filter coefficient 1/2 01_B **1_4**, Filter coefficient 1/4 10_B **1_8**, Filter coefficient 1/8 11_B **1_16**, Filter coefficient 1/16
COEF_A2	9:8	rwpt	**Filter Coefficient 0** 00_B **1_2**, Filter coefficient 1/2 01_B **1_4**, Filter coefficient 1/4 10_B **1_8**, Filter coefficient 1/8 11_B **1_16**, Filter coefficient 1/16
COEF_A3	13:12	rwpt	**Filter Coefficient 0** 00_B **1_2**, Filter coefficient 1/2 01_B **1_4**, Filter coefficient 1/4 10_B **1_8**, Filter coefficient 1/8 11_B **1_16**, Filter coefficient 1/16
COEF_A4	17:16	rwpt	**Filter Coefficient 0** 00_B **1_2**, Filter coefficient 1/2 01_B **1_4**, Filter coefficient 1/4 10_B **1_8**, Filter coefficient 1/8 11_B **1_16**, Filter coefficient 1/16
COEF_A5	21:20	rwpt	**Filter Coefficient 0** 00_B **1_2**, Filter coefficient 1/2 01_B **1_4**, Filter coefficient 1/4 10_B **1_8**, Filter coefficient 1/8 11_B **1_16**, Filter coefficient 1/16
COEF_A6	25:24	rw	**Filter Coefficient 0** 00_B **1_2**, Filter coefficient 1/2 01_B **1_4**, Filter coefficient 1/4 10_B **1_8**, Filter coefficient 1/8 11_B **1_16**, Filter coefficient 1/16

Field	Bits	Type	Description
COEF_A7	29:28	rw	**Filter Coefficient 0** 00_B **1_2**, Filter coefficient 1/2 01_B **1_4**, Filter coefficient 1/4 10_B **1_8**, Filter coefficient 1/8 11_B **1_16**, Filter coefficient 1/16

Table 234 Reset Values of FILTCFG

Reset Type	Reset Value	Note
RESET_TYPE_4	$0000\ 0000_H$	RESET_TYPE_4
TRIM_1	$0000\ 0000_H$	RESET

18 Current Sense Amplifier (CSA)

18.1 Features overview

The Current Sense Amplifier (CSA) in **Figure 223** can be used to measure near-ground differential voltages via ADC1. Its gain and output offset voltage are digitally programmable through internal control registers.

Linear calibration has to be applied to achieve high gain accuracy, e.g. end-of-line calibration including the shunt resistor.

Figure 223 shows how the current sense amplifier can be used as a low-side current sense amplifier where the motor current is converted to a voltage by means of a shunt resistor R_{SH}. A differential amplifier input is used in order to eliminate measurement errors due to voltage drop across the stray resistance R_{Stray} and differences between the external and internal ground. If the voltage at one or both inputs (CSAP/CSAN) is out of the operating range it has to be taken into account that the input circuit is overloaded and needs a certain specified recovery time.

In general, the external low pass filter should suppress electromagnetic interferences (EMI).

The CSA provides following features:

- The CSA amplifies a near-ground differential input voltage to a single-ended output voltage
- The CSA has programmable gain settings of G = 10, 20, 40, 60
- The CSA output voltage has a programmable offset
- The CSA output voltage can be measured by the ADC1
- The CSA output voltage offset can be measured by the ADC1 independently from the CSA input conditions
- The CSA output voltage offset is derived from the ADC1 reference voltage VAREF

18.2 Block diagram

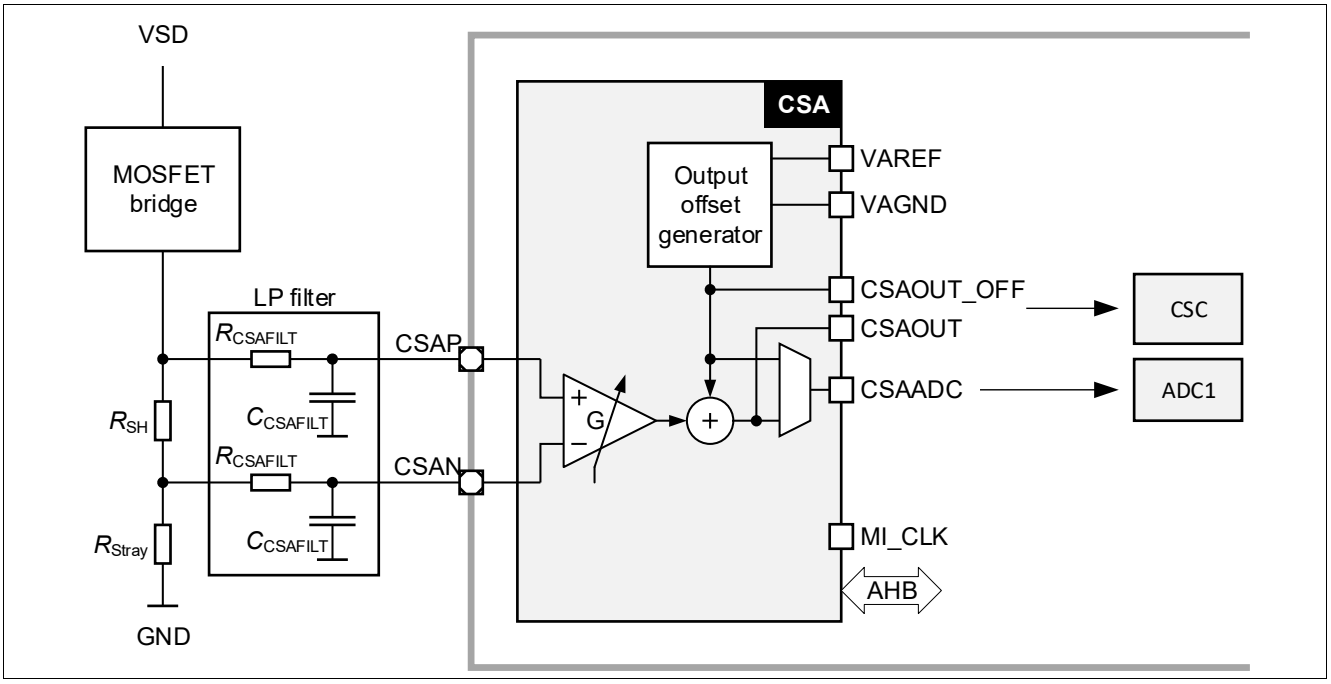


Figure 223 Block diagram CSA

18.3 Toplevel signals

Table 235 CSA clock domain

Signal	Direction	Description	From
MI_CLK	input	Peripheral clock	SCU

Table 236 CSA reference voltage

Signal	Direction	Description	From/To
VAREF	input	Analog reference voltage	VAREF pin
VAGND	input	Analog reference ground	VAGND pin

Table 237 CSA analog inputs

Signal	Direction	Description	From/To
CSAP	input	CSA positive input voltage	CSAP pin
CSAN	input	CSA negative input voltage	CSAN pin

Table 238 CSA outputs

Signal	Direction	Description	To
CSAOUT	output	CSA output voltage	CSC.INP
CSAOUT_OFF	output	CSA output voltage offset	CSC.INOFF
CSAADC	output	Multiplexed CSAOUT/CSAOUT_OFF	ADC1.AN[18]

18.4 Interrupts

Events

The CSA provides the following event:

- Gain or offset selection beyond the stored limits (SEL_ERR)

Interrupts

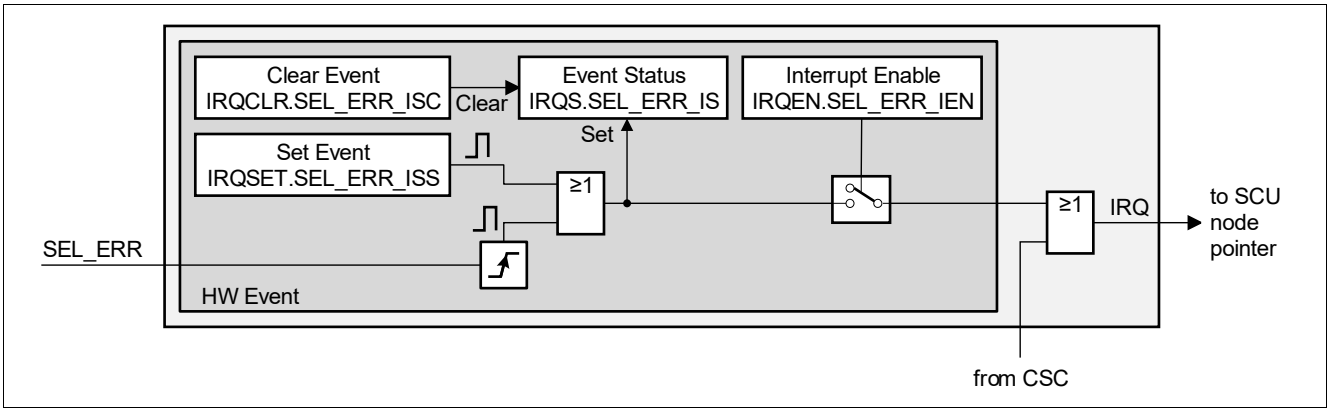


Figure 224 Interrupt handling

Notes

1. *The SEL_ERR event is also triggered by a CSC overcurrent threshold selection error.*
2. *The SEL_ERR event is only generated if CTRL1.CSAC_EN='1'.*

18.5 Operation mode behavior

Table 239 Operation mode behavior CSA

Reset	• RESET_TYPE_5 resets the CSA configuration settings
Power-up / power-down	• After power-up the CSA is disabled by default • At power-down the CSA is reset by RESET_TYPE_5
Active mode	• The CSA is disabled by default • The CSA can be enabled and configured by setting the AHB registers (start-up time t_{CSAPU})
Stop mode	• The CSA is disabled automatically when entering stop mode • The configuration of the CSA is lost • The CSA stays disabled after wake-up
Sleep mode	• The CSA is disabled automatically when entering sleep mode • The configuration of the CSA is lost • The CSA stays disabled after wake-up
Fail-sleep mode	• The CSA behaves like in sleep mode

18.6 Global control

The CSA is enabled together with the CSC by setting the CTRL1.CSAC_EN bit.

The configuration is done by the following register bit fields (for details see **Chapter 18.7**):

- CTRL2.GAIN_SEL for the gain
- CTRL2.OFFS_SEL for the output offset voltage

The possible range of settings for CTRL2.GAIN_SEL and CTRL2.OFFS_SEL can be limited by the following register bit fields (for details see **Chapter 18.8**):

- CTRL1.GAIN_MIN for the gain
- CTRL1.OFFS_MIN for the output offset voltage

Notes

1. *CTRL1.GAIN_MIN and CTRL1.OFFS_MIN must be written before enabling the CSA/CSC for the first time and are frozen afterwards.*
2. *After enabling the CSA a built-in self-test of the CSA and the CSC is started (see **Chapter 19**).*

In order to test the signal chain from the CSA inputs to the ADC results and CSC output an artificial offset voltage can be applied to the CSA inputs by setting CTRL2.ADD_INP_OFFS to '1'. This allows plausibility checks of the CSA settings without the need for current flowing through the external shunt resistor.

18.7 Transfer characteristics

The transfer characteristics of the CSA can be configured by the two register bit fields CTRL2.GAIN_SEL and CTRL2.OFFS_SEL. The gain selection by CTRL2.GAIN_SEL defines the size of the possible differential input voltage range (see **Figure 225**) while the output offset voltage selection by CTRL2.OFFS_SEL moves the possible differential input voltage range relative to 0 V (see **Figure 226**).

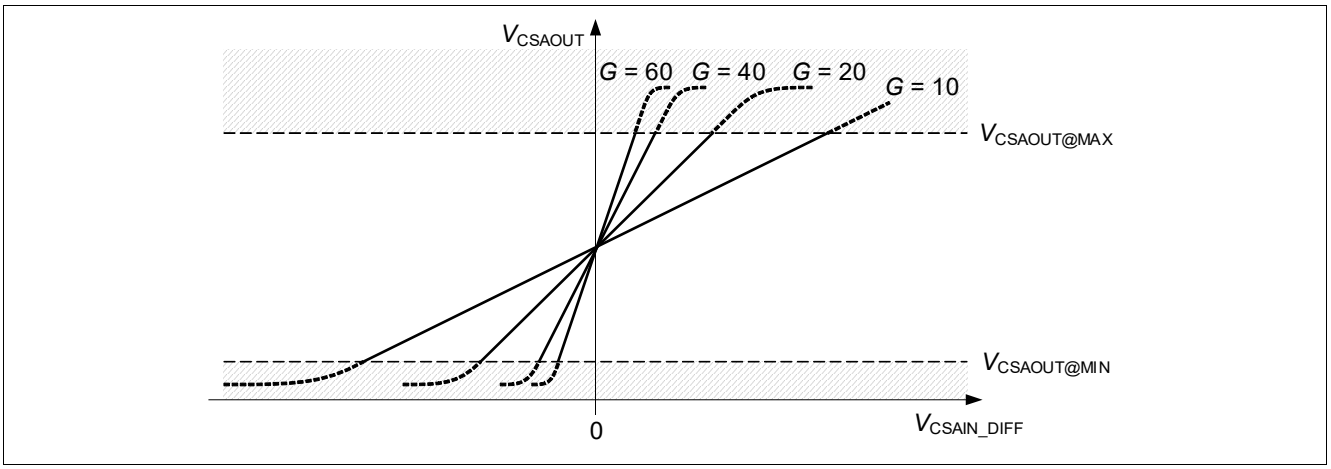


Figure 225 Transfer curves for different gain settings

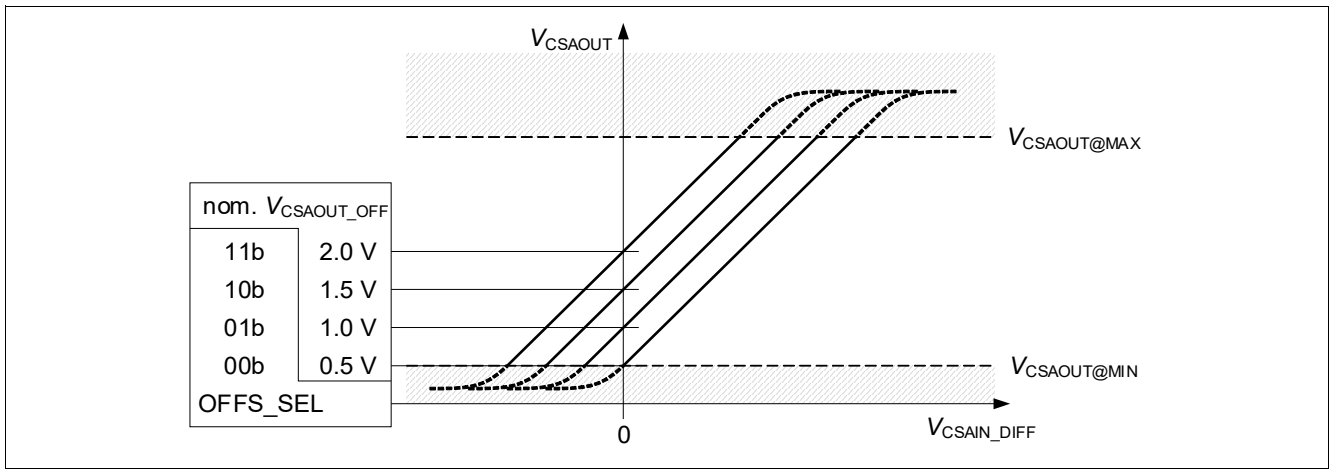


Figure 226 Transfer curves for different output offset voltage settings

Note: The output offset voltage is generated by scaling the analog reference voltage (VAREF vs. VAGND) appropriately.

18.8 Minimum gain and offset settings

The CSA is part of the functional safety relevant safe switch-off path. Therefore, a special protection mechanism is implemented to ensure a minimum sensitivity to detect shunt overcurrents. This is done by limiting the range of the gain resp. offset settings in the register bit fields CTRL1.GAIN_MIN resp. CTRL1.OFFS_MIN. Selecting values smaller than these limits in the bit fields CTRL2.GAIN_SEL resp. CTRL2.OFFS_SEL triggers a “selection error” (see IRQS.SEL_ERR_IS) and effectively sets the gain resp. offset to the values in CTRL1.GAIN_MIN resp. CTRL1.OFFS_MIN.

Additionally, a double-bit storage scheme is implemented to protect against single event upsets: Each bit field, CTRL1.GAIN_MIN and CTRL1.OFFS_MIN, is 4 bits wide but effectively decodes to a two-bit value corresponding to the minimum settings allowed for CTRL2.GAIN_SEL and CTRL2.OFFS_SEL. The 4-bit to 2-bit decoding is done by ORing the 2 MSBs and the 2 LSBs as shown in the following figure:

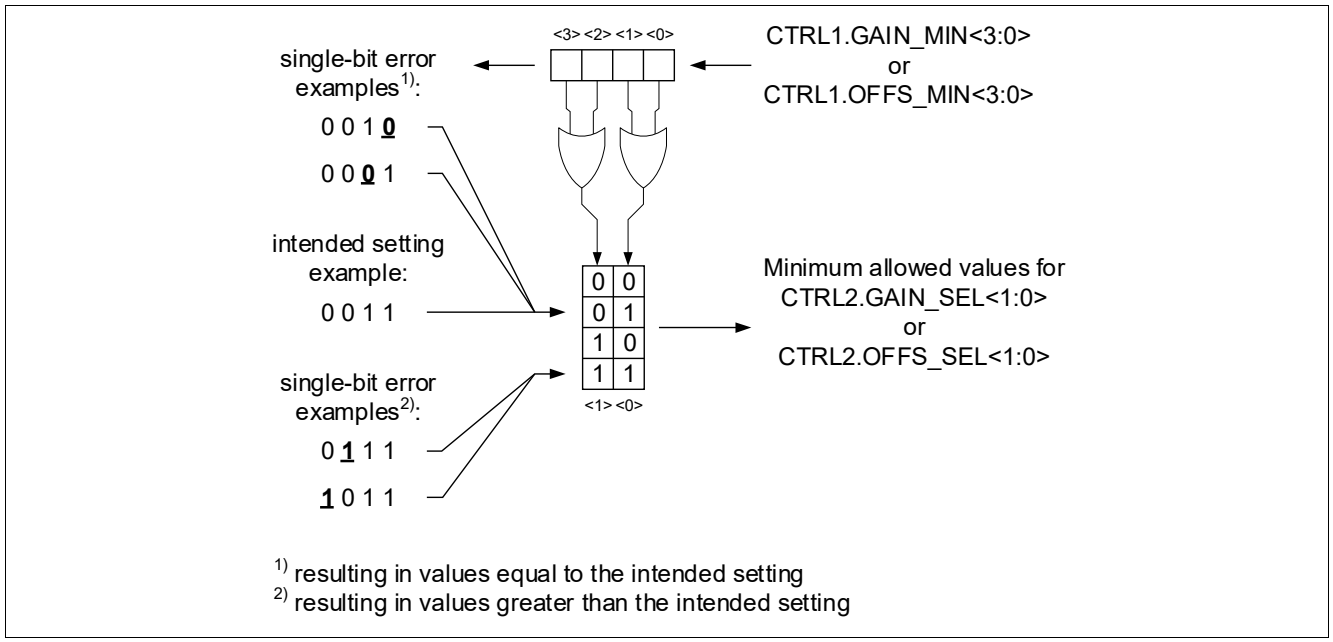


Figure 227 Double bit storage decoding scheme

The examples at the left in **Figure 227** show that if the recommended settings from **Table 240** are used, where both bits of a pair are either cleared or set, this double-bit storage scheme ensures that a single-bit error results in an effective value which is equal to or greater than the originally intended setting (and therefore potentially further limits the possible range of settings for CTRL2.GAIN_SEL and CTRL2.OFFS_SEL):

Table 240 Recommended settings and effective values

Recommended settings	Regular effective 2-bit value	Possible effective 2-bit values in the case of a single-bit error
0000_B	00_B	01_B (greater) or 10_B (greater)
0011_B	01_B	01_B (equal) or 11_B (greater)
1100_B	10_B	10_B (equal) or 11_B (greater)
1111_B	11_B	11_B (equal)

Note: CTRL1.GAIN_MIN and CTRL1.OFFS_MIN cannot be changed anymore once the CSA is enabled.

18.9 Output usage

The CSA output voltage is connected to the CSC. The voltage provided to the ADC1 can be selected by CTRL2.VOUT_SEL:

- '0': the CSA output voltage (CSAOUT) is selected for the ADC1
- '1': the CSA output offset voltage (CSAOUT_OFF) is selected for the ADC1

18.10 Programmer's guide

18.10.1 Initialization sequence

The initialization of the CSA is done together with the CSC. See **Chapter 19.11.1**.

18.10.2 ADC code calculation

The differential input voltage $V_{CSAIN\_DIFF} = V_{CSAP} - V_{CSAN}$ of the embedded CSA is converted to an ADC code by the following equation:

(18.1)

$$\mathrm{ADC1out} = \mathrm{floor}\left(\frac{V_{CSAOUT\_OFF} + (V_{CSAP} - V_{CSAN}) * G}{V_{LSB}} + 1\right)$$

wherein the parameters V_{CSAP} and V_{CSAN} are the voltages at the inputs of the amplifier, G is the configured gain (see CTRL2.GAIN_SEL), and V_{CSAOUT_OFF} is the configured output offset voltage (see CTRL2.OFFS_SEL). V_{LSB} is defined as follows:

(18.2)

$$V_{LSB} = \frac{\mathrm{VAREF}}{4095}$$

*Note: **Equation (18.1)** and **Equation (18.2)** show the nominal case without error calculation considerations.*

18.10.3 Recommended offset calibration

In order to perform a correct offset calibration the complete path from shunt resistor through the CSA to the ADC1 should be measured. For this purpose it should be ensured that there is no current flow (0 A) through the shunt. It is recommended to follow these steps when performing an offset calibration:

1. Connect the CSA output to the ADC1 Channel 1 input, CTRL2.VOUT_SEL='0'
2. Set the desired CSA gain and output offset, CTRL2.GAIN_SEL=x, CTRL2.OFFS_SEL=y
3. Ensure that there is no current flowing through the shunt, meaning that the differential voltage at the CSA inputs is 0 V
4. Perform an ADC1 measurement to get the digital value of the CSA output offset including the input offset
5. Repeat the measurement for other CSA gain and offset settings if needed

Figure 228 displays the effective signal path for the recommended offset calibration.

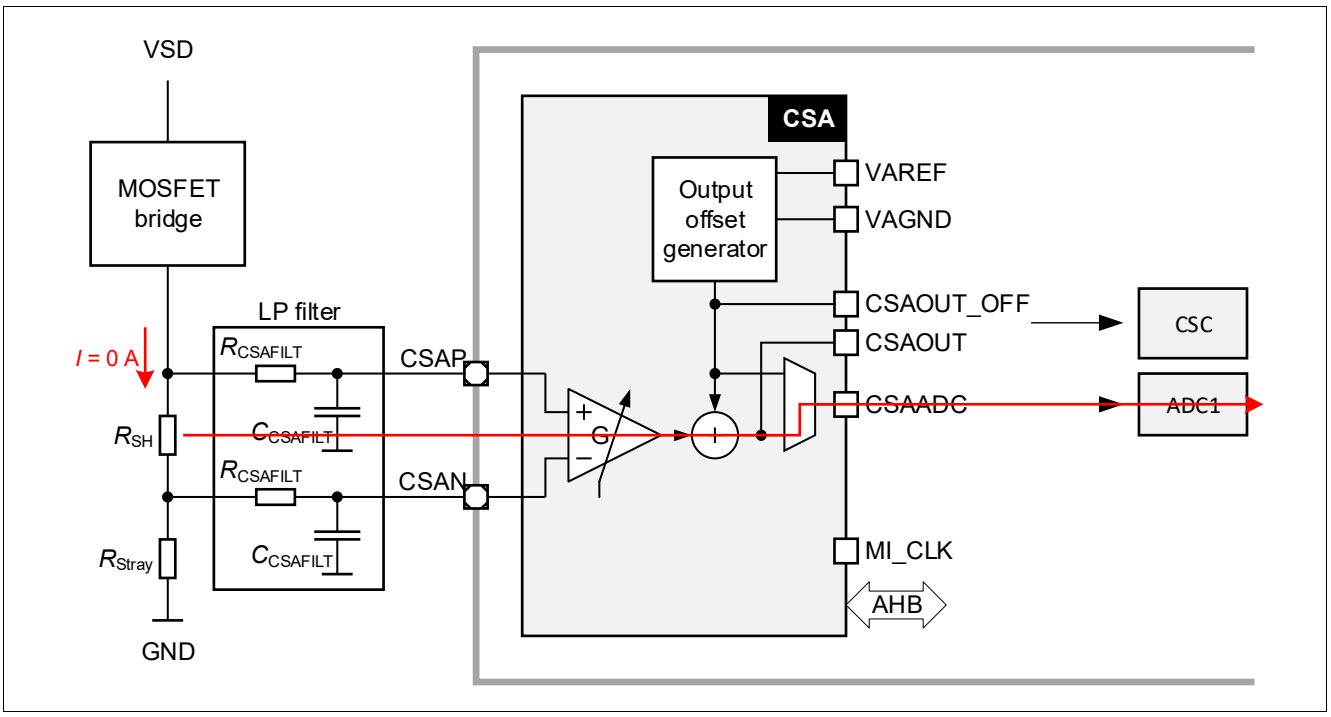


Figure 228 CSA offset calibration path

18.11 Register description CSA (and CSC)

The CSA and the CSC share the same register set. The description of the CSA/CSC registers see **Chapter 18.11**.

18.11.1 CSA/CSC Address Maps

Table 241 Register Address Space - CSACSC

Module	Base Address	End Address	Note
CSACSC	40010000_H	$40013FFF_H$	

Table 242 Register Overview - CSACSC (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CTRL1	Current sense amplifier and comparator control	0000_H	**813**
CTRL2	Current sense amplifier and comparator control	0004_H	**814**
IRQS	Current sense amplifier/comparator interrupt status	0008_H	**815**
IRQCLR	Current sense amplifier/comparator interrupt status clear	$000C_H$	**816**
IRQSET	Current sense amplifier/comparator interrupt status set	0010_H	**817**
IRQEN	Current sense amplifier/comparator interrupt enable	0014_H	**817**

19 Current Sense Comparator (CSC)

19.1 Features overview

The Current Sense Comparator (CSC) is used for fast detection and reaction on overcurrent on the shunt measurement by the CSA.

The CSC provides following features:

- The CSC compares the CSA output voltage against a programmable threshold voltage to detect positive overcurrents through the shunt
- The CSC threshold voltage is derived from the ADC1 reference voltage
- The CSC provides a programmable filter time
- The CSC event can trigger a CCU7.CTRAP event
- The CSC event can trigger an interrupt request
- The CSC event can switch off the bridge driver output safely (safe switch off in case of overcurrent)
- The CSC output status is indicated by a volatile level indication flag showing the actual status
- The CSC output status is indicated by a sticky status flag which must be cleared by software
- The CSC output status is indicated by an interrupt request flag

19.2 Block diagram

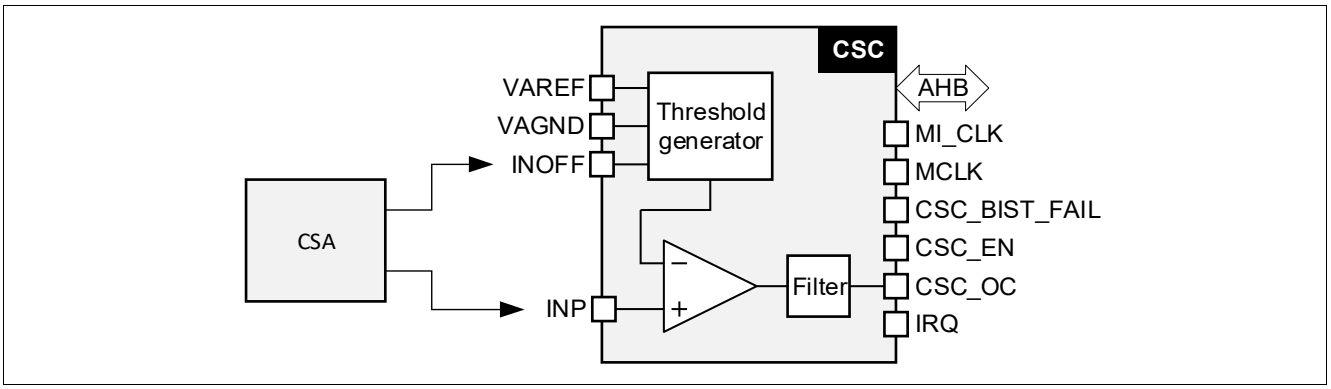


Figure 229 Block diagram CSC

19.3 Toplevel signals

Table 243 CSC clock domains

Signal	Direction	Description	From
MI_CLK	input	Peripheral clock	SCU
MCLK	input	BIST and filter clock	PMU

Table 244 CSC reference voltages

Signal	Direction	Description	From
VAREF	input	Analog reference voltage	VAREF pin
VAGND	input	Analog reference ground	VAGND pin
INOFF	input	CSA output voltage offset	CSA

Table 245 CSC analog input

Signal	Direction	Description	From
INP	input	CSA output voltage	CSA

Table 246 CSC digital outputs

Signal	Direction	Description	To
CSC_OC	output	Filtered CSC output signal	PMU, CCU7
CSC_EN	output	Comparator enable signal after self-test	PMU
CSC_BIST_FAIL	output	Result of built-in self-test	PMU
IRQ	output	Interrupt service request	SCU

19.4 Interrupts

Events

The CSC provides the following events:

- The CSA output voltage exceeds the CSC overcurrent threshold (CSC_OC)
- The CSC overcurrent threshold is selected beyond the stored limit (SEL_ERR)

Interrupts

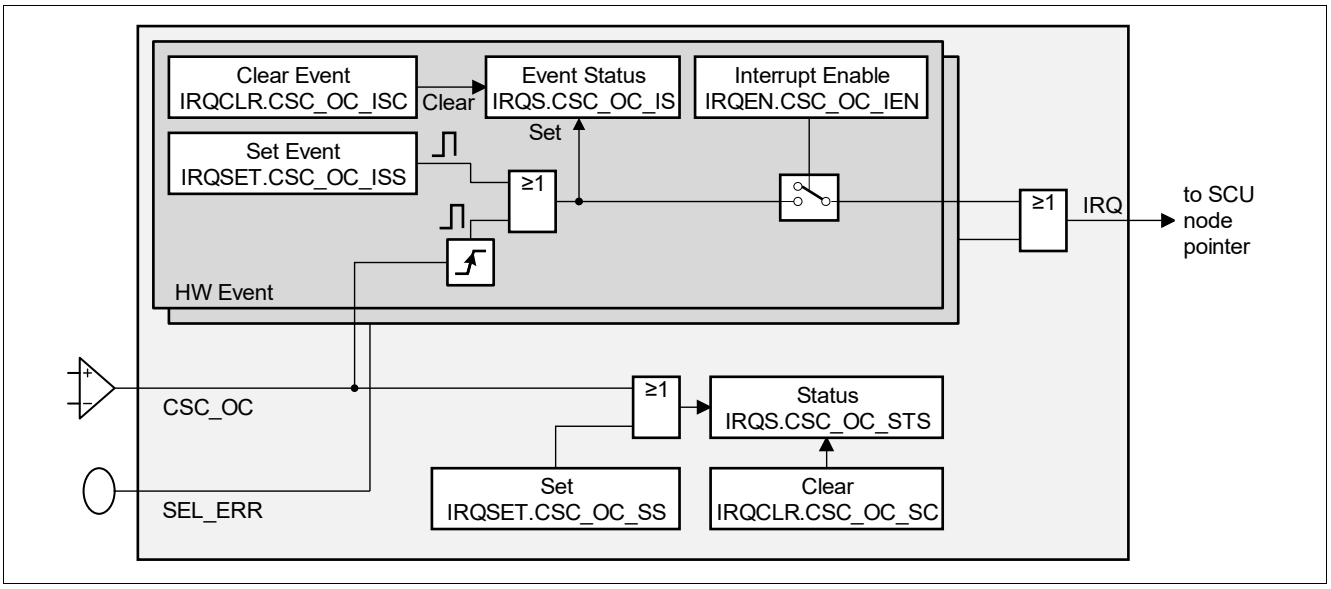


Figure 230 Interrupt handling

Note: The SEL_ERR event is also triggered by CSA gain and offset selection errors.

19.5 Operation mode behavior

Table 247 Operation mode behavior CSC

Reset	• RESET_TYPE_5 resets the CSC configuration settings
Power-up / power-down	• After power-up the CSC is disabled by default • At power-down the CSC is reset by RESET_TYPE_5
Active mode	• The CSC is disabled by default • The CSC can be enabled and configured by setting the AHB registers
Stop mode	• The CSC is disabled automatically when entering stop mode • The configuration of the CSC is lost • The CSC stays disabled after wake-up
Sleep mode	• The CSC is disabled automatically when entering sleep mode • The configuration of the CSC is lost • The CSC stays disabled after wake-up
Fail-sleep mode	• The CSC behave like in sleep mode

19.6 Global control

The CSC is enabled together with the CSA by setting the CTRL1.CSAC_EN bit.

The configuration is done in the following bit fields (for details see **Chapter 19.7**):

- CTRL2.THR_SEL for the overcurrent threshold
- CTRL2.TFILT_SEL for the filter time

The possible range of settings for CTRL2.THR_SEL can be limited by CTRL1.THR_MAX (for details see **Chapter 19.8**).

Notes

1. *CTRL1.THR_MAX must be written before enabling the CSA/CSC for the first time and is frozen afterwards.*
2. *After enabling the CSC a built-in self-test of the CSA and the CSC is started (see **Chapter 19.10**).*

19.7 Overcurrent threshold

The CSC compares the CSA output voltage V_{CSAOUT} (connected to the CSC input "INP") against a programmable threshold voltage V_{CSC_THRx} which is derived from V_{CSAOUT_OFF} (connected to the CSC input "INOFF") as shown in the following equation and figure:

(19.1)

$$V_{CSC\_THRx} = V_{CSAOUT\_OFF} + (x+1)\frac{V_{CSAOUT@MAX} - V_{CSAOUT\_OFF}}{32}$$

x = {0...31} defined by CTRL2.THR_SEL

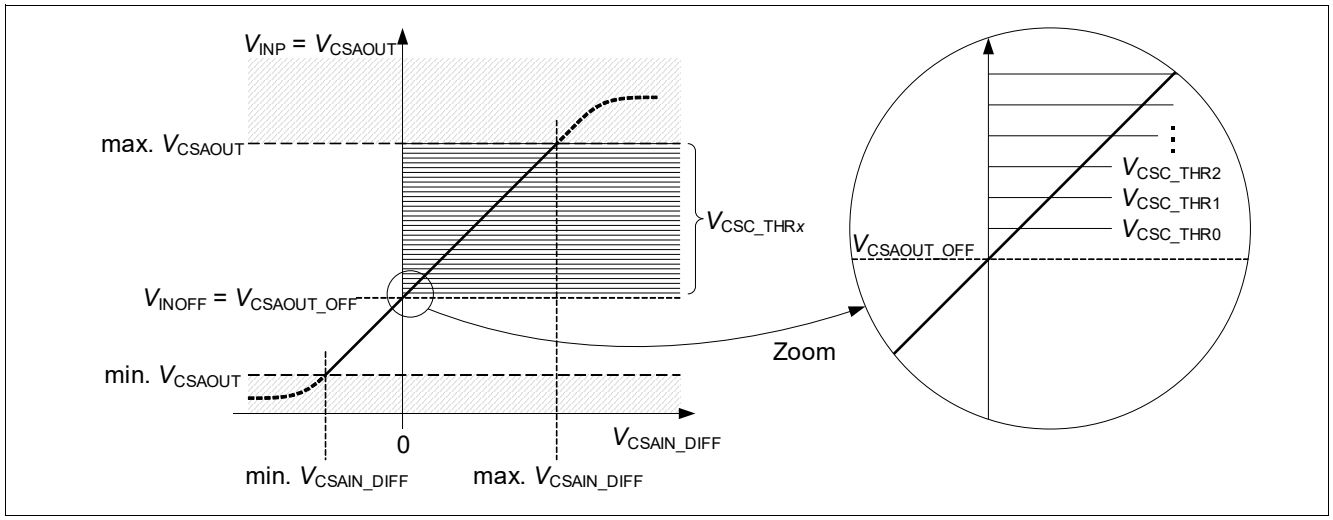


Figure 231 Programmable overcurrent threshold

Notes

1. *The overcurrent threshold is generated by scaling the analog reference voltage (VAREF vs. VAGND) appropriately.*
2. ***Equation (19.1)** and **Figure 231** show the nominal case without error calculation considerations.*

19.8 Maximum threshold setting

The CSC is part of the functional safety relevant safe switch-off path. Therefore, a special protection mechanism is implemented to ensure a minimum sensitivity to detect shunt overcurrents. This is done by

limiting the range of the overcurrent threshold settings in the bit field CTRL1.THR_MAX. Selecting values greater than this limit in the bit field CTRL2.THR_SEL triggers a “selection error” (see IRQS.SEL_ERR_IS) and effectively sets the overcurrent threshold to the value in CTRL1.THR_MAX.

Additionally, a double bit storage scheme is implemented to protect against single event upsets: The bit field CTRL1.THR_MAX is 10 bits wide but effectively decodes to a 5-bit value corresponding to the overcurrent threshold settings allowed for CTRL2.THR_SEL. The 10-bit to 5-bit decoding is done by ANDing pairs of bits as shown in the following figure:

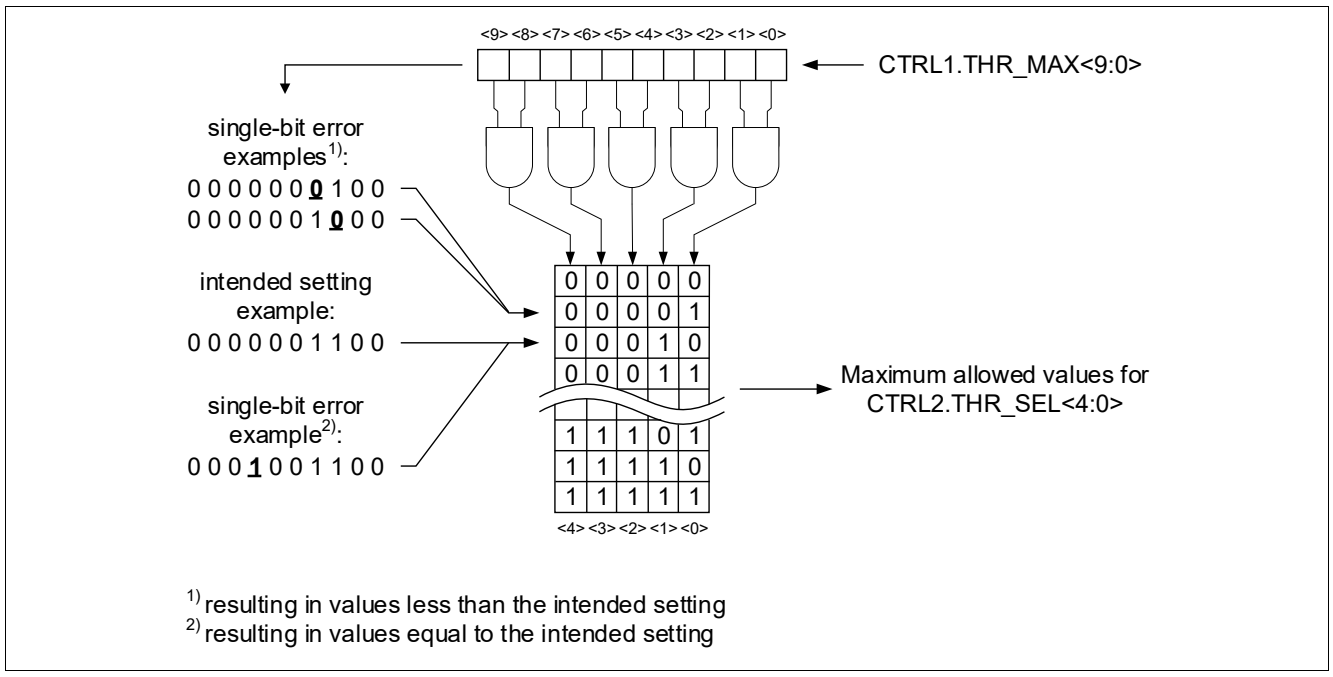


Figure 232 Double bit storage decoding scheme

The examples at the left in **Figure 232** show that if the recommended settings from **Table 248** are used, where both bits of a pair are either cleared or set, this double-bit storage scheme ensures that a single-bit error results in an effective value which is equal to or less than the originally intended setting (and therefore potentially further limits the possible range of settings for CTRL2.THR_SEL):

Table 248 Examples of recommended settings and their effective decoded values

Recommended settings	Regular effective 5-bit value	Possible effective 5-bit values in the case of a single-bit error
0000000000_B	00000_B	00000_B (equal)
0000000011_B	00001_B	00000_B (less) or 00001_B (equal)
0000001100_B	00010_B	00000_B (less) or 00010_B (equal)
0000001111_B	00011_B	00001_B (less) or 00010_B (less) or 00011_B (equal)
...	...	...
1111111111_B	11111_B	01111_B (less) or 10111_B (less) or 11011_B (less) or 11101_B (less) or 11110_B (less)

Note: CTRL1.THR_MAX cannot be changed anymore once the CSC is enabled.

19.9 Overcurrent status

The status of the overcurrent comparison is reported by:

- IRQS.CSC_OC_OUT: real-time status of the comparator output
- IRQS.CSC_OC_STS: latched overcurrent event (can be cleared by IRQCLR.CSC_OC_SC after the overcurrent condition disappeared)
- IRQS.CSC_OC_IS: latched overcurrent interrupt status (can be cleared by IRQCLR.CSC_OC_ISC)

19.10 Built-in self-test

A built-in self-test (BIST) is implemented to make sure that the overcurrent shutdown path is working as expected.

It is started, if the CSA and CSC are enabled by setting the CTRL1.CSAC_EN bit:

- The first time after a reset, or
- Each time after a previously failing built-in self-test

A self-test fail is processed in the PMU.

Note: The shunt current must be zero during the runtime of the built-in self-test.

The built-in self-test selects the following settings regardless of the actual register bit field contents in CTRL2:

- GAIN_SEL: 10_B
- OFFS_SEL: 00_B
- THR_SEL: 03_H
- TFILT_SEL: 11_B

Note: The actual register bit field contents are not overwritten by the built-in self-test and are again valid after the built-in self-test has finished.

After having applied the above settings the built-in self-test waits for nominal 4 µs to let the CSA and CSC start up and then runs in two steps:

Table 249 Built-in self-test steps

Step	Description	Additional input offset	Pass criterion: status of CSC_OC
1	Check normal operation	disabled	’0’ (no overcurrent)
2	Check overcurrent condition	enabled	’1’ (artificial overcurrent correctly detected)

In step 1 the signal chain is checked under normal operating conditions without shunt current. The BIST waits for nominal 12 µs to allow the CSA and CSC to settle and pass the digital filter. Afterwards, the status of CSC_OC is checked:

- CSC_OC = ‘0’ (PASS): expected status for prerequisite “no shunt current”
- CSC_OC = ‘1’ (FAIL): an overcurrent is wrongly detected and the CSC_BIST_FAIL output is set to ‘1’

In step 2 an artificial overcurrent condition is generated by activating the additional input offset at the CSA inputs. The BIST waits for nominal 12 µs to allow the CSA and CSC to settle and pass the digital filter. Afterwards, the status of CSC_OC is checked:

- CSC_OC = ‘0’ (FAIL): the artificial overcurrent has not been detected and the CSC_BIST_FAIL output is set to ‘1’
- CSC_OC = ‘1’ (PASS): expected status for the artificial overcurrent condition

If the build-in self-test has finished successfully with CSC_BIST_FAIL='0' the IRQS.CSC_BIST_STS flag and the CSC_EN output are set to '1'.

Notes

1. *During the BIST an artificial overcurrent event is generated which sets the flags IRQS.CS_OC_IS and IRQS.CS_OC_STS. These flags must be cleared by software.*
2. *The total runtime of the built-in self-test is nominal 28 µs (i.e. 4 µs start-up time at the beginning and 12 µs for each of the two steps) and depends on the accuracy of the MCLK (f_{MCLK}).*

19.11 Programmer's guide

19.11.1 Initialization sequence

This section summarizes the hints and constraints given throughout the detailed functional descriptions of the CSA and the CSC into a step-by-step initialization sequence.

Step 1	Ensure that the shunt current is zero in order not to disturb the built-in self-test which might be triggered automatically during the initialization sequence
Step 2	Ensure a valid reference voltage at VAREF/VAGND
Step 3 (optional)	Configure CTRL1.GAIN_MIN, CTRL1.OFFS_MIN, and CTRL1.THR_MAX
Step 4	Enable the CSA and CSC by setting CTRL1.CSAC_EN *Note: Step 3 can be done together with step 4 with only one write access to the CTRL1 register.*
Step 5	Wait until the built-in self-test is finished which might have been automatically started by step 4
Step 6 (optional)	Check the result of the built-in self-test reported by IRQS.CSC_BIST_STS. In the case of a fail the built-in self-test can be run again by disabling the CSA and CSC by clearing CTRL1.CSAC_EN and jumping back to step 4

19.12 Register description CSC (and CSA)

The CSA and the CSC share the same register set.

19.12.1 CSA/CSC Address Maps

Table 250 Register Address Space - CSACSC

Module	Base Address	End Address	Note
CSACSC	40010000_H	$40013FFF_H$	

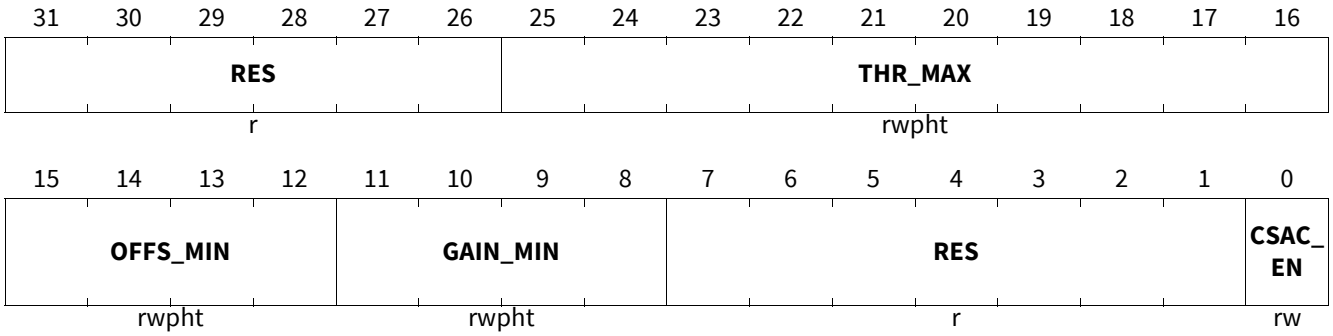
Table 251 Register Overview - CSACSC (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CTRL1	Current sense amplifier and comparator control	0000_H	813
CTRL2	Current sense amplifier and comparator control	0004_H	814
IRQS	Current sense amplifier/comparator interrupt status	0008_H	815
IRQCLR	Current sense amplifier/comparator interrupt status clear	$000C_H$	816
IRQSET	Current sense amplifier/comparator interrupt status set	0010_H	817
IRQEN	Current sense amplifier/comparator interrupt enable	0014_H	817

19.12.2 CSACSC Registers

Current sense amplifier and comparator control

CTRL1
Current sense amplifier and comparator control(0000_H) **RESET_TYPE_5 Value: $0000\ FF00_H$**



Field	Bits	Type	Description
CSAC_EN	0	rw	**Current sense amplifier and comparator enable** 0_B **DISABLE**, 1_B **ENABLE**,
RES	7:1, 31:26	r	**Reserved** Always read as 0
GAIN_MIN	11:8	rwpht	**Current sense amplifier minimum gain setting** *Note: These bits cannot be changed after enabling the CSA/CSC.* The bits are ORed in pairs of two and this way represent the two bit table of CTRL2.GAIN_SEL. Both bits of each pair need to be set/reset for the desired minimum gain value. 0_H **10**, Min. gain factor 10 1_H **NU**, Don't use this setting 2_H **NU**, Don't use this setting 3_H **20**, Min. gain factor 20 4_H **NU**, Don't use this setting … B_H **NU**, Don't use this setting C_H **40**, Min. gain factor 40 D_H **NU**, Don't use this setting E_H **NU**, Don't use this setting F_H **60**, Min. gain factor 60

Register description CSC (and CSA)

Field	Bits	Type	Description
OFFS_MIN	15:12	rwpht	**Current sense amplifier minimum output offset setting** *Note: These bits cannot be changed after enabling the CSA/CSC.* The bits are ORed in pairs of two and this way represent the two bit table of CTRL2.OFFS_SEL. Both bits of each pair need to be set/reset for the desired minimum output offset value. 0_H **0V5**, Min. output offset 0.5 V 1_H **NU**, Don't use this setting 2_H **NU**, Don't use this setting 3_H **1V0**, Min. output offset 1.0 V 4_H **NU**, Don't use this setting … B_H **NU**, Don't use this setting C_H **1V5**, Min. output offset 1.5 V D_H **NU**, Don't use this setting E_H **NU**, Don't use this setting F_H **2V0**, Min. output offset 2.0 V
THR_MAX	25:16	rwpht	**Current sense comparator maximum threshold setting** *Note: These bits cannot be changed after enabling the CSA/CSC.* The bits are ANDed in pairs of two and this way represent the five bit table of CTRL2.THR_SEL. Both bits of each pair need to be set/reset for the desired maximum comparator threshold value.

Current sense amplifier and comparator control

CTRL2

Current sense amplifier and comparator control(0004_H) **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						TFILT_SEL		RES			THR_SEL				
r						rw		r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES							VOUT_SEL	RES			OFFS_SEL		GAIN_SEL		ADD_INP_OFFS
r							rw	r			rw		rw		rw

Field	Bits	Type	Description
ADD_INP_OFFS	0	rw	**Additional input offset setting** 0_B **DISABLE**, No offset, normal operation 1_B **ENABLE**, +20 mV artificial input offset added
GAIN_SEL	2:1	rw	**Current sense amplifier gain setting** 00_B **10**, Gain factor 10 01_B **20**, Gain factor 20 10_B **40**, Gain factor 40 11_B **60**, Gain factor 60

Register description CSC (and CSA)

Field	Bits	Type	Description
OFFS_SEL	4:3	rw	**Current sense amplifier output offset setting** 00_B **0V5**, Output offset 0.5 V 01_B **1V0**, Output offset 1.0 V 10_B **1V5**, Output offset 1.5 V 11_B **2V0**, Output offset 2.0 V
RES	7:5, 15:9, 23:21, 31:26	r	**Reserved** Always read as 0
VOUT_SEL	8	rw	**Current sense output selection** 0_B **VOUT**, CSA output selected 1_B **VOFFS**, Offset voltage selected
THR_SEL	20:16	rw	**Current sense comparator threshold setting** $V_{CSC\_THR} = V_{CSAOUT\_OFF} + (THR\_SEL + 1) * (V_{CSAOUT@MAX} - V_{CSAOUT\_OFF}) / 32$ 00_H **MIN_THR**, Output offset voltage plus 1 LSB 01_H **THR1**, Threshold according to formula … $1E_H$ **THR30**, Threshold according to formula $1F_H$ **MAX_THR**, Maximum of output voltage linear range
TFILT_SEL	25:24	rw	**Filter time for current sense comparator** 00_B **2u**, 2 µs filter time 01_B **4u**, 4 µs filter time 10_B **6u**, 6 µs filter time 11_B **8u**, 8 µs filter time

Current sense amplifier/comparator interrupt status

IRQS

Current sense amplifier/comparator interrupt status(0008_H) **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						CSC_BIST_STS	CSC_OC_OUT	RES							CSC_OC_STS
r						r	r	r							rhxr

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														SEL_ERR_IS	CSC_OC_IS
r														rhxre	rhxre

Field	Bits	Type	Description
CSC_OC_IS	0	rhxre	**Overcurrent event interrupt status** 0_B **NO_OC**, No overcurrent event detected 1_B **OC**, Overcurrent event detected

Register description CSC (and CSA)

Field	Bits	Type	Description
SEL_ERR_IS	1	rhxre	**CSA minimum gain or offset selection error interrupt status** 0_B **SELOK**, CSA min. gain or max. offset selection fulfilled 1_B **SELERR**, CSA min. gain or max. offset selection violated
RES	15:2, 23:17, 31:26	r	**Reserved** Always read as 0
CSC_OC_STS	16	rhxr	**Overcurrent event status** 0_B **NO_OC**, No overcurrent event detected 1_B **OC**, Overcurrent event detected
CSC_OC_OUT	24	r	**Overcurrent comparator output** 0_B **NO_OC**, CSC output is low 1_B **OC**, CSC output is high
CSC_BIST_STS	25	r	**CSC built-in self test status** 0_B **BIST_OPEN**, CSC built-in self test not started, still running or failed 1_B **BIST_DONE**, CSC built-in self test successfully completed

Current sense amplifier/comparator interrupt status clear

IRQCLR

Current sense amplifier/comparator interrupt status clear($000C_H$) **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															CSC_OC_SC
r															w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														SEL_ERR_ISC	CSC_OC_ISC
r														w	w

Field	Bits	Type	Description
CSC_OC_ISC	0	w	**Overcurrent event interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
SEL_ERR_ISC	1	w	**CSA minimum gain or offset selection error interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
RES	15:2, 31:17	r	**Reserved** Always read as 0
CSC_OC_SC	16	w	**Overcurrent event status clear** 0_B **KEEP**, 1_B **CLEAR**,

Register description CSC (and CSA)

Current sense amplifier/comparator interrupt status set

IRQSET

Current sense amplifier/comparator interrupt status set(0010_H) **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															CSC_OC_SS
r															w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES														SEL_ERR_ISS	CSC_OC_ISS
r														w	w

Field	Bits	Type	Description
CSC_OC_ISS	0	w	**Overcurrent event interrupt status set** 0_B **KEEP**, 1_B **SET**,
SEL_ERR_ISS	1	w	**CSA minimum gain or offset selection error interrupt status set** 0_B **KEEP**, 1_B **SET**,
RES	15:2, 31:17	r	**Reserved** Always read as 0
CSC_OC_SS	16	w	**Overcurrent event status set** 0_B **KEEP**, 1_B **SET**,

Current sense amplifier/comparator interrupt enable

IRQEN

Current sense amplifier/comparator interrupt enable(0014_H) **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Register description CSC (and CSA)

Field	Bits	Type	Description
CSC_OC_IEN	0	rw	**Overcurrent event interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
SEL_ERR_IEN	1	rw	**CSA minimum gain or offset selection error interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
RES	31:2	r	**Reserved** Always read as 0

20 Temperature Sensor Unit (TMPSNS)

20.1 Features overview

The SoC integrates multiple temperature sensors spread across the die. These monitors are located and associated with critical blocks such as:

- PMU module (VDDP and VDDEXT linear regulators)
- CAN transceiver module
- Gate driver module (2-stage charge pump), TEMP0
- SoC core logic, TEMP1

This chapter describes the behavior of the temperature sensors TEMP0 and TEMP1. The overtemperature protection mechanisms associated with the PMU and the CAN transceiver are described in the respective module chapters.

Attention: Actual overtemperature warning and shutdown thresholds may differ. Please refer to the product datasheet and register definition for more accurate information.

The TMPSNS provides following features:

- Two dedicated temperature sensors to measure the on-chip temperature at different locations:
  - TEMP0 measures the die temperature in the charge pump of gate driver module
  - TEMP1 measures the die temperature in the center of the chip (i.e. system temperature)
- Positive output slope of 2.5 mV/°C typ. over the full T_j = -40°C to +175°C temperature range
- Temperature sensors connect internally to the multiple inputs analog-to-digital converter (ADC2). The ADC2 post-processing and digital comparator features are used for background temperature monitoring
- ADC2 raises an overtemperature shutdown flag (CP_OT) when the die temperature in the charge pump (TEMP0) exceeds the threshold value (190°C typ.)
- ADC2 raises an overtemperature warning flag (SYS_OTWARN) when the system temperature (TEMP1) exceeds the threshold value (135°C typ.)
- ADC2 raises an overtemperature shutdown flag (SYS_OT) when the system temperature (TEMP1) exceeds the threshold value (190°C typ.). This event will automatically transition the SoC into Fail-sleep system power mode (refer to WAKE_FAIL_STS.SYS_OT bit description)

20.2 Block diagram

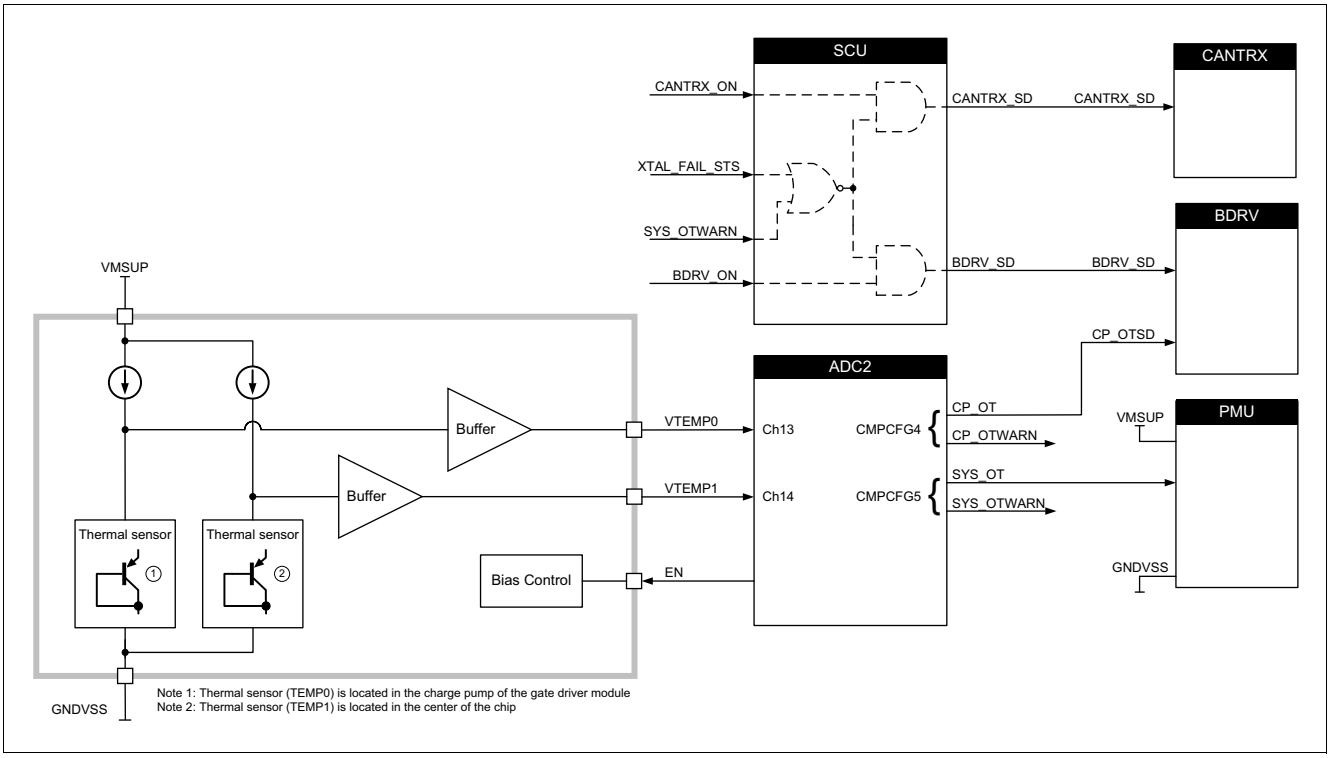


Figure 233 Block diagram TMPSNS

20.3 Toplevel signals

The TMPSNS module interconnects as specified in **Table 252**, **Table 253** and **Table 254**.

Table 252 TMPSNS power domain

Signal	Direction	Description	From/To
VMSUP	Power	1.5 V digital peripherals always-on supply	PMU
GNDVSS	Power	Supply ground	PMU

Table 253 TMPSNS outputs

Signal	Direction	Description	From/To
VTEMP0	Output	Analog voltage proportional to the die temp	ADC2
VTEMP1	Output	Analog voltage proportional to the die temp	ADC2

Table 254 TMPSNS control signals

Signal	Direction	Description	From/To
EN	Input	Module enable signal	ADC2

20.4 Interrupts

The TMPSNS does not directly support any interrupt node. Interrupt generation is handled by the ADC2 module. For more details, please refer to the **Monitoring Analog Digital Converter 2 (ADC2)** chapter.

20.5 Operation mode behavior

Table 255 TMPSNS state-of-operation vs. SoC system power modes

Reset	• TEMP0/TEMP1 sensors are disabled
Power-up / Power-down	• TEMP0/TEMP1 sensors are enabled during the power-up sequence • TEMP0/TEMP1 sensors are disabled at the end of the power-down sequence
Active mode	• TEMP0/TEMP1 sensors are enabled
Stop mode	• TEMP0/TEMP1 sensors are disabled
Sleep mode	• TEMP0/TEMP1 sensors are disabled
Fail-sleep mode	• TEMP0/TEMP1 sensors are disabled

20.6 TMPSNS state-of-operation

The TMPSNS performs on-chip temperature measurements at two distinct locations. The TEMP0 sensor measures the die temperature in the vicinity of the charge pump integrated in the gate driver module. The TEMP1 measures the die temperature in the center of the chip.

The temperature sensors connect internally to the multiple inputs analog-to-digital converter (ADC2). The voltage across the temperature sensor varies linearly with temperature according to the following formula.

$$V_{TEMP} = a + b \times T$$

where a = 678 mV (typ.) and b = 2.5 mV/°C (typ.). The die temperature (T) is given in degrees Celsius.

If the temperature sensor, ADC and voltage reference are considered ideal (i.e. without error), the ADC2 output will be 33_D at 0°C and 45_D at 100°C.

The ADC2 aims to run continuous discrete time measurements in the background. Its compare channels 4 and 5 are assigned to TEMP0 and TEMP1 and are preconfigured to operate in the so-called OV mode (i.e. to detect if a signal is above a lower or an upper threshold).

In case the measurement value exceeds the lower limit (i.e. temperature warning threshold), the CMP_LO_STSx (status flag) and CMP_LO_ISx (interrupt status) can be set automatically. Similarly, if the measurement value exceeds the upper limit (i.e. overtemperature threshold), the CMP_UP_STSx (status flag) and CMP_UP_ISx (interrupt status) can be set automatically.

This functionality facilitates continuous monitoring to protect the system from thermal overstress. As the system temperature (TEMP1) exceeds the warning threshold, it is recommended to actively reduce the SoC power dissipation by:

- Clocking down the microcontroller subsystem
- Reducing the motor control PWM frequency

The system will automatically transition from ACTIVE to FAIL-SLEEP system power mode as the system temperature (TEMP1) raises above the overtemperature threshold (190°C typ., SYS_OT is active). For more details, please refer to the **Power Management Unit (PMU)** chapter.

The System Control Unit (SCU) provides an option to deactivate the bridge driver (BDRV) and/or the CAN transceiver (CANTRX) TX/RX channels as the system temperature (TEMP1) exceeds the warning threshold (135°C typ.). For more details, please refer to the **System Control Unit (SCU)** chapter.

Note: The warning and overtemperature thresholds may differ in specific product variants. For more details, please refer to the electrical characteristics table in the datasheet.

20.7 Programmer's guide

20.7.1 TMPSNS module initialization

The temperature sensor module does not require any initialization sequence. Care should still be taken to initialize properly the ADC2 interrupt nodes.

21 BEMF Comparators (BEMFC)

21.1 Feature overview

For rotor position detection of a BLDC motor the BEMF (Back ElectroMotive Force) information can be used. This BEMF information is always sensed in the phase which is currently not active. Therefore, at each motor phase, a comparator compares the BEMF voltage against a virtual star point built by the other two motor phases and provides post-processing features to generate valid zero-crossing events.

The BEMFC provides following features:

- The BEMF comparator module consists of 3 BEMF comparators, one for each SHx pin
- The BEMF comparators compare the voltage at the corresponding SHx pin to a “virtual star point” voltage which is the average of the voltages at the remaining two SHy and SHz pins (see V_{BEMFC_TH})
- The BEMF comparators provide low settling time t_{BEMFC_D}
- The BEMF comparators can be switched off if not needed to avoid additional power consumption and undesired input currents in power down modes
- The BEMF comparator output signals are spike filtered with a programmable filter time
- The BEMF comparators have each a blanking filter which can be enabled to mask oscillations during a programmable time after switching the corresponding motor phases
- The BEMF comparators have each a demagnetisation filter which can be enabled to automatically remove demagnetisation pulses from the BEMF comparator output signal to get only valid zero-crossing events
- Interrupts can be triggered on BEMF comparator status changes at rising and/or falling edge

21.2 Block diagram

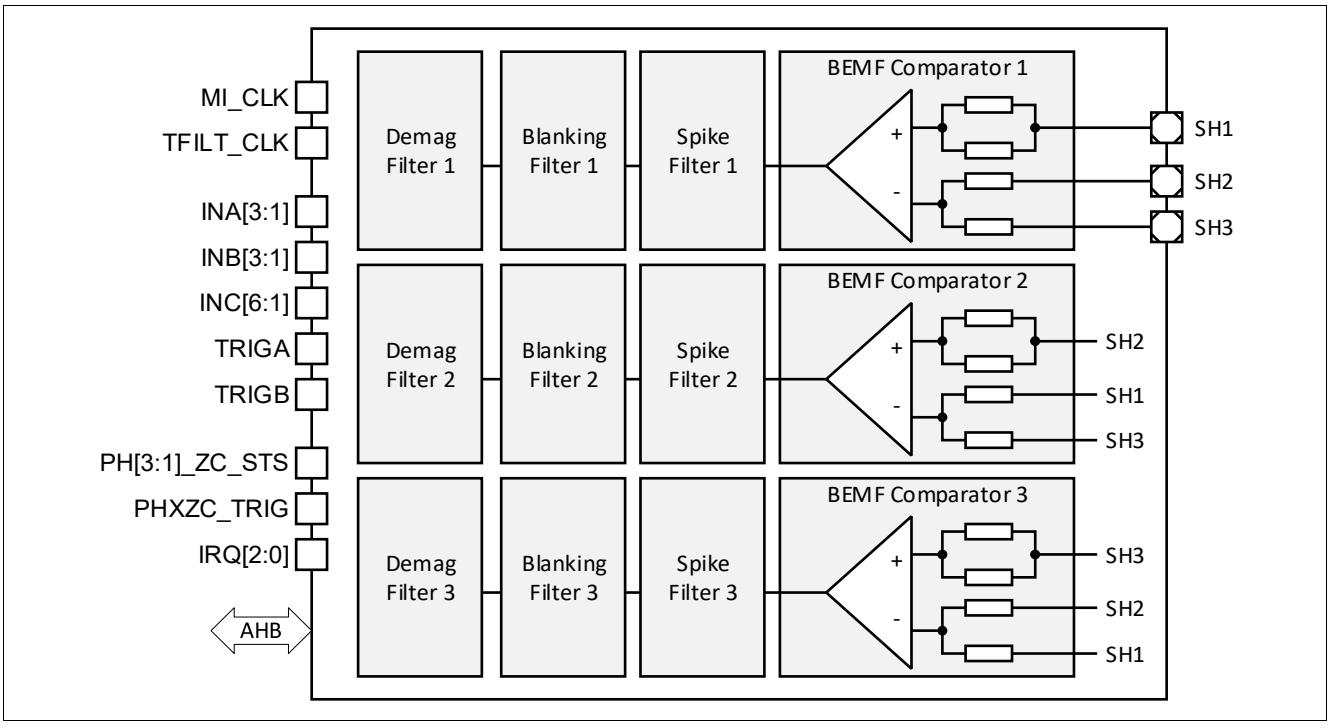


Figure 234 Block diagram BEMFC

21.3 Toplevel signals

Table 256 BEMF comparators clock domains

Signal	Direction	Description	From
MI_CLK	input	Peripheral clock	SCU
TFILT_CLK	input	Filter and blanking time clock	SCU

Table 257 BEMF comparators analog inputs

Signal	Direction	Description	From/to
SH1, SH2, SH3	input	High-side MOSFET source pins	SH1, SH2, SH3 pins

Table 258 BEMF comparators digital inputs

Signal	Direction	Description	From/to
INA[3:1], INB[3:1], TRIGA, TRIGB	input	Used to create sampling pulses	CCU7
INC[6:1]	input	Used to create sampling pulses	GPIOs

Table 259 BEMF comparators outputs

Signal	Direction	Description	From/to
PH[3:1]_ZC_STS	output	Phase zero-crossing status signals	CCU7
PHXZC_TRIG	output	Trigger signal for timer	GPT12

Table 260 BEMF comparators interrupt requests

Signal	Direction	Description	From/to
IRQ[2:0]	output	Interrupts from the PHx zero crossing edge detections	SCU

21.4 Interrupts

Events

The BEMF comparators provide the following events:

- Rising zero-crossing event on each motor phase (see BEMFC_IRQS.PHx_ZC_RISE_IS)
- Falling zero-crossing event on each motor phase (see BEMFC_IRQS.PHx_ZC_FALL_IS)

Interrupt requests

The BEMF comparators provide three interrupt nodes (one per motor phase):

- IRQ0: Interrupts from rising or falling zero-crossing event on motor phase 1
- IRQ1: Interrupts from rising or falling zero-crossing event on motor phase 2
- IRQ2: Interrupts from rising or falling zero-crossing event on motor phase 3

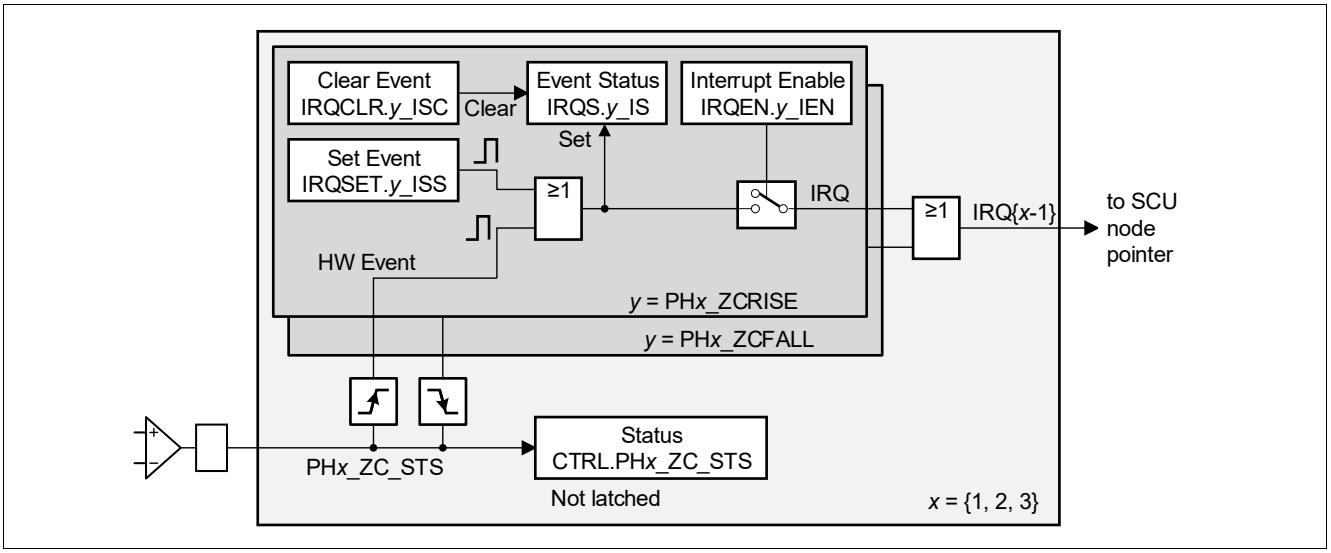


Figure 235 Interrupt handling

21.5 Operation mode behavior

Table 261 Operation mode behavior BEMFC

Reset	• RESET_TYPE_5 resets the BEMFC configuration settings
Power-up / power-down	• After power-up the BEMF comparators are disabled by default • At power-down the BEMF comparators are reset by RESET_TYPE_5
Active mode	• The BEMF comparators are disabled by default • The BEMF comparators can be enabled and configured by setting the AHB registers
Stop mode	• The BEMF comparators are disabled automatically when leaving active mode • The configuration of the BEMF comparators is lost • The BEMF comparators are disabled after wake-up
Sleep mode	• The BEMF comparators behave like in stop mode
Fail-sleep mode	• The BEMF comparators behave like in stop mode

21.6 Comparator

The back electromotive force (BEMF) can be used to detect the rotor position of a multi-phase motor when the corresponding phase pin SHx is not excited by the bridge driver, e.g. during block commutation. The rotating magnetic field of the rotor induces a voltage into the stator inductors that can be measured at the SHx pin relative to the virtual star point built by the SHx pins of the other phases.

The BEMF comparator of the pin SHx can be enabled by setting BEMFC_CTRL.PHx_COMP_EN to ‘1’. The BEMF comparator output is ‘1’ if the SHx voltage is greater than the arithmetic mean value of the SH voltages of the other phases, created by a resistive divider:

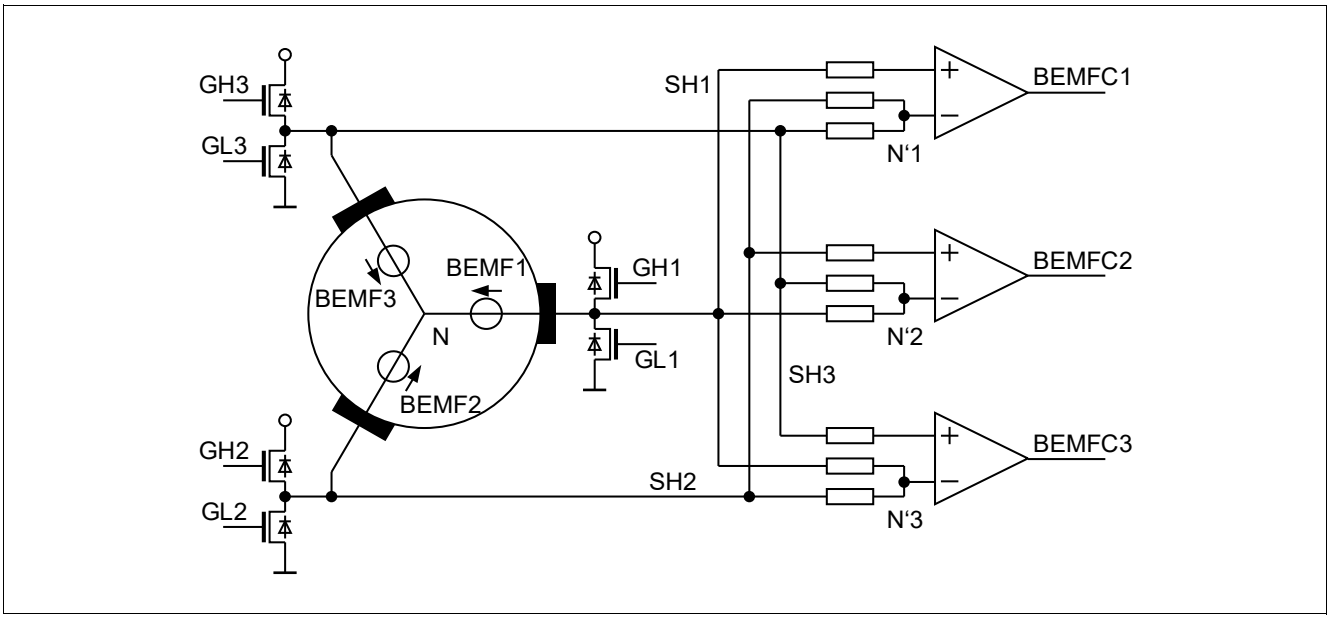


Figure 236 BEMF comparator connections to the motor

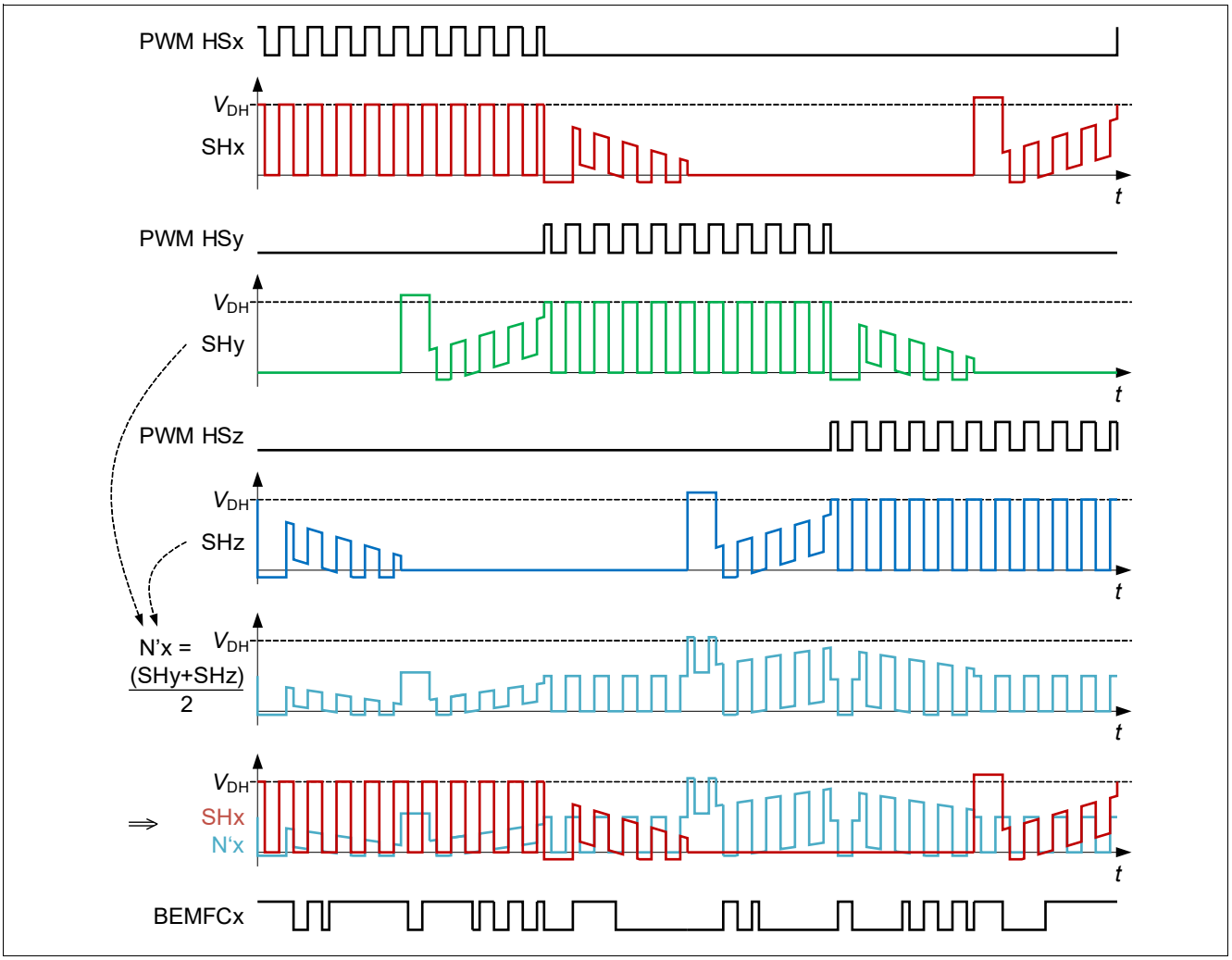


Figure 237 BEMF comparator output signal generation (example with PWM on high-side driver)

21.7 Post-processing

In order to provide suitable signals to trigger commutation events the outputs of the BEMF comparators need to be post-processed as shown in the figure below:

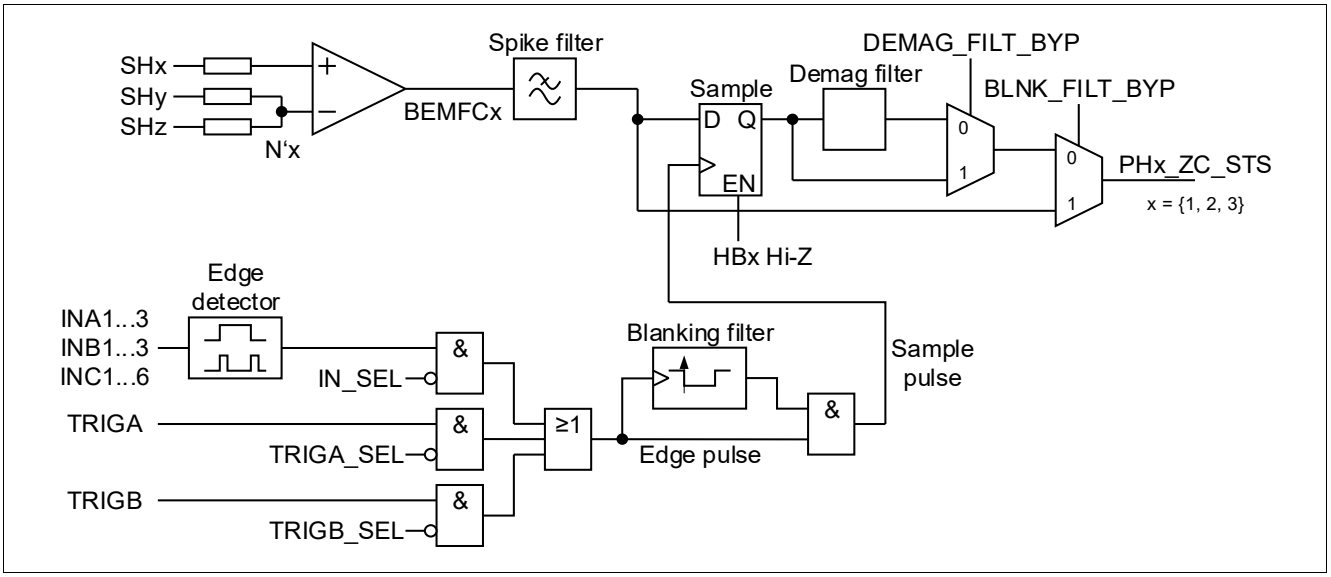


Figure 238 BEMF comparator output signal post-processing

The post-processing chain includes spike filters (see **Chapter 21.7.1**), demagnetisation filters (see **Chapter 21.7.2**), and blanking filters (see **Chapter 21.7.3**) for proper sampling pulse generation based on twelve level-sensitive PWM inputs (INA1...3, INB1...3, INC1...6) and two trigger pulse inputs (TRIGA and TRIGB).

21.7.1 Spike filter

The spike filter provides a low-pass filtered signal at the input of the sampling flip-flop. It is active at both transitions, from '0' to '1' and from '1' to '0', and its filter time can be configured by the bit field BEMFC_CTRL.CMP_TFILT_SEL.

21.7.2 Demagnetisation filter

After the half bridge stopped driving its connected motor phase, the coil demagnetisation forces the BEMF comparator output to the opposite of the previous filtered value. In this case, it generates a wrong zero-crossing event. The demagnetisation filter recognizes this opposite value and waits until the previous value appears again at the sampled comparator status. The zero-crossing status flag BEMFC_CTRL.PHx_ZC_STS (x = 1...3) is then set accordingly:

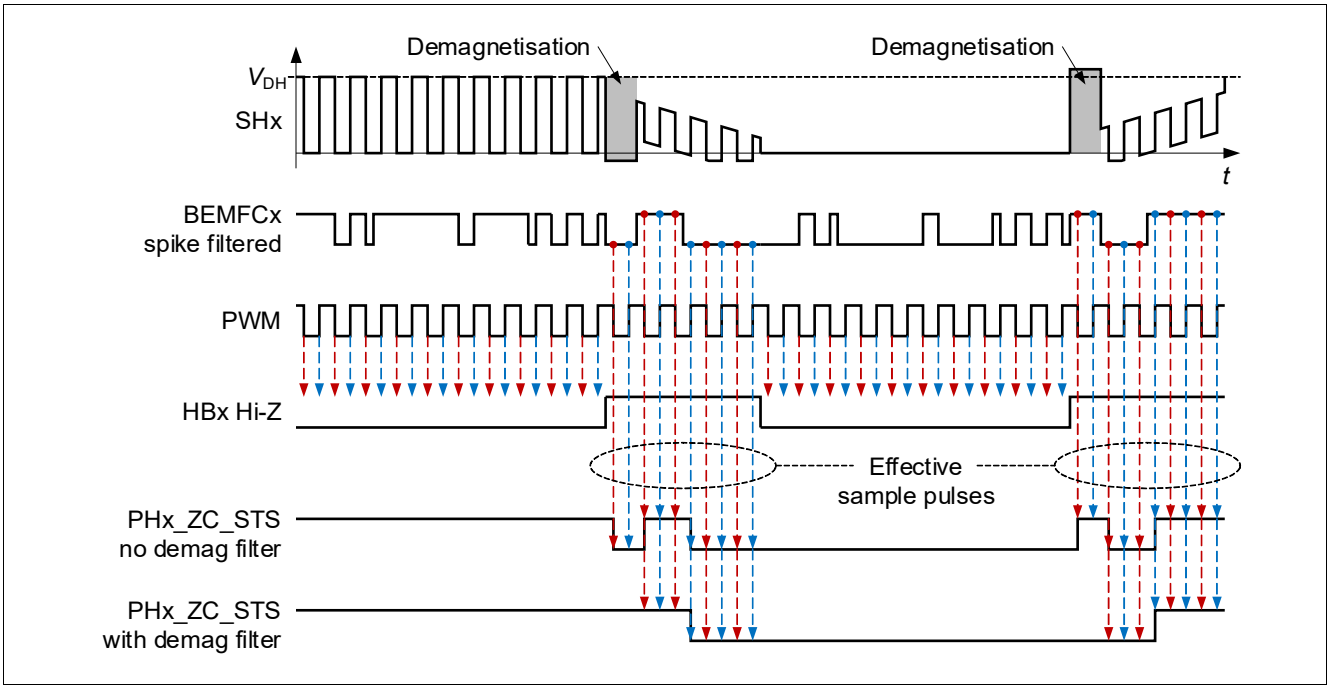


Figure 239 BEMF comparator demagnetisation filter

Note: The demagnetisation filter can be bypassed by setting BEMFC_CTRL.DEMAG_FILT_BYP to '1' (e.g. during start-up phase of the motor).

21.7.3 Blanking filter

Each BEMF comparator output signal is sampled at the end of a PWM "on" or "off" phase of the other half bridges. This ensures maximum possible settling times to achieve stable signals after switching of an external MOSFET.

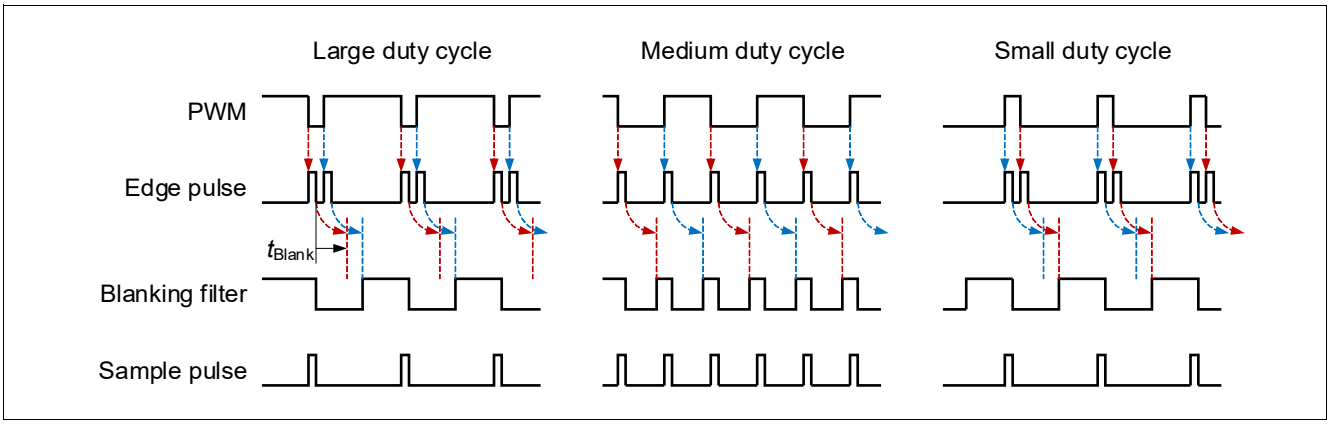


Figure 240 Blanking filter at different duty cycles

Figure 240 shows different cases of PWM duty cycles. An edge pulse is generated with any edge of the PWM control inputs INA1...3, INB1...3, and INC1...6 if BEMFC_CTRL.IN_SEL='0' or with the trigger pulse inputs TRIGA if BEMFC_CTRL.TRIGA_SEL='0' or TRIGB if BEMFC_CTRL.TRIGB_SEL='0'. This edge pulse starts the blanking filter time and becomes a sampling pulse only if the blanking filter has previously expired and therefore was set to '1'. The blanking filter time can be configured by BEMFC_CTRL.TBLNK_SEL.

With large duty cycles the "off" phase is too short for the back-EMF voltage to settle after the switching of one of the other half bridges. The comparator signal is only sampled at the end of the longer "on" phase. For small

duty cycles the situation is similar, now with the “on” phase being too short to settle. With medium duty cycles both “on” and “off” phases are long enough to sample the comparator signal at the end of each phase.

Notes

1. *For duty cycles of 0% or 100% no edge pulses can be generated from the corresponding INAx, INBx, and INCx signals. In this case, periodical trigger pulses must be provided to TRIGA and/or TRIGB (e.g. from the PWM base frequency generation).*
2. *The sampling pulses are only processed if the corresponding half bridge is not driven, i.e. the half bridge is in high-impedance state.*
3. *The blanking filter can be bypassed by setting BEMFC_CTRL.BLNK_FILT_BYP to ‘1’.*

21.8 Outputs

The BEMF comparator module provides the following outputs which are connected to other modules on the device (see section “BEMFC interconnections” in chapter “Product definitions”):

- PH1_ZC_STS: post-processed output signal of BEMF comparator at SH1
- PH2_ZC_STS: post-processed output signal of BEMF comparator at SH2
- PH3_ZC_STS: post-processed output signal of BEMF comparator at SH3
- PHXZC_TRIG: a configurable output according to **Figure 241**

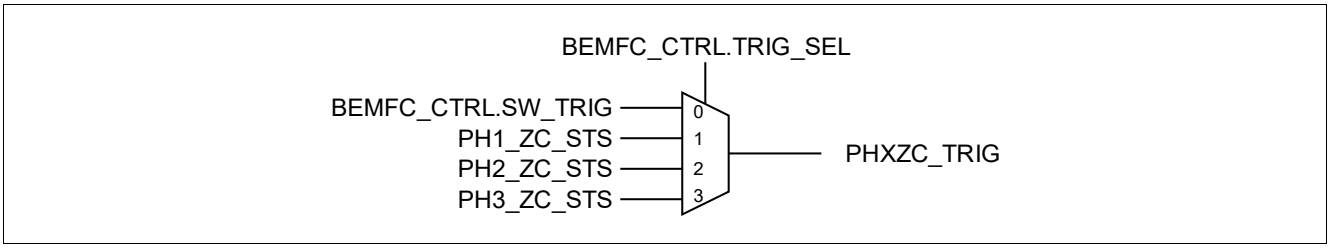


Figure 241 BEMF comparator output signal usage

Note: The status of the PHx_ZC_STS outputs can be defined by BEMFC_CTRL.PHx_COMP_DIS_SET in the case the corresponding BEMF comparator is disabled by BEMFC_CTRL.PHx_COMP_EN=’0’.

21.9 Register description BEMFC

The BEMFC and the BDRV share the same address space.

21.9.1 BEMFC/BDRV Address Maps

Table 262 Register Address Space - BDRV

Module	Base Address	End Address	Note
BDRV	$4000C000_H$	$4000FFFF_H$	

Table 263 Register Overview - BDRV (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
BEMFC_CTRL	BEMF comparator control and status	$012C_H$	832
BEMFC_IRQS	BEMF comparator interrupt status	0130_H	834
BEMFC_IRQCLR	BEMF comparator interrupt status clear	0134_H	835
BEMFC_IRQSET	BEMF comparator interrupt status set	0138_H	836
BEMFC_IRQEN	BEMF comparator interrupt enable	$013C_H$	837

21.9.2 BDRV Registers

BEMF comparator control and status

BEMFC_CTRL
BEMF comparator control and status **($012C_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES				PH3_ZC_STS	PH2_ZC_STS	PH1_ZC_STS	SW_TRIG	RES			TRIGB_SEL	TRIGA_SEL	IN_SEL	TRIG_SEL	
r				r	r	r	rw	r			rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	BLNK_FILT_BYP	DEMAG_FILT_BYP	TBLNK_SEL			CMP_TFILT_SEL		RES	PH3_COMP_DIS_SET	PH2_COMP_DIS_SET	PH1_COMP_DIS_SET	RES	PH3_COMP_EN	PH2_COMP_EN	PH1_COMP_EN
r	rw	rw	rw			rw		r	rw	rw	rw	r	rw	rw	rw

Field	Bits	Type	Description
PH1_COMP_EN	0	rw	**Phase 1 comparator enable** *Note: For a proper operation of BEMF functionality all three comparators need to be enabled.* 0_B **DISABLE**, 1_B **ENABLE**,
PH2_COMP_EN	1	rw	**Phase 2 comparator enable** *Note: For a proper operation of BEMF functionality all three comparators need to be enabled.* 0_B **DISABLE**, 1_B **ENABLE**,
PH3_COMP_EN	2	rw	**Phase 3 comparator enable** *Note: For a proper operation of BEMF functionality all three comparators need to be enabled.* 0_B **DISABLE**, 1_B **ENABLE**,
RES	3, 7, 15, 23:21, 31:28	r	**Reserved** Always read as 0
PH1_COMP_DIS_SET	4	rw	**Phase 1 comparator output status value if disabled** 0_B **LOW**, PH1_ZC_STS is '0' if disabled 1_B **HIGH**, PH1_ZC_STS is '1' if disabled
PH2_COMP_DIS_SET	5	rw	**Phase 2 comparator output status value if disabled** 0_B **LOW**, PH2_ZC_STS is '0' if disabled 1_B **HIGH**, PH2_ZC_STS is '1' if disabled

Field	Bits	Type	Description
PH3_COMP_DIS_SET	6	rw	**Phase 3 comparator output status value if disabled** 0_B **LOW**, PH3_ZC_STS is '0' if disabled 1_B **HIGH**, PH3_ZC_STS is '1' if disabled
CMP_TFILT_SEL	9:8	rw	**Symmetrical spike filter time for BEMF comparators** 00_B **1u**, 1 µs filter time 01_B **2u**, 2 µs filter time 10_B **4u**, 4 µs filter time 11_B **8u**, 8 µs filter time
TBLNK_SEL	12:10	rw	**Blanking time for BEMF comparators** 000_B **6u**, 6 µs blanking time 001_B **8u**, 8 µs blanking time 010_B **12u**, 12 µs blanking time 011_B **16u**, 16 µs blanking time 100_B **3u**, 3 µs blanking time 101_B **NU**, Not used (3 µs blanking time selected) 110_B **NU**, Not used (3 µs blanking time selected) 111_B **NU**, Not used (3 µs blanking time selected)
DEMAG_FILT_BYP	13	rw	**Demagnetisation filter bypass** 0_B **ENABLE**, Filter enabled 1_B **BYPASS**, Filter bypassed
BLNK_FILT_BYP	14	rw	**Blanking time and demagnetisation filter bypass** 0_B **ENABLE**, Filter enabled 1_B **BYPASS**, Filter bypassed, direct signal ouptut
TRIG_SEL	17:16	rw	**Trigger output selector** 00_B **SWTRIG**, Trigger from BEMFC_CTRL.SW_TRIG 01_B **PH1TRIG**, Trigger from BEMFC_CTRL.PH1_ZC_STS 10_B **PH2TRIG**, Trigger from BEMFC_CTRL.PH2_ZC_STS 11_B **PH3TRIG**, Trigger from BEMFC_CTRL.PH3_ZC_STS
IN_SEL	18	rw	**Deactivate INA/B/C as sample pulse sources** 0_B **ACT**, INA/B/C sample inputs are used 1_B **NOTACT**, INA/B/C are not used
TRIGA_SEL	19	rw	**Deactivate TRIGA as sample pulse source** 0_B **ACT**, TRIGA sample input is used 1_B **NOTACT**, TRIGA is not used
TRIGB_SEL	20	rw	**Deactivate TRIGB as sample pulse source** 0_B **ACT**, TRIGB sample input is used 1_B **NOTACT**, TRIGB is not used
SW_TRIG	24	rw	**Software trigger for output to timer** TRIG_SEL needs to be set to '00' for this bit to become relevant. 0_B **VALUE_0**, Software trigger reset 1_B **VALUE_1**, Software trigger set

Register description BEMFC

Field	Bits	Type	Description
PH1_ZC_STS	25	r	**Phase 1 zero crossing comparator status** 0_B **NEGATIVE**, Phase voltage is below the average of the other phases 1_B **POSITIVE**, Phase voltage is above the average of the other phases
PH2_ZC_STS	26	r	**Phase 2 zero crossing comparator status** 0_B **NEGATIVE**, Phase voltage is below the average of the other phases 1_B **POSITIVE**, Phase voltage is above the average of the other phases
PH3_ZC_STS	27	r	**Phase 3 zero crossing comparator status** 0_B **NEGATIVE**, Phase voltage is below the average of the other phases 1_B **POSITIVE**, Phase voltage is above the average of the other phases

BEMF comparator interrupt status

BEMFC_IRQS

BEMF comparator interrupt status **(0130_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										PH3_ZCRISE_IS	PH3_ZCFALL_IS	PH2_ZCRISE_IS	PH2_ZCFALL_IS	PH1_ZCRISE_IS	PH1_ZCFALL_IS
r										rhxre	rhxre	rhxre	rhxre	rhxre	rhxre

Field	Bits	Type	Description
PH1_ZCFALL_IS	0	rhxre	**Phase 1 zero crossing falling interrupt status** 0_B **NOZCFALL**, No falling zero crossing detected 1_B **ZCFALL**, Falling zero crossing detected
PH1_ZCRISE_IS	1	rhxre	**Phase 1 zero crossing rising interrupt status** 0_B **NOZCRISE**, No rising zero crossing detected 1_B **ZCRISE**, Rising zero crossing detected
PH2_ZCFALL_IS	2	rhxre	**Phase 2 zero crossing falling interrupt status** 0_B **NOZCFALL**, No falling zero crossing detected 1_B **ZCFALL**, Falling zero crossing detected
PH2_ZCRISE_IS	3	rhxre	**Phase 2 zero crossing rising interrupt status** 0_B **NOZCRISE**, No rising zero crossing detected 1_B **ZCRISE**, Rising zero crossing detected

Register description BEMFC

Field	Bits	Type	Description
PH3_ZCFALL_IS	4	rhxre	**Phase 3 zero crossing falling interrupt status** 0_B **NOZCFALL**, No falling zero crossing detected 1_B **ZCFALL**, Falling zero crossing detected
PH3_ZCRISE_IS	5	rhxre	**Phase 3 zero crossing rising interrupt status** 0_B **NOZCRISE**, No rising zero crossing detected 1_B **ZCRISE**, Rising zero crossing detected
RES	31:6	r	**Reserved** Always read as 0

BEMF comparator interrupt status clear

BEMFC_IRQCLR
BEMF comparator interrupt status clear **(0134_H)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										PH3_ZCRISE_ISC	PH3_ZCFALL_ISC	PH2_ZCRISE_ISC	PH2_ZCFALL_ISC	PH1_ZCRISE_ISC	PH1_ZCFALL_ISC
r										w	w	w	w	w	w

Field	Bits	Type	Description
PH1_ZCFALL_ISC	0	w	**Phase 1 zero crossing falling interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
PH1_ZCRISE_ISC	1	w	**Phase 1 zero crossing rising interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
PH2_ZCFALL_ISC	2	w	**Phase 2 zero crossing falling interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
PH2_ZCRISE_ISC	3	w	**Phase 2 zero crossing rising interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
PH3_ZCFALL_ISC	4	w	**Phase 3 zero crossing falling interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
PH3_ZCRISE_ISC	5	w	**Phase 3 zero crossing rising interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
RES	31:6	r	**Reserved** Always read as 0

Register description BEMFC

BEMF comparator interrupt status set

BEMFC_IRQSET
BEMF comparator interrupt status set **(0138_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										PH3_ZCRISE_ISS	PH3_ZCFALL_ISS	PH2_ZCRISE_ISS	PH2_ZCFALL_ISS	PH1_ZCRISE_ISS	PH1_ZCFALL_ISS
r										w	w	w	w	w	w

Field	Bits	Type	Description
PH1_ZCFALL_ISS	0	w	**Phase 1 zero crossing falling interrupt status set** 0_B **KEEP**, 1_B **SET**,
PH1_ZCRISE_ISS	1	w	**Phase 1 zero crossing rising interrupt status set** 0_B **KEEP**, 1_B **SET**,
PH2_ZCFALL_ISS	2	w	**Phase 2 zero crossing falling interrupt status set** 0_B **KEEP**, 1_B **SET**,
PH2_ZCRISE_ISS	3	w	**Phase 2 zero crossing rising interrupt status set** 0_B **KEEP**, 1_B **SET**,
PH3_ZCFALL_ISS	4	w	**Phase 3 zero crossing falling interrupt status set** 0_B **KEEP**, 1_B **SET**,
PH3_ZCRISE_ISS	5	w	**Phase 3 zero crossing rising interrupt status set** 0_B **KEEP**, 1_B **SET**,
RES	31:6	r	**Reserved** Always read as 0

Register description BEMFC

BEMF comparator interrupt enable

BEMFC_IRQEN
BEMF comparator interrupt enable **($013C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										PH3_ZCRISE_IEN	PH3_ZCFALL_IEN	PH2_ZCRISE_IEN	PH2_ZCFALL_IEN	PH1_ZCRISE_IEN	PH1_ZCFALL_IEN
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PH1_ZCFALL_IEN	0	rw	**Phase 1 zero crossing falling interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
PH1_ZCRISE_IEN	1	rw	**Phase 1 zero crossing rising interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
PH2_ZCFALL_IEN	2	rw	**Phase 2 zero crossing falling interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
PH2_ZCRISE_IEN	3	rw	**Phase 2 zero crossing rising interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
PH3_ZCFALL_IEN	4	rw	**Phase 3 zero crossing falling interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
PH3_ZCRISE_IEN	5	rw	**Phase 3 zero crossing rising interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
RES	31:6	r	**Reserved** Always read as 0

22 Sigma Delta ADC (SDADC)

22.1 Features overview

There is one Sigma Delta ADC (SDADC) module with two independent channels with input stage, 2nd order modulator, 3rd order CIC filter, result handling and synchronization feature. The SDADC is optimized for the usage of external AMR/GMR/TMR type sensors. The application configuration is shown in **Figure 242**, the block diagram in **Figure 243**.

The SDADC has following features:

- Performance
  - Sampling frequency up to 20 MHz (typ.), *MCLK*, f_S
  - Input frequency of up to 1 kHz (typ.), f_{IN}
  - Linear input range of ±3.75 V (typ.), V_{DIFF_lin}
  - RMS noise of less than 1 mV (typ.), V_{rms}
  - SNDR of 72 dB (typ.), *SNDR*
- Input stages
  - Configurable for differential or single ended input types
  - Two possible inputs selectable for usage of two sensors in time multiplex
  - Offset compensation feature
- Modulator (2nd order type)
  - Normal mode (use modulator and demodulator together)
  - External demodulator mode (modulator's output as alternate function, demodulator external)
  - External modulator mode (modulator bypassed, demodulator inputs via GPIOs)
  - Dither unit for dead zone cancellation and idle tone reduction
- Demodulator (3rd order CIC filter type)
  - Linear programmable decimation factor (DECF) from 16 to 512 with automatic result scaling
  - 16-bit signed filter result (s16 format, internally s29)
  - Two filter modes: continues or triggered (synchronization feature to PWM)
  - Timestamping upon external trigger to capture the age of a result (synchronization feature to PWM)
  - Programmable digital comparator thresholds three modes (range, over-, undervoltage)
- Interrupts, DMA and events
  - SDADC events can be mapped to 2 interrupt node pointers (with 2 IRQ lines)
  - Result events can be mapped to 2 DMA requests
  - Compare events are connected to GPIOs, CCU7 and GPT12

22.2 Block diagram

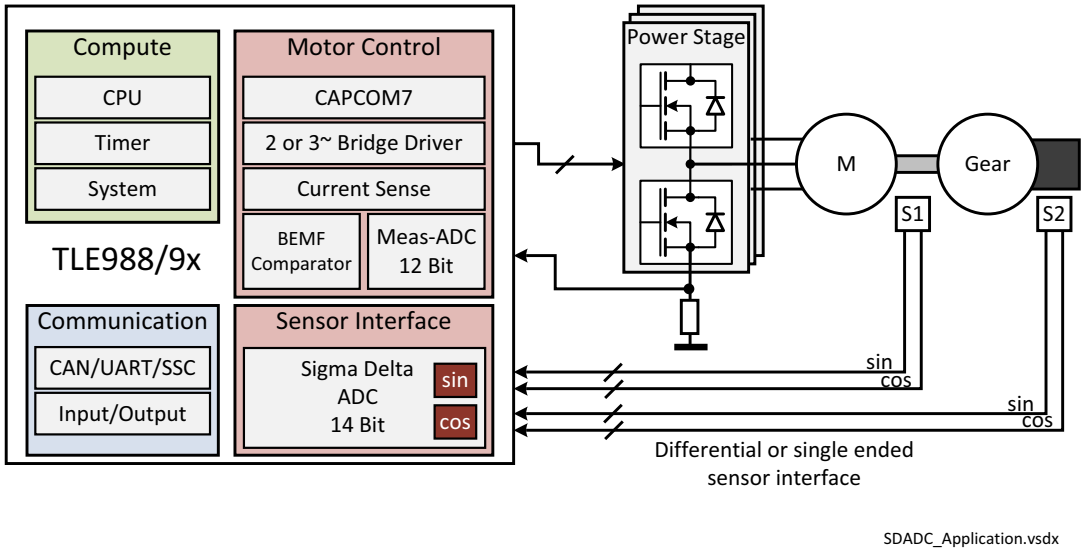


Figure 242 Application diagram SDADC

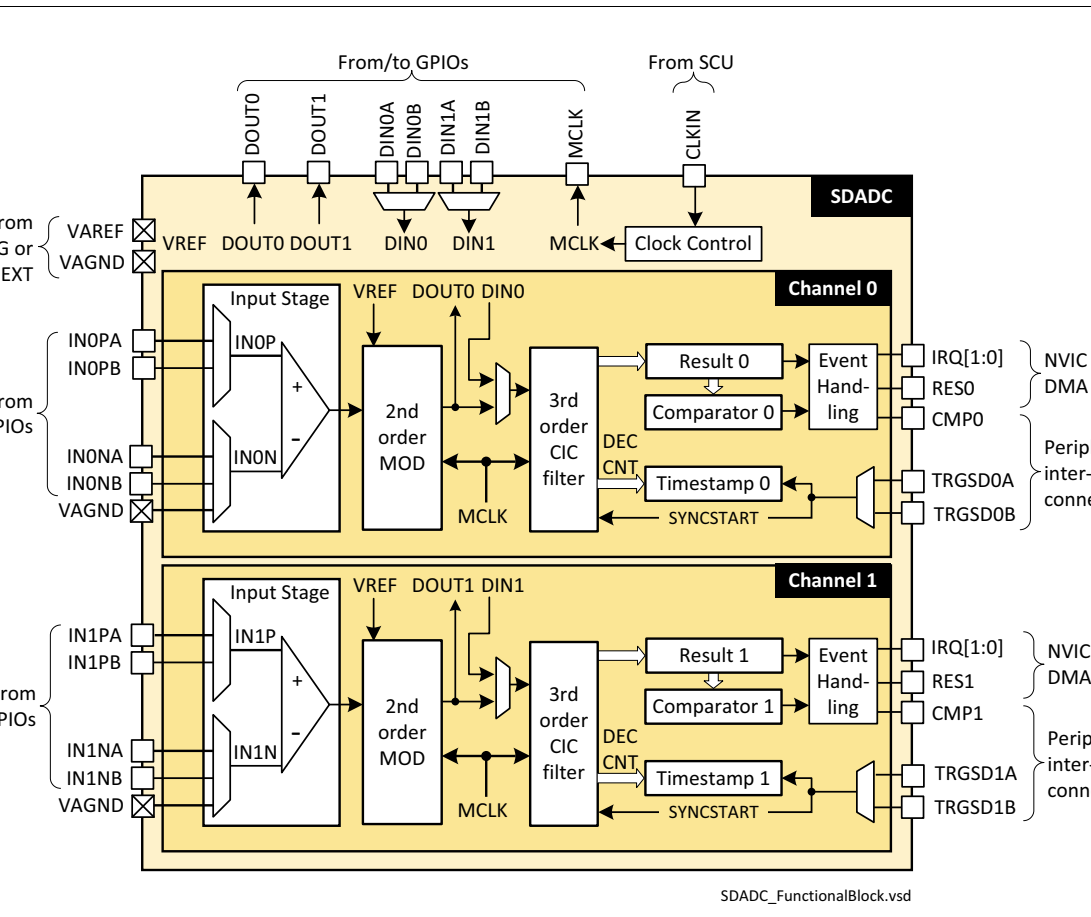


Figure 243 Block diagram SDADC

22.3 Toplevel signals

Table 264 SDADC clock domain

Signal	Direction	Description	From/To
CLKIN	Input	Clock for digital logic	SCU

Table 265 SDADC toplevel block signal connection (see SDADC interconnections)

Signal	Direction	Description	From/To
VAREF	Input	Analog reference voltage	VAREF pin
VAGND	Input	Analog reference ground	VAGND pin
IN0PA	Input	Positive input channel 0 option A	Analog port pin
IN0PB	Input	Positive input channel 0 option B	Analog port pin
IN0NA	Input	Negative input channel 0 option A	Analog port pin
IN0NB	Input	Negative input channel 0 option B	Analog port pin
IN0NC	Input	Negative input channel 0 is grounded	VAGND pin
IN1PA	Input	Positive input channel 1 option A	Analog port pin
IN1PB	Input	Positive input channel 1 option B	Analog port pin
IN1NA	Input	Negative input channel 1 option A	Analog port pin
IN1NB	Input	Negative input channel 1 option B	Analog port pin
IN1NC	Input	Negative input channel 1is grounded	VAGND pin
DIN0A	Input	Data input for CIC filter channel 0 option A	Digital port pin
DIN0B	Input	Data input for CIC filter channel 0 option B	Digital port pin
DIN1A	Input	Data input for CIC filter channel 1 option A	Digital port pin
DIN1B	Input	Data input for CIC filter channel 1 option B	Digital port pin
DOUT00	Output	Modulator data output channel 0 option 0	Digital port pin
DOUT01	Output	Modulator data output channel 0 option 1	Digital port pin
DOUT10	Output	Modulator data output channel 1 option 0	Digital port pin
DOUT11	Output	Modulator data output channel 1 option 1	Digital port pin
MCLK0	Output	Modulator clock output option 0	Digital port pin
MCLK1	Output	Modulator clock output option 1	Digital port pin
CMP0	Output	Comparator output channel 0	GPT12/T2/T21
CMP1	Output	Comparator output channel 1	GPT12/T2/T21
TRGSDA	Input	External trigger option A for timestamp or syncstart	CCU7
TRGSDB	Input	External trigger option B for timestamp or syncstart	CCU7
AHB	Bidirectional	Bus interface	CPU

Table 266 SDADC event connection

Event	Direction	Description	From/To
IRQ[1:0]	output	interrupt service request event	NVIC
DMA[1:0]	output	DMA service request	DMA

22.4 Interrupts

Events

The SDADC provides following events:

- Result event **IS**.RESx_IS (x = 0, 1)
- Compare event **IS**.CMPx_UP_IS, **IS**. CMPx_LO_IS(x = 0, 1)

The events are propagated to the SCU, DMA and other peripherals.

Interrupts and DMA handling

There are two interrupt node pointers which can be assigned to the SDADC events via a node pointer scheme, see **Figure 244**.

Following events can be mapped to Interrupt outputs with **INP**:

- Result event **IS**.RESx_IS (x = 0, 1)
- Compare event **IS**.CMPx_UP_IS, **IS**. CMPx_LO_IS(x = 0, 1)

Following events can be mapped to DMA outputs:

- Result event **IS**.RESx_IS (x = 0, 1)

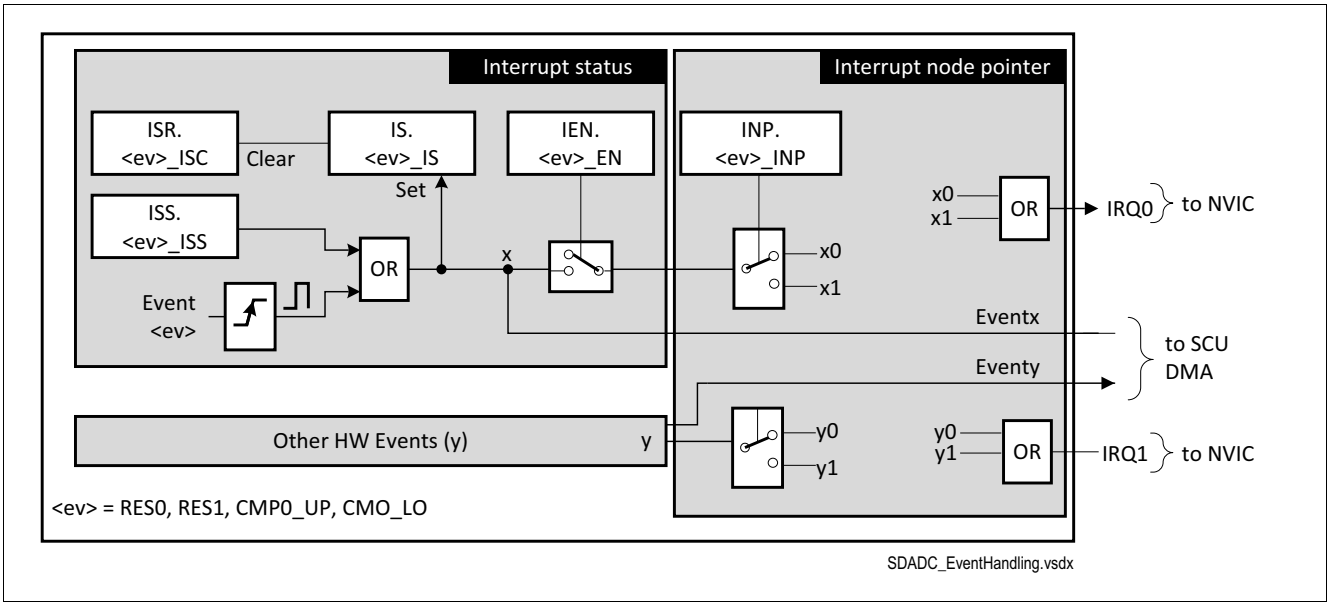


Figure 244 Interrupt and DMA handling

Status handling

Following events implement the status handling, see **Figure 245**:

- Compare event **STS**.CMPx_UP_STS, **STS**. CMPx_LO_STS(x = 0, 1)
- Wait for read Error **STS**.WFRx_STS (x=0,1)

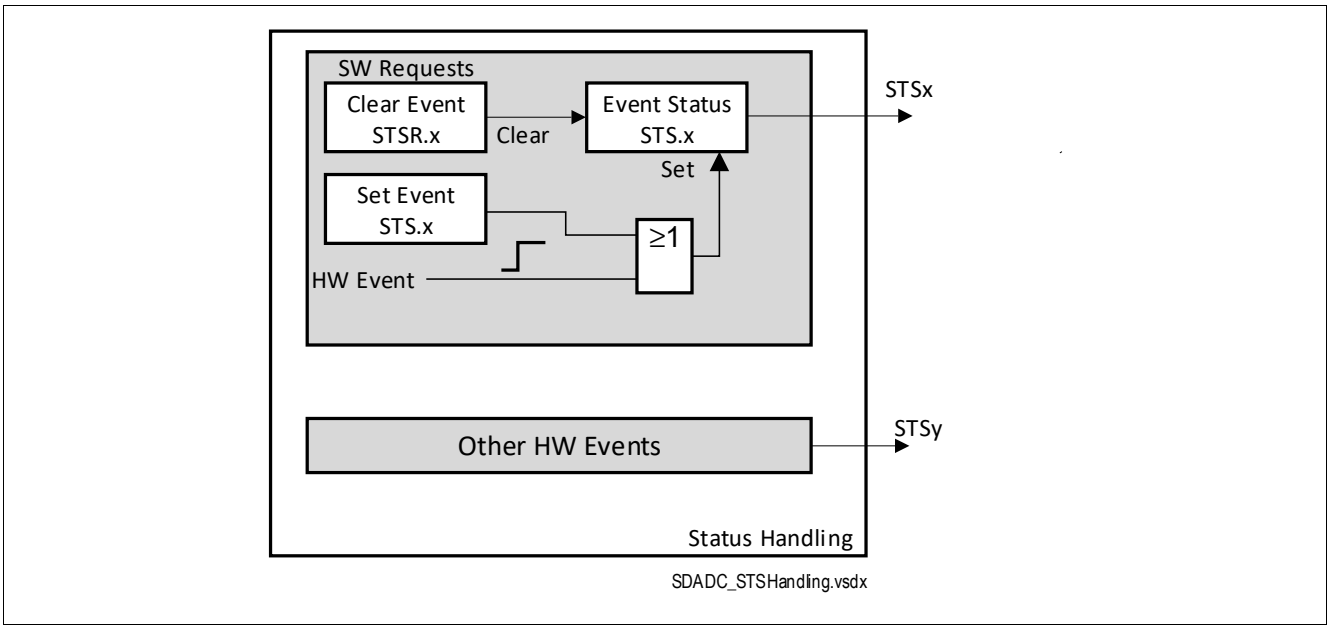


Figure 245 Status handling

22.5 Operation mode behavior

- The SDADC module can enabled disabled with **CFG0**.ENx (x=0,1) in active mode
- The SDADC module is powered off in Stop Mode, Sleep Mode and Fail Sleep Mode

Table 267 Operation mode behavior SDADC

Reset	The SDADC module is reset via RESET_TYPE_3. An activated reset has following effects: • All SFRs of the SDADC are reset • The SD modulator is set to inactive
Power-up/ Power-down	At power-up or power-down the SDADC has following behavior: • The SDADC is kept in reset state by the module reset as long as the supply voltages are not in specified operating range • Power-up: the SDADC is released from reset state by the module reset when the supply voltages are in the specified operating range. The analog parts power up and its biasing starts. The analog module is in operating state after the power up time (t_{up}) from RESET inactive. The module is disabled by default • Power-down: the SDADC is set into reset state by the module reset when the supply voltages are not in the specified operating range. The analog parts power down and its biasing is disabled • At power down the SDADC analog parts power down and its biasing is disabled
Active mode	In active mode the SDADC has following behavior: • Once the SDADC module is enabled the module clock starts • Once the SDADC module is disabled the module clock stops and the module reset is asserted
Stop mode	In stop mode the SDADC has following behavior: • When entering stop mode the SDADC is powered down automatically (see power-down behavior), all information is lost • When leaving stop mode the SDADC powers up automatically (see power-up behavior)
Sleep mode/ Fail-sleep mode	In sleep mode the SDADC has following behavior: • When entering sleep mode the SDADC is powered down automatically (see power-down behavior), all information is lost • When leaving sleep mode the SDADC powers up automatically (see power-up behavior)

22.6 Clock control

The modulator and digital filter run with the same sample frequency f_S (MCLK) which is derived from the module clock f_{SDADC} (CLK_SDADC).

There is a clock control with programmable prescaler, see register **CFG0**.PRE.

It is not recommended to disable the module (**CFG0**.ENx) while the clock is gated off.

The module clock (mod_clk) of the SDADC can be programmed with **CFG0**.PRE

22.7 Modulator channels

There are two 2nd order sigma delta modulators (channel 0 and channel 1). Each channel converts an analog input signal into a digital bitstream, see **Figure 246**.

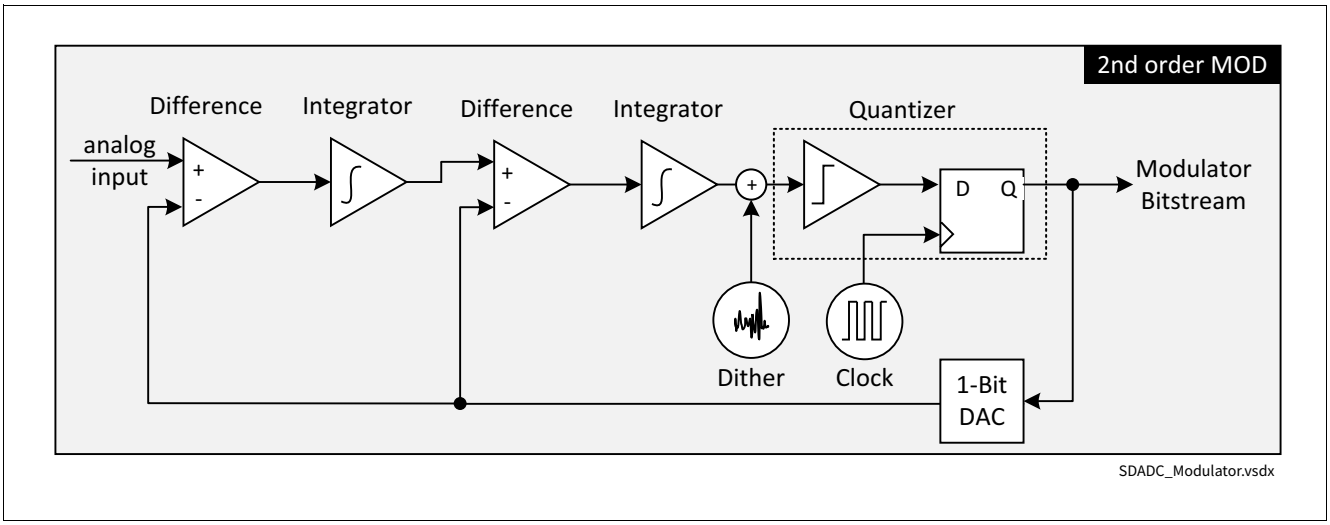


Figure 246 Principle of a 2nd order modulator

22.7.1 Reference selection

The modulators are referenced to the VAREF pin voltage.

The reference selection is done outside the SDADC. The references can be selected from following sources:

- VDDEXT for ratiometric measurement when used as sensor supply and reference
- VREF5V for supply independent measurement

22.7.2 Modulator inputs

Each channel has one pair of fully differential capacitive inputs.

Each channel has an input gain (see *GAIN*) which adapts the differential input range to the reference voltage.

Each input has a multiplexer for alternate port pin selection (**INSEL**.INxPSEL/**INSEL**.INxNSEL), see **Figure 247**.

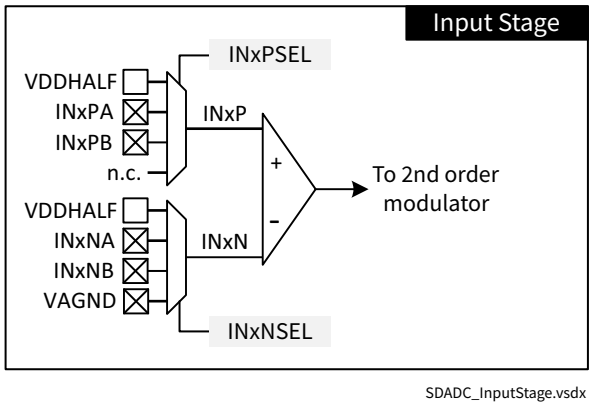


Figure 247 Input stage and selection

22.7.3 Differential and single ended mode

The inputs can be used in differential and single ended mode. In both modes the comparator compares the difference between the positive input INxP to the negative INxN.

The differential mode has following options and parameters:

- The negative input (INxN) can be selected via **INSEL**.INxNSEL
- The positive input (INxP) can be selected via **INSEL**.INxPSEL

The single ended mode has following options and parameters:

- The negative input (INxN) has to be connected to VAGND via **INSEL**.INxNSEL
- The positive input (INxP) can be selected via **INSEL**.INxPSEL

22.7.4 Input ranges

The modulators have two input ranges (differential voltage and single ended voltage)

- The differential voltage ($V_{DIFF} = V_{INxP} - V_{INxN}$) has to be within a symmetrical range around the common mode voltage $V_{IN,com}$. Within the linear range ($V_{DIFF, lin}$) the modulator's output is linear to the input, whereas it is nonlinear in the nonlinear range ($V_{DIFF, nonlin}$). Outside the nonlinear range the modulator saturates. The differential voltage V_{DIFF} can have double the amplitude of its input signals V_{INxP} and V_{INxN}. This results in a result range which covers positive and negative values.
- The single ended voltage ($V_{SNGL} = V_{INxP}$) has to be within a range. Within the linear range ($V_{SNGL, lin}$) the modulator's output is linear to the input, whereas it is nonlinear in the nonlinear range ($V_{SNGL, nonlin}$). Outside the nonlinear range the modulator saturates. The single ended voltage V_{SNGL} has the amplitude of V_{INxP}. This results in a result range which covers only positive values, i.e. the single ended mode has only half of the resolution compared to the differential mode.

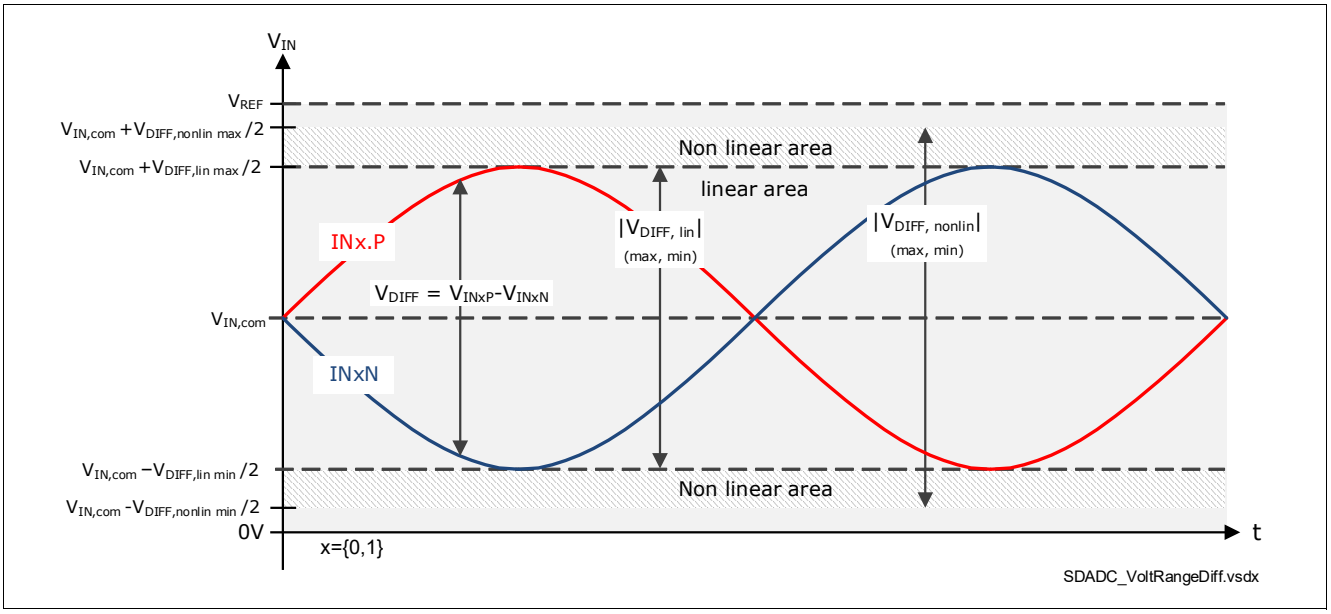


Figure 248 Input voltage range for differential mode

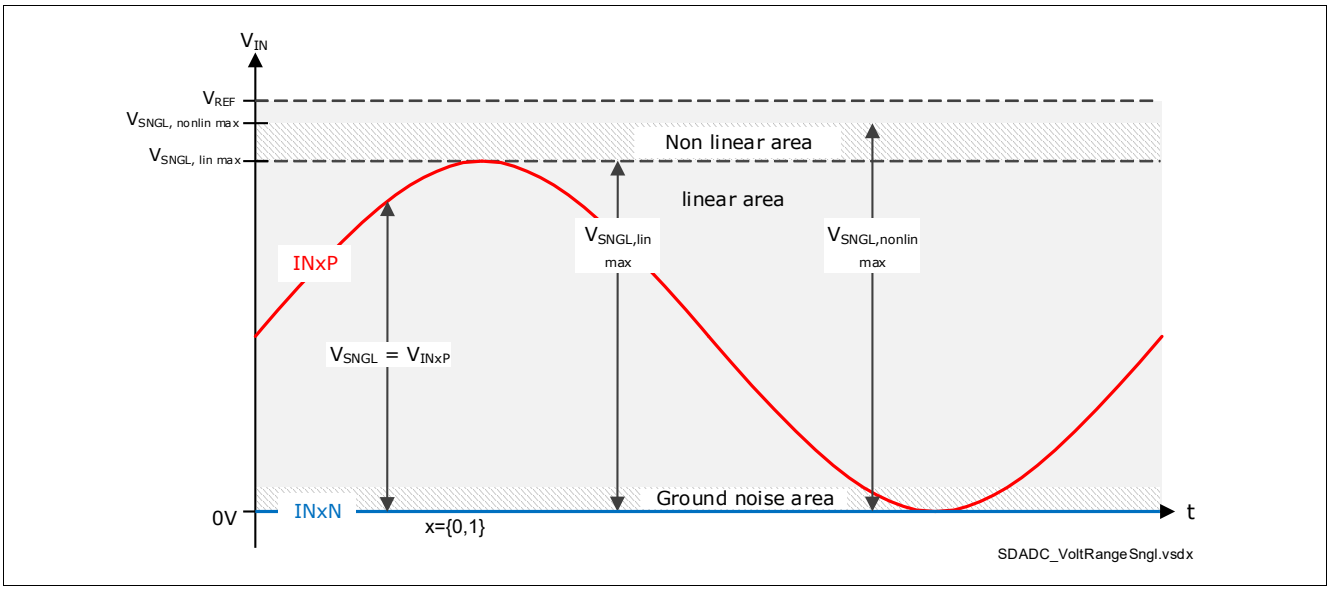


Figure 249 Input voltage range for single ended mode

22.7.5 Offset compensation

The offset compensation can be achieved by connecting the positive (INxP) and negative input (INxN) to the common mode voltage via selecting “VDDHALF” via **INSEL**.INxPSEL and **INSEL**.INxNSEL (x=0,1), see **Figure 247**. Via this configuration the differential input voltage is zero and the filtered output bit stream of the modulator represents the offset.

22.7.6 Dither unit

The dither unit adds a pseudo random noise to the input of the quantizer within the modulator, see **Figure 246**. This is useful for two cases:

- Cancel dead zones if the input signal frequency is very low
- Reduce idle tones if the input signal level is an integer ratio of the reference voltage

The dither function is enabled in SFR CFG0.DITHEREN.

The dither voltage is selected in DITHCFG.DITH_VOLT_SEL.

(22.1)

$$\mathrm{Tdith} = (2^{20} - 1) \times \mathrm{Tmclk}$$

Note: The dithering function reduces the SNDR performance.

22.8 External modulator mode

In some application cases the sensor element has a different ground potential than the computing element. In this case an external modulator can be used with an isolated digital interface to the TLE988/9x.

Therefore modulator channels can be bypassed. In this case the bitstream of an external modulator can be directly routed to the CIC filter. The external sensor has to be clock slave, i.e. the clock is provided via MCLK.

In bypass mode the data for the CIC filter is taken from DIN0/1 via **INSEL**.DINxSEL (x= 0, 1).

The clock for the external modulator is MCLK, which is available on port pins via alternate function selection of MCLK0/1.

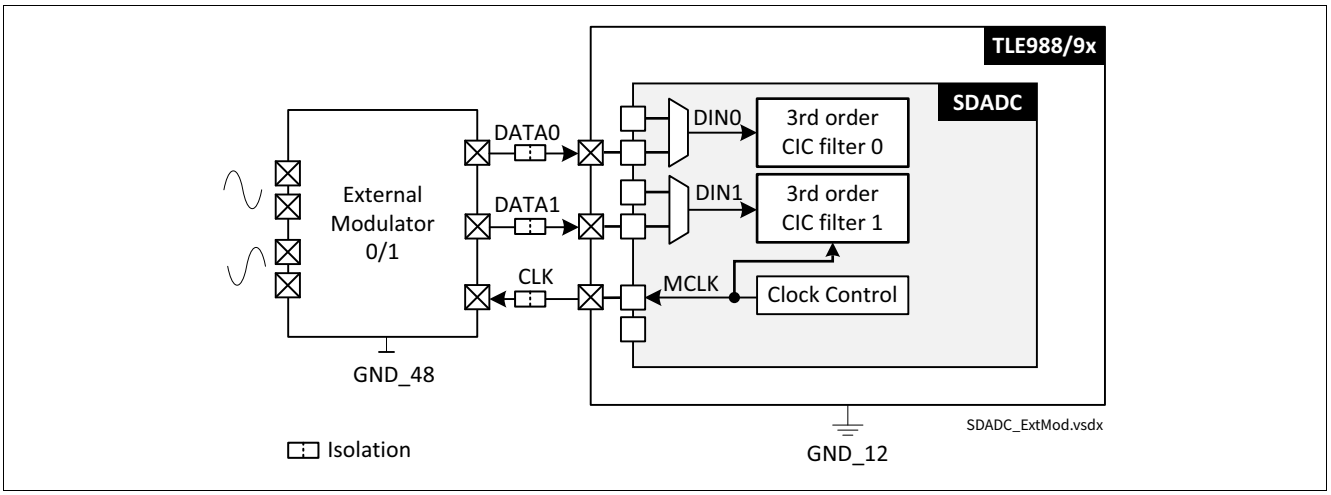


Figure 250 External modulator mode

22.9 External demodulator mode

In some application cases the sensor element has a different ground potential than the computing element. In this case an external demodulator can be used with an isolated digital interface to the TLE988/9x.

Therefore the modulator bit stream can be routed to the port pins, an external demodulator (filter) can be used:

- Data outputs are available on port pins via alternate function selection (**ALTSEL**) of DOUTxy (x for the channel number, y for the option).
- The modulator clock is available on the port pins via alternate function selection (**ALTSEL**) of MCLK0/1.

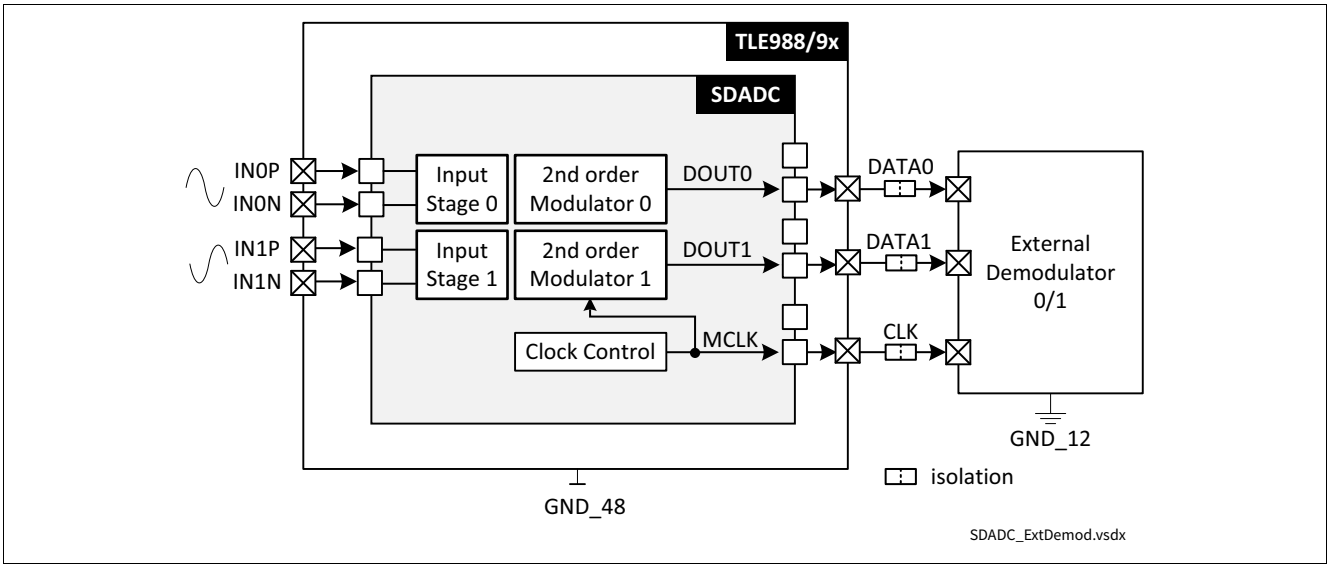


Figure 251 External demodulator mode

22.10 Demodulator channels (digital filter)

There are two 3rd order CIC (Cascaded-Integrator-Comb filter, aka. SINC3) filter which demodulate the modulator's bitstream. There is one CIC filter for each modulator channel with following features.

22.10.1 Programmable decimation

Each filter has a programmable decimation counter, which defines the decimation factor (DECFx). The decimation can be programmed in linear steps from 16 to 512 in register **CFG1**.DECFx (x= 0, 1).

The decimation counter counts down from the value DECFx. When the counter underflows a new result is transferred to the RESULTx and the RESx event is generated (x = 0, 1).

22.10.2 Filter start

The two filter channels can work synchronously (same DECF, same starting point) or independently (different DECF, different starting point).

The filter for the channels can be enabled by software via setting bit **CFG0**.ENx (x = 0, 1).

There are two filter modes which can be selected in **CFG0**.CICMODEx (x=0,1):

- CIC Filter continues mode: **CFG0**.CICMODEx = 0, the filter continuously runs once **CFG0**.ENx =1 (x = 0,1)
- CIC Filter triggered mode: **CFG0**.CICMODEx = 1, the filter channel x starts with the external event selected by **INSEL**.TRGSDxSEL and stops automatically after three decimation counter overflows (which equals a valid SINC3 result), see figure **Figure 252**. This allows a sampled conversion synchronized to another timebase, e. g. a PWM period.

Note: A started filter cannot be restarted.

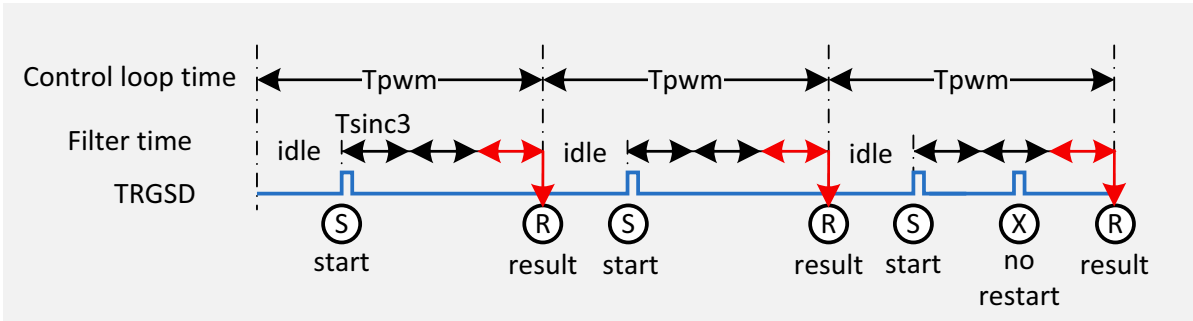


Figure 252 Sychronized filter start by signal TRGSTARTx

22.10.3 Filter time

The filter settling time, which is needed to remove the full filter history is calculated according to following formula:

(22.2)

$$\text{filter settling time} = [3 \times (\text{DECFx} + 1) + 3] \times \text{TMCLK}$$

The filter group delay time, which is the delay from a change of the input signal visible at the output signal is calculated according following formula:

(22.3)

$$\text{group delay time} = 1.5 \times (\text{DECFx} - 1) \times \text{TMCLK}$$

22.11 Result generation

22.11.1 Result event

Following software actions are possible on the result event flag:

- Event enable, **IEN**.RESx_EN (x = 0, 1)
- Event status read, **IS**.RESx_IS (x = 0, 1)
- Event status set, **IS**.RESx_IS (x = 0, 1)
- Event status clear, **ISR**.RESx_ISC (x = 0, 1)

The result events are available as interrupt request and DMA request according to the scheme in **Figure 244**.

22.11.2 Filter output word size

The output word is generated in the Result Generation block, see **Figure 253**. The number of significant bits in the output word of the CIC filter depends on the decimation factor setting. The output word can be adapted to the user defined presentation. Every decimation counter underflow a new result is generated.

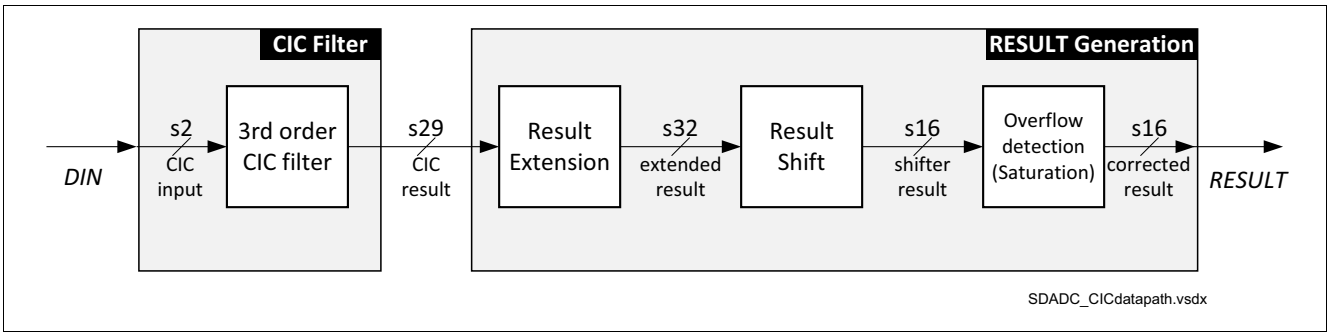


Figure 253 Data path CIC and result generation

For a SINC3 filter type the raw output word size is calculated according to following formula:

(22.4)

$$\text{raw filter output word size} = 3 \times \log 2(\text{DECFx} + 1) + 1 + 1(\text{sign})$$

The maximum selectable decimation factor (DECF = 512) results in a filter output word size of 29 bits. The effective resolution of the modulator is lower, therefore the width of the result register is 16 bits.

22.11.3 Result scaling

Ideally the result range is independent of the decimation factor. This means that for a given input voltage measured with different decimation factors the result range (visible on the higher significant bits) stays the same. Only the result resolution (visible on the lower significant bits) is higher with the higher decimation factor.

The internal result is 32 bit wide. Consisting of the 29 bit raw filter result of the CIC filter plus 3 extension bits.

The 16 bit result register readable by the CPU is provided out of this 32 bit internal register.

Which part of the internal result register is taken for the 16 bit result register is defined by **CFG1**.RESSHIFTx (x= 0,1). See **Figure 254**.

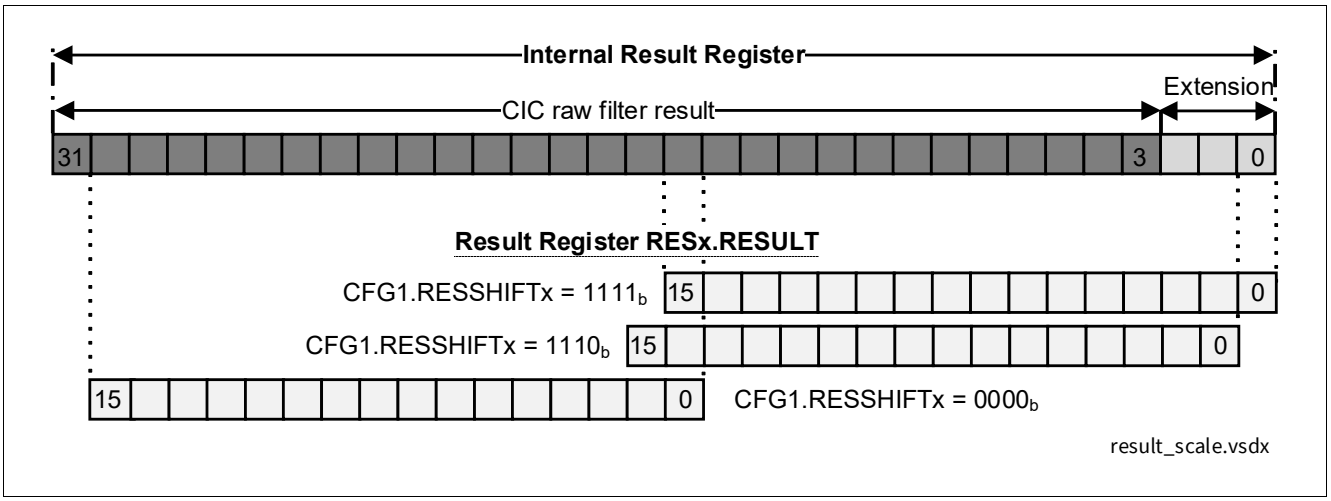


Figure 254 Result register scaling

To prevent the result register from overflowing there is a certain dependency of DECFx and the RESSHIFTx which needs to be respected. The recommended settings for DECF, RESSHIFTx are shown in **Table 268**.

Table 268 Dependency DECF and RESSHIFT

DECF	RESSHIFT
16	15
17-20	14
21-25	13
26- 32	12
33 - 40	11
41 -50	10
51 - 64	9
65 - 80	8
81 -101	7
102 - 128	6
129 - 161	5
162 - 203	4
204 - 256	3
257 - 322	2
323 - 406	1
407 - 512	0

(22.5)

$$CFG1.RESSHIFTx = 27 - round\_up(3 \times log2(CFG1.DECFx + 1))$$

22.11.4 Result calculation

The result generation offers the result in following format:

- 16 bit two's complement (left aligned, sign bit is bit 15)
- The result is calculated according to formula 1.6:

(22.6)

$$result = \frac{V_{DIFF,lin}}{V_{AREF}} \times GAIN \times (2^{14} - 1)$$

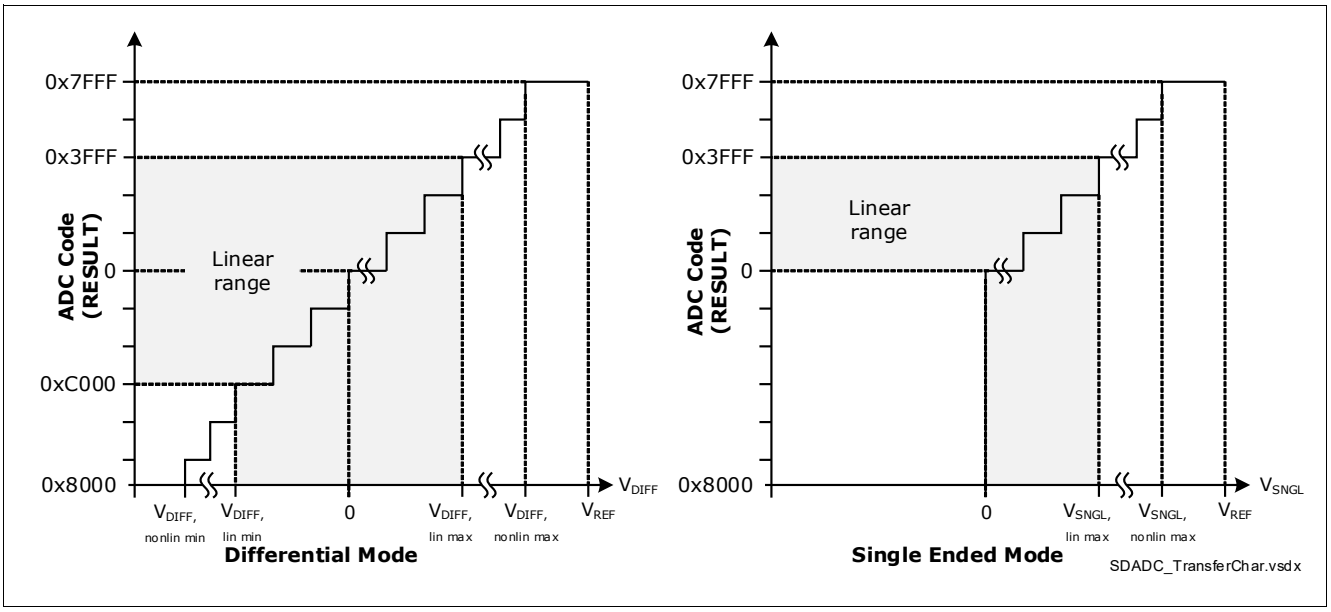


Figure 255 Transfer characteristic

- The result is written to RESx.RESULT (x = 0, 1).
- Both result register can be read with one CPU/DMA access (32 bit).

22.11.5 Correction factor for linear decimation values

In order to adapt the filter time to a fixed timebase (e.g. PWM period) it is possible to program the DECF in linear steps (e.g. DECF = 125 instead of 128 = 2^7). As the result shifter can only shift in power of two steps, the result must be corrected by user software. The correction factor is deterministic and can be calculated according to following formula:

(22.7)

$$Result_{corrected} = Result_{uncorrected} \times Gain_{corrected}$$

(22.8)

$$GAIN_{correction} = \frac{MAX\_VAL}{round\_up((CFG1.DECFx + 1)^3 * 2^{CFG1.RESSHIFTx-12})} \quad *)$$

*) set CFG1.RESSHIFTx value according to (21.5)
condition: $2^{15} >= (CFG1.DECFx + 1)^3 * 2^{CFG1.RESSHIFTx-12}$

where ±MAX_VAL are the boundaries of the desired output range. For example, if $Result_{corrected}$ is a 16 bit value (two's complement) then MAX_VAL should be equal to 32767. In this scenario $Result_{uncorrected}$ is mapped in the range of ±32767.

Setting CFG1.RESSHIFTx with a value that differs from the one obtained with the **Equation (22.5)** might lead to saturation effects in $Result_{uncorrected}$. This phenomena could occur if the condition of **Equation (22.8)** is not verified. In such a scenario, it is recommended to set the suggested CFG1.RESSHIFTx value or at least its value if the output exhibits a saturation.

DECF	SINC3-Amplitude	Number of Bits	RESSHIFT	Shifted Amplitude	GAINCORR
16	4096	12	15	32768	1,0000
17	4913	13	14	19652	1,6674
20	8000	13	14	32000	1,0240
31	29791	15	12	29791	1,0999
32	32768	15	12	32768	1,0000
33	35937	16	11	17969	1,8235
45	91125	17	10	22782	1,4383
62	238328	18	9	29791	1,0999
63	250047	18	9	31256	1,0483
64	262144	18	9	32768	1,0000
65	274625	19	8	17165	1,9089
70	343000	19	8	21438	1,5285
80	512000	19	8	32000	1,0240
100	1000000	20	7	31250	1,0485
127	2048383	21	6	32006	1,0238
128	2097152	21	6	32768	1,0000
129	2146689	22	5	16772	1,9537
256	16777216	24	3	32768	1,0000
512	134217728	27	0	32768	1,0000

Figure 256 Example several GAINCORR factors for different DECF

22.11.6 Wait for read mechanism

The wait for read mechanism protects valid results from being overwritten before read. It optionally discards a new filter result in case the old filter result register is not yet read by CPU/DMA (RESx.RESVALID = 1).

The wait for read is configured in **CFG0**.WFRENx (x=0,1).

Once a result is discarded it is displayed in **STS**.WFRx_STS. Every time a result is discarded the RESx.VALCNT is increased. This allows to calculate the age of the last read result.

A set valid flag indicates that the result register RESx.RESULT was updated from the hardware since the last read of the result(x= 0,1).

A cleared valid flag indicates that there was no update of the result register RESx.RESULT since the last read of the result.

22.11.7 Initial result latency of a SINC3 filter

The SINC3 filter has an initial result latency. Once the filter clock is enabled the first reasonable result is available after three decimation filter times. The first two filter results contain unwanted filter history, only the third filter result is a meaningful.

22.12 Timestamp

The timestamp value is 9 bit wide and can be read from register RESx.TIMVAL and CTIMx.TIMVAL (x=0, 1).

Each filter channel offers two timestamp values:

- The actual value of the decimation counter value within the result Register (RESx.TIMVAL) (x= 0,1). Also within the RESx Register there is the VALCNT which represents the number of discarded result updates in case the wait for read feature is activated. With the 2 bit fields TIMVAL and VALCNT it is possible to determine the age of the read RESULT within each readout. A value of 7 of the VALCNT will indicate an overflow (result is too old for further processing). The VALCNT is cleared when the RESx Register is updated.
- A captured value from the decimation counter value. The capturing will be done from the decimation counter based on the rising edge of the external event TRGTMPx (x= 0,1) into CTIMx.TIMVAL (x= 0,1). In this way the age of a filter result related to an external timebase is available.

Both versions of the timestamp value can be used to implement a software extrapolation algorithm.

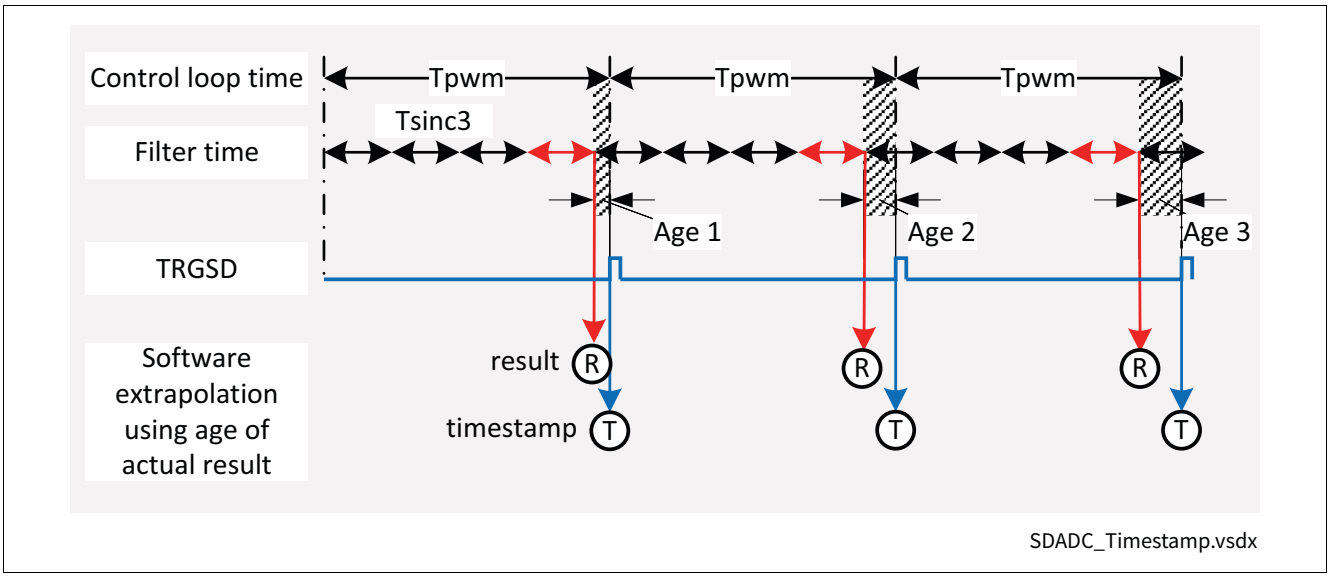


Figure 257 Timestamping upon an external trigger

22.13 Coherent read of result information

In order to allow a coherent read of all result information (result values of both channels in different views, timestamp and valid information) it is recommended to use the wait for read mechanism.

All registers are in a regular order and can be read with one DMA transfer. It has to be noted that a DMA transfer takes several clocks and data coherency cannot be guaranteed if wait for read mechanism is not used.

22.14 Digital comparators

There are two digital result comparators with programmable compare values for limit checking:

- The result comparators have an upper and a lower threshold with 9 bit width each: CMPx_CTRL.UPPER, CMPx_CTRL.LOWER (x = 0, 1)
- The interpretation of the CMPx_CTRL.UPPER and CMPx_CTRL.LOWER values is signed.(x = 0, 1)
- The CMPx_CTRL.UPPER and CMPx_CTRL.LOWER values are compared aligned to the MSB of RESx.RESULT (E.g.: RESULTx [15:7] > UPPERx [8:0] respecting the signed notation; RESULT[6:0] are cut off for the comparison) (x = 0, 1)

- The result comparators have an hysteresis setting for the upper and lower compare values: CMPx_CTRL.HYST_UP, CMPx_CTRL_HYST_LO (x = 0, 1)
- The result comparators provide a compare output for each compare threshold: UPx, LOx (x= 0,1)
- The compare mode for the upper and lower result compares can be configured: CMPx_CTRL.MODE (x= 0,1)

There are 3 modes available (see also **Figure 259**):

1. Range (range control)

Compare output	Compare output actual value	Comparison	
UPx	set	RESx.RESULT > CMPx_CTRL.UPPER	x= 0,1
	clear	RESx.RESULT ≤ (CMPx_CTRL.UPPER - CMPx_CTRL.HYST_UP)	x= 0,1
LOx	set	RESx.RESULT < CMPx_CTRL.LOWER	x= 0,1
	clear	RESx.RESULT ≥ (CMPx_CTRL.LOWER + CMPx_CTRL.HYST_LO)	x= 0,1

2. Overvoltage mode

Compare output	Compare output actual value	Comparison	
UPx	set	RESx.RESULT > CMPx_CTRL.UPPER	x= 0,1
	clear	RESx.RESULT ≤ (CMPx_CTRL.UPPER - CMPx_CTRL.HYST_UP)	x= 0,1
LOx	set	RESx.RESULT > CMPx_CTRL.LOWER	x= 0,1
	clear	RESx.RESULT ≤ (CMPx_CTRL.LOWER - CMPx_CTRL.HYST_LO)	x= 0,1

Undervoltage mode

Compare output	Compare output actual value	Comparison	
UPx	set	RESx.RESULT < CMPx_CTRL.UPPER	x= 0,1
	clear	RESx.RESULT ≥ (CMPx_CTRL.UPPER + CMPx_CTRL.HYST_UP)	x= 0,1
LOx	set	RESx.RESULT < CMPx_CTRL.LOWER	x= 0,1
	clear	RESx.RESULT ≥ (CMPx_CTRL.LOWER + CMPx_CTRL.HYST_LO)	x= 0,1

The 2 compare outputs can be logical combined to a combined compare output: UPLOx (x = 0, 1).

Following assignments are possible for the combined output and are controlled by CMPx_CTRL (x= 0,1):

- UPLOx = UPx OR LOx

- UPLOx = UPx XOR LOx
- UPLOx = UPx
- UPLOx = LOx

The compare events are available for following software actions:

- Event enable, **IEN**.CMPx_UP_EN, CMPx_LO_EN (x = 0, 1)
- Event status read/set, **IS**.CMPx_UP_IS, **IS**.CMPx_LO_IS(x = 0, 1)
- Event status clear, **ISR**.CMPx_UP_IS, **ISR**.CMPx_LO_IS (x = 0, 1)

The compare events can be mapped to the two service request nodes SDADC.IRQ[1:0] via **INP**.CMPx_UP_INP, **INP**.CMPx_LO_INP (x = 0, 1).

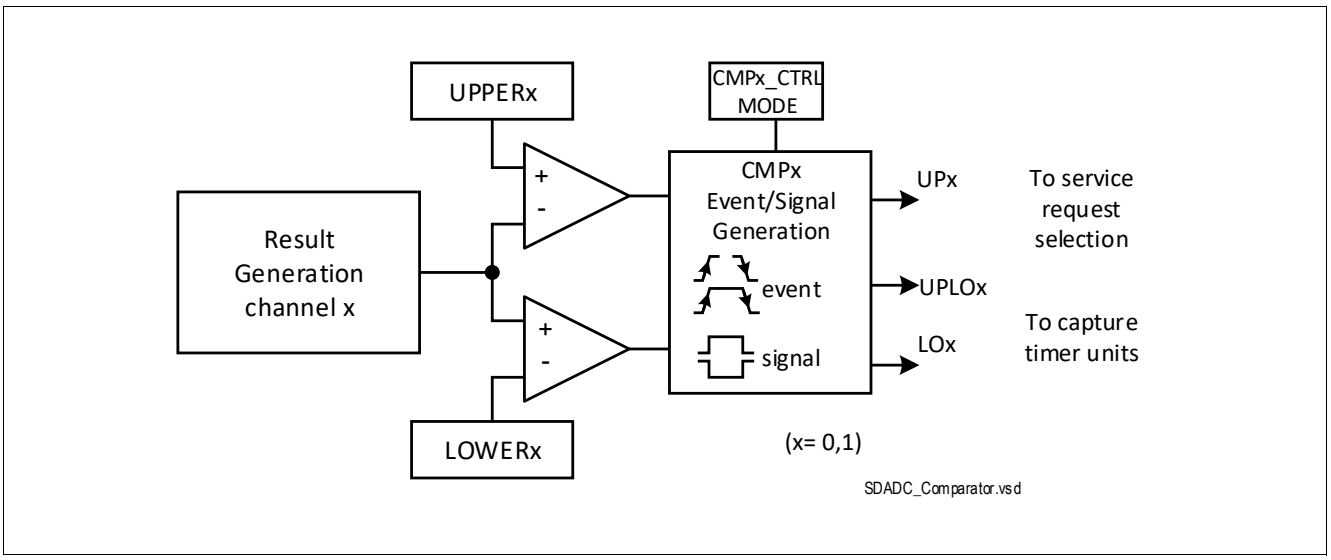


Figure 258 Comparator block diagram

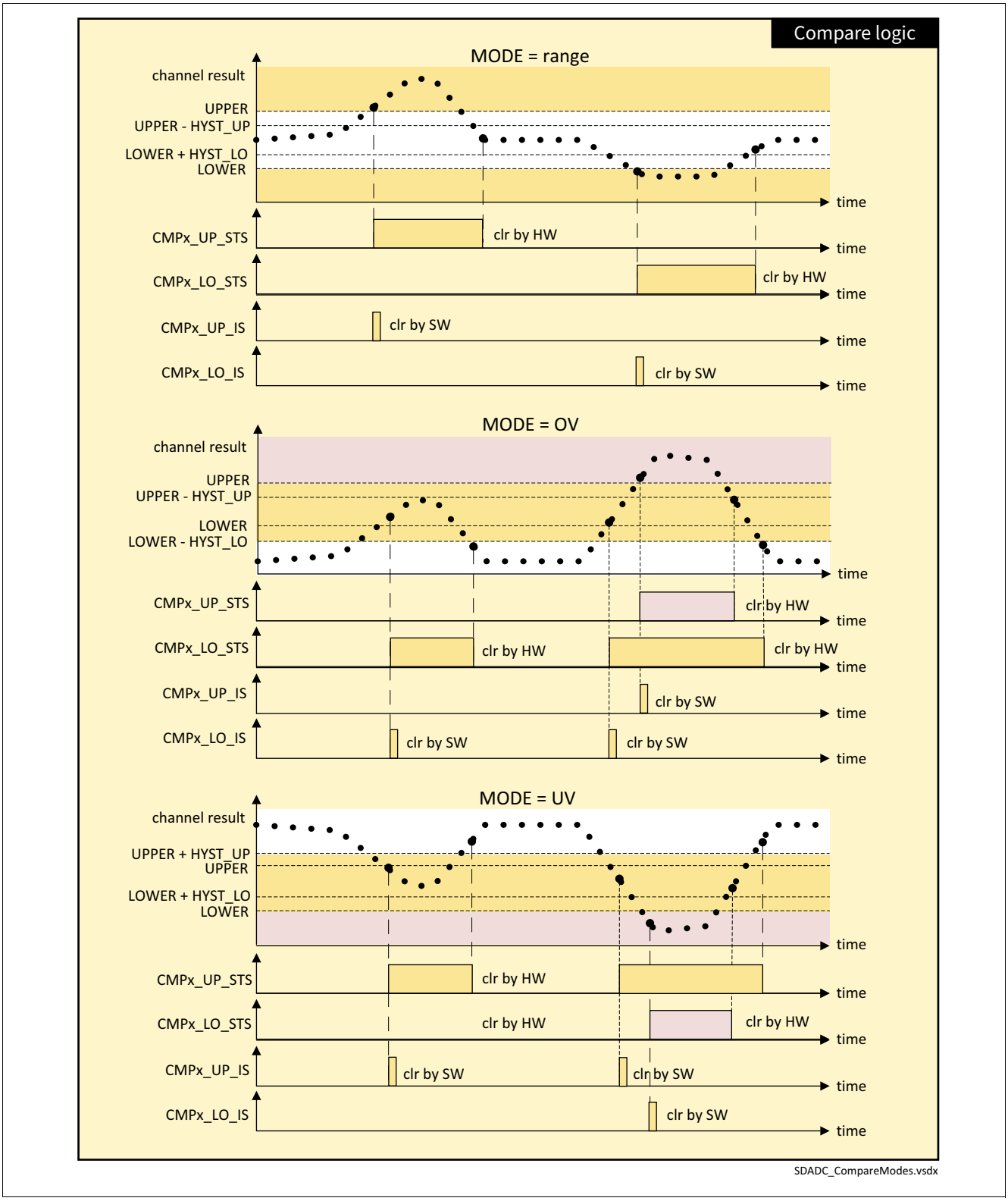


Figure 259 Comparator use cases

22.15 Suspend mode

For debugging purpose the module can be suspended. In suspend mode the clock to the demodulator channels is halted but the bus access is possible and debugger can read the registers.

The suspend mode is requested from the CPU by the external input mod_susp_i (active high).

The suspend mode must be enabled with SUSCTR.EN in order to be effective.
There is only one suspend mode available. In case the suspend event occurs the clock is halted.
The suspend status can be read from SUSCTR.STAT.
It is not recommended to disable the module (**CFG0**.ENx) while suspend mode is active.

22.16 Register description SDADC

22.16.1 SDADC Address Maps

Table 269 Register Address Space - SDADC

Module	Base Address	End Address	Note
SDADC	40004000_H	$40007FFF_H$	

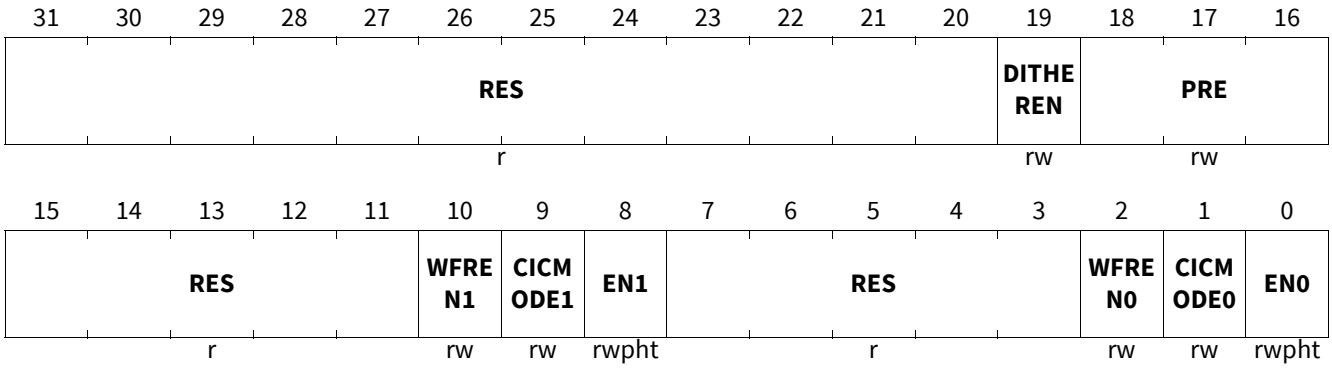
Table 270 Register Overview - SDADC (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CFG0	Configuration Register 0	0000_H	861
CFG1	Configuration Register 1	0004_H	862
ALTSEL	Alternate function selection	0008_H	863
INSEL	Input Port Selection	$000C_H$	864
IEN	Interrupt Enable Register	0010_H	865
IS	Event Register	0014_H	866
ISR	Event Clear Register	0018_H	867
ISS	Event Set Register	$001C_H$	868
INP	Interrupt Node Pointer	0020_H	869
SUSCTR	Suspend Control	0024_H	870
SUSSTAT	Suspend Status	0028_H	870
CMP0_CTRL	Compare Channel 0 Control Register	$002C_H$	871
CMP1_CTRL	Compare Channel 1 Control Register	0030_H	872
RES0	Result Register channel 0	0034_H	873
RES1	Result Register channel 1	0038_H	873
CTIM0	Captured Timestamp channel 0	$003C_H$	874
CTIM1	Captured Timestamp channel 1	0040_H	874
DITHCFG	Dither Configuration Register	0044_H	875
STSR	Status Clear Register	$004C_H$	875
STS	Status Register	0050_H	876
STSS	Status Set Register	0054_H	877

22.16.2 SDADC Registers

Configuration Register 0

CFG0
Configuration Register 0 **(0000_H)** **RESET_TYPE_5 Value: 0000 0000_H**



Field	Bits	Type	Description
EN0	0	rwpht	**Enable Channel 0** 0_B **Disable**, Channel 0 is disabled 1_B **Enable**, Channel 0 is enabled
CICMODE0	1	rw	**CIC Filter Mode channel 0** 0_B **CONT**, Continuous Mode - CIC Filter is clocked when EN0 = 1 1_B **TRIG**, Triggered Mode - CIC Filter starts with External Event selected by TRGSDxSEL
WFREN0	2	rw	**Wait for Read of Result Register Channel 0** 0_B **Disable**, Result Register is overwritten regardless if it was read 1_B **Enable**, Wait for read is enabled; Result Register is not overwritten until its read
RES	7:3, 15:11, 31:20	r	**Reserved - 0** Read as 0; should be written with 0.
EN1	8	rwpht	**Enable Channel 1** 0_B **Disable**, Channel 1 is disabled 1_B **Enable**, Channel 1 is enabled
CICMODE1	9	rw	**CIC Filter Mode channel 1** 0_B **CONT**, Continuous Mode - CIC Filter is clocked when EN1 = 1 1_B **TRIG**, Triggered Mode - CIC Filter starts with External Event selected by TRGSDxSEL
WFREN1	10	rw	**Wait for Read of Result Register Channel 1** 0_B **Disable**, Result Register is overwritten regardless if it was read 1_B **Enable**, Wait for read is enabled; Result Register is not overwritten until its read
PRE	18:16	rw	**Prescaler for Modulator and CIC clock** 000_B **DIV1**, fsdadc / 1 … 111_B **DIV8**, fsdadc / 8

Register description SDADC

Field	Bits	Type	Description
DITHEREN	19	rw	**Dithering of SD ADC Clock for channel 0/1** 0_B **Disable**, Dithering disabled 1_B **Enable**, Dithering enabled

Configuration Register 1

CFG1
Configuration Register 1 **(0004_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESSHIFT1				**RES**			**DECF1**								
rw				r			rw								

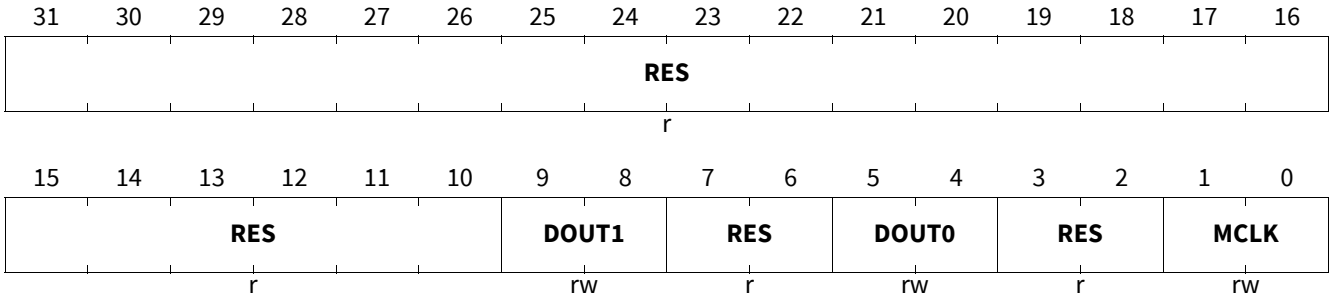
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESSHIFT0				**RES**			**DECF0**								
rw				r			rw								

Field	Bits	Type	Description
DECF0	8:0	rw	**Decimation Factor channel 0** 000_H **DECF**, 16 (not used) … $00E_H$ **DECF**, 16 (not used) 010_H **DECF17**, 17 … $1FF_H$ **DECF512**, 512
RES	11:9, 27:25	r	**Reserved - 0** Read as 0; should be written with 0.
RESSHIFT0	15:12	rw	**Result Left shift channel 0** 0_H **LSHIFT0**, 0 left shifts … F_H **LSHIFT15**, 15 left shifts
DECF1	24:16	rw	**Decimation factor channel 1** 000_H **DECF**, 16 (not used) … $00E_H$ **DECF**, 16 (not used) 010_H **DECF17**, 17 … $1FF_H$ **DECF512**, 512
RESSHIFT1	31:28	rw	**Result Left shift channel 1** 0_H **LSHIFT0**, 0 left shifts … F_H **LSHIFT15**, 15 left shifts

Register description SDADC

Alternate function selection

ALTSEL
Alternate function selection **(0008_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
MCLK	1:0	rw	**MCLK output mapping** 00_B **OFF**, MCLK output is off 01_B **MCLK0**, MCLK connected to MCLK0 10_B **MCLK1**, MCLK connected to MCLK1 11_B **reserved**,
RES	3:2, 7:6, 15:10, 31:16	r	**Reserved - 0** Read as 0; should be written with 0.
DOUT0	5:4	rw	**DOUT0 mapping** 00_B **OFF**, DOUT0 output is off 01_B **DOUT00**, Channel 0 analog modulator output available at pin DOUT01 10_B **DOUT10**, Channel 1 analog modulator output available at pin DOUT10 11_B **reserved**,
DOUT1	9:8	rw	**DOUT1 mapping** 00_B **OFF**, DOUT0 output is off 01_B **DOUT01**, Channel 0 analog modulator output available at pin DOUT01 10_B **DOUT11**, Channel 1 analog modulator output available at pin DOUT11 11_B **reserved**,

Input Port Selection

INSEL
Input Port Selection **($000C_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES									TRG1SEL	DIN1SEL		IN1NSEL		IN1PSEL	
r									rw	rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES									TRG0SEL	DIN0SEL		IN0NSEL		IN0PSEL	
r									rw	rw		rw		rw	

Field	Bits	Type	Description
IN0PSEL	1:0	rw	**Selection for positive analog input of channel 0** In case Bit TEST.DIG_TEST_EN is set this setting is overruled and the input of modulator is set according the setting of TEST.DIG_TEST_EN 00_B **VDDHALF**, Input disconnected; internally connected to VDD/2 01_B **IN0PA**, Input connected to IN0PA 10_B **IN0PB**, Input connected to IN0PB 11_B **reserved**, reserved (disconnected)
IN0NSEL	3:2	rw	**Selection for negative analog input of channel 0** In case Bit TEST.DIG_TEST_EN is set this setting is overruled and the input of modulator is set according the setting of TEST.DIG_TEST_EN 00_B **VDDHALF**, Input disconnected 01_B **IN0NA**, Input connected to IN0NA 10_B **IN0NB**, Input connected to IN0NB 11_B **GND**, Input connected to GND
DIN0SEL	5:4	rw	**Selection of digital input of channel 0** 00_B **MOD0**, from Modulator channel 0 01_B **DIN0A**, DIN0A 10_B **DIN0B**, DIN0B 11_B **reserved**, reserved
TRG0SEL	6	rw	**Trigger Event Selection** 0_B **TRGA**, Event TRG0SDA is selected 1_B **TRGB**, Event TRG0SDB is selected
RES	15:7, 31:23	r	**Reserved - 0** Read as 0; should be written with 0.
IN1PSEL	17:16	rw	**Selection for positive analog input of channel 0** In case Bit TEST.DIG_TEST_EN is set this setting is overruled and the input of modulator is set according the setting of TEST.DIG_TEST_EN 00_B **VDDHALF**, Input disconnected; internally connected to VDD/2 01_B **IN1PA**, Input connected to IN1PA 10_B **IN1PB**, Input connected to IN1PB 11_B **reserved**, reserved (disconnected)

Register description SDADC

Field	Bits	Type	Description
IN1NSEL	19:18	rw	**INN Input Select 1** In case Bit TEST.DIG_TEST_EN is set this setting is overruled and the input of modulator is set according the setting of TEST.DIG_TEST_EN 00_B **VDDHALF**, Input disconnected; internally connected to VDD/2 01_B **IN1NA**, Input connected to IN1NA 10_B **IN1NB**, Input connected to IN1NB 11_B **GND**, Input connected to GND
DIN1SEL	21:20	rw	**External Digital Input Selection 1** 00_B **MOD1**, from Modulator channel 1 01_B **DIN1A**, DIN1A 10_B **DIN1B**, DIN1B 11_B **reserved**,
TRG1SEL	22	rw	**Trigger Event Selection** 0_B **TRGA**, Event TRG1SDA is selected 1_B **TRGB**, Event TRG1SDB is selected

Interrupt Enable Register

IEN
Interrupt Enable Register **(0010_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES									CMP1_LO_EN	CMP1_UP_EN	RES1_EN	RES	CMP0_LO_EN	CMP0_UP_EN	RES0_EN
r									rw	rw	rw	r	rw	rw	rw

Field	Bits	Type	Description
RES0_EN	0	rw	**Result interrupt channel 0** 0_B **Disable**, Result Interrupt disabled 1_B **Enable**, Result Interrupt enabled
CMP0_UP_EN	1	rw	**Upper Compare Level Interrupt channel 0** 0_B **Disable**, Upper compare Interrupt disabled 1_B **Enable**, Upper compare Interrupt enabled
CMP0_LO_EN	2	rw	**Lower compare level Interrupt channel 0** 0_B **Disable**, Lower compare Interrupt disabled 1_B **Enable**, Lower compare Interrupt enabled
RES	3, 31:7	r	**Reserved - 0** Read as 0; should be written with 0.

Register description SDADC

Field	Bits	Type	Description
RES1_EN	4	rw	**Result interrupt channel 1** 0_B **Disable**, Result Interrupt disabled 1_B **Enable**, Result Interrupt enabled
CMP1_UP_EN	5	rw	**Upper Compare Level Interrupt channel 1** 0_B **Disable**, Upper compare Interrupt disabled 1_B **Enable**, Upper compare Interrupt enabled
CMP1_LO_EN	6	rw	**Lower compare level Interrupt channel 1** 0_B **Disable**, Lower compare Interrupt disabled 1_B **Enable**, Lower compare Interrupt enabled

Event Register

IS

Event Register **(0014_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES									CMP1_LO_IS	CMP1_UP_IS	RES1_IS	RES	CMP0_LO_IS	CMP0_UP_IS	RES0_IS
r									rhxre	rhxre	rhxre	r	rhxre	rhxre	rhxre

Field	Bits	Type	Description
RES0_IS	0	rhxre	**Result Interupt channel 0** Result available channel 0 0_B **Inactive**, Interrupt has not occured 1_B **Active**, Interrupt occured
CMP0_UP_IS	1	rhxre	**Upper Compare level Interupt channel 0** 0_B **Inactive**, Interrupt has not occured 1_B **Active**, Interrupt occured
CMP0_LO_IS	2	rhxre	**Lower compare level interupt channel 0** 0_B **Inactive**, Interrupt has not occured 1_B **Active**, Interrupt occured
RES	3, 31:7	r	**Reserved - 0** Read as 0; should be written with 0.
RES1_IS	4	rhxre	**Result interupt channel 1** Result available channel 1 0_B **Inactive**, Interrupt has not occured 1_B **Active**, Interrupt occured
CMP1_UP_IS	5	rhxre	**Upper Compare level Interupt channel 1** 0_B **Inactive**, Interrupt has not occured 1_B **Active**, Interrupt occured

Register description SDADC

Field	Bits	Type	Description
CMP1_LO_IS	6	rhxre	**Lower compare level interupt channel 1** 0_B **Inactive**, Interrupt has not occured 1_B **Active**, Interrupt occured

Event Clear Register

ISR
Event Clear Register **(0018_H)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES									CMP1_LO_ISC	CMP1_UP_ISC	RES1_ISC	RES	CMP0_LO_ISC	CMP0_UP_ISC	RES0_ISC
r									w	w	w	r	w	w	w

Field	Bits	Type	Description
RES0_ISC	0	w	**Result Interupt clear channel 0** 0_B **NoClear**, Interrupt Status flag not cleared 1_B **Clear**, Interrupt status flag cleared
CMP0_UP_ISC	1	w	**Upper Compare Level Interupt clear channel 0** 0_B **NoClear**, Interrupt Status flag not cleared 1_B **Clear**, Interrupt status flag cleared
CMP0_LO_ISC	2	w	**Lower Compare Level Interupt clear channel 0** 0_B **NoClear**, Interrupt Status flag not cleared 1_B **Clear**, Interrupt status flag cleared
RES	3, 31:7	r	**Reserved - 0** Read as 0; should be written with 0.
RES1_ISC	4	w	**Result Interupt clear channel 1** 0_B **NoClear**, Interrupt Status flag not cleared 1_B **Clear**, Interrupt status flag cleared
CMP1_UP_ISC	5	w	**Upper Compare Level Interupt clear channel 1** 0_B **NoClear**, Interrupt Status flag not cleared 1_B **Clear**, Interrupt status flag cleared
CMP1_LO_ISC	6	w	**Lower Compare Level Interupt clear channel 1** 0_B **NoClear**, Interrupt Status flag not cleared 1_B **Clear**, Interrupt status flag cleared

Register description SDADC

Event Set Register

ISS

Event Set Register **(001C$_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES									CMP1_LO_ISS	CMP1_UP_ISS	RES1_ISS	RES	CMP0_LO_ISS	CMP0_UP_ISS	RES0_ISS
r									w	w	w	r	w	w	w

Field	Bits	Type	Description
RES0_ISS	0	w	**Result Interupt set channel 0** 0_B **NoClear**, Interrupt Status flag not cleared 1_B **Clear**, Interrupt status flag cleared
CMP0_UP_ISS	1	w	**Upper Compare Level Interupt set channel 0** 0_B **NoClear**, Interrupt Status flag not cleared 1_B **Clear**, Interrupt status flag cleared
CMP0_LO_ISS	2	w	**Lower Compare Level Interrupt set channel 0** 0_B **NoClear**, Interrupt Status flag not cleared 1_B **Clear**, Interrupt status flag cleared
RES	3, 31:7	r	**Reserved - 0** Read as 0; should be written with 0.
RES1_ISS	4	w	**Result Interupt set channel 1** 0_B **NoClear**, Interrupt Status flag not cleared 1_B **Clear**, Interrupt status flag cleared
CMP1_UP_ISS	5	w	**Upper Compare Level Interupt set channel 1** 0_B **NoClear**, Interrupt Status flag not cleared 1_B **Clear**, Interrupt status flag cleared
CMP1_LO_ISS	6	w	**Lower Compare Level Interupt set channel 1** 0_B **NoClear**, Interrupt Status flag not cleared 1_B **Clear**, Interrupt status flag cleared

Register description SDADC

Interrupt Node Pointer

INP
Interrupt Node Pointer **(0020_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES									CMP1_LO_INP	CMP1_UP_INP	RES1_INP	RES	CMP0_LO_INP	CMP0_UP_INP	RES0_INP
r									rw	rw	rw	r	rw	rw	rw

Field	Bits	Type	Description
RES0_INP	0	rw	**Result Interupt Node Pointer channel 0** 0_B **IRQ0**, mapped to IRQ0 1_B **IRQ1**, mapped to IRQ1
CMP0_UP_INP	1	rw	**Upper Compare Level Interupt Pointer channel 0** 0_B **IRQ0**, mapped to IRQ0 1_B **IRQ1**, mapped to IRQ1
CMP0_LO_INP	2	rw	**Lower Compare Level Interupt Pointer channel 0** 0_B **IRQ0**, mapped to IRQ0 1_B **IRQ1**, mapped to IRQ1
RES	3, 31:7	r	**Reserved - 0** Read as 0; should be written with 0.
RES1_INP	4	rw	**Result Interupt Node Pointer channel 1** 0_B **IRQ0**, mapped to IRQ0 1_B **IRQ1**, mapped to IRQ1
CMP1_UP_INP	5	rw	**Upper Compare Level Interupt Pointer channel 1** 0_B **IRQ0**, mapped to IRQ0 1_B **IRQ1**, mapped to IRQ1
CMP1_LO_INP	6	rw	**Lower Compare Level Interupt Pointer channel 1** 0_B **IRQ0**, mapped to IRQ0 1_B **IRQ1**, mapped to IRQ1

Register description SDADC

Suspend Control

SUSCTR
Suspend Control **(0024_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
EN	0	rw	**Suspend Mode Enable** 0_B **Disable**, Entering Suspend Mode is disabled 1_B **Enable**, Entering Suspend Mode is enabled
RES	31:1	r	**Reserved - 0** Read as 0; should be written with 0.

Suspend Status

SUSSTAT
Suspend Status **(0028_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
STAT	0	r	**Suspend Status** 0_B **NoSuspend**, SD ADC is not suspended 1_B **Suspend**, SD ADC is suspended
RES	31:1	r	**Reserved - 0** Read as 0; should be written with 0.

Register description SDADC

Compare Channel 0 Control Register

CMP0_CTRL
Compare Channel 0 Control Register **($002C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE		HYST_UP		RES			UPPER								
rw		rw		r			rw								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UPLO_OUTSEL		HYST_LO		RES			LOWER								
rw		rw		r			rw								

Field	Bits	Type	Description
LOWER	8:0	rw	**Lower Compare Level for channel 0**
RES	11:9, 27:25	r	**Reserved - 0** Read as 0; should be written with 0.
HYST_LO	13:12	rw	**Hysteresis setting for lower compare threshold channel 0** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
UPLO_OUTSEL	15:14	rw	**Signal output selection for UPLO output of channel 0** 00_B **UP_OR_LO**, Output OR-Combination of UP/LO signal 01_B **UP_XOR_LO**, Output XOR-Combination of UP/LO signal 10_B **UP**, Output UP signal 11_B **LO**, Output LO signal
UPPER	24:16	rw	**Upper Compare Level for channel 0**
HYST_UP	29:28	rw	**Hysteresis setting for upper compare threshold channel 0** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
MODE	31:30	rw	**Compare Mode channel 0** 00_B **RANGE**, Upper and Lower voltage/limit measurement 01_B **UV**, undervoltage/-limit measurement 10_B **OV**, overvoltage/-limit measurement 11_B **RES**, Reserved (RANGE)

Register description SDADC

Compare Channel 1 Control Register

CMP1_CTRL
Compare Channel 1 Control Register **(0030_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

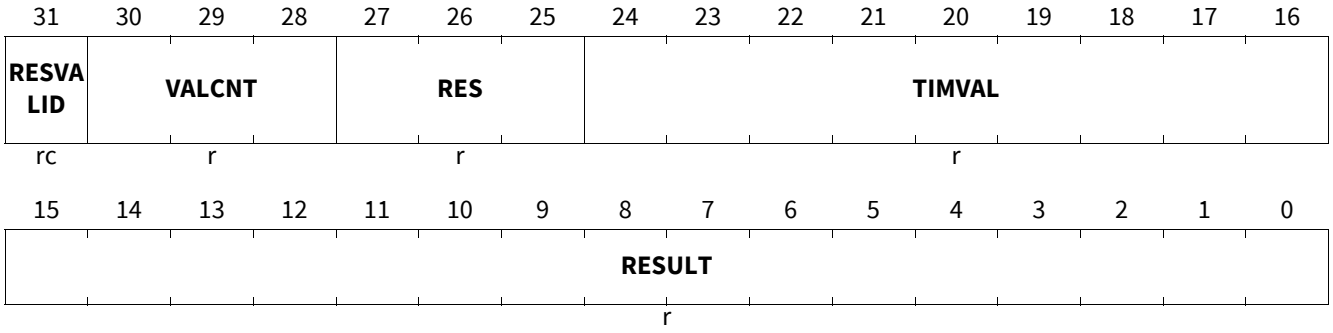
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE		HYST_UP		RES			UPPER								
rw		rw		r			rw								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UPLO_OUTSEL		HYST_LO		RES			LOWER								
rw		rw		r			rw								

Field	Bits	Type	Description
LOWER	8:0	rw	**Lower Compare Level for channel 1**
RES	11:9, 27:25	r	**Reserved - 0** Read as 0; should be written with 0.
HYST_LO	13:12	rw	**Hysteresis setting for lower compare threshold channel 1** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
UPLO_OUTSEL	15:14	rw	**Signal output selection for UPLO output of channel 1** 00_B **UP_OR_LO**, Output OR-Combination of UP/LO signal 01_B **UP_XOR_LO**, Output XOR-Combination of UP/LO signal 10_B **UP**, Output UP signal 11_B **LO**, Output LO signal
UPPER	24:16	rw	**Upper Compare Level for channel 1**
HYST_UP	29:28	rw	**Hysteresis setting for upper compare threshold channel 1** 00_B **OFF**, 01_B **HYST4**, Hysteresis=4 10_B **HYST8**, Hysteresis=8 11_B **HYST16**, Hysteresis=16
MODE	31:30	rw	**Compare Mode channel 1** 00_B **RANGE**, Upper and Lower voltage/limit measurement 01_B **UV**, undervoltage/-limit measurement 10_B **OV**, overvoltage/-limit measurement 11_B **RES**, Reserved (RANGE)

Result Register channel 0

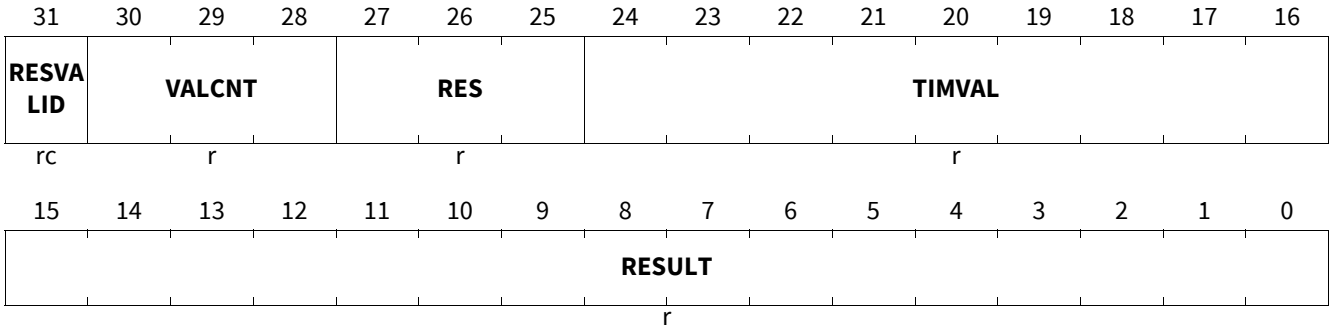
RES0
Result Register channel 0 **(0034_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
RESULT	15:0	r	**Result Register 2s complement**
TIMVAL	24:16	r	**Timestamp Value**
RES	27:25	r	**Reserved - 0** Read as 0; should be written with 0.
VALCNT	30:28	r	**Valid Counter** The counter value indicates the number of discarded results due to the wait for read feature since the last result read.
RESVALID	31	rc	**Result valid** 0_B **INVALID**, The result Register has not been updated 1_B **VALID**, The result Register has been updated

Result Register channel 1

RES1
Result Register channel 1 **(0038_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

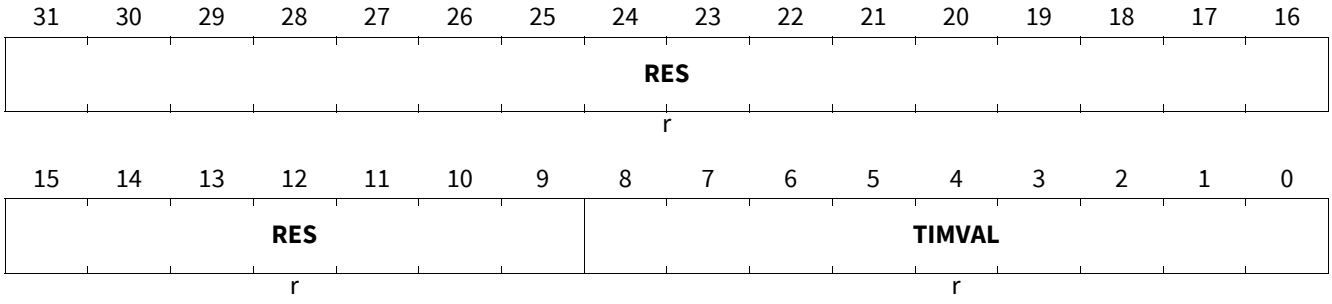


Field	Bits	Type	Description
RESULT	15:0	r	**Result Register 2s complement**
TIMVAL	24:16	r	**Timestamp Value**
RES	27:25	r	**Reserved - 0** Read as 0; should be written with 0.

Field	Bits	Type	Description
VALCNT	30:28	r	**Valid Counter** The counter value indicates the number of discarded results due to the wait for read feature since the last result read.
RESVALID	31	rc	**Result valid** 0_B **INVALID**, The result Register has not been updated 1_B **VALID**, The result Register has been updated

Captured Timestamp channel 0

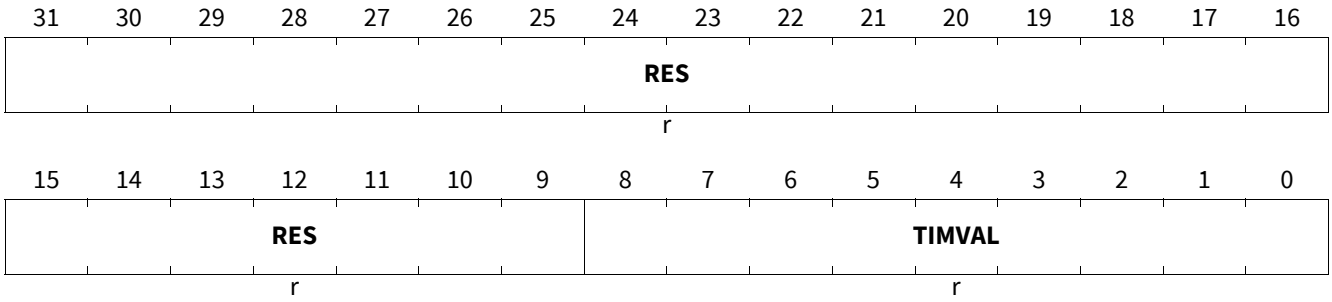
CTIM0
Captured Timestamp channel 0 ($003C_H$) RESET_TYPE_5 Value: 0000 0000$_H$



Field	Bits	Type	Description
TIMVAL	8:0	r	**Captured Timestamp Value**
RES	31:9	r	**Reserved - 0** Read as 0; should be written with 0.

Captured Timestamp channel 1

CTIM1
Captured Timestamp channel 1 (0040_H) RESET_TYPE_5 Value: 0000 0000$_H$



Field	Bits	Type	Description
TIMVAL	8:0	r	**Captured Timestamp Value**
RES	31:9	r	**Reserved - 0** Read as 0; should be written with 0.

Register description SDADC

Dither Configuration Register

DITHCFG
Dither Configuration Register **(0044_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

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

Field	Bits	Type	Description
DITH_VOLT_SEL	3:0	rw	**Dither voltage Selection** 0_H SEL0 - 0.0000 V 1_H SEL1 - 0.0125 V 2_H SEL2 - 0.0250 V F_H SEL15 - 0.1875 V
RES	31:4	r	**Reserved - 0** Read as 0; should be written with 0.

Status Clear Register

STSR
Status Clear Register **($004C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								WFR1_SC	CMP1_LO_SC	CMP1_UP_SC	RES	WFR0_SC	CMP0_LO_SC	CMP0_UP_SC	RES
r								w	w	w	r	w	w	w	r

Field	Bits	Type	Description
RES	0, 4, 31:8	r	**Reserved - 0** 0_B **Inactive**, Interupt has not occured 1_B **Active**, Interupt occured
CMP0_UP_SC	1	w	**Upper Compare level Status clear channel 0** 0_B **Inactive**, Interupt is not cleared 1_B **Active**, Interupt is cleared

Register description SDADC

Field	Bits	Type	Description
CMP0_LO_SC	2	w	**Lower compare level Status clear channel 0** 0_B **Inactive**, Interupt is not cleared 1_B **Active**, Interupt is cleared
WFR0_SC	3	w	**Wait for Read Status clear channel 0** Read as 0; should be written with 0. 0_B **Inactive**, Interupt is not cleared 1_B **Active**, Interupt is cleared
CMP1_UP_SC	5	w	**Upper Compare level Status clear channel 1** 0_B **Inactive**, Interupt is not cleared 1_B **Active**, Interupt is cleared
CMP1_LO_SC	6	w	**Lower compare level Status clear channel 1** 0_B **Inactive**, Interupt is not cleared 1_B **Active**, Interupt is cleared
WFR1_SC	7	w	**Wait for Read Status clear channel 1** Read as 0; should be written with 0. 0_B **Inactive**, Interupt is not cleared 1_B **Active**, Interupt is cleared

Status Register

STS
Status Register **(0050_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								WFR1_STS	CMP1_LO_STS	CMP1_UP_STS	RES	WFR0_STS	CMP0_LO_STS	CMP0_UP_STS	RES
r								rhxr	rhxr	rhxr	r	rhxr	rhxr	rhxr	r

Field	Bits	Type	Description
RES	0, 4, 31:8	r	**Reserved - 0** 0_B **Inactive**, Interupt has not occured 1_B **Active**, Interupt occured
CMP0_UP_STS	1	rhxr	**Upper Compare level Status channel 0** 0_B **Inactive**, Interupt has not occured 1_B **Active**, Interupt occured
CMP0_LO_STS	2	rhxr	**Lower compare level Status channel 0** 0_B **Inactive**, Interupt has not occured 1_B **Active**, Interupt occured

Field	Bits	Type	Description
WFR0_STS	3	rhxr	**Result discarded channel 0** This flag indicates that a result was discarded due to the wait for read feature
CMP1_UP_STS	5	rhxr	**Upper Compare level Status channel 1** 0_B **Inactive**, Interupt has not occured 1_B **Active**, Interupt occured
CMP1_LO_STS	6	rhxr	**Lower compare level Status channel 1** 0_B **Inactive**, Interupt has not occured 1_B **Active**, Interupt occured
WFR1_STS	7	rhxr	**Result discarded channel 1** This flag indicates that a result was discarded due to the wait for read feature

Status Set Register

STSS
Status Set Register **(0054$_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								WFR1_SS	CMP1_LO_SS	CMP1_UP_SS	RES	WFR0_SS	CMP0_LO_SS	CMP0_UP_SS	RES
r								w	w	w	r	w	w	w	r

Field	Bits	Type	Description
RES	0, 4, 31:8	r	**Reserved - 0** 0_B **Inactive**, Interupt has not occured 1_B **Active**, Interupt occured
CMP0_UP_SS	1	w	**Upper Compare level Status set channel 0** 0_B **Inactive**, Interupt is not cleared 1_B **Active**, Interupt is cleared
CMP0_LO_SS	2	w	**Lower compare level Status set channel 0** 0_B **Inactive**, Interupt is not cleared 1_B **Active**, Interupt is cleared
WFR0_SS	3	w	**Wait for Read Status set channel 0** 0_B **Inactive**, Interupt is not cleared 1_B **Active**, Interupt is cleared
CMP1_UP_SS	5	w	**Upper Compare level Status set channel 1** 0_B **Inactive**, Interupt is not cleared 1_B **Active**, Interupt is cleared

Field	Bits	Type	Description
CMP1_LO_SS	6	w	**Lower compare level Status set channel 1** 0_B **Inactive**, Interupt is not cleared 1_B **Active**, Interupt is cleared
WFR1_SS	7	w	**Wait for Read Status set channel 1** 0_B **Inactive**, Interupt is not cleared 1_B **Active**, Interupt is cleared

23 General Purpose Timer Units (GPT12)

23.1 Features overview

The General Purpose Timer Unit blocks GPT1 and GPT2 have very flexible multifunctional timer structures which may be used for timing, event counting, pulse width measurement, pulse generation, frequency multiplication, and other purposes.

They incorporate five 16-bit timers that are grouped into the two timer blocks GPT1 and GPT2. Each timer in each block may operate independently in a number of different modes such as Gated timer or Counter mode, or may be concatenated with another timer of the same block.

Each block has alternate input/output functions and specific interrupts associated with it. Input signals can be selected from several sources by register **PISEL**.

The GPT module is clocked with clock f_{GPT_CLK}.

The GPT12 provides following features:

- Features block GPT1:
  - f_{GPT_CLK}/4 maximum resolution
  - 3 independent timers/counters
  - Timers/counters can be concatenated
  - 4 operating modes:
    - Timer mode
    - Gated Timer mode
    - Counter Mode
    - Incremental Interface mode
  - Reload and Capture functionality
- Features block GPT2:
  - f_{GPT_CLK}/2 maximum resolution
  - 2 independent timers/counters
  - Timers/counters can be concatenated
  - 3 operating modes:
    - Timer mode
    - Gated Timer mode
    - Counter mode
  - Extended capture/reload functions via 16-bit capture/reload register **CAPREL**

23.2 Block diagram

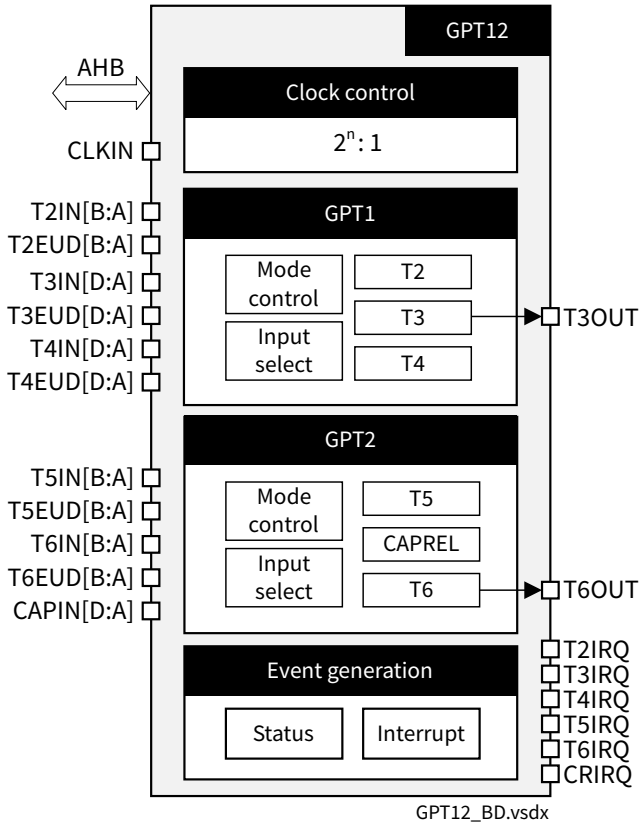


Figure 260 Block diagram GPT12

Block GPT1

Block GPT1 contains three timers/counters: The core timer T3 and the two auxiliary timers T2 and T4. The maximum resolution is $f_{GPT\_CLK}/4$. The auxiliary timers of GPT1 may optionally be configured as reload or capture registers for the core timer.

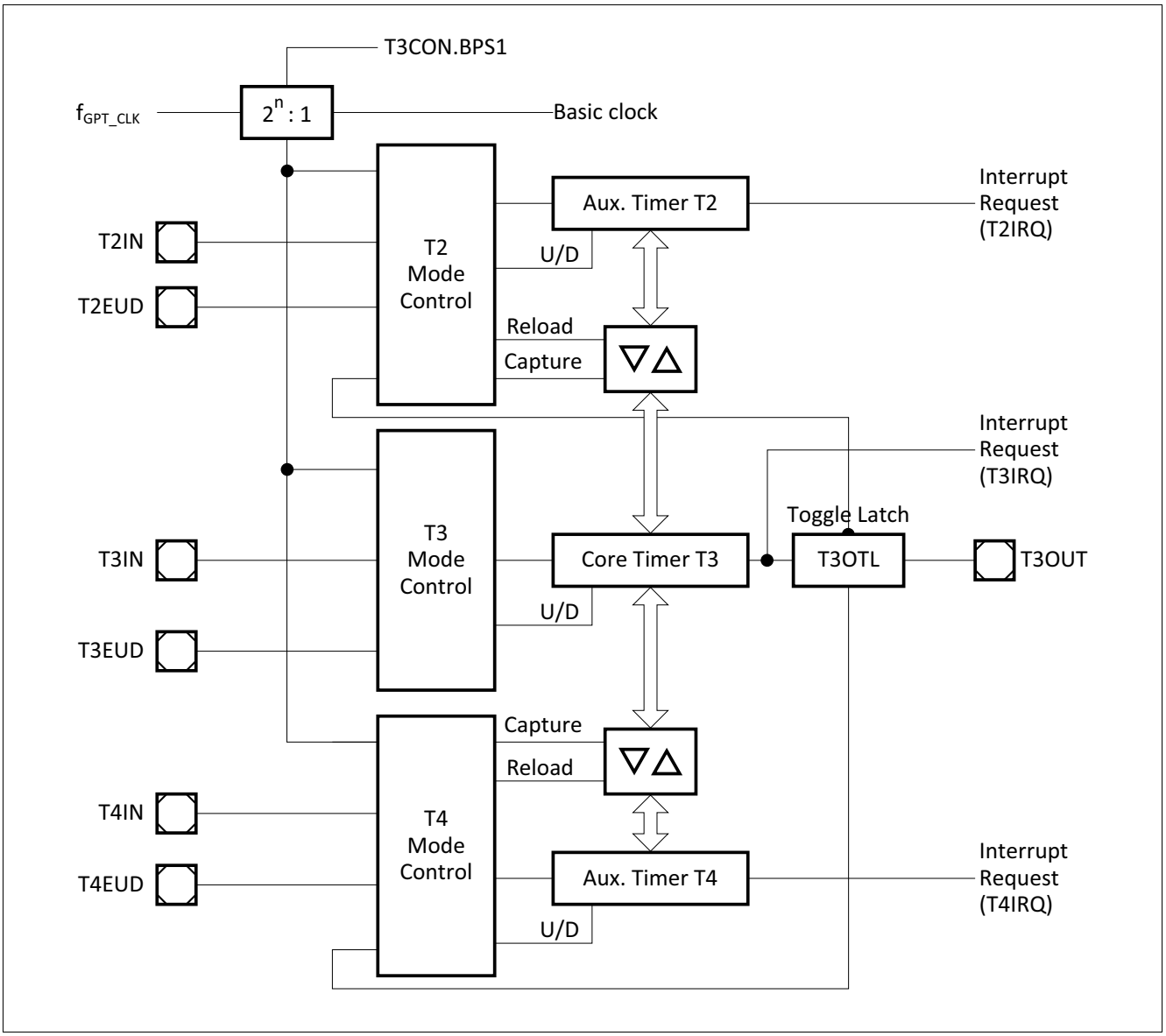


Figure 261 GPT1 block (n = 2 ... 5)

Block GPT2

Block GPT2 contains two timers/counters: The core timer T6 and the auxiliary timer T5. The maximum resolution is $f_{GPT\_CLK}/2$. An additional Capture/Reload register (**CAPREL**) supports capture and reload operation with extended functionality.

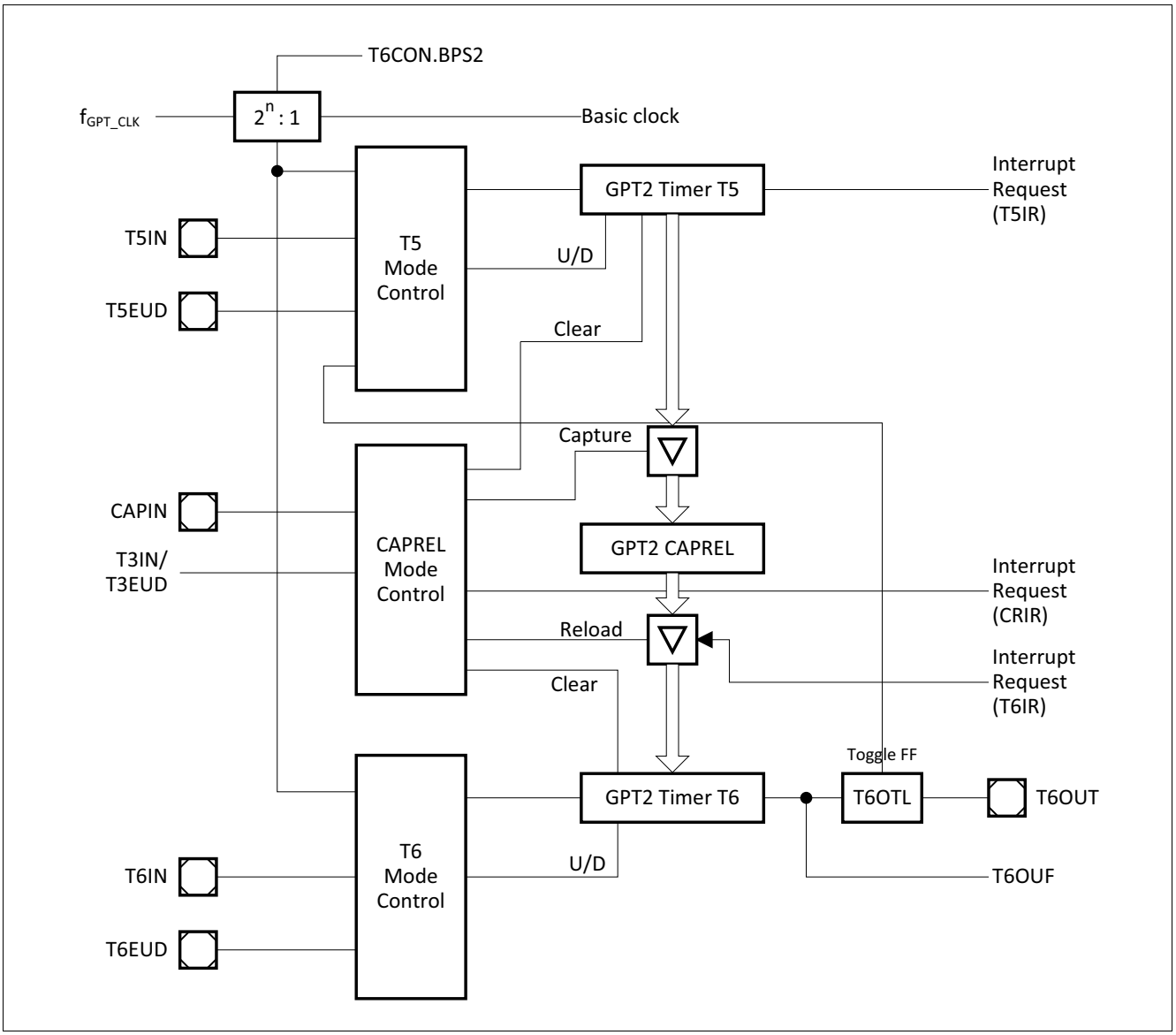


Figure 262 GPT2 block

23.3 Toplevel signals

Table 271 Toplevel connection

Signal	Direction	Description	From/To
T2IN	Input	Count input signals for timer T2	See **Product definitions**, **GPT12 interconnections**
T2EUD	Input	Direction input signals for timer T2	
T2IRQ	Output	Interrupt request from timer T2	
T3IN	Input	Count input signals for timer T3	
T3EUD	Input	Direction input signals for timer T3	
T3IRQ	Output	Interrupt request from timer T3	
T3OUT	Output	Count output signal for timer T3	
T4IN	Input	Count input signals for timer T4	
T4EUD	Input	Direction input signals for timer T4	
T4IRQ	Output	Interrupt request from timer T4	
T5IN	Input	Count input signals for timer T5	
T5EUD	Input	Direction input signals for timer T5	
T5IRQ	Output	Interrupt request from timer T5	
T6IN	Input	Count input signals for timer T6	
T6EUD	Input	Direction input signals for timer T6	
T6IRQ	Output	Interrupt request from timer T6	
T6OUT	Output	Count output signal for timer T6	
CAPIN	Input	Input capture signals	
CRIRQ	Output	Interrupt request from capture control	
fGPT_CLK	Input	Clock for GPT12	
GPT12_DIS	Input	GPT12disable	SCU
GPT12SUS	Input	GPT12suspend	SCU

23.4 Interrupts

Interrupt control for GPT1 timers

When a timer overflows from $FFFF_H$ to 0000_H (when counting up), or when it underflows from 0000_H to $FFFF_H$ (when counting down), its interrupt request flag in register **T2**, **T3**, or **T4** will be set. This will cause an interrupt to the respective timer interrupt vector, if the respective interrupt enable bit is set.

- In Reload mode, upon a trigger signal, T3 is loaded with the contents of the respective timer (T2 or T4) and the respective interrupt request flag in register **T2** or **T4** is set
- In Incremental Interface mode, the interrupt request generation can be selected as follows:
  - In Rotation Detection mode (T3M = 110_B), an interrupt request is generated each time the count direction of T3 changes
  - In Edge Detection mode (T3M = 111_B), an interrupt request is generated each time a count edge for T3 is detected
- In Capture mode, upon a trigger (selected transition) at the corresponding input pin the content of the core timer T3 are loaded into the auxiliary timer register Tx and the associated interrupt request flag in register SCU.GPTIS will be set

Interrupt control for GPT2 timers and CAPREL

When a timer overflows from $FFFF_H$ to 0000_H (when counting up), or when it underflows from 0000_H to $FFFF_H$ (when counting down), its interrupt request flag in register SCU.GPTIS will be set. This will cause an interrupt to the respective timer interrupt vector, if the respective interrupt enable bit is set.

Whenever a transition according to the selection in bit field CI is detected at pin CAPIN, interrupt request flag in register SCU.GPTIS is set. Setting any request flag will cause an interrupt to the respective timer or **CAPREL** interrupt vector, if the respective interrupt enable bit is set.

There is an interrupt control register for each of the two timers (T5, T6) and for the **CAPREL** register. All interrupt control registers have the same structure described in **Chapter** .

GPT12 interrupt control registers

The GPT12 module itself doesn't contain the corresponding interrupt registers and logic, it was implemented in the SCU INT_CTRL module.

For each GPT12 interrupt event input the INT_CTRL module implements a status, clear and set register bit (SCU.GPTIS, SCU.GPTISC, SCU.GPTISS). The interrupts can be enabled by SCU.GPTIEN, see **Figure 263**.

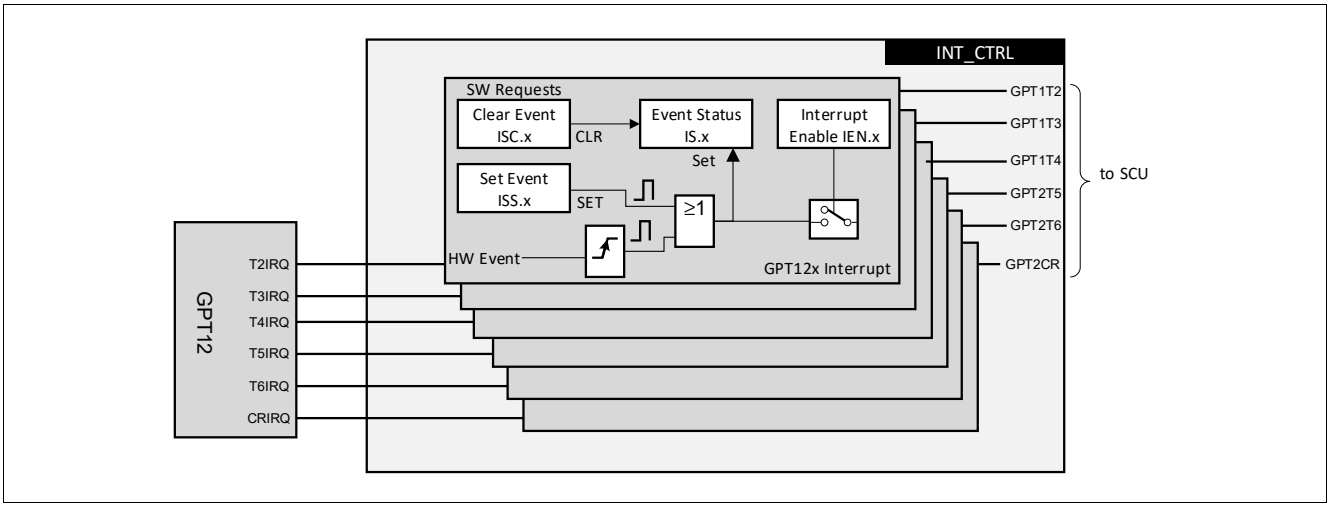


Figure 263 GPT12 interrupt generation

23.5 Timer block GPT1

From a programmer's point of view, the GPT1 block is composed of a set of SFRs as summarized below. Those portions of port and direction registers which are used for alternate functions by the GPT1 block are shaded.

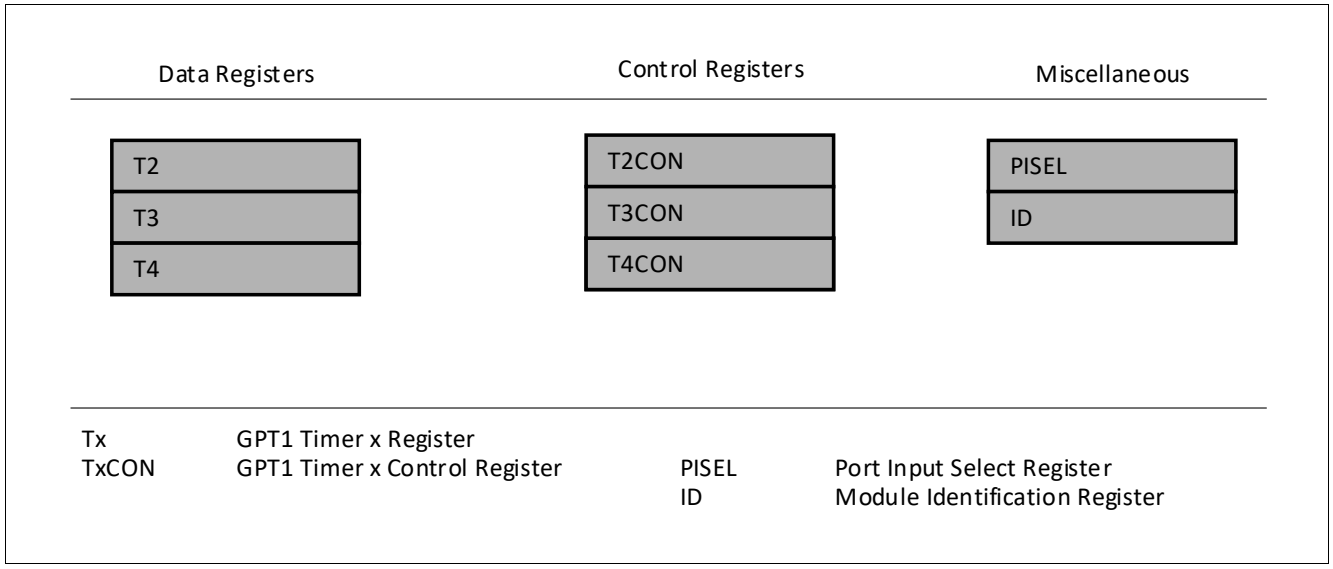


Figure 264 SFRs associated with timer block GPT1

All three timers of block GPT1 (T2, T3, T4) can run in one of 4 basic modes: Timer mode, Gated Timer mode, Counter mode, or Incremental Interface mode. All timers can count up or down. Each timer of GPT1 is controlled by a separate control register TxCON.

Each timer has an input pin TxIN (alternate pin function) associated with it, which serves as the gate control in Gated Timer mode, or as the count input in Counter mode. The count direction (up/down) may be programmed via software or may be dynamically altered by a signal at the External Up/Down control input TxEUD (alternate pin function). An overflow/underflow of core timer T3 is indicated by the Output Toggle Latch T3OTL, whose state may be output on the associated pin T3OUT (alternate pin function). The auxiliary timers T2 and T4 may additionally be concatenated with the core timer T3 (through T3OTL) or may be used as capture or reload registers for the core timer T3.

The current contents of each timer can be read or modified by the CPU by accessing the corresponding timer count registers **T2**, **T3**, or **T4**, located in the non-bitaddressable SFR space (see **Chapter 23.7**). When any of the timer registers is written to by the CPU in the state immediately preceding a timer increment, decrement, reload, or capture operation, the CPU write operation has priority in order to guarantee correct results.

The interrupts of GPT1 are controlled through the SCU.GPTIEN, SCU.GPTIS, SCU.GPTISC and SCU.GPTISS. These registers are not part of the GPT1 block.

The input and output lines of GPT1 are connected to pins. The control registers for the port functions are located in the respective port modules.

*Note: The timing requirements for external input signals can be found in **Chapter 23.5.5**, **Chapter 23.3** summarizes the module interface signals.*

23.5.1 GPT1 core timer T3 control

The current contents of the core timer T3 are reflected by its count register **T3**. This register can also be written to by the CPU, for example, to set the initial start value.

The core timer T3 is configured and controlled via its control register **T3CON**.

23.5.1.1 Timer T3 run control

The core timer T3 can be started or stopped by software through bit T3R (Timer T3 Run Bit). This bit is relevant in all operating modes of T3. Setting bit T3R will start the timer, clearing bit T3R stops the timer.

In Gated Timer mode, the timer will only run if T3R = 1 and the gate is active (high or low, as programmed).

*Note: When bit T2RC or T4RC in timer control register **T2CON** or **T4CON** is set, bit T3R will also control (start and stop) the auxiliary timer(s) T2 and/or T4.*

23.5.1.2 Count direction control

The count direction of the GPT1 timers (core timer and auxiliary timers) can be controlled either by software or by the external input pin TxEUD (Timer Tx External Up/Down Control Input). These options are selected by bits TxUD and TxUDE in the respective control register TxCON. When the up/down control is provided by software (bit TxUDE = 0), the count direction can be altered by setting or clearing bit TxUD. When bit TxUDE = 1, pin TxEUD is selected to be the controlling source of the count direction. However, bit TxUD can still be used to reverse the actual count direction, as shown in **Table 276**. The count direction can be changed regardless of whether or not the timer is running.

Note: When pin TxEUD is used as external count direction control input, it must be configured as input.

23.5.1.3 Timer T3 output toggle latch

The overflow/underflow signal of timer T3 is connected to a block named ‘Toggle Latch’, shown in the Timer mode diagrams. **Figure 265** illustrates the details of this block. An overflow or underflow of T3 will clock two latches: The first latch represents bit T3OTL in control register **T3CON**. The second latch is an internal latch toggled by T3OTL’s output. Both latch outputs are connected to the input control blocks of the auxiliary timers T2 and T4. The output level of the shadow latch will match the output level of T3OTL, but is delayed by one clock cycle. When the T3OTL value changes, this will result in a temporarily different output level from T3OTL and the shadow latch, which can trigger the selected count event in T2 and/or T4.

When software writes to T3OTL, both latches are set or cleared simultaneously. In this case, both signals to the auxiliary timers carry the same level and no edge will be detected. Bit T3OE (overflow/underflow output enable) in register **T3CON** enables the state of T3OTL to be monitored via an external pin T3OUT. When T3OTL is linked to an external port pin (must be configured as output), T3OUT can be used to control external HW. If T3OE = 1, pin T3OUT outputs the state of T3OTL. If T3OE = 0, pin T3OUT outputs a high level (as long as the T3OUT alternate function is selected for the port pin).

The trigger signals can serve as an input for the counter function or as a trigger source for the reload function of the auxiliary timers T2 and T4.

As can be seen from **Figure 265**, when latch T3OTL is modified by software to determine the state of the output line, also the internal shadow latch is set or cleared accordingly. Therefore, no trigger condition is detected by T2/T4 in this case.

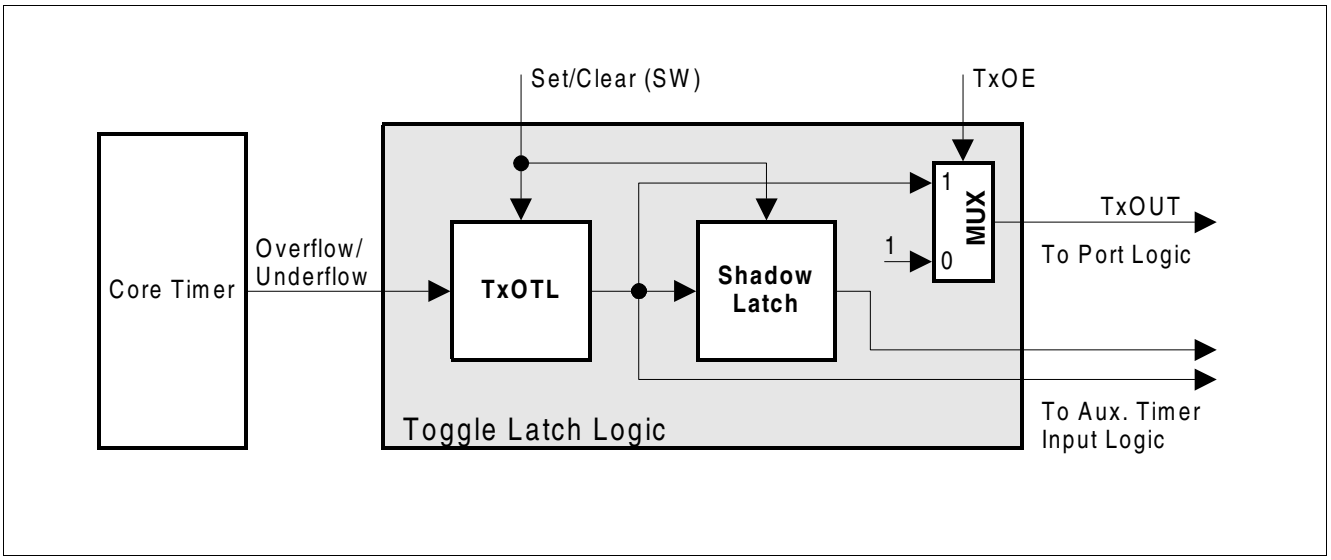


Figure 265 Block diagram of the toggle latch logic of core timer T3 (x = 3)

23.5.2 GPT1 core timer T3 operating modes

Timer T3 can operate in one of the following modes:

- **Timer T3 in Timer mode**
- **Timer T3 in Gated Timer mode**
- **Timer T3 in Counter mode**
- **Timer T3 in Incremental Interface mode**

23.5.2.1 Timer T3 in Timer mode

Timer mode for the core timer T3 is selected by setting bitfield T3M in register **T3CON** to 000_B. In Timer mode, T3 is clocked with the module's input clock f_{GPT_CLK} divided by two programmable prescalers controlled by bitfields BPS1 and T3I in register **T3CON**. Please see **Chapter 23.5.5** for details on the input clock options.

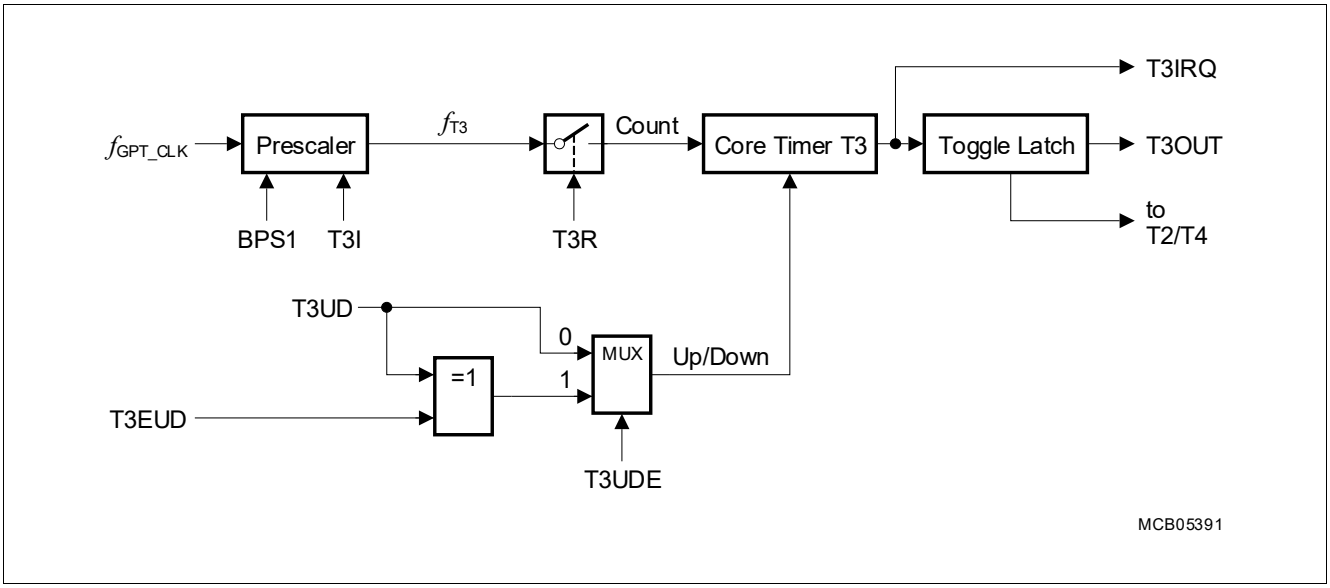


Figure 266 Block diagram of core timer T3 in Timer mode

23.5.2.2 Timer T3 in Gated Timer mode

Gated Timer mode for the core timer T3 is selected by setting bitfield T3M in register **T3CON** to 010_B or 011_B. Bit T3M.0 (T3CON.3) selects the active level of the gate input. The same options for the input frequency are available in Gated Timer mode as in Timer mode (see **Chapter 23.5.5**). However, the input clock to the timer in this mode is gated by the external input pin T3IN (Timer T3 External Input).
To enable this operation, the associated pin T3IN must be configured as input.

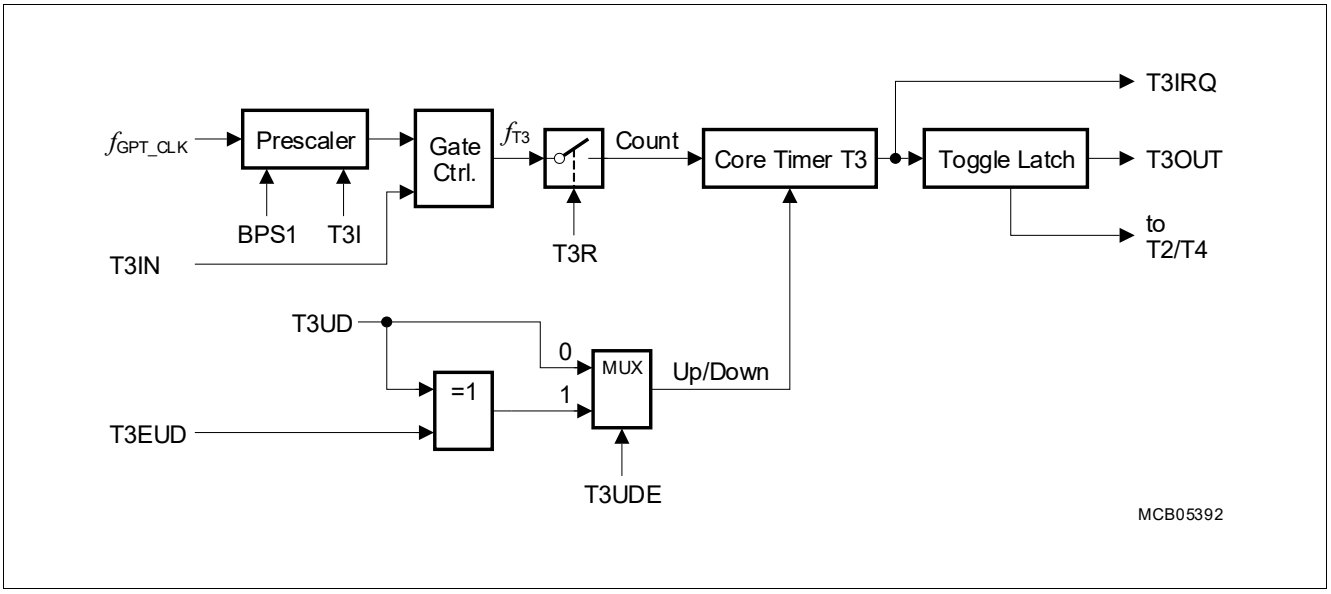


Figure 267 Block diagram of core timer T3 in Gated Timer mode

If T3M = 010_B, the timer is enabled when T3IN shows a low level. A high level at this line stops the timer. If T3M = 011_B, line T3IN must have a high level in order to enable the timer. Additionally, the timer can be turned on or off by software using bit T3R. The timer will only run if T3R is 1 and the gate is active. It will stop if either T3R is 0 or the gate is inactive.

Note: A transition of the gate signal at pin T3IN does not cause an interrupt request.

23.5.2.3 Timer T3 in Counter mode

Counter mode for the core timer T3 is selected by setting bitfield T3M in register **T3CON** to 001_B. In Counter mode, timer T3 is clocked by a transition at the external input pin T3IN. The event causing an increment or decrement of the timer can be a positive, a negative, or both a positive and a negative transition at this line. Bitfield T3I in control register **T3CON** selects the triggering transition (see **Table 278**).

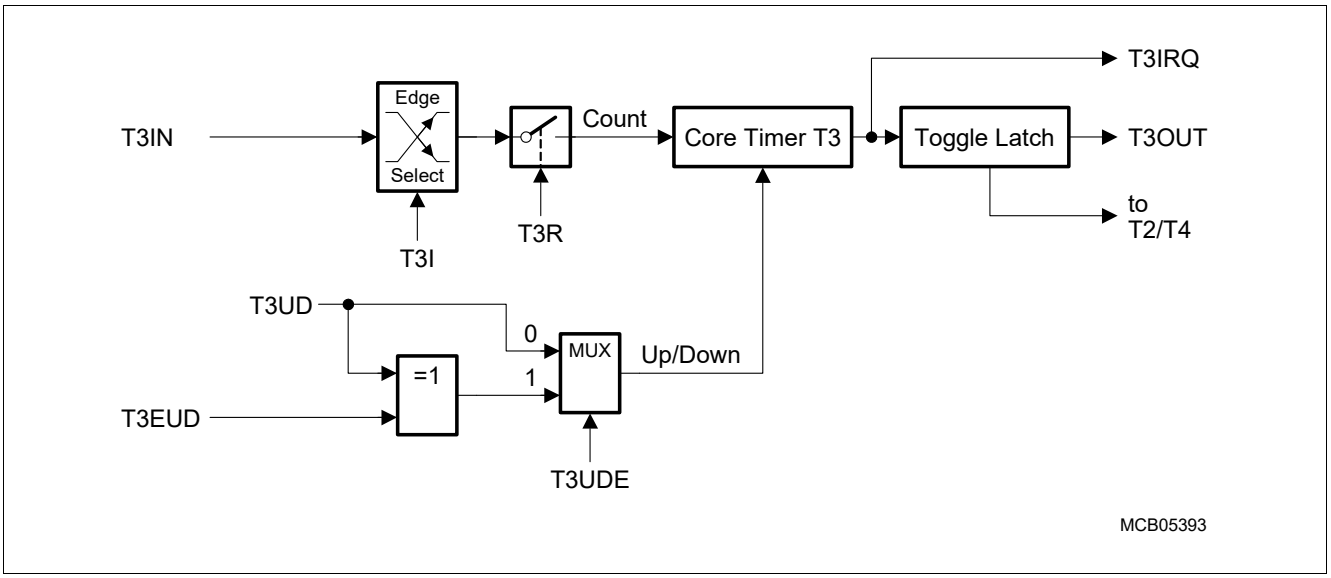


Figure 268 Block diagram of core timer T3 in Counter mode

For Counter mode operation, pin T3IN must be configured as input. The maximum input frequency allowed in Counter mode depends on the selected prescaler value. To ensure that a transition of the count input signal applied to T3IN is recognized correctly, its level must be held high or low for a minimum number of module clock cycles before it changes. This information can be found in **Chapter 23.5.5**.

23.5.2.4 Timer T3 in Incremental Interface mode

Incremental Interface mode for the core timer T3 is selected by setting bitfield T3M in register **T3CON** to 110_B or 111_B. In Incremental Interface mode, the two inputs associated with core timer T3 (T3IN, T3EUD) are used to interface to an incremental encoder. T3 is clocked by each transition on one or both of the external input pins to provide 2-fold or 4-fold resolution of the encoder input.

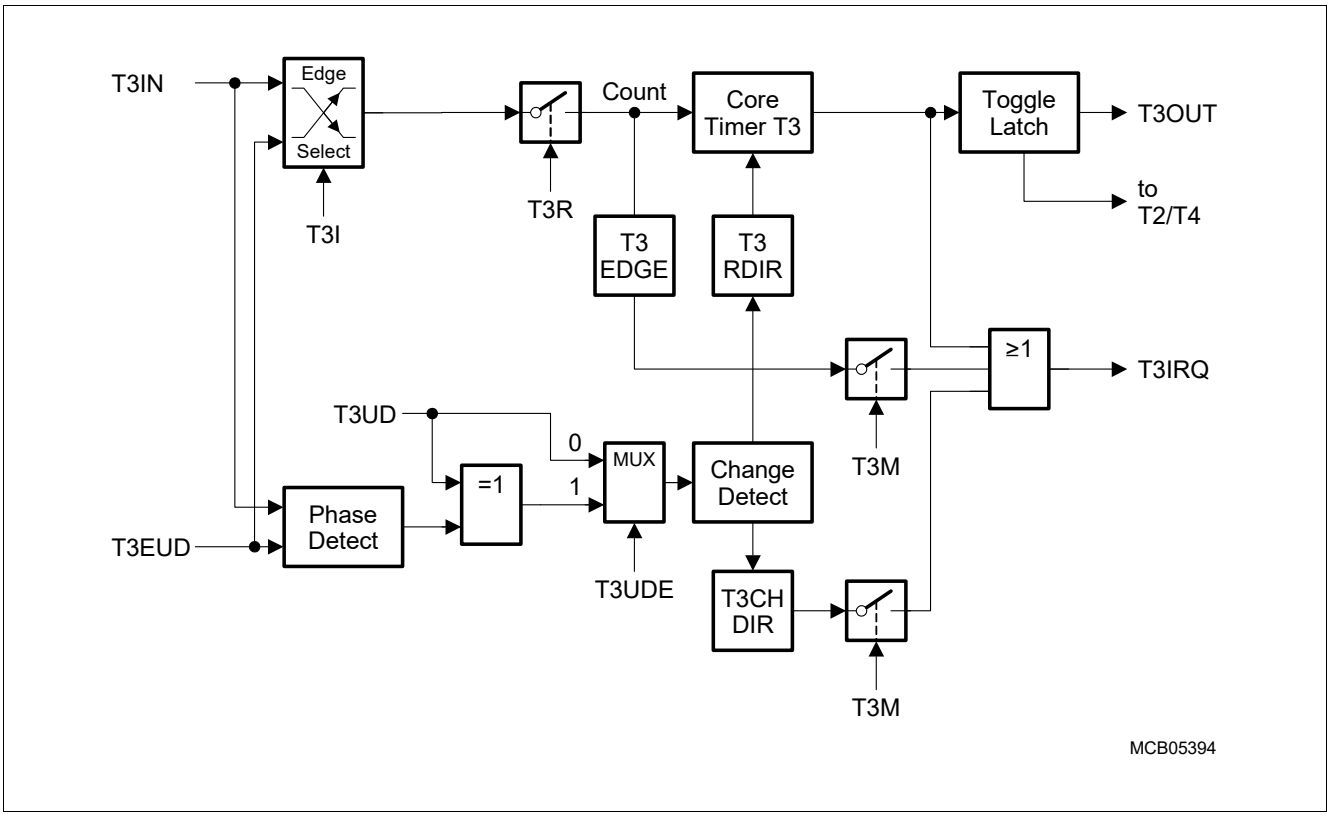


Figure 269 Block diagram of core timer T3 in Incremental Interface mode

Bitfield T3I in control register **T3CON** selects the triggering transitions (see **Table 281**). The sequence of the transitions of the two input signals is evaluated and generates count pulses as well as the direction signal. So T3 is modified automatically according to the speed and the direction of the incremental encoder and, therefore, its contents always represent the encoder's current position.

The interrupt request (T3IRQ) generation mode can be selected: In Rotation Detection mode (T3M = 110_B), an interrupt request is generated each time the count direction of T3 changes. In Edge Detection mode (T3M = 111_B), an interrupt request is generated each time a count edge for T3 is detected. Count direction, changes in the count direction, and count requests are monitored by status bits T3RDIR, T3CHDIR, and T3EDGE in register **T3CON**.

The incremental encoder can be connected directly to the TLE989x/TLE988x without external interface logic. In a standard system, however, comparators will be employed to convert the encoder's differential outputs (such as A, $\overline{A}$) to digital signals (such as A). This greatly increases noise immunity.

Note: The third encoder output T0, which indicates the mechanical zero position, may be connected to an external interrupt input and trigger a reset of timer T3.
If input T4IN is available, T0 can be connected there and clear T3 automatically without requiring an interrupt.

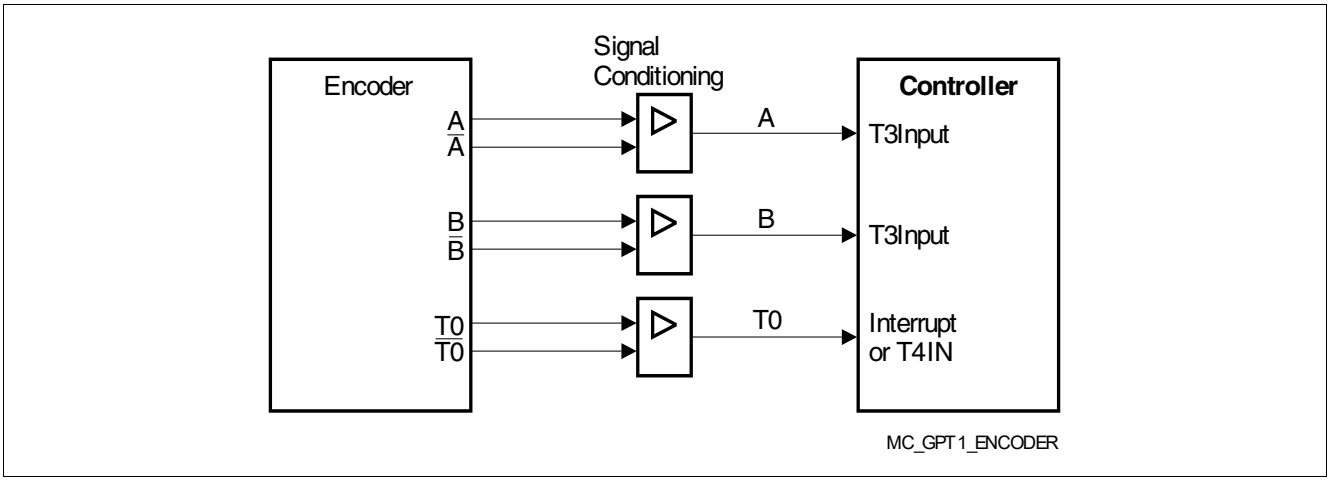


Figure 270 Connection of the encoder to the TLE989x/TLE988x

For incremental interface operation, the following conditions must be met:

- Bitfield T3M must be 110_B or 111_B
- Both pins T3IN and T3EUD must be configured as input
- Pin T4IN must be configured as input, if used for T0
- Bit T3UDE must be 1 to enable automatic external direction control

The maximum count frequency allowed in Incremental Interface mode depends on the selected prescaler value. To ensure that a transition of any input signal is recognized correctly, its level must be held high or low for a minimum number of module clock cycles before it changes. This information can be found in **Chapter 23.5.5**.

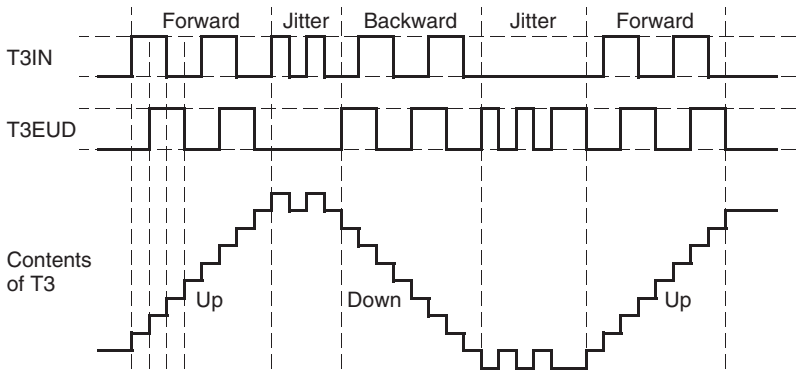
As in Incremental Interface mode two input signals with a 90° phase shift are evaluated, their maximum input frequency can be half the maximum count frequency.

In Incremental Interface mode, the count direction is automatically derived from the sequence in which the input signals change, which corresponds to the rotation direction of the connected sensor. **Table 272** summarizes the possible combinations.

Table 272 GPT1 core timer T3 (Incremental Interface mode) count direction

Level on respective other input	T3IN input		T3EUD input	
	Rising ↑	**Falling ↓**	**Rising ↑**	**Falling ↓**
High	Down	Up	Up	Down
Low	Up	Down	Down	Up

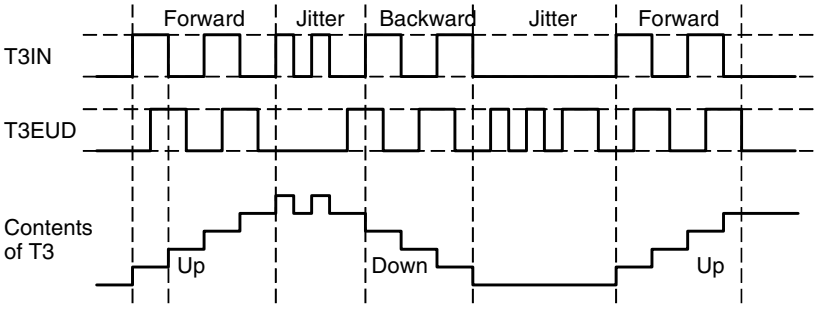
Figure 271 and **Figure 272** give examples of T3's operation, visualizing count signal generation and direction control. They also show how input jitter is compensated, which might occur if the sensor rests near to one of its switching points.



Note: This example shows the timer behaviour assuming that T3 counts upon any transition on input, i.e. T3I = '011 $_B$'.

MCT04373

Figure 271 Evaluation of incremental encoder signals, 2 count inputs



Note: This example shows the timer behaviour assuming that T3 counts upon any transition on input T3IN, i.e. T3I = '001$_B$'.

MCT04374

Figure 272 Evaluation of incremental encoder signals, 1 count input

*Note: Timer T3 operating in Incremental Interface mode automatically provides information on the sensor's current position. Dynamic information (speed, acceleration, deceleration) may be obtained by measuring the incoming signal periods (see **Combined Capture modes**).*

23.5.3 GPT1 auxiliary timers T2/T4 control

Auxiliary timers T2 and T4 have exactly the same functionality. They can be configured for Timer mode, Gated Timer mode, Counter mode, or Incremental Interface mode with the same options for the timer frequencies and the count signal as the core timer T3. In addition to these 4 counting modes, the auxiliary timers can be concatenated with the core timer, or they may be used as reload or capture registers in conjunction with the core timer. The start/stop function of the auxiliary timers can be remotely controlled by the T3 run control bit. Several timers may thus be controlled synchronously.

The current contents of an auxiliary timer are reflected by its count register **T2** or **T4**, respectively. These registers can also be written to by the CPU, for example, to set the initial start value.

The individual configurations for timers T2 and T4 are determined by their control registers **T2CON** and **T4CON**, which are organized identically. Note that functions which are present in all 3 timers of block GPT1 are controlled in the same bit positions and in the same manner in each of the specific control registers.

Note: The auxiliary timers have no output toggle latch and no alternate output function.

23.5.3.1 Timer T2/T4 run control

Each of the auxiliary timers T2 and T4 can be started or stopped by software in two different ways:

- Through the associated timer run bit (T2R or T4R). In this case it is required that the respective control bit TxRC = 0
- Through the core timer's run bit (T3R). In this case the respective remote control bit must be set (TxRC = 1)

The selected run bit is relevant in all operating modes of T2/T4. Setting the bit will start the timer, clearing the bit stops the timer.

In Gated Timer mode, the timer will only run if the selected run bit is set and the gate is active (high or low, as programmed).

Note: If remote control is selected T3R will start/stop timer T3 and the selected auxiliary timer(s) synchronously.

23.5.3.2 Count direction control

The count direction of the GPT1 timers (core timer and auxiliary timers) is controlled in the same way, either by software or by the external input pin TxEUD. Please refer to the description in **Table 276**.

Note: When pin TxEUD is used as external count direction control input, it must be configured as input.

23.5.4 GPT1 auxiliary timers T2/T4 operating modes

The operation of the auxiliary timers in the basic operating modes is almost identical with the core timer's operation, with very few exceptions. Additionally, some combined operating modes can be selected.

23.5.4.1 Timers T2 and T4 in Timer mode

Timer mode for an auxiliary timer Tx is selected by setting its bitfield TxM in register TxCON to 000_B.

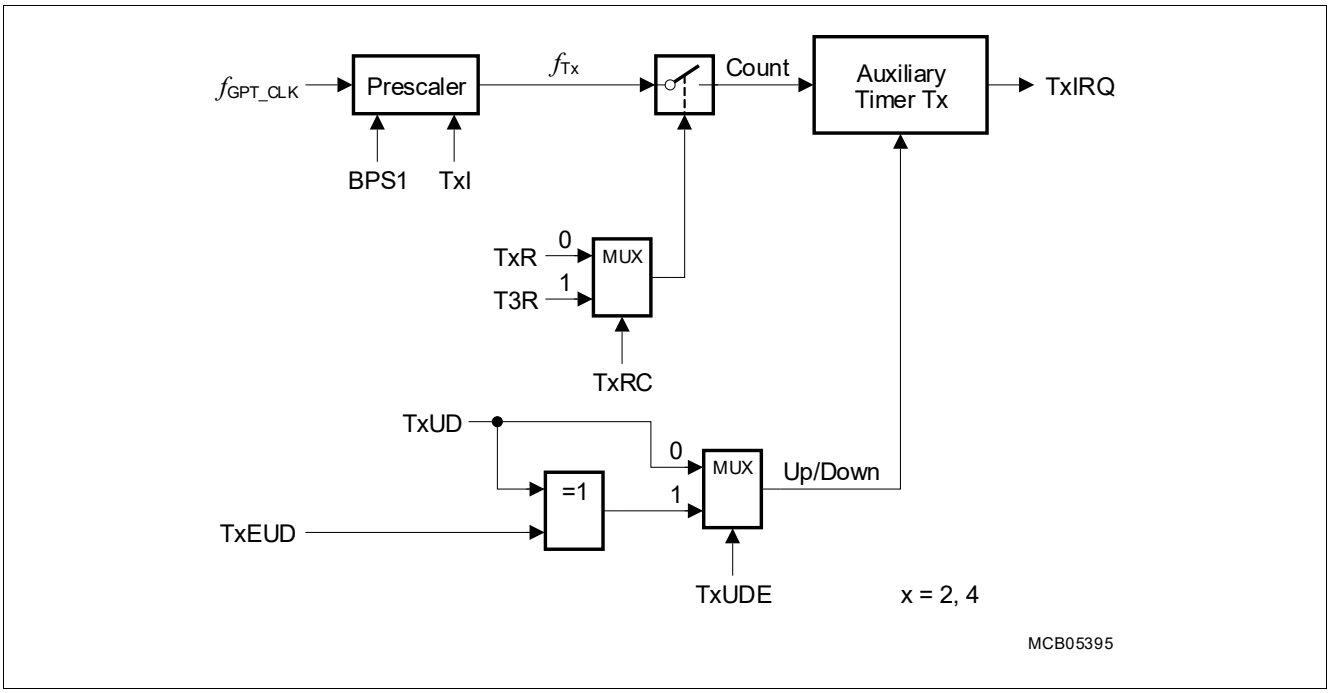


Figure 273 Block diagram of an auxiliary timer in Timer mode

23.5.4.2 Timers T2 and T4 in Gated Timer mode

Gated Timer mode for an auxiliary timer Tx is selected by setting bitfield TxM in register TxCONto 010_B or 011_B. Bit TxM.0 (TxCON.3) selects the active level of the gate input.

Note: A transition of the gate signal at line TxIN does not cause an interrupt request.

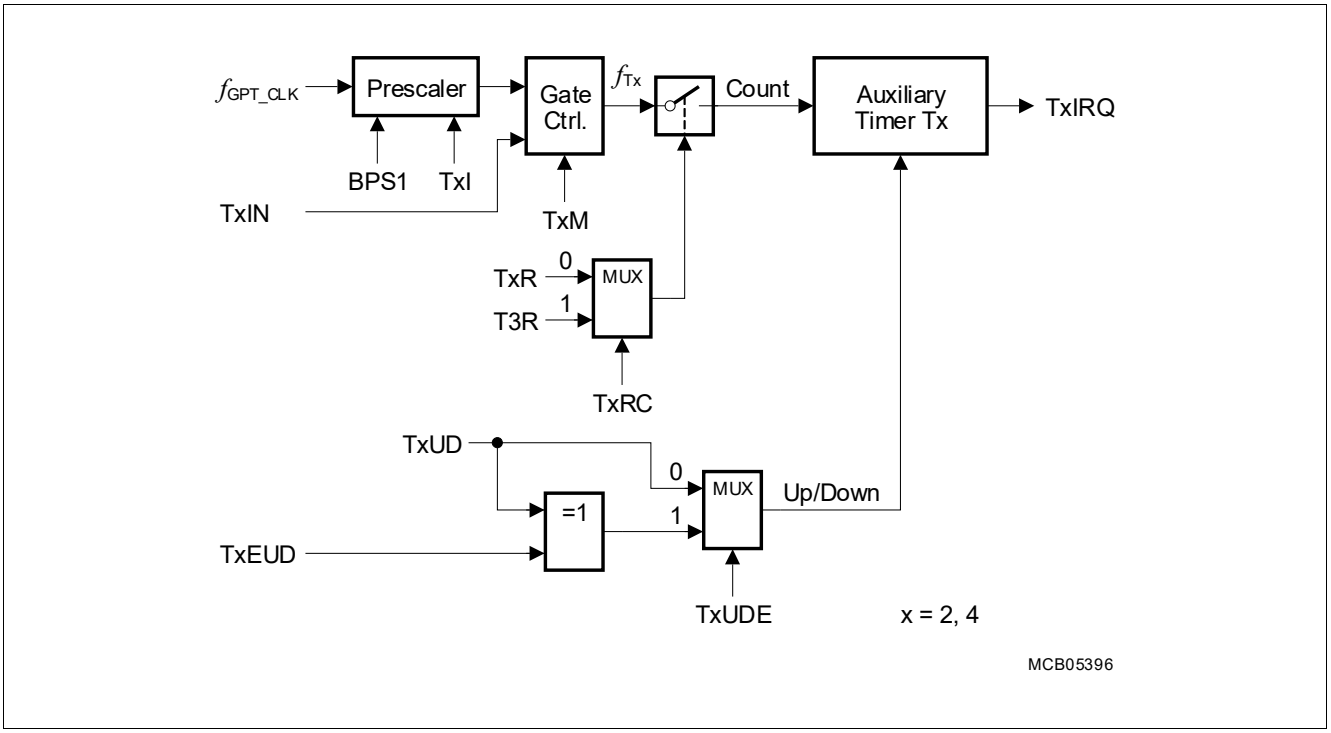


Figure 274 Block diagram of an auxiliary timer in Gated Timer mode

Note: There is no output toggle latch for T2 and T4.
Start/stop of an auxiliary timer can be controlled locally or remotely.

23.5.4.3 Timers T2 and T4 in Counter mode

Counter mode for an auxiliary timer Tx is selected by setting bitfield TxM in register TxCON to 001_B. In Counter mode, an auxiliary timer can be clocked either by a transition at its external input line TxIN, or by a transition of timer T3's toggle latch T3OTL. The event causing an increment or decrement of a timer can be a positive, a negative, or both a positive and a negative transition at either the respective input pin or at the toggle latch. Bitfield TxI in control register TxCON selects the triggering transition (see **Table 280**).

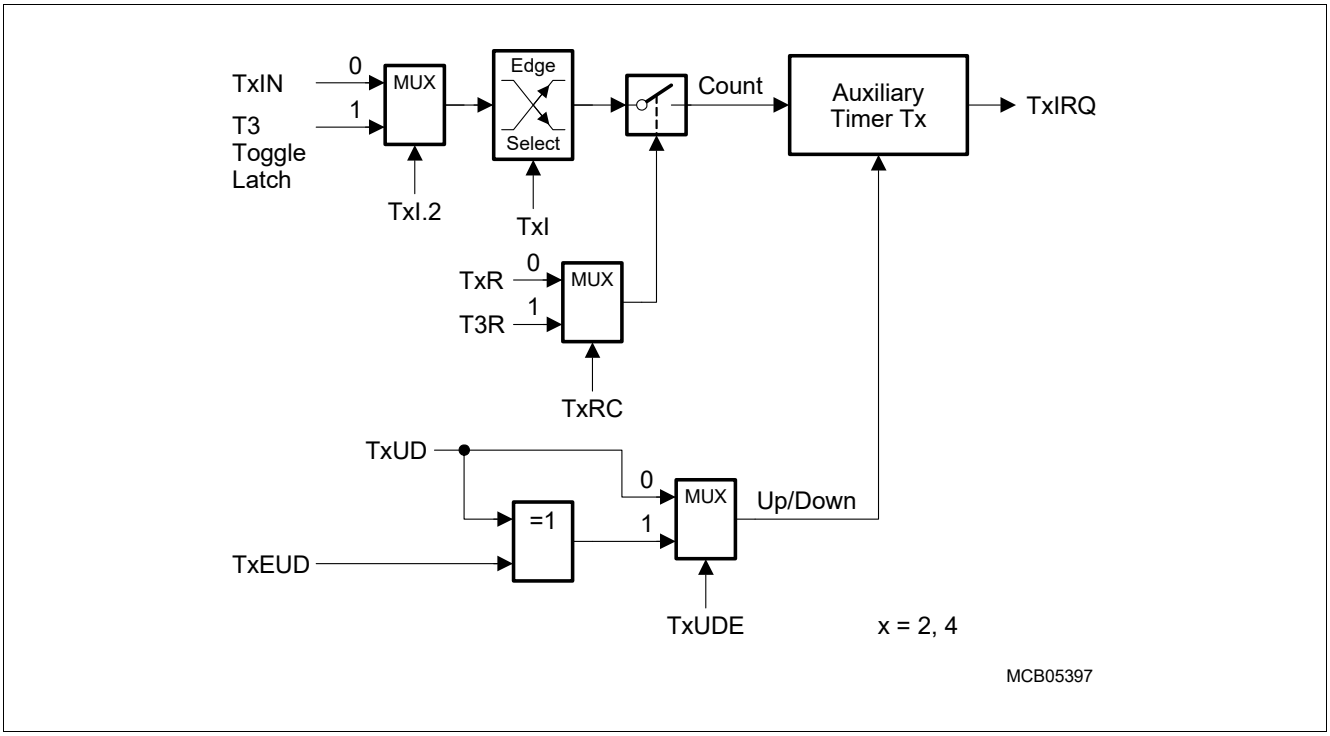


Figure 275 Block diagram of an auxiliary timer in Counter mode

Note: Only state transitions of T3OTL which are caused by the overflows/underflows of T3 will trigger the counter function of T2/T4. Modifications of T3OTL via software will NOT trigger the counter function of T2/T4.

For counter operation, pin TxIN must be configured as input. The maximum input frequency allowed in Counter mode depends on the selected prescaler value. To ensure that a transition of the count input signal applied to TxIN is recognized correctly, its level must be held high or low for a minimum number of module clock cycles before it changes. This information can be found in **Chapter 23.5.5**.

23.5.4.4 Timer concatenation

Using the toggle bit T3OTL as a clock source for an auxiliary timer in Counter mode concatenates the core timer T3 with the respective auxiliary timer. This concatenation forms either a 32-bit or a 33-bit timer/counter, depending on which transition of T3OTL is selected to clock the auxiliary timer.

- 32-bit timer/counter:
  If both a positive and a negative transition of T3OTL are used to clock the auxiliary timer, this timer is clocked on every overflow/underflow of the core timer T3. Thus, the two timers form a 32-bit timer.
- 33-bit timer/counter:
  If either a positive or a negative transition of T3OTL is selected to clock the auxiliary timer, this timer is clocked on every second overflow/underflow of the core timer T3. This configuration forms a 33-bit timer (16-bit core timer + T3OTL + 16-bit auxiliary timer).
  As long as bit T3OTL is not modified by software, it represents the state of the internal toggle latch, and can be regarded as part of the 33-bit timer.

The count directions of the two concatenated timers are not required to be the same. This offers a wide variety of different configurations.

T3, which represents the low-order part of the concatenated timer, can operate in Timer mode, Gated Timer mode or Counter mode in this case.

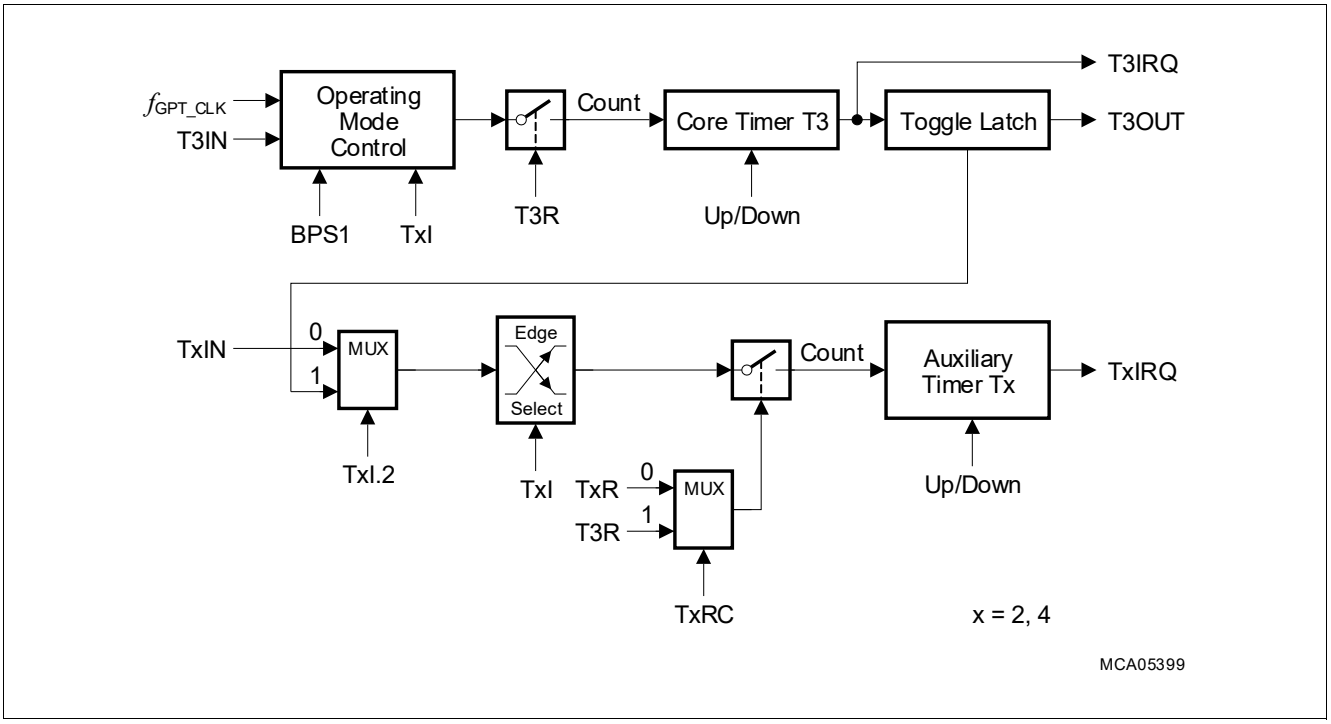


Figure 276 Concatenation of core timer T3 and an auxiliary timer

For measuring longer time periods, the core timer T3 may be concatenated with an auxiliary timer (T2/T4). The core timer contains the low part, and the auxiliary timer contains the high part of the extended timer value.

23.5.4.5 Timers T2 and T4 in Capture mode

Capture mode for an auxiliary timer Tx is selected by setting bitfield TxM in the respective register TxCON to 101_B. In Capture mode, the contents of the core timer T3 are latched into an auxiliary timer register in response to a signal transition at the respective auxiliary timer's external input pin TxIN. The capture trigger signal can be a positive, a negative, or both a positive and a negative transition.

The two least significant bits of bitfield TxI select the active transition (see **Table 279**). Bit 2 of TxI is irrelevant for Capture mode and must be cleared (TxI.2 = 0).

Note: When programmed for Capture mode, the respective auxiliary timer (T2 or T4) stops independently of its run flag T2R or T4R.

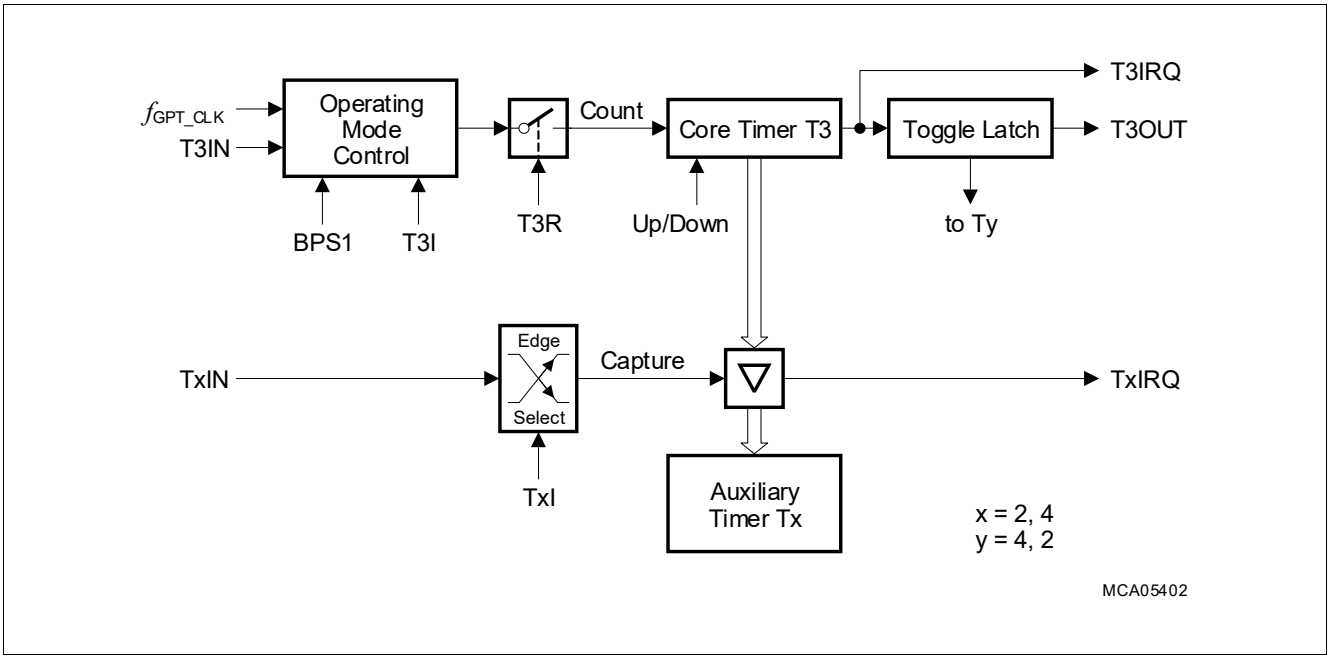


Figure 277 GPT1 auxiliary timer in Capture mode

Upon a trigger (selected transition) at the corresponding input pin TxIN the contents of the core timer are loaded into the auxiliary timer register and the associated interrupt request flag TxIR will be set.

For Capture mode operation, the respective timer input pin TxIN must be configured as input. To ensure that a transition of the capture input signal applied to TxIN is recognized correctly, its level must be held high or low for a minimum number of module clock cycles, detailed in **Chapter 23.5.5**.

23.5.4.6 Timers T2 and T4 in Incremental Interface mode

Incremental Interface mode for an auxiliary timer Tx is selected by setting bitfield TxM in the respective register TxCON to 110_B or 111_B. In Incremental Interface mode, the two inputs associated with an auxiliary timer Tx (TxIN, TxEUD) are used to interface to an incremental encoder. Tx is clocked by each transition on one or both of the external input pins to provide 2-fold or 4-fold resolution of the encoder input.

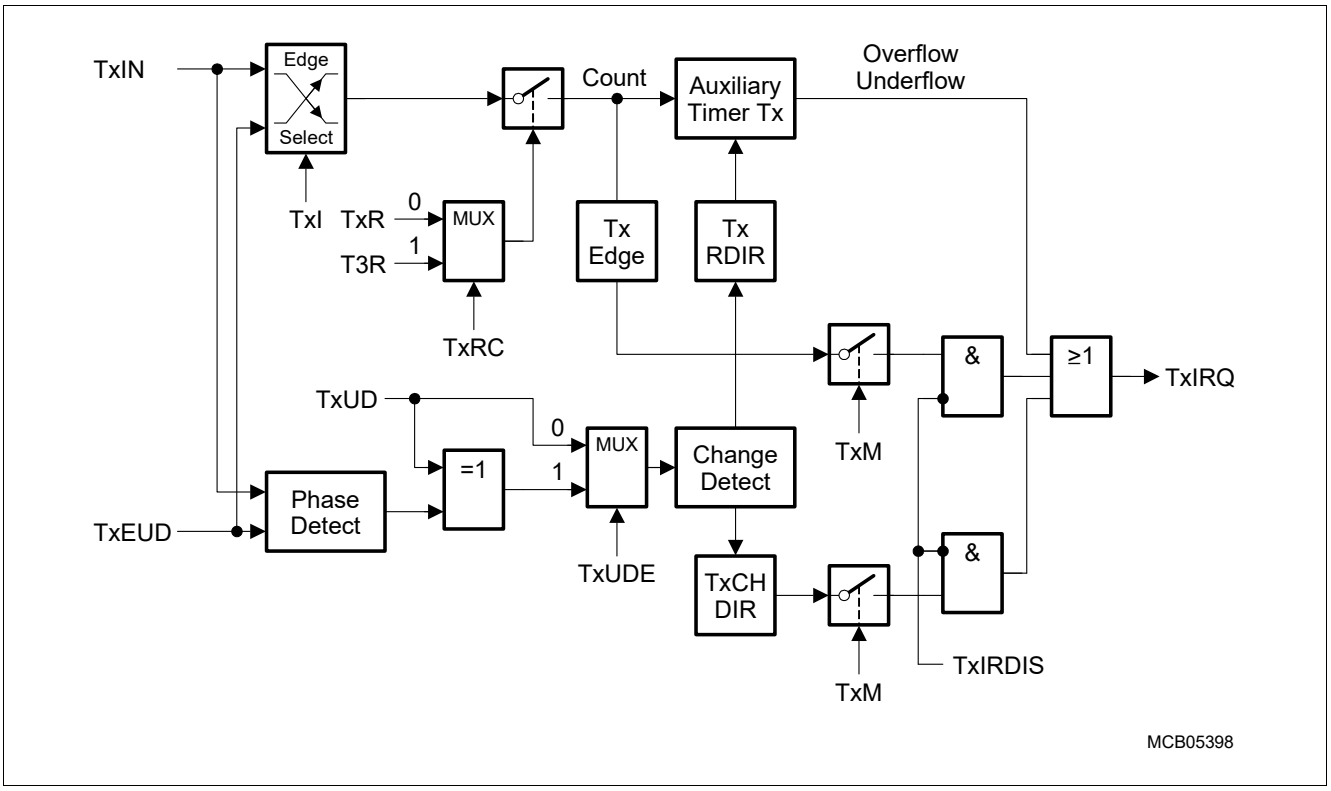


Figure 278 Block diagram of an auxiliary timer in Incremental Interface mode

The operation of the auxiliary timers T2 and T4 in Incremental Interface mode and the interrupt generation are the same as described for the core timer T3. The descriptions, figures and tables apply accordingly.

*Note: Timers T2 and T4 operating in Incremental Interface mode automatically provide information on the sensor's current position. For dynamic information (speed, acceleration, deceleration) see **Combined Capture modes**).*

23.5.4.7 Timers T2 and T4 in Reload mode

Reload mode for an auxiliary timer Tx is selected by setting bitfield TxM in the respective register TxCON to 100_B. In Reload mode, the core timer T3 is reloaded with the contents of an auxiliary timer register, triggered by one of two different signals. The trigger signal is selected the same way as the clock source for Counter mode (see **Table 280**), i.e. a transition of the auxiliary timer's input TxIN or the toggle latch T3OTL may trigger the reload.

Note: When programmed for Reload mode, the respective auxiliary timer (T2 or T4) stops independently of its run flag T2R or T4R.
The timer input pin TxIN must be configured as input if it shall trigger a reload operation.

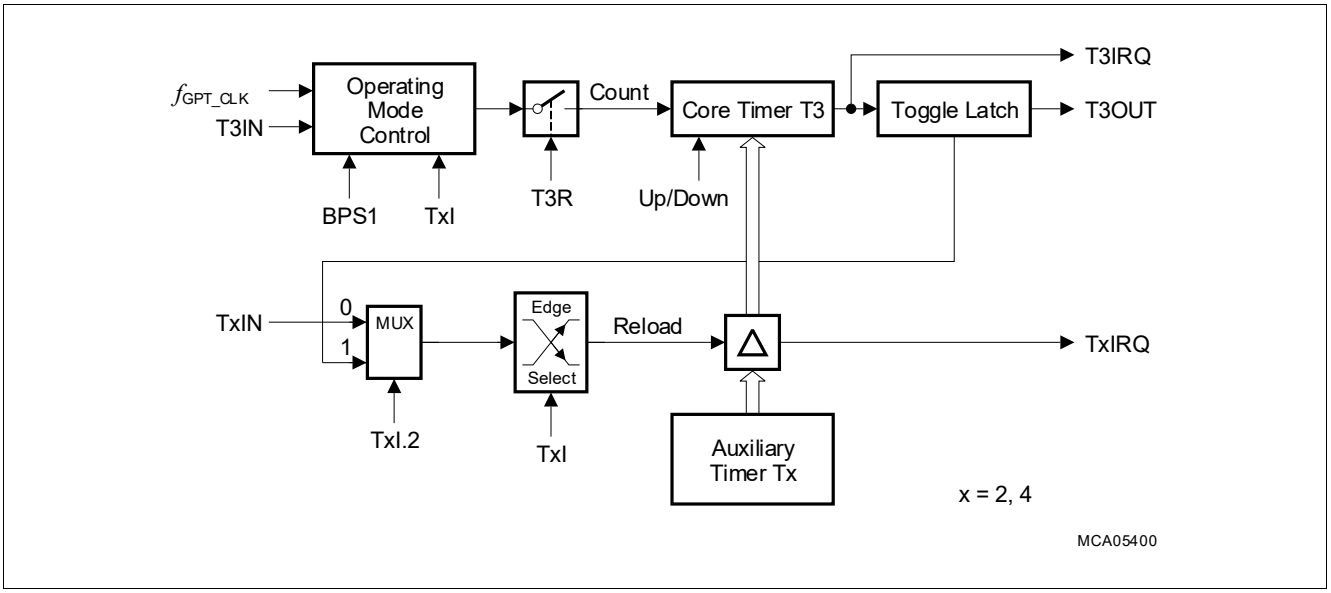


Figure 279 GPT1 auxiliary timer in Reload mode

Upon a trigger signal, T3 is loaded with the contents of the respective timer register (**T2** or **T4**) and the respective interrupt request flag (T2IR or T4IR) is set.

Note: When a T3OTL transition is selected for the trigger signal, the interrupt request flag T3IR will also be set upon a trigger, indicating T3's overflow or underflow. Modifications of T3OTL via software will NOT trigger the counter function of T2/T4.

To ensure that a transition of the reload input signal applied to TxIN is recognized correctly, its level must be held high or low for a minimum number of module clock cycles, detailed in **Chapter 23.5.5**.

The Reload mode triggered by the T3 toggle latch can be used in a number of different configurations. The following functions can be performed, depending on the selected active transition:

- If both a positive and a negative transition of T3OTL are selected to trigger a reload, the core timer will be reloaded with the contents of the auxiliary timer each time it overflows or underflows. This is the standard Reload mode (reload on overflow/underflow).
- If either a positive or a negative transition of T3OTL is selected to trigger a reload, the core timer will be reloaded with the contents of the auxiliary timer on every second overflow or underflow.
- Using this "single-transition" mode for both auxiliary timers allows to perform very flexible Pulse Width Modulation (PWM). One of the auxiliary timers is programmed to reload the core timer on a positive transition of T3OTL, the other is programmed for a reload on a negative transition of T3OTL. With this combination the core timer is alternately reloaded from the two auxiliary timers.

Figure 280 shows an example for the generation of a PWM signal using the “single-transition” reload mechanism. T2 defines the high time of the PWM signal (reloaded on positive transitions) and T4 defines the low time of the PWM signal (reloaded on negative transitions). The PWM signal can be output on pin T3OUT if T3OE = 1. With this method, the high and low time of the PWM signal can be varied in a wide range.

Note: The output toggle latch T3OTL is accessible via software and may be changed, if required, to modify the PWM signal.
However, this will NOT trigger the reloading of T3.

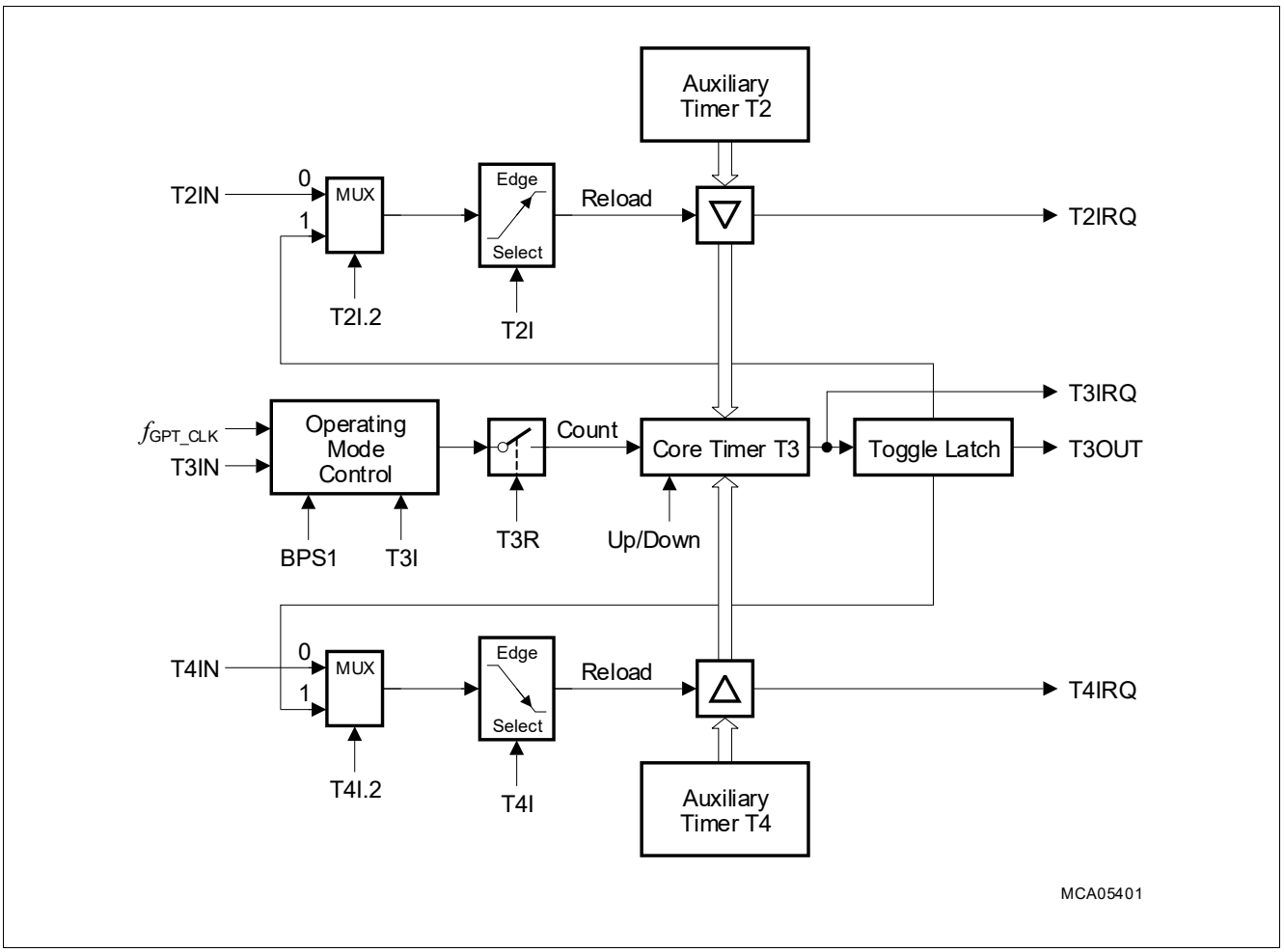


Figure 280 GPT1 timer reload configuration for PWM generation

Note: Although possible, selecting the same reload trigger event for both auxiliary timers should be avoided. In such a case, both reload registers would try to load the core timer at the same time. If this combination is selected, T2 is disregarded and the contents of T4 is reloaded.

23.5.5 GPT1 clock signal control

All actions within the timer block GPT1 are triggered by transitions of its basic clock. This basic clock is derived from the system clock by a basic block prescaler, controlled by bitfield BPS1 in register **T3CON** (see **Figure 261**). The count clock can be generated in two different ways:

- Internal count clock: derived from GPT1's basic clock via a programmable prescaler, is used for (Gated) Timer mode.
- External count clock: derived from the timer's input pin(s), is used for Counter mode.

For both ways, the basic clock determines the maximum count frequency and the timer's resolution:

Table 273 Basic clock selection for block GPT1

Block prescaler[1]	**BPS1 = 01_B**	**BPS1 = 00_B**[2]	**BPS1 = 11_B**	**BPS1 = 10_B**
Prescaling factor for GPT1: F(BPS1)	F(BPS1) = 4	F(BPS1) = 8	F(BPS1) = 16	F(BPS1) = 32
Maximum external count frequency	$f_{GPT\_CLK}/8$	$f_{GPT\_CLK}/16$	$f_{GPT\_CLK}/32$	$f_{GPT\_CLK}/64$
Input signal stable time	$4 \times t_{GPT\_CLK}$	$8 \times t_{GPT\_CLK}$	$16 \times t_{GPT\_CLK}$	$32 \times t_{GPT\_CLK}$

1) Please note the non-linear encoding of bitfield BPS1.
2) Default after reset.

*Note: When initializing the GPT1 block, and the block prescaler BPS1 in register **T3CON** needs to be set to a value different from its reset value (00_B), it must be initialized first before any mode involving external trigger signals is configured. These modes include counter, incremental interface, capture, and Reload mode. Otherwise, unintended count/capture/reload events may occur.*
*In this case (e.g. when changing BPS1 during operation of the GPT1 block), disable related interrupts before modification of BPS1, and afterwards clear the corresponding service request flags and re-initialize those registers (**T2**, **T3**, **T4**) that might be affected by a count/capture/reload event.*

23.5.5.1 Internal count clock generation

In Timer mode and Gated Timer mode, the count clock for each GPT1 timer is derived from the GPT1 basic clock by a programmable prescaler, controlled by bitfield TxI in the respective timer's control register TxCON.

The count frequency f_{Tx} for a timer Tx and its resolution r_{Tx} are scaled linearly with lower clock frequencies, as can be seen from the following formula:

$$f_{Tx} = \frac{f_{GPT\_CLK}}{F(BPS1) \bullet 2^{<TxI>}} \qquad r_{Tx}[\mu s] = \frac{F(BPS1) \bullet 2^{<TxI>}}{f_{GPT\_CLK}[MHz]} \tag{23.1}$$

The effective count frequency depends on the common module clock prescaler factor F(BPS1) as well as on the individual input prescaler factor $2^{<TxI>}$. **Table 277** summarizes the resulting overall divider factors for a GPT1 timer that result from these cascaded prescalers.

Table 274 lists GPT1 timer's parameters (such as count frequency, resolution, and period) resulting from the selected overall prescaler factor and the module clock f_{GPT}. Note that some numbers may be rounded.

Table 274 GPT1 timer parameters

Module clock $f_{GPT_}$CLK = 10 MHz			Overall prescaler factor	Module clock f_{GPT_CLK} = 40 MHz		
Frequency	Resolution	Period		Frequency	Resolution	Period
2.5 MHz	400 ns	26.21 ms	4	10.0 MHz	100 ns	6.55 ms
1.25 MHz	800 ns	52.43 ms	8	5.0 MHz	200 ns	13.11 ms
625.0 kHz	1.6 µs	104.9 ms	16	2.5 MHz	400 ns	26.21 ms
312.5 kHz	3.2 µs	209.7 ms	32	1.25 MHz	800 ns	52.43 ms
156.25 kHz	6.4 µs	419.4 ms	64	625.0 kHz	1.6 µs	104.9 ms
78.125 kHz	12.8 µs	838.9 ms	128	312.5 kHz	3.2 µs	209.7 ms
39.06 kHz	25.6 µs	1.678 s	256	156.25 kHz	6.4 µs	419.4 ms
19.53 kHz	51.2 µs	3.355 s	512	78.125 kHz	12.8 µs	838.9 ms
9.77 kHz	102.4 µs	6.711 s	1024	39.06 kHz	25.6 µs	1.678 s
4.88 kHz	204.8 µs	13.42 s	2048	19.53 kHz	51.2 µs	3.355 s
2.44 kHz	409.6 µs	26.84 s	4096	9.77 kHz	102.4 µs	6.711 s

23.5.5.2 External count clock input

The external input signals of the GPT1 block are sampled with the GPT1 basic clock (see **Figure 261**). To ensure that a signal is recognized correctly, its current level (high or low) must be held active for at least one complete sampling period, before changing. A signal transition is recognized if two subsequent samples of the input signal represent different levels. Therefore, a minimum of two basic clock periods are required for the sampling of an external input signal. Thus, the maximum frequency of an input signal must not be higher than half the basic clock.

Table 275 summarizes the resulting requirements for external GPT1 input signals.

Table 275 GPT1 external input signal limits

GPT1 basic clock = 10 MHz		Input frequ. factor	GPT1 divider BPS1	Input phase duration	GPT1 basic clock = 40 MHz	
Max. input frequency	Min. level hold time				Max. input frequency	Min. level hold time
1.25 MHz	400 ns	$f_{GPT\_CLK}/8$	01_B	$4 \times t_{GPT\_CLK}$	5.0 MHz	100 ns
625.0 kHz	800 ns	$f_{GPT\_CLK}/16$	00_B	$8 \times t_{GPT\_CLK}$	2.5 MHz	200 ns
312.5 kHz	1.6 µs	$f_{GPT\_CLK}/32$	11_B	$16 \times t_{GPT\_CLK}$	1.25 MHz	400 ns
156.25 kHz	3.2 µs	$f_{GPT\_CLK}/64$	10_B	$32 \times t_{GPT\_CLK}$	625.0 kHz	800 ns

These limitations are valid for all external input signals to GPT1, including the external count signals in Counter Mode and Incremental Interface mode, the gate input signals in Gated Timer Mode, and the external direction signals.

23.5.6 GPT1 encoding

23.5.6.1 Encoding of GPT1 timer count direction control

Table 276 GPT1 timer count direction control

Pin TxEUD	Bit TxUDE	Bit TxUD	Count direction	Bit TxRDIR
X	0	0	Count up	0
X	0	1	Count down	1
0	1	0	Count up	0
1	1	0	Count down	1
0	1	1	Count down	1
1	1	1	Count up	0

23.5.6.2 Timer mode and Gated Timer mode: encoding of GPT1 overall prescaler factor

Table 277 GPT1 overall prescaler factors for internal count clock (Timer mode and Gated Timer mode)

Individual prescaler for Tx	Common prescaler for module clock[1)]			
	BPS1 = 01_B	**BPS1 = 00_B**	**BPS1 = 11_B**	**BPS1 = 10_B**
TxI = 000_B	4	8	16	32
TxI = 001_B	8	16	32	64
TxI = 010_B	16	32	64	128
TxI = 011_B	32	64	128	256
TxI = 100_B	64	128	256	512
TxI = 101_B	128	256	512	1024
TxI = 110_B	256	512	1024	2048
TxI = 111_B	512	1024	2048	4096

1) Please note the non-linear encoding of bitfield BPS1.

23.5.6.3 Counter mode: encoding of GPT1 input edge selection

Table 278 GPT1 core timer T3 input edge selection (Counter mode)

T3I	Triggering edge for counter increment/decrement
000_B	None. Counter T3 is disabled
001_B	Positive transition (rising edge) on T3IN
010_B	Negative transition (falling edge) on T3IN
011_B	Any transition (rising or falling edge) on T3IN
$1XX_B$	Reserved. Do not use this combination

Table 279 GPT1 auxiliary timers T2/T4 input edge selection (Capture mode)

T2I/T4I	Triggering edge for counter increment/decrement
000_B	None. Counter Tx is disabled
001_B	Positive transition (rising edge) on TxIN
010_B	Negative transition (falling edge) on TxIN
011_B	Any transition (rising or falling edge) on TxIN
$1XX_B$	Reserved. Do not use this combination

Table 280 GPT1 auxiliary timers T2/T4 input edge selection (Counter mode, Reload mode)

T2I/T4I	Triggering edge for counter increment/decrement
$X00_B$	None. Counter Tx is disabled
001_B	Positive transition (rising edge) on TxIN
010_B	Negative transition (falling edge) on TxIN
011_B	Any transition (rising or falling edge) on TxIN
101_B	Positive transition (rising edge) of T3 toggle latch T3OTL
110_B	Negative transition (falling edge) of T3 toggle latch T3OTL
111_B	Any transition (rising or falling edge) of T3 toggle latch T3OTL

23.5.6.4 Incremental Interface mode: encoding of input edge selection

Table 281 GPT1 core timer T3 input edge selection (Incremental Interface mode)

T3I	Triggering edge for counter increment/decrement
000_B	None. Counter T3 stops
001_B	Any transition (rising or falling edge) on T3IN
010_B	Any transition (rising or falling edge) on T3EUD
011_B	Any transition (rising or falling edge) on any T3 input (T3IN or T3EUD)
$1XX_B$	Reserved. Do not use this combination

23.6 Timer block GPT2

From a programmer's point of view, the GPT2 block is represented by a set of SFRs as summarized below. Those portions of port and direction registers which are used for alternate functions by the GPT2 block are shaded.

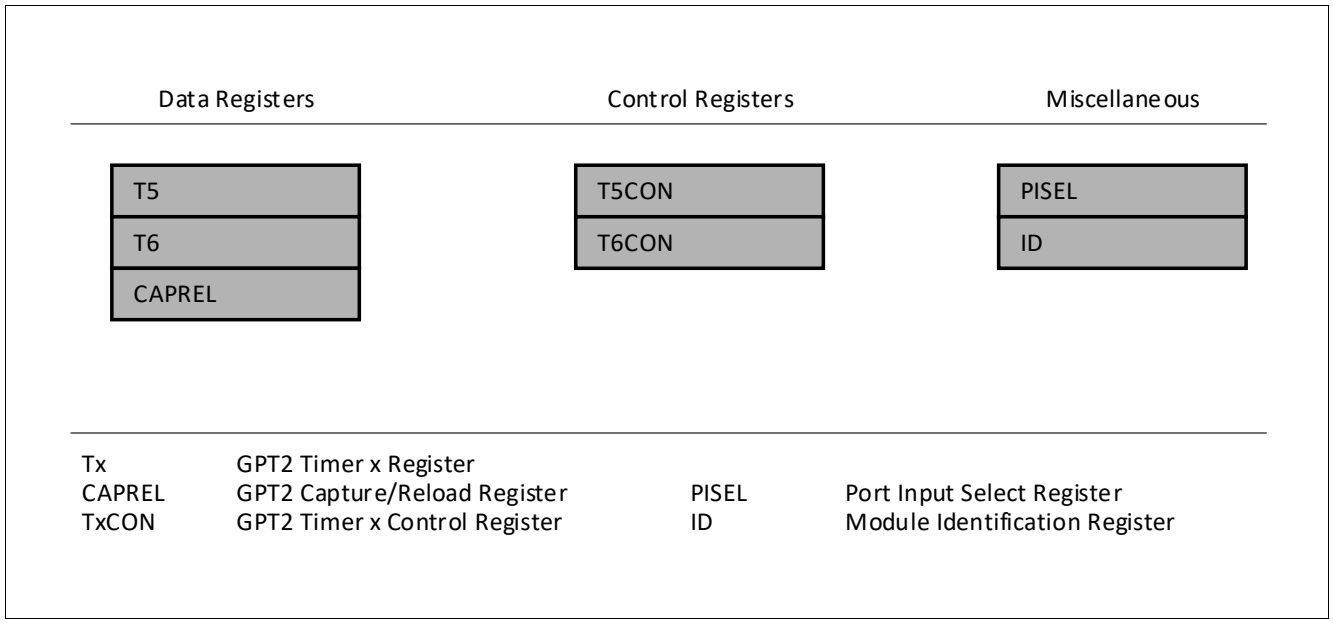


Figure 281 SFRs associated with timer block GPT2

Both timers of block GPT2 (T5, T6) can run in one of 3 basic modes: Timer mode, Gated Timer mode, or Counter mode. All timers can count up or down. Each timer of GPT2 is controlled by a separate control register TxCON.

Each timer has an input pin TxIN (alternate pin function) associated with it, which serves as the gate control in Gated Timer mode, or as the count input in Counter mode. The count direction (up/down) may be programmed via software or may be dynamically altered by a signal at the External Up/Down control input TxEUD (alternate pin function). An overflow/underflow of core timer T6 is indicated by the Output Toggle Latch T6OTL, whose state may be output on the associated pin T6OUT (alternate pin function). The auxiliary timer T5 may additionally be concatenated with core timer T6 (through T6OTL).

The Capture/Reload register **CAPREL** can be used to capture the contents of timer T5, or to reload timer T6. A special mode facilitates the use of register **CAPREL** for both functions at the same time. This mode allows frequency multiplication. The capture function is triggered by the input pin CAPIN, or by GPT1 timer's T3 input lines T3IN and T3EUD. The reload function is triggered by an overflow or underflow of timer T6. Overflows/underflows of timer T6 may also clock the timers of the CAPCOM units.

The current contents of each timer can be read or modified by the CPU by accessing the corresponding timer count registers **T5** or **T6**, located in the SFR space (see **Chapter 23.7**). When any of the timer registers is written to by the CPU in the state immediately preceding a timer increment, decrement, reload, or capture operation, the CPU write operation has priority in order to guarantee correct results.

The interrupts of GPT2 are controlled through the SCU.GPTIEN, SCU.GPTIS, SCU.GPTISC and SCU.GPTISS. These registers are not part of the GPT2 block.

The input and output lines of GPT2 are connected to pins. The control registers for the port functions are located in the respective port modules.

*Note: The timing requirements for external input signals can be found in **Chapter 23.6.6**, **Chapter 23.3** summarizes the module interface signals, including pins.*

23.6.1 GPT2 core timer T6 control

The current contents of the core timer T6 are reflected by its count register **T6**. This register can also be written to by the CPU, for example, to set the initial start value.

The core timer T6 is configured and controlled via its control register **T6CON**.

23.6.1.1 Timer T6 run control

The core timer T6 can be started or stopped by software through bit T6R (timer T6 run bit). This bit is relevant in all operating modes of T6. Setting bit T6R will start the timer, clearing bit T6R stops the timer.

In Gated Timer mode, the timer will only run if T6R = 1 and the gate is active (high or low, as programmed).

*Note: When bit T5RC in timer control register **T5CON** is set, bit T6R will also control (start and stop) the Auxiliary Timer T5.*

23.6.1.2 Count direction control

The count direction of the GPT2 timers (core timer and auxiliary timer) can be controlled either by software or by the external input pin TxEUD (Timer Tx External Up/Down Control Input). These options are selected by bits TxUD and TxUDE in the respective control register TxCON. When the up/down control is provided by software (bit TxUDE = 0), the count direction can be altered by setting or clearing bit TxUD. When bit TxUDE = 1, pin TxEUD is selected to be the controlling source of the count direction. However, bit TxUD can still be used to reverse the actual count direction, as shown in **Table 286**. The count direction can be changed regardless of whether or not the timer is running.

Note: When pin TxEUD is used as external count direction control input, it must be configured as input.

23.6.1.3 Timer T6 output toggle latch

The overflow/underflow signal of timer T6 is connected to a block named ‘Toggle Latch’, shown in the Timer mode diagrams. **Figure 282** illustrates the details of this block. An overflow or underflow of T6 will clock two latches: The first latch represents bit T6OTL in control register **T6CON**. The second latch is an internal latch toggled by T6OTL’s output. Both latch outputs are connected to the input control block of the auxiliary timer T5. The output level of the shadow latch will match the output level of T6OTL, but is delayed by one clock cycle. When the T6OTL value changes, this will result in a temporarily different output level from T6OTL and the shadow latch, which can trigger the selected count event in T5.

When software writes to T6OTL, both latches are set or cleared simultaneously. In this case, both signals to the auxiliary timers carry the same level and no edge will be detected. Bit T6OE (overflow/underflow output enable) in register **T6CON** enables the state of T6OTL to be monitored via an external pin T6OUT. When T6OTL is linked to an external port pin (must be configured as output), T6OUT can be used to control external HW. If T6OE = 1, pin T6OUT outputs the state of T6OTL. If T6OE = 0, pin T6OUT outputs a high level (while it selects the timer output signal).

As can be seen from **Figure 282**, when latch T6OTL is modified by software to determine the state of the output line, also the internal shadow latch is set or cleared accordingly. Therefore, no trigger condition is detected by T5 in this case.

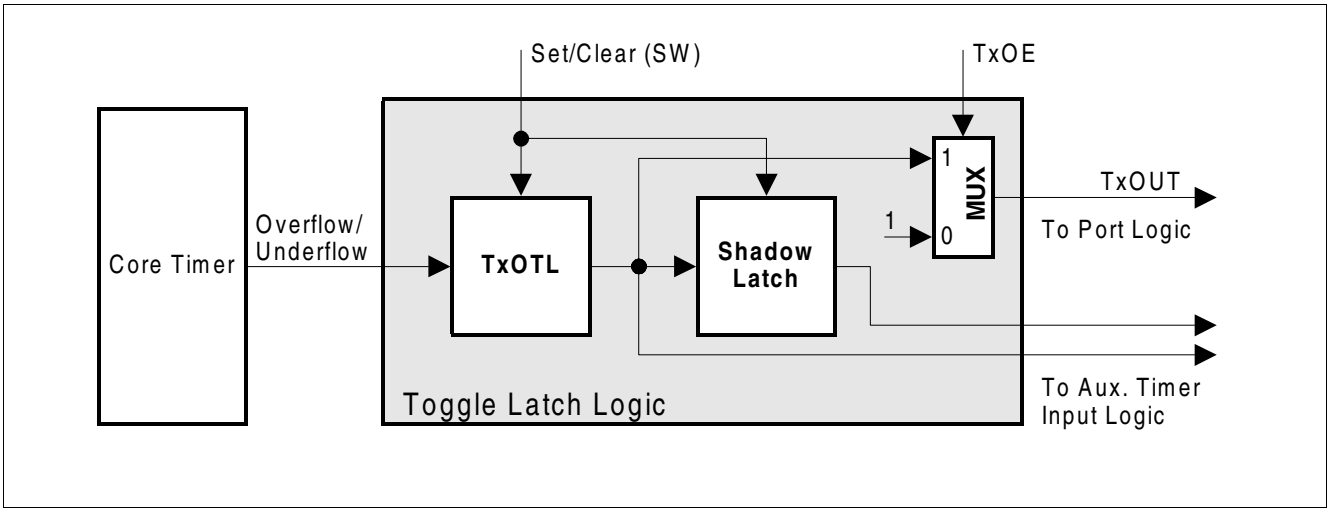


Figure 282 Block diagram of the toggle latch logic of core timer T6 (x = 6)

Note: T6 is also used to clock the timers in the CAPCOM units. For this purpose, there is a direct internal connection between the T6 overflow/underflow line and the CAPCOM timers (signal T6OUF).

23.6.2 GPT2 core timer T6 operating modes

Timer T6 can operate in one of the following modes:

- **Timer T6 in Timer mode**
- **Timer T6 in Gated Timer mode**
- **Timer T6 in Counter mode**

23.6.2.1 Timer T6 in Timer mode

Timer mode for the core timer T6 is selected by setting bitfield T6M in register **T6CON** to 000_B. In this mode, T6 is clocked with the module's input clock f_{GPT} divided by two programmable prescalers controlled by bitfields BPS2 and T6I in register **T6CON**. Please see **Chapter 23.6.6** for details on the input clock options.

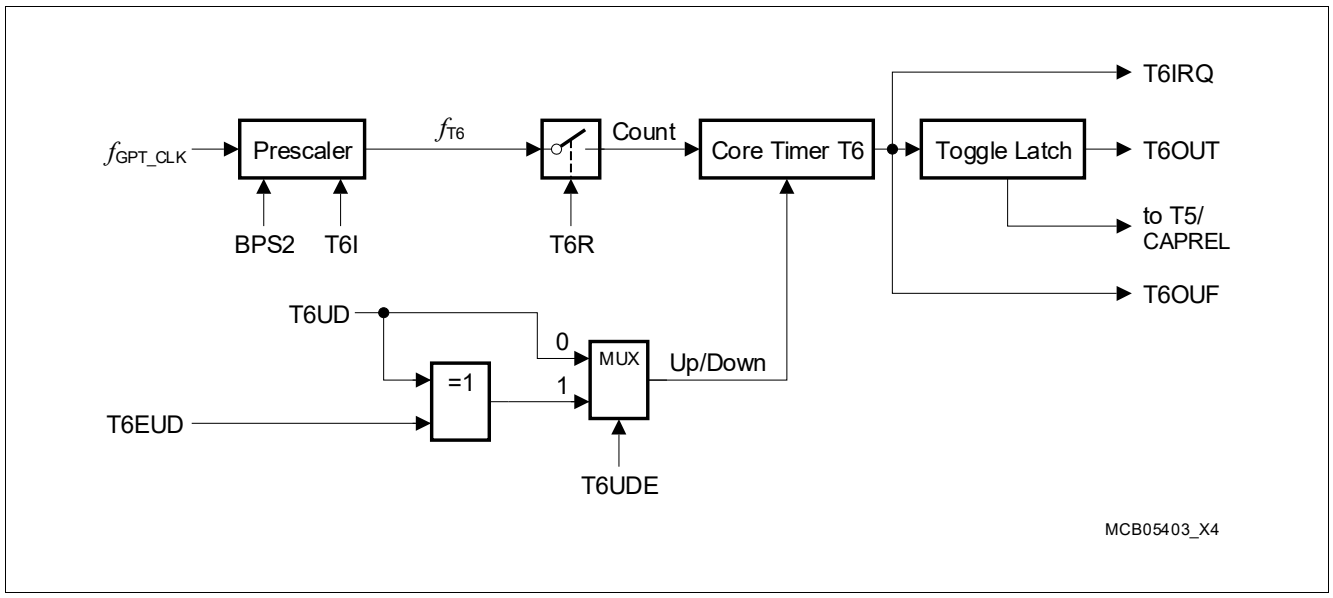


Figure 283 Block diagram of core timer T6 in Timer mode

23.6.2.2 Timer T6 in Gated Timer mode

Gated Timer mode for the core timer T6 is selected by setting bitfield T6M in register **T6CON** to 010_B or 011_B. Bit T6M.0 (T6CON.3) selects the active level of the gate input. The same options for the input frequency are available in Gated Timer mode as in Timer mode (see **Chapter 23.6.6**). However, the input clock to the timer in this mode is gated by the external input pin T6IN (Timer T6 External Input).
To enable this operation, the associated pin T6IN must be configured as input.

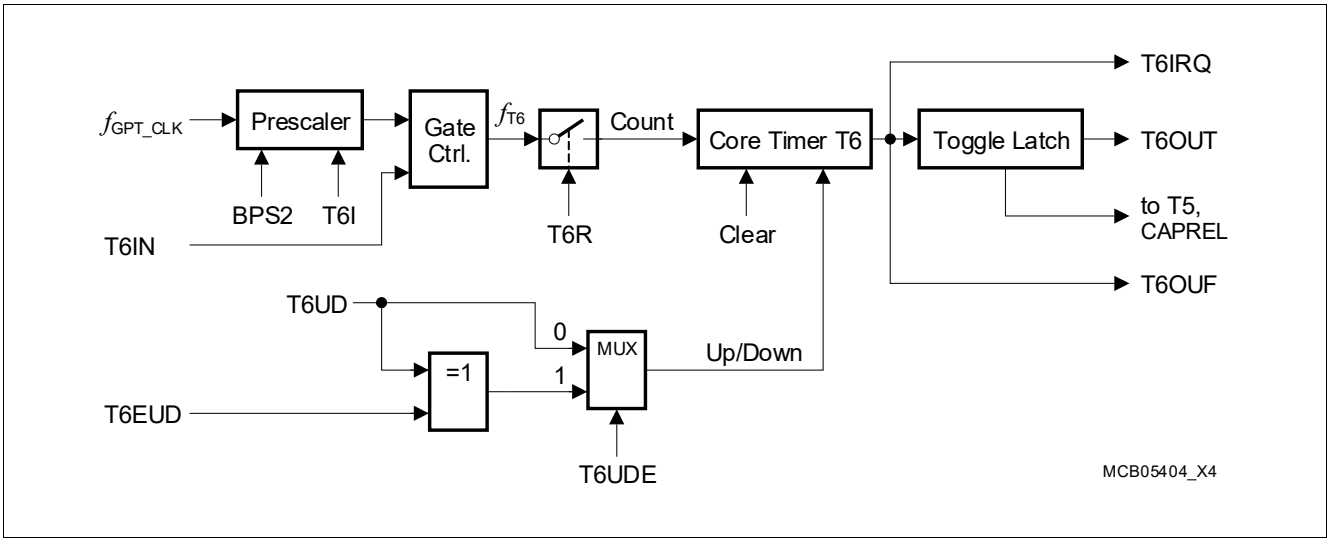


Figure 284 Block diagram of core Timer T6 in Gated Timer mode

If T6M = 010_B, the timer is enabled when T6IN shows a low level. A high level at this line stops the timer. If T6M = 011_B, line T6IN must have a high level in order to enable the timer. Additionally, the timer can be turned on or off by software using bit T6R. The timer will only run if T6R is 1 and the gate is active. It will stop if either T6R is 0 or the gate is inactive.

Note: A transition of the gate signal at pin T6IN does not cause an interrupt request.

23.6.2.3 Timer T6 in Counter mode

Counter mode for the core timer T6 is selected by setting bitfield T6M in register **T6CON** to 001_B. In Counter mode, timer T6 is clocked by a transition at the external input pin T6IN. The event causing an increment or decrement of the timer can be a positive, a negative, or both a positive and a negative transition at this line. Bitfield T6I in control register **T6CON** selects the triggering transition (see **Table 289**).

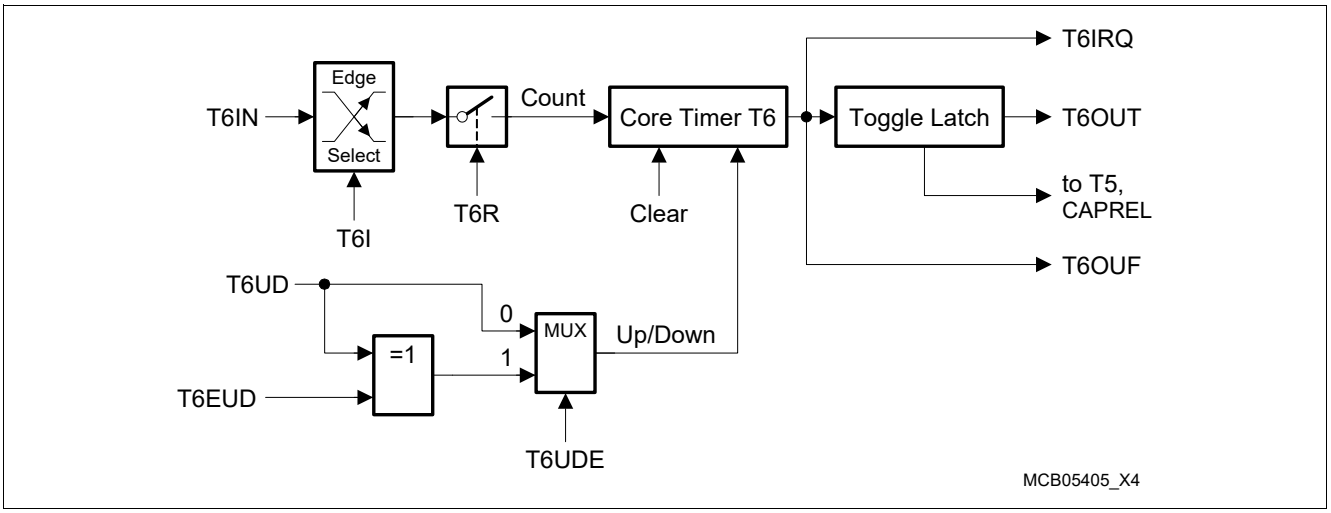


Figure 285 Block diagram of core timer T6 in Counter mode

For Counter mode operation, pin T6IN must be configured as input. The maximum input frequency allowed in Counter mode depends on the selected prescaler value. To ensure that a transition of the count input signal applied to T6IN is recognized correctly, its level must be held high or low for a minimum number of module clock cycles before it changes. This information can be found in **Chapter 23.6.6**.

23.6.3 GPT2 auxiliary timer T5 control

Auxiliary timer T5 can be configured for Timer mode, Gated Timer mode, or Counter mode with the same options for the timer frequencies and the count signal as the core timer T6. In addition to these 3 counting modes, the auxiliary timer can be concatenated with the core timer. The contents of T5 may be captured to register **CAPREL** upon an external or an internal trigger. The start/stop function of the auxiliary timers can be remotely controlled by the T6 run control bit. Several timers may thus be controlled synchronously.

The current contents of the auxiliary timer are reflected by its count register **T5**. This register can also be written to by the CPU, for example, to set the initial start value.

The individual configurations for timer T5 are determined by its control register **T5CON**. Some bits in this register also control the function of the **CAPREL** register. Note that functions which are present in all timers of block GPT2 are controlled in the same bit positions and in the same manner in each of the specific control registers.

Note: The auxiliary timer has no output toggle latch and no alternate output function.

Timer T5 run control

The auxiliary timer T5 can be started or stopped by software in two different ways:

- Through the associated timer run bit (T5R). In this case it is required that the respective control bit T5RC = 0
- Through the core timer's run bit (T6R). In this case the respective remote control bit must be set (T5RC = 1).

The selected run bit is relevant in all operating modes of T5. Setting the bit will start the timer, clearing the bit stops the timer

In Gated Timer mode, the timer will only run if the selected run bit is set and the gate is active (high or low, as programmed)

Note: If remote control is selected T6R will start/stop timer T6 and the auxiliary timer T5 synchronously.

23.6.4 GPT2 auxiliary timer T5 operating modes

The operation of the auxiliary timer in the basic operating modes is almost identical with the core timer's operation, with very few exceptions. Additionally, some combined operating modes can be selected.

23.6.4.1 Timer T5 in Timer mode

Timer mode for the auxiliary timer T5 is selected by setting its bitfield T5M in register **T5CON** to 000_B.

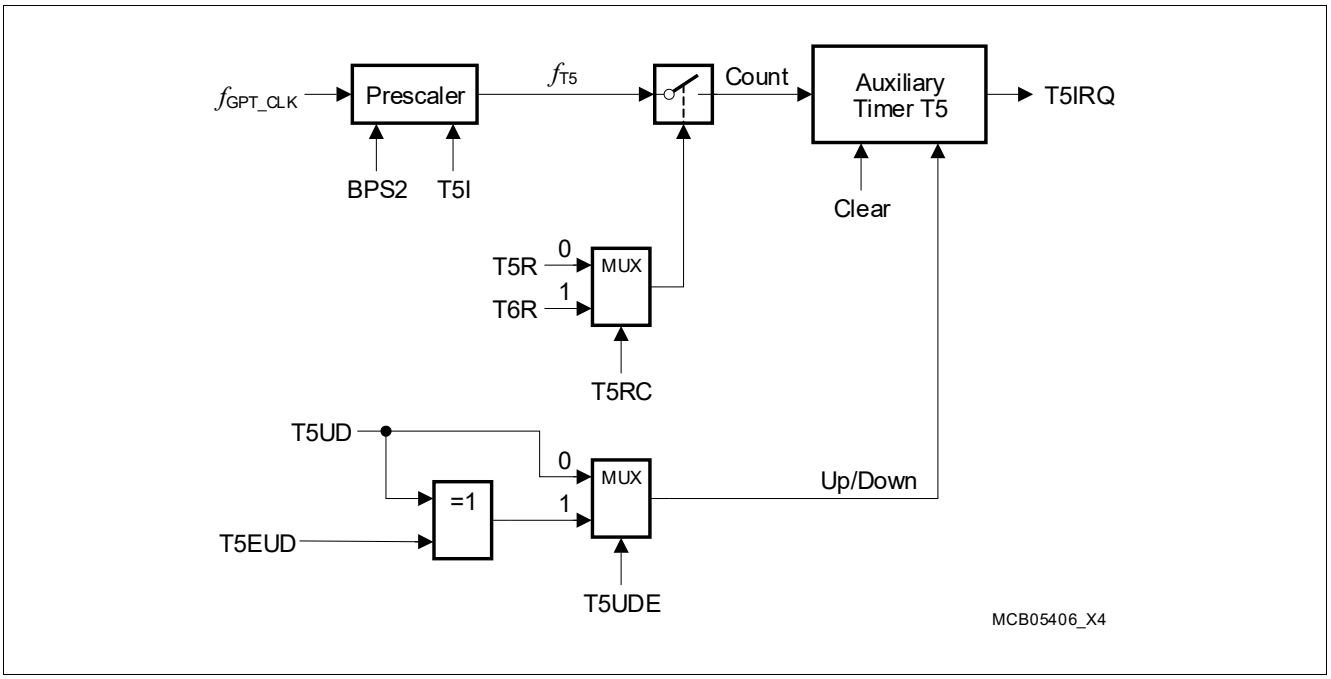


Figure 286 Block diagram of auxiliary timer T5 in Timer mode

23.6.4.2 Timer T5 in Gated Timer mode

Gated Timer mode for the auxiliary timer T5 is selected by setting bitfield T5M in register **T5CON** to 010_B or 011_B. Bit T5M.0 (T5CON.3) selects the active level of the gate input.

Note: A transition of the gate signal at line T5IN does not cause an interrupt request.

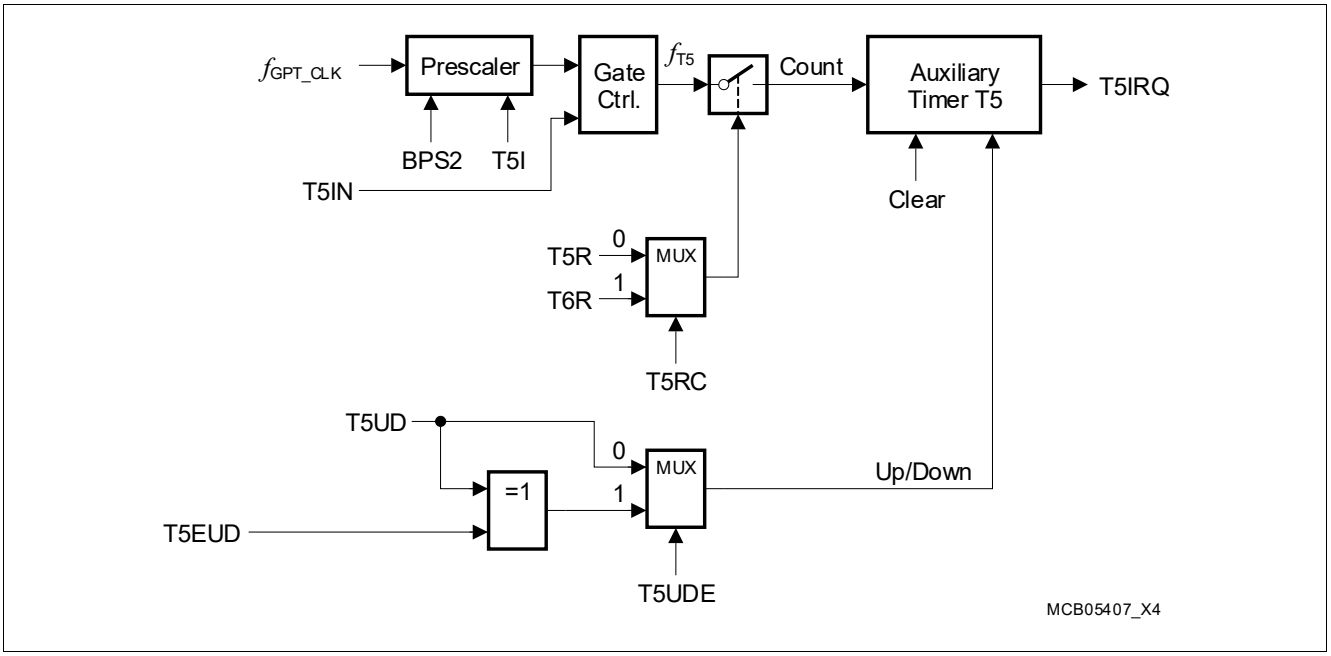


Figure 287 Block diagram of auxiliary timer T5 in Gated Timer mode

Note: There is no output toggle latch for T5.
Start/stop of the auxiliary timer can be controlled locally or remotely.

23.6.4.3 Timer T5 in Counter mode

Counter mode for auxiliary timer T5 is selected by setting bitfield T5M in register **T5CON** to 001_B. In Counter mode, the auxiliary timer can be clocked either by a transition at its external input line T5IN, or by a transition of timer T6's toggle latch T6OTL. The event causing an increment or decrement of a timer can be a positive, a negative, or both a positive and a negative transition at either the respective input pin or at the toggle latch. Bitfield T5I in control register **T5CON** selects the triggering transition (see **Table 288**).

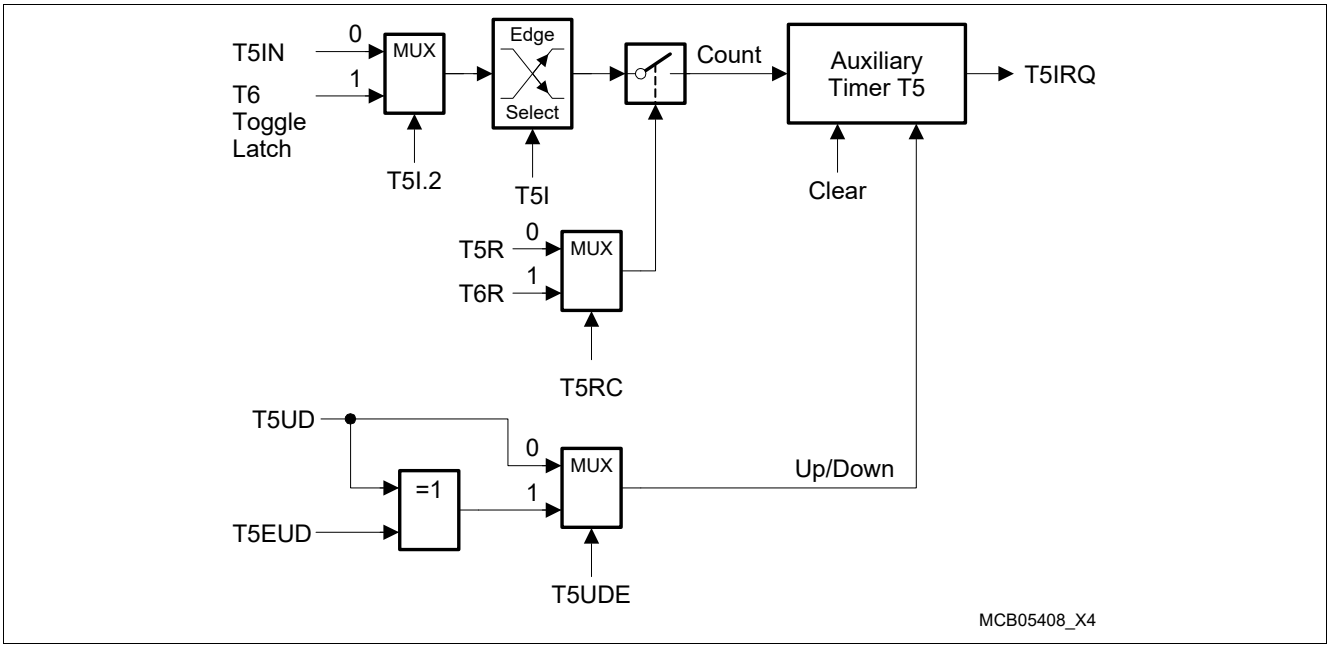


Figure 288 Block diagram of auxiliary timer T5 in Counter mode

Note: Only state transitions of T6OTL which are caused by the overflows/underflows of T6 will trigger the counter function of T5. Modifications of T6OTL via software will NOT trigger the counter function of T5.

For counter operation, pin T5IN must be configured as input. The maximum input frequency allowed in Counter mode depends on the selected prescaler value. To ensure that a transition of the count input signal applied to T5IN is recognized correctly, its level must be held high or low for a minimum number of module clock cycles before it changes. This information can be found in **Chapter 23.6.6**.

23.6.4.4 Timer concatenation

Using the toggle bit T6OTL as a clock source for the auxiliary timer in Counter Mode concatenates the core timer T6 with the auxiliary timer T5. This concatenation forms either a 32-bit or a 33-bit timer/counter, depending on which transition of T6OTL is selected to clock the auxiliary timer.

- 32-bit timer/counter
  If both a positive and a negative transition of T6OTL are used to clock the auxiliary timer, this timer is clocked on every overflow/underflow of the core timer T6. Thus, the two timers form a 32-bit timer.
- 33-bit timer/counter
  If either a positive or a negative transition of T6OTL is selected to clock the auxiliary timer, this timer is clocked on every second overflow/underflow of the core timer T6. This configuration forms a 33-bit timer (16-bit core timer + T6OTL + 16-bit auxiliary timer).
  As long as bit T6OTL is not modified by software, it represents the state of the internal toggle latch, and can be regarded as part of the 33-bit timer.

The count directions of the two concatenated timers are not required to be the same. This offers a wide variety of different configurations.

T6, which represents the low-order part of the concatenated timer, can operate in Timer mode, Gated Timer mode or Counter mode in this case.

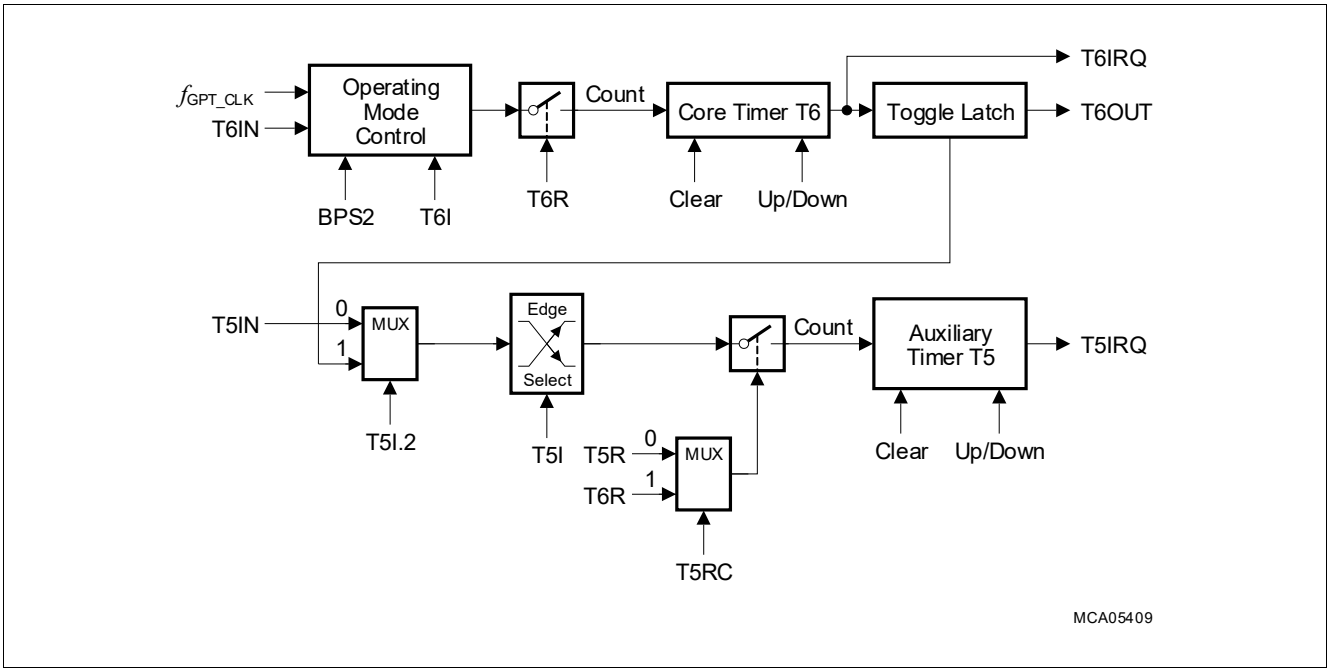


Figure 289 Concatenation of core timer T6 and auxiliary timer T5

23.6.5 GPT2 register CAPREL operating modes

The Capture/Reload register **CAPREL** can be used to capture the contents of timer T5, or to reload timer T6. A special mode facilitates the use of register **CAPREL** for both functions at the same time. This mode allows frequency multiplication. The capture function is triggered by CAPIN, by T3IN and T3EUD, or by read GPT1 timers. The reload function is triggered by an overflow or underflow of timer T6.

In addition to the capture function, the capture trigger signal can also be used to clear the contents of timers T5 and T6 individually.

The functions of register **CAPREL** are controlled via several bit(field)s in the timer control registers **T5CON** and **T6CON**.

23.6.5.1 Capture/Reload register CAPREL in Capture mode

Capture mode for register **CAPREL** is selected by setting bit T5SC in control register **T5CON** (set bitfield CI in register **T5CON** to a non-zero value to select a trigger signal). In Capture mode, the contents of the auxiliary timer T5 are latched into register **CAPREL** in response to a signal transition at the selected external input pin(s). Bit CT3 selects the external input line CAPIN or the input lines T3IN and/or T3EUD of GPT1 timer T3 as the source for a capture trigger. Either a positive, a negative, or both a positive and a negative transition at line CAPIN can be selected to trigger the capture function, or transitions on input T3IN or input T3EUD or both inputs, T3IN and T3EUD. The active edge is controlled by bitfield CI in register **T5CON**. **Table 282** summarizes these options.

Table 282 CAPREL register input edge selection

CT3	CI	Triggering signal/edge for Capture mode
X	00_B	None. Capture mode is disabled
0	01_B	Positive transition (rising edge) on CAPIN[1)]
0	10_B	Negative transition (falling edge) on CAPIN
0	11_B	Any transition (rising or falling edge) on CAPIN
1	01_B	Any transition (rising or falling edge) on T3IN
1	10_B	Any transition (rising or falling edge) on T3EUD
1	11_B	Any transition (rising or falling edge) on T3IN or T3EUD

1) Rising edge must be selected if capturing is triggered by the internal GPT1 read signals (see register **PISEL** and **Combined Capture modes**).

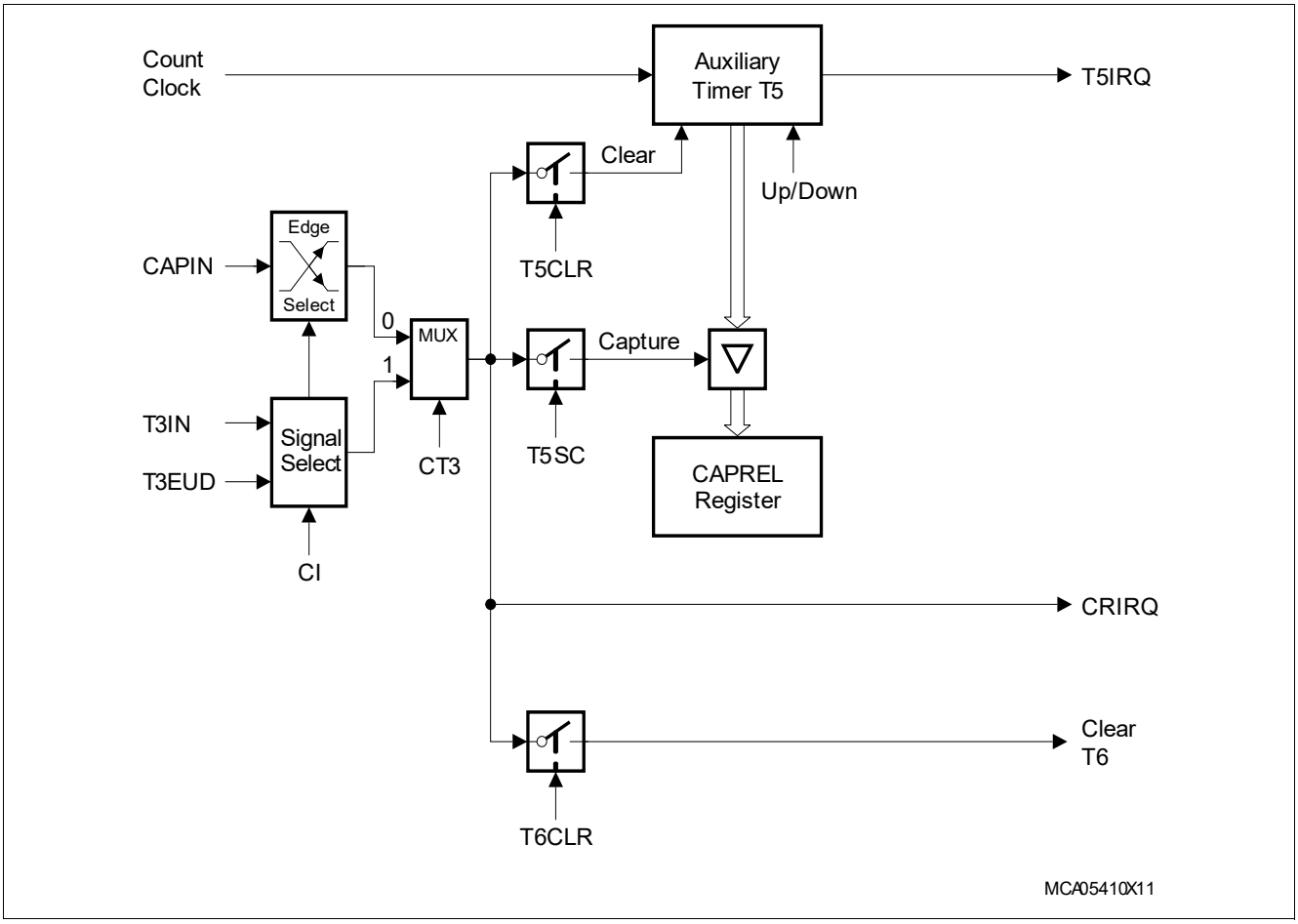


Figure 290 Capture/Reload register CAPREL in Capture mode

When a selected trigger is detected, the contents of the auxiliary timer T5 are latched into register **CAPREL** and the interrupt request line CRIRQ is activated. The same event can optionally clear timer T5 and/or timer T6. This option is enabled by bit T5CLR in register **T5CON** and bit T6CLR in register **T6CON**, respectively. If TxCLR = 0 the contents of timer Tx is not affected by a capture. If TxCLR = 1 timer Tx is cleared after the current timer T5 value has been latched into register **CAPREL**.

*Note: Bit T5SC only controls whether or not a capture is performed. If T5SC is cleared the external input pin(s) can still be used to clear timer T5 and/or T6, or as external interrupt input(s). This interrupt is controlled by the **CAPREL** interrupt control register SCU.GPTIEN, SCU.GPTIS, SCU.GPTISC and SCU.GPTISS.*

When capture triggers T3IN or T3EUD are enabled (CT3 = 1), register **CAPREL** captures the contents of T5 upon transitions of the selected input(s). These values can be used to measure T3's input signals. This is useful, for example, when T3 operates in Incremental Interface mode, in order to derive dynamic information (speed, acceleration) from the input signals.

For Capture mode operation, the selected pins CAPIN, T3IN, or T3EUD must be configured as input. To ensure that a transition of a trigger input signal applied to one of these inputs is recognized correctly, its level must be held high or low for a minimum number of module clock cycles, detailed in **Chapter 23.6.6**.

23.6.5.2 Capture/Reload register CAPREL in Reload mode

Reload mode for register **CAPREL** is selected by setting bit T6SR in control register **T6CON**. In Reload mode, the core timer T6 is reloaded with the contents of register **CAPREL**, triggered by an overflow or underflow of T6. This will not activate the interrupt request line CRIRQ associated with the **CAPREL** register. However, interrupt request line T6IRQ will be activated, indicating the overflow/underflow of T6.

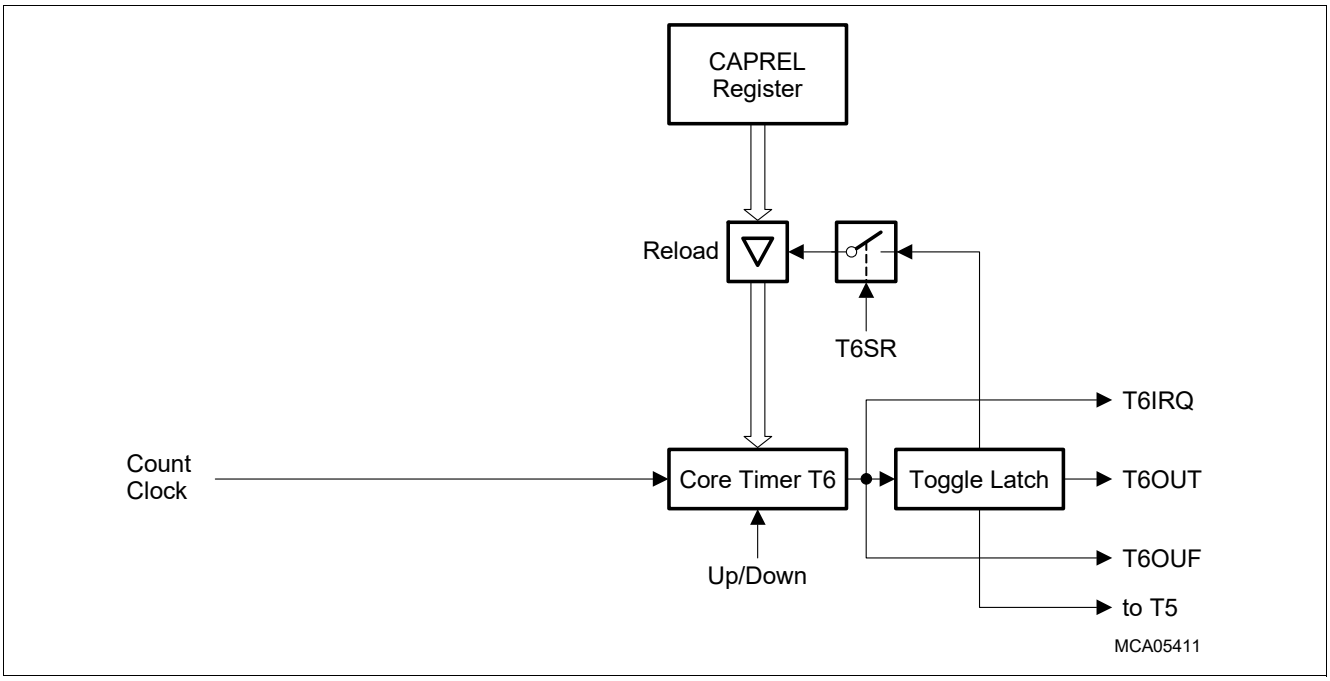


Figure 291 Capture/Reload register CAPREL in Reload mode

23.6.5.3 Capture/Reload register CAPREL in Capture-And-Reload mode

Since the reload function and the capture function of register **CAPREL** can be enabled individually by bits T5SC and T6SR, the two functions can be enabled simultaneously by setting both bits. This feature can be used to generate an output frequency that is a multiple of the input frequency.

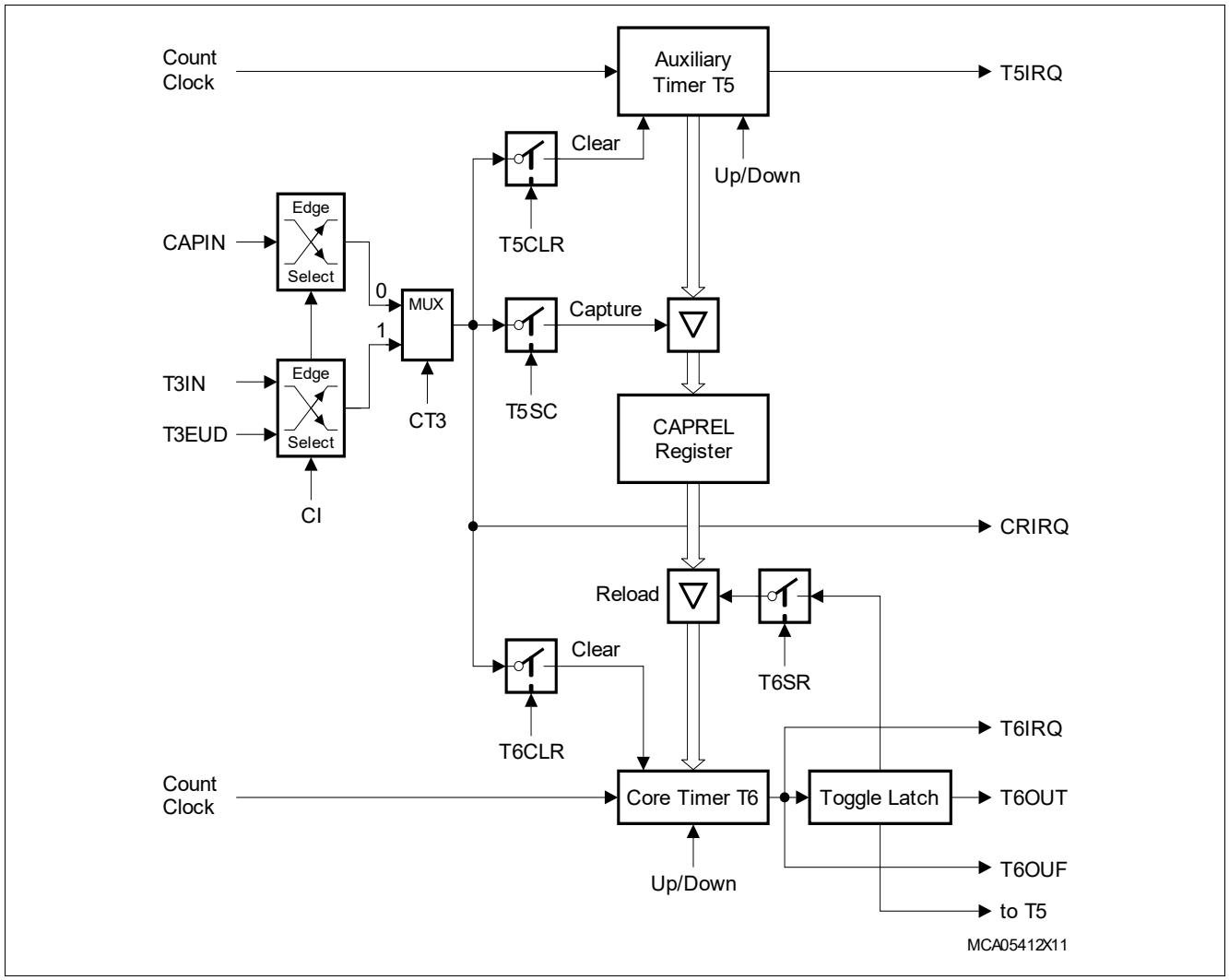


Figure 292 Capture/Reload register CAPREL in Capture-And-Reload mode

This combined mode can be used to detect consecutive external events which may occur aperiodically, but where a finer resolution, that means, more ‘ticks’ within the time between two external events is required.

For this purpose, the time between the external events is measured using timer T5 and the **CAPREL** register. Timer T5 runs in Timer mode counting up with a frequency of e.g. $f_{GPT}/32$. The external events are applied to pin CAPIN. When an external event occurs, the contents of timer T5 are latched into register **CAPREL** and timer T5 is cleared (T5CLR = 1). Thus, register always contains the correct time between two events, measured in timer T5 increments. Timer T6, which runs in Timer mode counting down with a frequency of e.g. $f_{GPT}/4$, uses the value in register **CAPREL** to perform a reload on underflow. This means, the value in register **CAPREL** represents the time between two underflows of timer T6, now measured in timer T6 increments. Since (in this example) timer T6 runs 8 times faster than timer T5, it will underflow 8 times within the time between two external events. Thus, the underflow signal of timer T6 generates 8 ‘ticks’. Upon each underflow, the interrupt request line T6IRQ will be activated and bit T6OTL will be toggled. The state of T6OTL may be output on pin T6OUT. This signal has 8 times more transitions than the signal which is applied to pin CAPIN.

Note: The underflow signal of Timer T6 can furthermore be used to clock one or more of the timers of the CAPCOM units, which gives the user the possibility to set compare events based on a finer resolution than that of the external events. This connection is accomplished via signal T6OUF.

23.6.5.4 Capture correction

A certain deviation of the output frequency is generated by the fact that timer T5 will count actual time units (e.g. T5 running at 1 MHz will count up to the value $64_H/100_D$ for a 10 kHz input signal), while T6OTL will only toggle upon an underflow of T6 (i.e. the transition from 0000_H to $FFFF_H$). In the above mentioned example, T6 would count down from 64_H, so the underflow would occur after 101 timing ticks of T6. The actual output frequency then is 79.2 kHz, instead of the expected 80 kHz.

This deviation can be compensated for by using T6 overflows. In this case, T5 counts down and T6 counts up. Upon a signal transition on pin CAPIN, the count value in T5 is captured into **CAPREL** and T5 is cleared to 0000_H. In its next clock cycle, T5 underflows to $FFFF_H$, and continues to count down with the following clocks. T6 is reloaded from **CAPREL** upon an overflow, and continues to count up with its following clock cycles (8 times faster in the above example). In this case, T5 and T6 count the same number of steps with their respective internal count frequency.

In the above example, T5 running at 1 MHz will count down to the value $FF9C_H/-100_D$ for a 10 kHz input signal applied at CAPIN, while T6 counts up from $FF9C_H$ through $FFFF_H$ to 0000_H. So the overflow occurs after 100 timing ticks of T6, and the actual output frequency at T6OUT then is the expected 80 kHz.

However, in this case **CAPREL** does not directly contain the time between two CAPIN events, but rather its 2's complement. Software will have to convert this value, if it is required for the operation.

23.6.5.5 Combined Capture modes

For incremental interface applications in particular, several timer features can be combined to obtain dynamic information such as speed, acceleration, or deceleration. The current position itself can be obtained directly from the timer register (**T2**, **T3**, **T4**).

The time information to determine the dynamic parameters is generated by capturing the contents of the free-running timer T5 into register **CAPREL**. Two trigger sources for this event can be selected:

- Capture trigger on sensor signal transitions
- Capture trigger on position read operations

Capturing on sensor signal transitions is available for timer T3 inputs. This mode is selected by setting bit CT3 and selecting the intended signal(s) via bitfield CI in register **T5CON**. **CAPREL** then indicates the time between two selected transitions (measured in T5 counts).

Capturing on position read operations is available for timers T2, T3, and T4. This mode is selected by clearing bit CT3 and selecting the rising edge via bitfield CI in register **T5CON**. Bitfield ISCAPIN in register **PISEL** then selects either a read access from T3 or a read access from any of T2 or T3 or T4. **CAPREL** then indicates the time between two read accesses.

These operating modes directly support the measurement of position and rotational speed. Acceleration and deceleration can then be determined by evaluating subsequent speed measurements.

23.6.6 GPT2 clock signal control

All actions within the timer block GPT2 are triggered by transitions of its basic clock. This basic clock is derived from the module clock f_{GPT_CLK} by a basic block prescaler, controlled by bitfield BPS2 in register **T6CON** (see **Figure 262**). The count clock can be generated in two different ways:

- Internal count clock: derived from GPT2's basic clock via a programmable prescaler, is used for (Gated) Timer mode
- External count clock: derived from the timer's input pin(s), is used for Counter mode

For both ways, the basic clock determines the maximum count frequency and the timer's resolution:

Table 283 Basic clock selection for block GPT2

Block prescaler[1]	BPS2 = 01_B	BPS2 = 00_B[2]	BPS2 = 11_B	BPS2 = 10_B
Prescaling factor for GPT2: F(BPS2)	F(BPS2) = 2	F(BPS2) = 4	F(BPS2) = 8	F(BPS2) = 16
Maximum external count frequency	$f_{GPT\_CLK}/4$	$f_{GPT\_CLK}/8$	$f_{GPT\_CLK}/16$	$f_{GPT\_CLK}/32$
Input signal stable time	$2 \times t_{GPT\_CLK}$	$4 \times t_{GPT\_CLK}$	$8 \times t_{GPT\_CLK}$	$16 \times t_{GPT\_CLK}$

1) Please note the non-linear encoding of bitfield BPS2.
2) Default after reset.

Note: When initializing the GPT2 block, and the block prescaler BPS2 in T6CON needs to be set to a value different from its reset value (00_B), it must be initialized first before any mode involving external trigger signals is configured. These modes include counter, capture, and Reload mode. Otherwise, unintended count/capture/reload events may occur.
*In this case (e.g. when changing BPS2 during operation of the GPT2 block), disable related interrupts before modification of BPS2, and afterwards clear the corresponding service request flags and re-initialize those registers (**T5**, **T6**, **CAPREL**) that might be affected by a count/capture/reload event.*

23.6.6.1 Internal count clock generation

In Timer mode and Gated Timer mode, the count clock for each GPT2 timer is derived from the GPT2 basic clock by a programmable prescaler, controlled by bitfield TxI in the respective timer's control register TxCON.

The count frequency f_{Tx} for a timer Tx and its resolution r_{Tx} are scaled linearly with lower clock frequencies, as can be seen from the following formula:

$$f_{Tx} = \frac{f_{GPT\_CLK}}{F(BPS2) \bullet 2^{<TxI>}} \qquad r_{Tx}[\mu s] = \frac{F(BPS2) \bullet 2^{<TxI>}}{f_{GPT\_CLK}[MHz]} \tag{23.2}$$

The effective count frequency depends on the common module clock prescaler factor F(BPS2) as well as on the individual input prescaler factor $2^{<TxI>}$. **Table 287** summarizes the resulting overall divider factors for a GPT2 timer that result from these cascaded prescalers.

Table 284 lists GPT2 timer's parameters (such as count frequency, resolution, and period) resulting from the selected overall prescaler factor and the module clock f_{GPT_CLK}. Note that some numbers may be rounded.

Table 284 GPT2 timer parameters

System clock = 10 MHz			Overall divider factor	System clock = 40 MHz		
Frequency	Resolution	Period		Frequency	Resolution	Period
5.0 MHz	200 ns	13.11 ms	2	20.0 MHz	50 ns	3.28 ms
2.5 MHz	400 ns	26.21 ms	4	10.0 MHz	100 ns	6.55 ms
1.25 MHz	800 ns	52.43 ms	8	5.0 MHz	200 ns	13.11 ms
625.0 kHz	1.6 µs	104.9 ms	16	2.5 MHz	400 ns	26.21 ms
312.5 kHz	3.2 µs	209.7 ms	32	1.25 MHz	800 ns	52.43 ms
156.25 kHz	6.4 µs	419.4 ms	64	625.0 kHz	1.6 µs	104.9 ms
78.125 kHz	12.8 µs	838.9 ms	128	312.5 kHz	3.2 µs	209.7 ms
39.06 kHz	25.6 µs	1.678 s	256	156.25 kHz	6.4 µs	419.4 ms
19.53 kHz	51.2 µs	3.355 s	512	78.125 kHz	12.8 µs	838.9 ms
9.77 kHz	102.4 µs	6.711 s	1024	39.06 kHz	25.6 µs	1.678 s
4.88 kHz	204.8 µs	13.42 s	2048	19.53 kHz	51.2 µs	3.355 s

23.6.6.2 External count clock input

The external input signals of the GPT2 block are sampled with the GPT2 basic clock (see **Figure 262**). To ensure that a signal is recognized correctly, its current level (high or low) must be held active for at least one complete sampling period, before changing. A signal transition is recognized if two subsequent samples of the input signal represent different levels. Therefore, a minimum of two basic clock periods are required for the sampling of an external input signal. Thus, the maximum frequency of an input signal must not be higher than half the basic clock.

Table 285 summarizes the resulting requirements for external GPT2 input signals.

Table 285 GPT2 external input signal limits

GPT2 basic clock = 10 MHz		Input frequ. factor	GPT2 divider BPS2	Input phase duration	GPT2 basic clock = 40 MHz	
Max. input frequency	Min. level Hold time				Max. input frequency	Min. level hold time
2.5 MHz	200 ns	$f_{GPT\_CLK}/4$	01_B	$2 \times t_{GPT\_CLK}$	10.0 MHz	50 ns
1.25 MHz	400 ns	$f_{GPT\_CLK}/8$	00_B	$4 \times t_{GPT\_CLK}$	5.0 MHz	100 ns
625.0 kHz	800 ns	$f_{GPT\_CLK}/16$	11_B	$8 \times t_{GPT\_CLK}$	2.5 MHz	200 ns
312.5 kHz	1.6 µs	$f_{GPT\_CLK}/32$	10_B	$16 \times t_{GPT\_CLK}$	1.25 MHz	400 ns

These limitations are valid for all external input signals to GPT2, including the external count signals in Counter mode and the gate input signals in Gated Timer mode.

23.6.7 GPT2 encoding

23.6.7.1 Encoding of timer count direction control

Table 286 GPT2 timer count direction control

Pin TxEUD	Bit TxUDE	Bit TxUD	Count direction
X	0	0	Count up
X	0	1	Count down
0	1	0	Count up
1	1	0	Count down
0	1	1	Count down
1	1	1	Count up

23.6.7.2 Timer mode and Gated Timer mode: encoding of overall prescaler factor

Table 287 GPT2 overall prescaler factors for internal count clock (Timer mode and Gated Timer mode)

Individual prescaler for Tx	Common prescaler for module clock[1)]			
	BPS2 = 01_B	**BPS2 = 00_B**	**BPS2 = 11_B**	**BPS2 = 10_B**
TxI = 000_B	2	4	8	16
TxI = 001_B	4	8	16	32
TxI = 010_B	8	16	32	64
TxI = 011_B	16	32	64	128
TxI = 100_B	32	64	128	256
TxI = 101_B	64	128	256	512
TxI = 110_B	128	256	512	1024
TxI = 111_B	256	512	1024	2048

1) Please note the non-linear encoding of bitfield BPS2.

23.6.7.3 Counter mode: Encoding of input edge selection

Table 288 GPT2 auxiliary timer T5 input edge selection (Counter mode)

T5I	Triggering edge for counter increment/decrement
$X00_B$	None. Counter T5 is disabled
001_B	Positive transition (rising edge) on T5IN
010_B	Negative transition (falling edge) on T5IN
011_B	Any transition (rising or falling edge) on T5IN
101_B	Positive transition (rising edge) of T6 toggle latch T6OTL
110_B	Negative transition (falling edge) of T6 toggle latch T6OTL
111_B	Any transition (rising or falling edge) of T6 toggle latch T6OTL

Table 289 GPT2 core timer T6 input edge selection (Counter mode)

T6I	Triggering edge for counter increment/decrement
000_B	None. Counter T6 is disabled
001_B	Positive transition (rising edge) on T6IN
010_B	Negative transition (falling edge) on T6IN
011_B	Any transition (rising or falling edge) on T6IN
$1XX_B$	Reserved. Do not use this combination

23.7 Register description GPT12

23.7.1 GPT12 Address Maps

Table 290 Register Address Space - GPT12

Module	Base Address	End Address	Note
GPT12	40014000_H	$40017FFF_H$	

Table 291 Register Overview - GPT12 (ascending Offset Address)

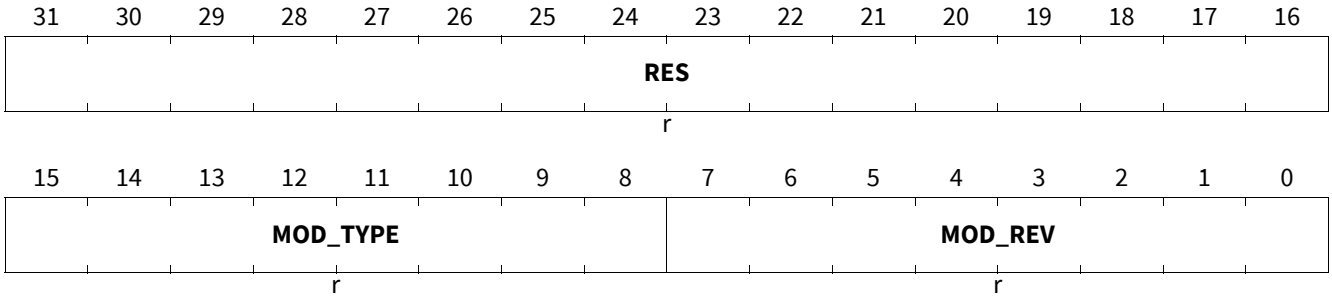
Short Name	Long Name	Offset Address	Page Number
ID	Module Identification Register	0000_H	928
PISEL	Port Input Select Register	0004_H	928
T2CON	Timer T2 Control Register	0008_H	930
T3CON	Timer T3 Control Register	$000C_H$	931
T4CON	Timer T4 Control Register	0010_H	933
T5CON	Timer T5 Control Register	0014_H	934
T6CON	Timer T6 Control Register	0018_H	936
CAPREL	Capture/Reload Register	$001C_H$	937
T2	Timer T2 Count Register	0020_H	937
T3	Timer T3 Count Register	0024_H	938
T4	Timer T4 Count Register	0028_H	938
T5	Timer 5 Count Register	$002C_H$	939
T6	Timer 6 Count Register	0030_H	939

23.7.2 GPT12 Registers

Module Identification Register

Register ID indicates the module version.

ID
Module Identification Register **(0000_H)** **RESET_TYPE_5 Value: $0000\ 5804_H$**

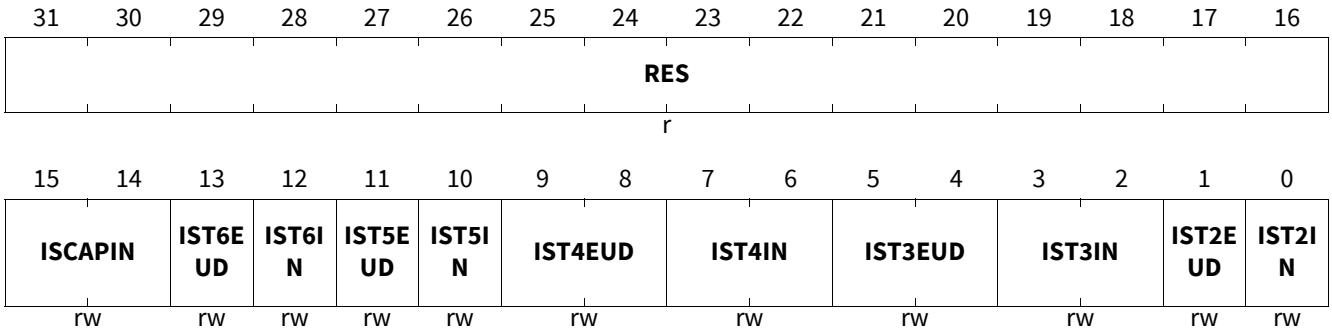


Field	Bits	Type	Description
MOD_REV	7:0	r	**Module Revision Number** MOD:_REV defines the revision number. The value of a module revision starts with 01_H (first revision)
MOD_TYPE	15:8	r	**Module Identification Number** This bitfield defines the module identification number (58_H = GPT12E)
RES	31:16	r	**Reserved**

Port Input Select Register

Register PISEL selects timer input signal from several sources under software control.

PISEL
Port Input Select Register **(0004_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
IST2IN	0	rw	**Input Select for T2IN** 0_B **T2INA**, Signal T2INA is selected 1_B **T2INB**, Signal T2INB is selected
IST2EUD	1	rw	**Input Select for T2EUD** 0_B **T2EUDA**, Signal T2EUDA is selected 1_B **T2EUDB**, Signal T2EUDB is selected

Field	Bits	Type	Description
IST3IN	3:2	rw	**Input Select for T3IN** 00_B **T3INA**, Signal T3INA is selected 01_B **T3INB**, Signal T3INB is selected 10_B **T3INC**, Signal T3INC is selected 11_B **T3IND**, Signal T3IND is selected
IST3EUD	5:4	rw	**Input Select for T3EUD** 00_B **T3EUDA**, Signal T3EUDA is selected 01_B **T3EUDB**, Signal T3EUDB is selected 10_B **T3EUDC**, Signal T3EUDC is selected 11_B **T3EUDD**, Signal T3EUDD is selected
IST4IN	7:6	rw	**Input Select for T4IN** 00_B **T4INA**, Signal T4INA is selected 01_B **T4INB**, Signal T4INB is selected 10_B **T4INC**, Signal T4INC is selected 11_B **T4IND**, Signal T4IND is selected
IST4EUD	9:8	rw	**Input Select for TEUD** 00_B **T4EUDA**, Signal T4EUDA is selected 01_B **T4EUDB**, Signal T4EUDB is selected 10_B **T4EUDC**, Signal T4EUDC is selected 11_B **T4EUDD**, Signal T4EUDD is selected
IST5IN	10	rw	**Input Select for T5IN** 0_B **T5INA**, Signal T5INA is selected 1_B **T5INB**, Signal T5INB is selected
IST5EUD	11	rw	**Input Select for T5EUD** 0_B **T5EUDA**, Signal T5EUDA is selected 1_B **T5EUDB**, Signal T5EUDB is selected
IST6IN	12	rw	**Input Select for T6IN** 0_B **T6INA**, Signal T6INA is selected 1_B **T6INB**, Signal T6INB is selected
IST6EUD	13	rw	**Input Select for T6EUD** 0_B **T6EUDA**, Signal T6EUDA is selected 1_B **T6EUDB**, Signal T6EUDB is selected
ISCAPIN	15:14	rw	**Input Select for CAPIN** 00_B **CAPINA**, Signal CAPINA is selected 01_B **CAPINB**, Signal CAPINB is selected 10_B **CAPINC**, Signal CAPINC (Read trigger from T3) is selected 11_B **CAPIND**, Signal CAPIND (Read trigger from T2 or T3 or T4) is selected
RES	31:16	r	**Reserved**

Timer T2 Control Register

T2CON
Timer T2 Control Register **(0008_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T2DIR	T2CHDIR	T2EDGE	T2IRIDIS	RES		T2RC	T2UDE	T2UD	T2R	T2M			T2I		
rh	rwh	rwh	rw	r		rw	rw	rw	rw	rw			rw		

Field	Bits	Type	Description
T2I	2:0	rw	**Timer T2 Input Parameter Selection** Depends on the operating mode, see respective sections for encoding: **Table 277** for Timer Mode and Gated Timer Mode **Table 280** for Counter Mode **Table 281** for Incremental Interface Mode
T2M	5:3	rw	**Timer T2 Input Mode Control** 000_B **TimerMode**, 001_B **CounterMode**, 010_B **Gatedlow**, Gated Timer Mode with gate active low 011_B **Gatedhigh**, Gated Timer Mode with gate active high 100_B **ReloadMode**, 101_B **CaptureMode**, 110_B **IncrementalInterfaceMode**, (Rotation Detection Mode) 111_B **IncrementalInterfaceMode**, (Edge Detection Mode)
T2R	6	rw	**Timer T2 Input Run Bit** 0_B **Stop**, Timer T2 stops 1_B **Run**, Timer T2 runs
T2UD	7	rw	**Timer T2 Up/Down Control** 0_B **Up**, Timer T2 counts up 1_B **Down**, Timer T2 counts down
T2UDE	8	rw	**Timer T2 External Up/Down Enable** 0_B **T2UD**, Count direction is controlled by bit T2UD; input T2EUD is disconnected 1_B **T2EUD**, Count direction is controlled by input T2EUD
T2RC	9	rw	**Timer T2 Remote Control** 0_B **T2R**, Timer T2 is controlled by its own run bit T2R 1_B **T3R**, Timer T2 is controlled by the run bit T3R of core timer T3, not by bit T2R
RES	11:10, 31:16	r	**Reserved** Read as 0; should be written with 0.

Field	Bits	Type	Description
T2IRIDIS	12	rw	**Timer T2 Interrupt Disable** 0_B **Enabled**, Interrupt generation for T2CHDIR and T2EDGE interrupts in Incremental Interface Mode is enabled 1_B **Disabled**, Interrupt generation for T2CHDIR and T2EDGE interrupts in Incremental Interface Mode is disabled
T2EDGE	13	rwh	**Timer T2 Edge Detection** The bit is set each time a count edge is detected.T2EDGE must be cleared by software. 0_B **Nocount**, No count edge was detected 1_B **Count**, A count edge was detected
T2CHDIR	14	rwh	**Timer T2 Count Direction Change** This bit is set each time the count direction of timer T2 changes. T2CHDIR must be cleared by software. 0_B **Nochange**, No change of count direction was detected 1_B **Change**, A change of count direction was detected
T2DIR	15	rh	**Timer T2 Rotation Direction** 0_B **Up**, Timer T2 counts up 1_B **Down**, Timer T2 counts down

Timer T3 Control Register

T3CON
Timer T3 Control Register **($000C_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T3DIR	T3CHDIR	T3EDGE	BPS1		T3OTL	T3OE	T3UDE	T3UD	T3R	T3M			T3I		
rh	rwh	rwh	rw		rwh	rw	rw	rw	rw	rw			rw		

Field	Bits	Type	Description
T3I	2:0	rw	**Timer T3 Input Parameter Selection** Depends on the operating mode, see respective sections for encoding: **Table 277** for Timer Mode and Gated Timer Mode **Table 278** for Counter Mode **Table 281** for Incremental Interface Mode

Field	Bits	Type	Description
T3M	5:3	rw	**Timer T3 Input Mode Control** 000_B **TimerMode**, 001_B **CounterMode**, 010_B **Gatedlow**, Gated Timer Mode with gate active low 011_B **Gatedhigh**, Gated Timer Mode with gate active high 100_B **Reserved**, Do not use this combination 101_B **Reserved**, Do not use this combination 110_B **IncrementalInterfaceMode**, (Rotation Detection Mode) 111_B **IncrementalInterfaceMode**, (Edge Detection Mode)
T3R	6	rw	**Timer T3 Input Run Bit** 0_B **Stop**, Timer T3 stops 1_B **Run**, Timer T3 runs
T3UD	7	rw	**Timer T3 Up/Down Control** 0_B **Up**, Timer T3 counts up 1_B **Down**, Timer T3 counts down
T3UDE	8	rw	**Timer T3 External Up/Down Enable** 0_B **T3UD**, Count direction is controlled by bit T3UD; input T3EUD is disconnected 1_B **T3EUD**, Count direction is controlled by input T3EUD
T3OE	9	rw	**Overflow/Underflow Output Enable** 0_B **Disabled**, Alternate Output Function Disabled 1_B **T3OUT**, State of T3 toggle latch is output on pin T3OUT
T3OTL	10	rwh	**Timer T3 Overflow Toggle Latch** Toggles on each overflow/underflow of T3. Can be set or cleared by software (see separate description)
BPS1	12:11	rw	**GPT1 Block Prescaler Control** Select basic clock for block GPT1 00_B **8**, GPT_CLK/8 01_B **4**, GPT_CLK/4 10_B **32**, GPT_CLK/32 11_B **16**, GPT_CLK/16
T3EDGE	13	rwh	**Timer T3 Edge Detection Flag** The bit is set each time a count edge is detected. T3EDGE must be cleared by software. 0_B **Nocount**, No count edge was detected 1_B **Count**, A count edge was detected
T3CHDIR	14	rwh	**Timer T3 Count Direction Change Flag** This bit is set each time the count direction of timer T3 changes. T3CHDIR must be cleared by software. 0_B **Nochange**, No change of count direction was detected 1_B **Change**, A change of count direction was detected
T3DIR	15	rh	**Timer T3 Rotation Direction Flag** 0_B **Up**, Timer T3 counts up 1_B **Down**, Timer T3 counts down
RES	31:16	r	**Reserved** Read as 0; should be written with 0.

Timer T4 Control Register

T4CON
Timer T4 Control Register **(0010_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T4RDIR	T4CHDIR	T4EDGE	T4IRDIS	CLRT3EN	CLRT2EN	T4RC	T4UDE	T4UD	T4R	T4M			T4I		
rh	rwh	rwh	rw	rw	rw	rw	rw	rw	rw	rw			rw		

Field	Bits	Type	Description
T4I	2:0	rw	**Timer T4 Input Parameter Selection** Depends on the operating mode, see respective sections for encoding: **Table 277** for Timer Mode and Gated Timer Mode **Table 280** for Counter Mode **Table 281** for Incremental Interface Mode
T4M	5:3	rw	**Timer T4 Mode Control (Basic Operating Mode)** 000_B **TimerMode**, 001_B **CounterMode**, 010_B **Gatedlow**, Gated Timer Mode with gate active low 011_B **Gatedhigh**, Gated Timer Mode with gate active high 100_B **ReloadMode**, 101_B **CaptureMode**, 110_B **IncrementalInterfaceMode**, (Rotation Detection Mode) 111_B **IncrementalInterfaceMode**, (Edge Detection Mode)
T4R	6	rw	**Timer T4 Input Run Bit** 0_B **Stop**, Timer T4 stops 1_B **Run**, Timer T4 runs
T4UD	7	rw	**Timer T4 Up/Down Control** 0_B **Up**, Timer T4 counts up 1_B **Down**, Timer T4 counts down
T4UDE	8	rw	**Timer T4 External Up/Down Enable** 0_B **T4UD**, Count direction is controlled by bit T4UD; input T4EUD is disconnected 1_B **T4EUD**, Count direction is controlled by input T4EUD
T4RC	9	rw	**Timer T4 Remote Control** 0_B **T4R**, Timer T4 is controlled by its own run bit T4R 1_B **T3R**, Timer T4 is controlled by the run bit T3R of core timer T3, but not by bit T4R

Field	Bits	Type	Description
CLRT2EN	10	rw	**Clear Timer T2 Enable** Enables the automatic clearing of timer T2 upon a falling edge of the selected T4EUD input. 0_B **Noeffect**, No effect of T4EUD on timer T2 1_B **Clear**, A falling edge on T4EUD clears timer T2
CLRT3EN	11	rw	**Clear Timer T3 Enable** Enables the automatic clearing of timer T3 upon a falling edge of the selected T4In input. 0_B **Noeffect**, No effect of T4IN on Timer T3 1_B **Clear**, A falling edge on T4In clears timer T3
T4IRDIS	12	rw	**Timer T4 Interrupt Disable** 0_B **Enabled**, Interrupt generation for T4CHDIR and T4EDGE interrupts in Incremental Interface Mode is enabled 1_B **Disabled**, Interrupt generation for T4CHDIR and T4EDGE interrupts in Incremental Interface Mode is disabled
T4EDGE	13	rwh	**Timer T4 Edge Direction** The bit is set each time a count edge is detected. T4EDGE has to be cleared by software 0_B **Nocount**, No count edge was detected 1_B **Count**, A count edge was detected
T4CHDIR	14	rwh	**Timer T4 Count Direction Change** The bit is set each time a count direction of timer T4 changes. T4CHDIR must be cleared by software 0_B **Nochange**, No change in count direction was detected 1_B **Change**, A change in count direction was detected
T4RDIR	15	rh	**Timer T4 Rotation Direction** 0_B **Up**, Timer T4 counts up 1_B **Down**, Timer T4 counts down
RES	31:16	r	**Reserved** Read as 0; should be written with 0.

Timer T5 Control Register

T5CON
Timer T5 Control Register **(0014_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T5SC	**T5CLR**	**CI**		**RES**	**CT3**	**T5RC**	**T5UDE**	**T5UD**	**T5R**	**RES**	**T5M**		**T5I**		
rw	rw	rw		r	rw	rw	rw	rw	rw	r	rw		rw		

Field	Bits	Type	Description
T5I	2:0	rw	**Timer T5 Input Parameter Selection** Depends on the operating mode, see respective sections for encoding: **Table 287** for Timer Mode and Gated Timer Mode **Table 288** for Counter Mode
T5M	4:3	rw	**Timer T5 Input Mode Control** 00_B **TimerMode**, 01_B **CounterMode**, 10_B **Gatedlow**, Gated Timer Mode with gate active low 11_B **Gatedhigh**, Gated Timer Mode with gate active high
RES	5, 11, 31:16	r	**Reserved** Contains the current value of the CAPREL register
T5R	6	rw	**Timer T5 Run Bit** 0_B **Stop**, Timer T5 stops 1_B **Run**, Timer T5 runs
T5UD	7	rw	**Timer T5 Up/Down Control** 0_B **Up**, Timer T5 counts up 1_B **Down**, Timer T5 counts down
T5UDE	8	rw	**Timer T5 External Up/Down Enable** 0_B **T5UD**, Count direction is controlled by bit T5UD; input T5EUD is disconnected 1_B **T5EUD**, Count direction is controlled by input T5EUD
T5RC	9	rw	**Timer T5 Remote Control** 0_B **T5R**, Timer T5 is controlled by its own run bit T5R 1_B **T6R**, Timer T5 is controlled by the run bit T6R of core timer T6, not by bit T5R
CT3	10	rw	**Timer T3 Capture Trigger Enable** 0_B **CAPIN**, Capture trigger from input line CAPIN 1_B **T3IN**, Capture trigger from T3 input lines T3IN and/or T3EUD
CI	13:12	rw	**Register CAPREL Capture Trigger Selection** 00_B **Disabled**, Capture disabled 01_B **Positive**, Positive transition (rising edge) on CAPIN 10_B **Negative**, Negative transition (falling edge) on CAPIN or any transition on T3EUD 11_B **Any**, Any transition (rising or falling edge) on CAPIN or any transition on T3IN or T3EUD
T5CLR	14	rw	**Timer T5 Clear Enable Bit** 0_B **Notcleared**, Timer T5 is not cleared on a capture event 1_B **Cleared**, Timer T5 is cleared on a capture event
T5SC	15	rw	**Timer T5 Capture Mode Enable** 0_B **Disabled**, Capture into register CAPREL disabled 1_B **Enabled**, Capture into register CAPREL enabled

Timer T6 Control Register

T6CON
Timer T6 Control Register **(0018_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T6SR	T6CLR	RES	BPS2		T6OTL	T6OE	T6UDE	T6UD	T6R	T6M			T6I		
rw	rw	r	rw		rwh	rw	rw	rw	rw	rw			rw		

Field	Bits	Type	Description
T6I	2:0	rw	**Timer T6 Input Parameter Selection** Depends on the operating mode, see respective sections for encoding: **Table 287** for Timer Mode and Gated Timer Mode **Table 289** for Counter Mode
T6M	5:3	rw	**Timer T6 Mode Control** 000_B **TimerMode**, 001_B **CounterMode**, 010_B **Gatedlow**, Gated Timer Mode with gate active low 011_B **Gatedhigh**, Gated Timer Mode with gate active high 100_B **Reserved**, Do not use this combination … 111_B **Reserved**, Do not use this combination
T6R	6	rw	**Timer T6 Input Run Bit** 0_B **Stop**, Timer T3 stops 1_B **Run**, Timer T3 runs
T6UD	7	rw	**Timer T6 Up/Down Control** 0_B **Up**, Timer T3 counts up 1_B **Down**, Timer T3 counts down
T6UDE	8	rw	**Timer T6 External Up/Down Enable** 0_B **T6UD**, Count direction is controlled by bit T6UD; input T6EUD is disconnected 1_B **T6EUD**, Count direction is controlled by input T6EUD
T6OE	9	rw	**Overflow/Underflow Output Enable** 0_B **Disabled**, Alternate Output Function Disabled 1_B **T6OUT**, State of T6 toggle latch is output on pin T6OUT
T6OTL	10	rwh	**Timer T6 Overflow Toggle Latch** Toggles on each overflow/underflow of T6. Can be set or cleared by software (see separate description)

Field	Bits	Type	Description
BPS2	12:11	rw	**GPT2 Block Prescaler Control** Select basic clock for block GPT2 00_B **4**, GPT_CLK/4 01_B **2**, GPT_CLK/2 10_B **16**, GPT_CLK/16 11_B **8**, GPT_CLK/8
RES	13, 31:16	r	**Reserved**
T6CLR	14	rw	**Timer T6 Clear Enable Bit** 0_B **Notcleared**, Timer T6 is not cleared on a capture event 1_B **Cleared**, Timer T6 is cleared on a capture event
T6SR	15	rw	**Timer T6 Reload Mode Enable** 0_B **Disabled**, Reload from register CAPREL disabled 1_B **Enabled**, Reload from register CAPREL enabled

Capture/Reload Register

CAPREL
Capture/Reload Register **($001C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPREL															
rwh															

Field	Bits	Type	Description
CAPREL	15:0	rwh	**Current reload value or Captured value** Contains the current value of the timer CAPREL register
RES	31:16	r	**Reserved**

Timer T2 Count Register

T2
Timer T2 Count Register **(0020_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

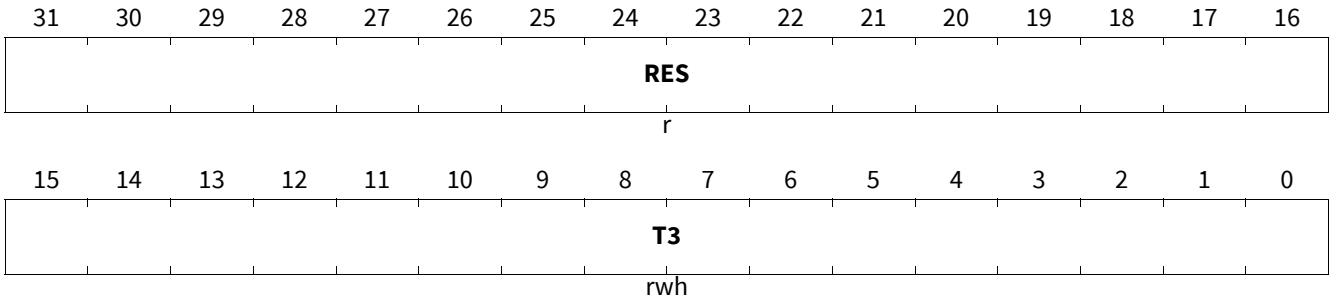
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T2															
rwh															

Field	Bits	Type	Description
T2	15:0	rwh	**Timer T2 Current Value** Contains the current value of the timer T2
RES	31:16	r	**Reserved**

Timer T3 Count Register

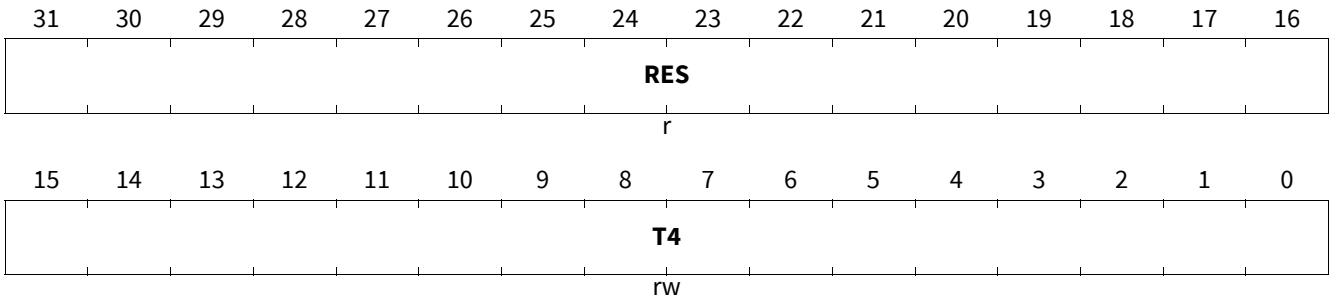
T3
Timer T3 Count Register **(0024$_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**



Field	Bits	Type	Description
T3	15:0	rwh	**Timer T3 Current Value** Contains the current value of the timer T3
RES	31:16	r	**Reserved**

Timer T4 Count Register

T4
Timer T4 Count Register **(0028$_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**



Field	Bits	Type	Description
T4	15:0	rw	**Timer T4 Current Value** Contains the current value of the timer T4
RES	31:16	r	**Reserved**

Timer 5 Count Register

T5
Timer 5 Count Register **($002C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T5															
rwh															

Field	Bits	Type	Description
T5	15:0	rwh	**Timer T5 Current Value** Contains the current value of the timer T5
RES	31:16	r	**Reserved**

Timer 6 Count Register

T6
Timer 6 Count Register **(0030_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T6															
rwh															

Field	Bits	Type	Description
T6	15:0	rwh	**Timer T6 Current Value** Contains the current value of the timer T6
RES	31:16	r	**Reserved**

24 Timer20 (T20) and Timer21 (T21)

24.1 Features overview

Two functionally identical timers are implemented: Timer20 and Timer21. The description also use the name as Timer2.

The timer modules are general purpose 16-bit timers. Timer2 can function as a timer or counter in each of its modes. As a timer, it counts with an input clock of $f_{T2\_CLK}/12$ (if prescaler is disabled). As a counter, Timer2 counts 1-to-0 transitions on pin T2. In the counter mode, the maximum resolution for the count is $f_{T2\_CLK}/24$ (if prescaler is disabled).

The T20 and T21 provides following features:

- 16-bit auto-reload mode
  - selectable up or down counting
- One channel 16-bit capture mode
- T20 and T21 can be configured as trigger source for ADC1

24.1.1 Block diagram

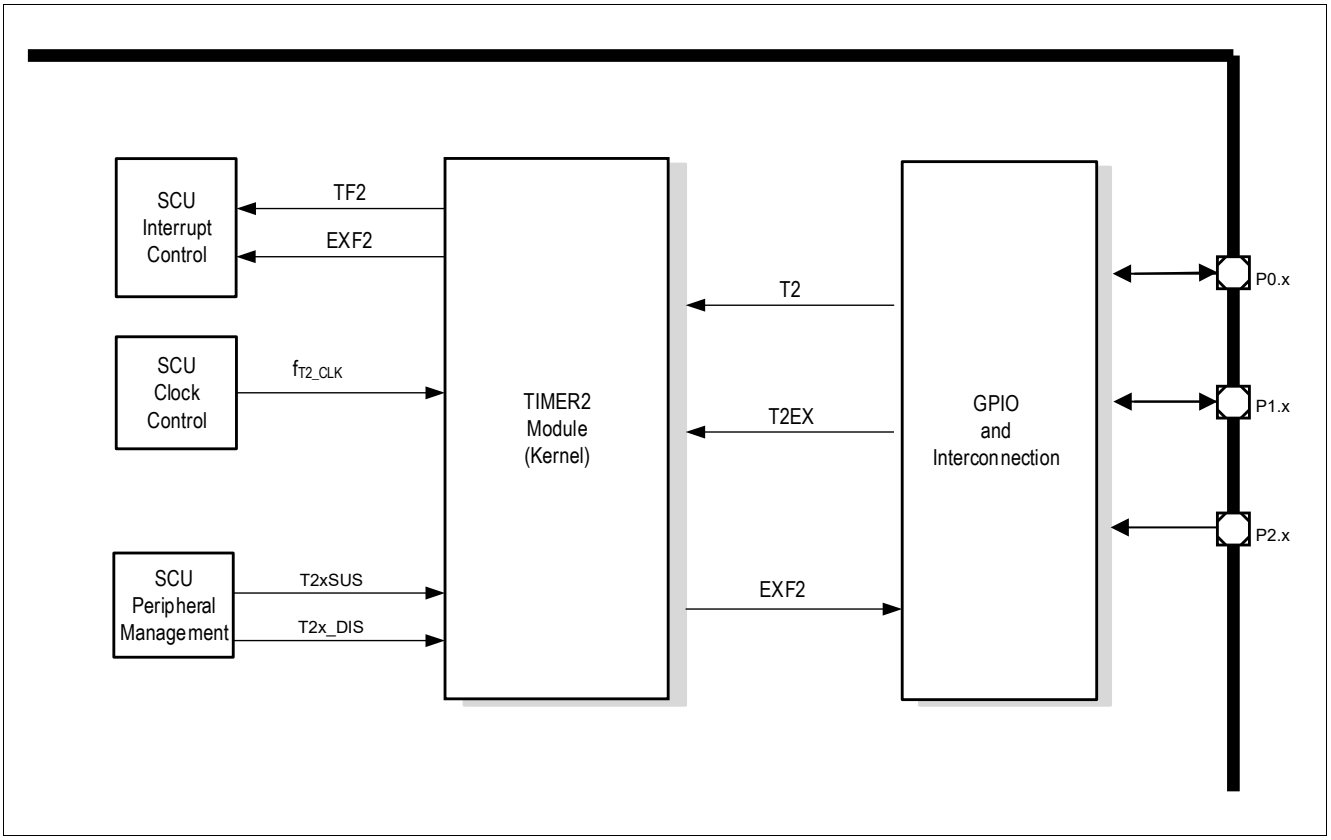


Figure 293 Block diagram Timer2

24.1.2 Toplevel signals

Table 292 Toplevel connection

Signal	Direction	Description	From/To
T2	Input	Count input	See **Product definitions**, **T20 interconnections** and **T21 interconnections**
T2EX	Input	External input	
EXF2	Output	External flag	
TF2/EXF2	Output	Interrupt	
f_{T2_CLK}	Input	Clock for T2	
T2x_DIS	Input	T2 disable	SCU
T2SUS	Input	T2 suspend	SCU

24.1.3 Interrupts

When an interrupt event happened, the corresponding interrupt flag bit EXF2/TF2 is set. If enabled by the related interrupt enable bit EXF2EN/TF2EN in register CON1, an interrupt for the interrupt event EXF2/TF2 will be generated. An overview is given in **Table 293**.

Table 293 Timer2 interrupt sources

Interrupt	Interrupt enable bit	Interrupt status flag	Interrupt status clear bit
External interrupt	CON1.EXF2EN	CON.EXF2	ICLR.EXF2CLR
Overflow/underflow interrupt	CON1.TF2EN	CON.TF2	ICLR.TF2CLR

Note: When the timer/counter is stopped and while the module remains enabled, it is possible for an external event at T2EX to generate an interrupt. For this to occur, bit EXEN2 in SFR T2CON must be set. In this case, a dummy reload or capture happens depending on the CP_RL2 bit selection. The resulting interrupt could therefore be used in the product as an external falling/rising edge triggered interrupt.

24.2 Timer20 and Timer21 modes

Table 294 Timer20 and Timer21 modes overview

Mode	Description
Auto-reload, Up/Down Count Disabled	• Count up only • Start counting from 16-Bit reload value, overflow at $FFFF_H$ • Reload event configurable for trigger by overflow condition only, or by negative/positive edge at input pin T2EX as well • Programmable reload value in register RC2 • Interrupt is generated with reload events.
Auto-reload, Up/Down Count Enabled	• Count up or down, direction determined by level at input pin T2EX • No interrupt is generated • Count up – Start counting from 16-Bit reload value, overflow at $FFFF_H$ – Reload event triggered by overflow condition – Programmable reload value in register RC2 • Count down – Start counting from $FFFF_H$, underflow at value defined in register RC2 – Reload event triggered by underflow condition – Reload value fixed at $FFFF_H$
Channel capture	• Count up only • Start counting from 0000_H, overflow at $FFFF_H$ • Reload event triggered by overflow condition • Reload value fixed at 0000_H • Capture event triggered by falling/rising edge at pin T2EX • Captured timer value stored in register RC2 • Interrupt is generated with reload or capture event

Timer2 can be started by using TR2 bit by hardware or software. Timer2 can be started by setting TR2 bit by software. If bit T2RHEN is set, Timer2 can be started by hardware. Bit T2REGS defines the event on pin T2EX: falling edge or rising edge, that can set the run bit TR2 by hardware. Timer2 can only be stopped by resetting TR2 bit by software.

24.2.1 Auto-reload mode

The auto-reload mode is selected when the bit CP_RL2 in register CON is zero. In the auto-reload mode, Timer2 counts to an overflow value and then reloads its register contents with a 16-bit start value for a fresh counting sequence. The overflow condition is indicated by setting bit TF2 in the CON register. This will then generate an interrupt request to the core. The overflow flag TF2 must be cleared by software.

The auto-reload mode is further classified into two categories depending upon the DCEN control bit.

24.2.1.1 Up/down count disabled

If DCEN = 0, the up-down count selection is disabled. The timer, therefore, functions as a pure up counter/timer only. The operational block diagram is shown in **Figure 294**.

In this mode, if EXEN2 = 0, the timer starts to count up to a maximum of $FFFF_H$, once TR2 is set. Upon overflow, bit TF2 is set and the timer register is reloaded with the 16-bit reload value of the RC2 register. This reload value is chosen by software, prior to the occurrence of an overflow condition. A fresh count sequence is started and the timer counts up from this reload value as in the previous count sequence.

If EXEN2 = 1, the timer counts up to a maximum of $FFFF_H$ once TR2 is set. A 16-bit reload of the timer registers from register RC2 is triggered either by an overflow condition or by a negative/positive edge (chosen by MOD.EDGESEL) at input pin T2EX. If an overflow caused the reload, the overflow flag TF2 is set. If a negative/positive transition at pin T2EX caused the reload, bit EXF2 is set. In either case, an interrupt is generated to the core and the timer proceeds to its next count sequence. The EXF2 flag, similar to the TF2, must be cleared by software.

If bit T2RHEN is set, Timer2 is started by first falling edge/rising edge at pin T2EX, which is defined by bit T2REGS. If bit EXEN2 is set, bit EXF2 is also set at the same point when Timer2 is started with the same falling edge/rising edge at pin T2EX, which is defined by bit EDGESEL. The reload will happen with the following negative/positive transitions at pin T2EX, which is defined by bit EDGESEL.

Note: In counter mode, if the reload via T2EX and the count clock T2 are detected simultaneously, the reload takes precedence over the count. The counter increments its value with the following T2 count clock.

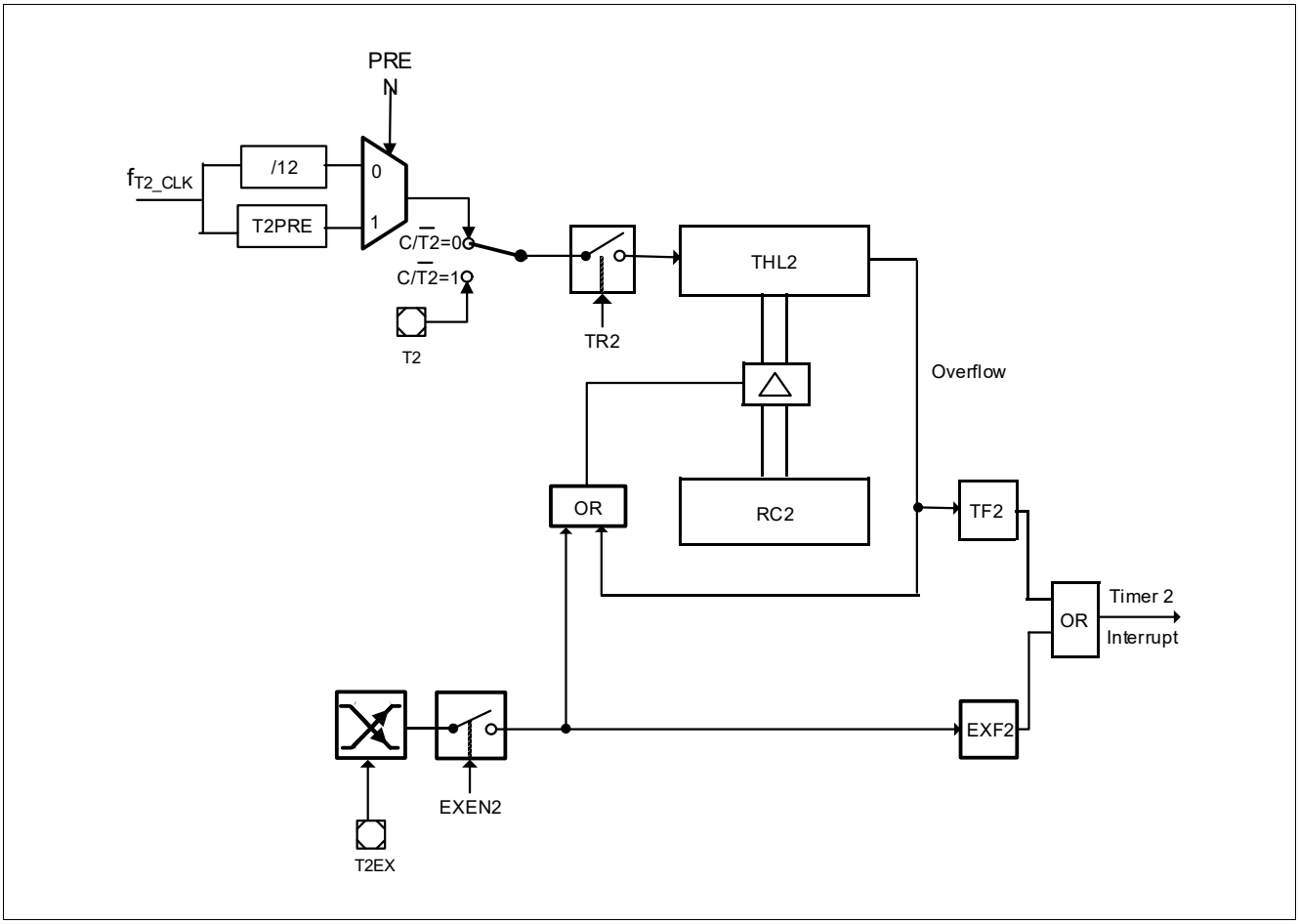


Figure 294 Auto-reload mode (DCEN = 0)

24.2.1.2 Up/down count enabled

If DCEN = 1, the up-down count selection is enabled. The direction of count is determined by the level at input pin T2EX. The operational block diagram is shown in **Figure 295**.

A logic 1 at pin T2EX sets the Timer2 to up counting mode. The timer, therefore, counts up to a maximum of $FFFF_H$. Upon overflow, bit TF2 is set and the timer register is reloaded with a 16-bit reload value of the RC2 register. A fresh count sequence is started and the timer counts up from this reload value as in the previous count sequence. This reload value is chosen by software, prior to the occurrence of an overflow condition.

A logic 0 at pin T2EX sets the Timer2 to down counting mode. The timer counts down and underflows when the THL2 value reaches the value stored at register RC2. The underflow condition sets the TF2 flag and causes $FFFF_H$ to be reloaded into the THL2 register. A fresh down counting sequence is started and the timer counts down as in the previous counting sequence.

If bit T2RHEN is set, Timer2 can only be started either by rising edge (T2REGS = 1) at pin T2EX and then do the up counting, or be started by falling edge (T2REGS = 0) at pin T2EX and then do the down counting.

In this mode, bit EXF2 toggles whenever an overflow or an underflow condition is detected. This flag, however, does not generate an interrupt request.

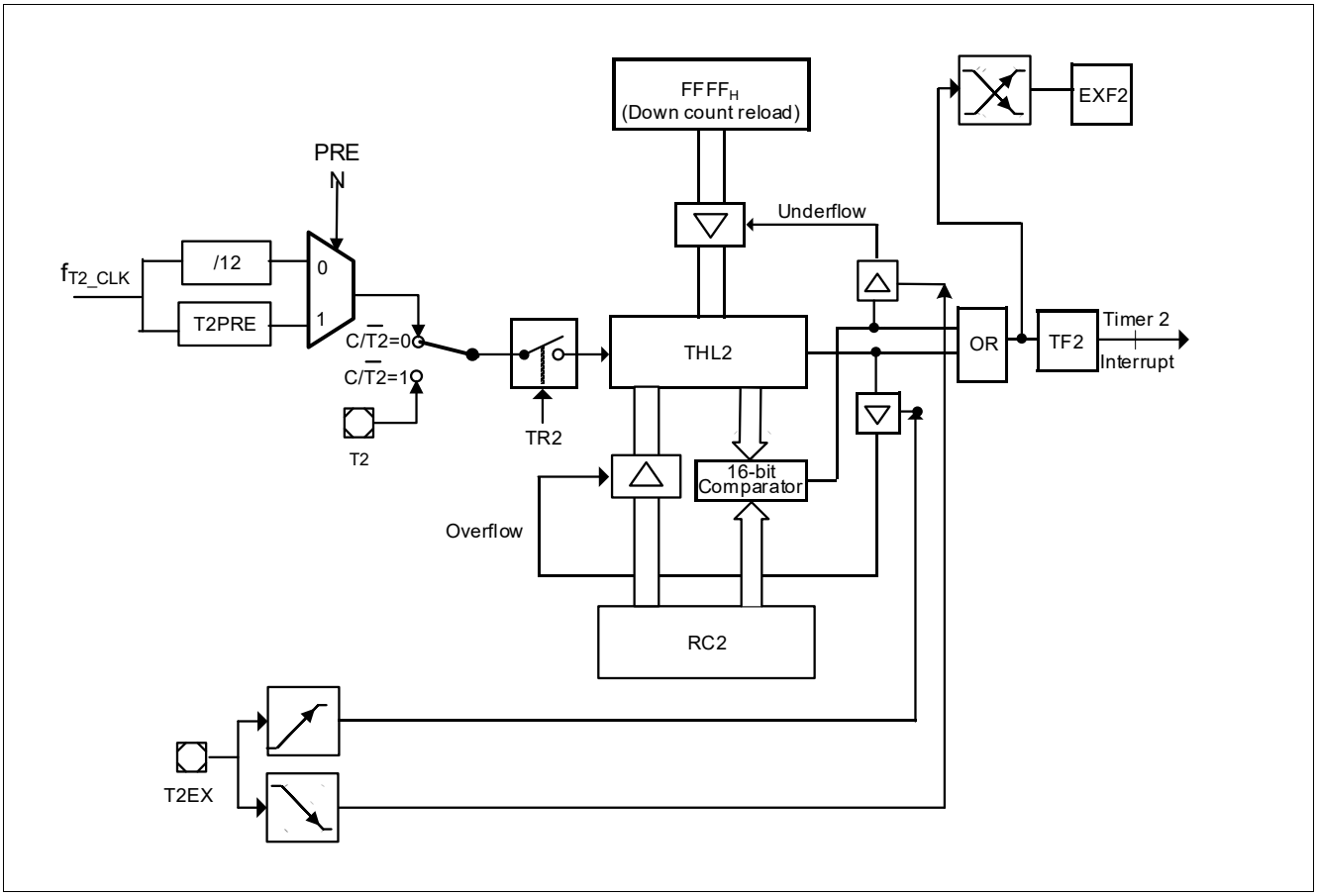


Figure 295 Auto-reload mode (DCEN = 1)

24.2.2 Capture mode

In order to enter the 16-bit capture mode, bits CP_RL2 and EXEN2 in register CON must be set. In this mode, the down count function must remain disabled. The timer functions as a 16-bit timer or counter and always counts up to $FFFF_H$ and overflows. Upon an overflow condition, bit TF2 is set and the timer reloads its registers with 0000_H. The setting of TF2 generates an interrupt request to the core.

Additionally, with a falling/rising edge on pin T2EX (chosen by MOD.EDGESEL) the contents of the timer register (THL2) are captured into the RC2 register. The external input is sampled in every f_{T2_CLK} cycle. When a sampled input shows a low (high) level in one f_{T2_CLK} cycle and a high (low) in the next f_{T2_CLK} cycle, a transition is recognized. If the capture signal is detected while the counter is being incremented, the counter is first incremented before the capture operation is performed. This ensures that the latest value of the timer register is always captured.

If bit T2RHEN is set, Timer2 is started by first falling edge/rising edge at pin T2EX, which is defined by bit T2REGS. If bit EXEN2 is set, bit EXF2 is also set at the same point when Timer2 is started with the same falling edge/rising edge at pin T2EX, which is defined by bit EDGESEL. The capture will happen with the following negative/positive transitions at pin T2EX, which is defined by bit EDGESEL.

When the capture operation is completed, bit EXF2 is set and can be used to generate an interrupt request. **Figure 296** describes the capture function of Timer2.

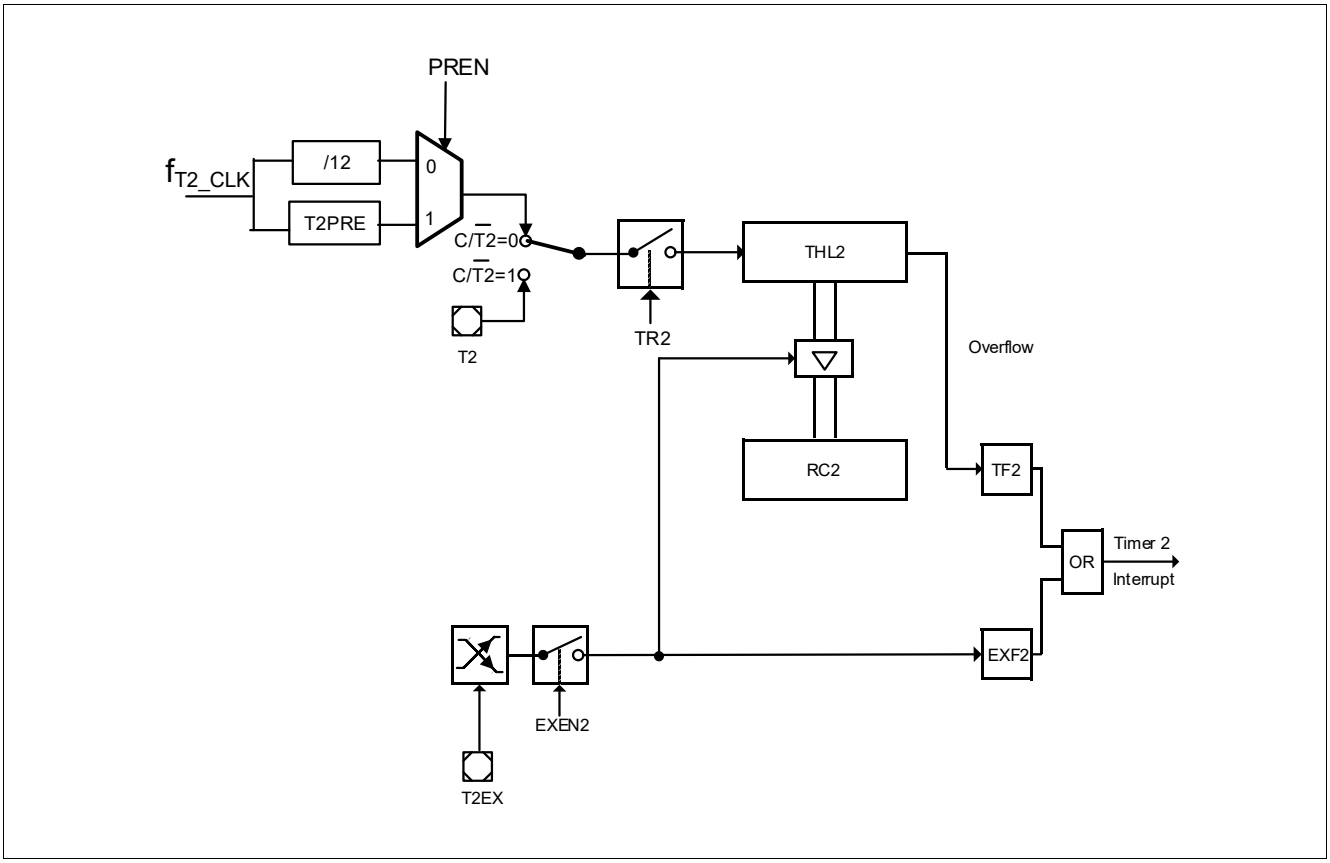


Figure 296 Capture mode

24.3 Count clock

The count clock for the auto-reload mode is chosen by the bit C_T2 in register CON. If C_T2 = 0, a count clock of $f_{T2\_CLK}/12$ (if prescaler is disabled) is used for the count operation.

If C_T2 = 1, Timer2 behaves as a counter that counts 1-to-0 transitions of input pin T2. The counter samples pin T2 over 2 f_{T2_CLK} cycles. If a 1 was detected during the first clock and a 0 was detected in the following clock, then the counter increments by one. Therefore, the input levels should be stable for at least 1 clock.

If bit T2RHEN is set, Timer2 can be started by the falling edge/rising edge on pin T2EX, which is defined by bit T2REGS.

Note: If pin T2 is not connected, counting clock function on pin T2 cannot be used.

24.4 Input control

The external trigger and counter inputs of the two Timer2 modules can be selected from several different sources. This selection is performed via the corresponding input control and select bits in SFR CON. See **Figure 297**.

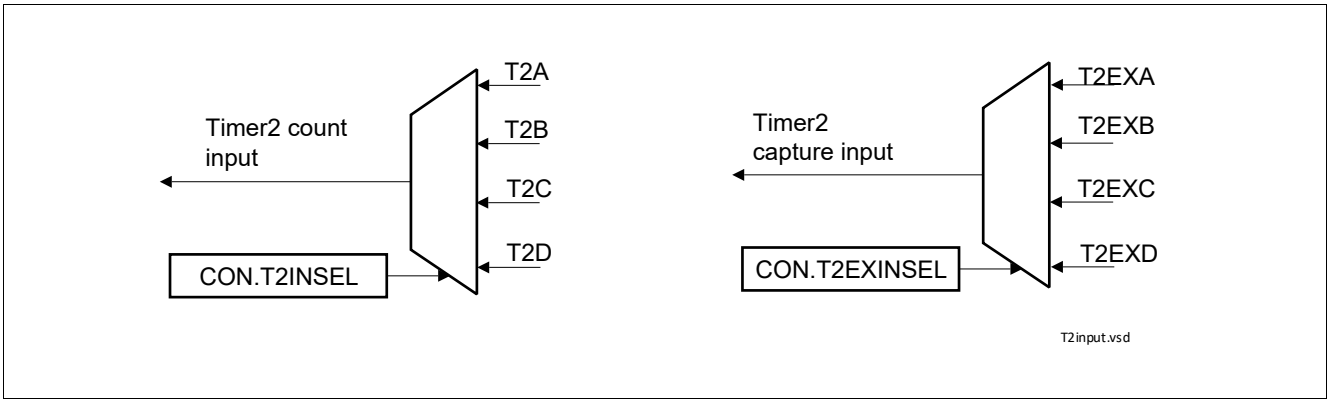


Figure 297 Input control

24.5 Register description T20 and T21

All Timer20 and Timer21 register names described in the following sections will be referenced in other chapters with the module name prefix “T20_” and “T21_”, respectively.

24.5.1 T20/T21 Address Maps

Table 295 Register Address Space

Module	Base Address	End Address	Note
T20	48028000_H	$4802BFFF_H$	Timer20
T21	$4802C000_H$	$4802FFFF_H$	Timer21

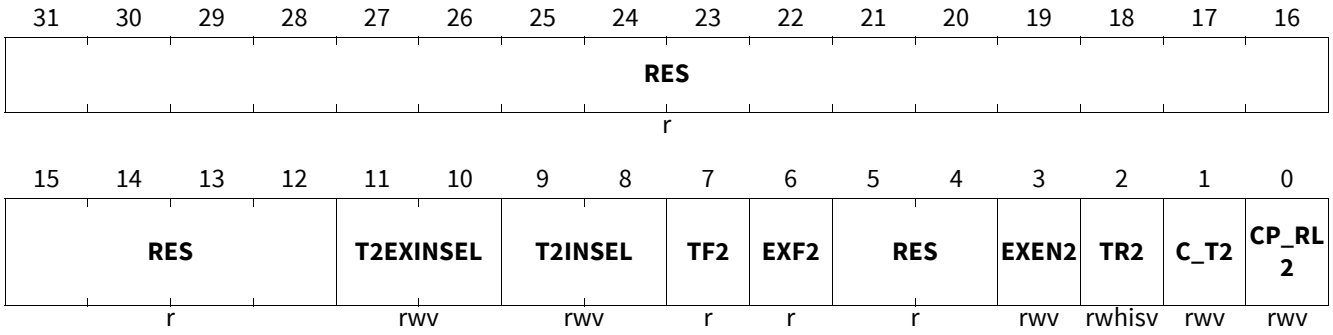
Table 296 Register Overview - T2 (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CON	Timer 2 Control Register	0000_H	948
MOD	Timer 2 Mode Register	0004_H	949
RC	Timer 2 Reload/Capture Register	0008_H	950
CNT	Timer 2 Count Register	0010_H	951
ICLR	Timer 2 Interrupt Clear Register	0018_H	952
CON1	Timer 2 Control Register 1	$001C_H$	952

24.5.2 T2 Registers

Timer 2 Control Register

CON
Timer 2 Control Register **(0000_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
CP_RL2	0	rwv	**Capture/Reload Select** If TCLK/RCLK = 1, this bit is ignored. 0_B **Reload**, upon overflow or upon negative/positive transition at pin T2EX (when EXEN2=1). 1_B **Capture**, Timer 2 data register contents on the negative/positive transition at pin T2EX, provided EXEN2=1.The negative or positive transition at Pin T2EX is selected by bit EDGESEL.
C_T2	1	rwv	**Timer or Counter Select** 0_B **Timer**, function selected. 1_B **Count**, upon negative edge at pin T2.
TR2	2	rwhisv	**Timer 2 Start/Stop Control** 0_B **STOP**, Timer 2. 1_B **START**, Timer 2.
EXEN2	3	rwv	**Timer 2 External Enable Control** 0_B **DISABLED**, External events are disabled. 1_B **ENABLED**, External events are enabled in Capture/Reload/Baud-rate Generator Mode.
RES	5:4, 31:12	r	**Reserved**
EXF2	6	r	**Timer 2 External Flag** In Capture/Reload/Baud-rate Generator Mode, this bit is set by hardware when a negative/positive transition occurs at pin T2EX, if bit EXEN2=1. This bit must be cleared by software.

Register description T20 and T21

Field	Bits	Type	Description
TF2	7	r	**Timer 2 Overflow/Underflow Flag** Set by a Timer 2 overflow/underflow. Must be cleared by software. TF2 will not be set when either RCLK = 1 or TCLK = 1.
T2INSEL	9:8	rwv	**Timer 2 count input selection** 00_B **SEL_A**, T2A input selected 01_B **SEL_B**, T2B input selected 10_B **SEL_C**, T2C input selected 11_B **SEL_D**, T2D input selected
T2EXINSEL	11:10	rwv	**Timer 2 capture input selection** 00_B **SEL_A**, T2EXA input selected 01_B **SEL_B**, T2EXB input selected 10_B **SEL_C**, T2EXC input selected 11_B **SEL_D**, T2EXD input selected

Timer 2 Mode Register

MOD

Timer 2 Mode Register **(0004_H)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								T2REGS	T2RHEN	EDGESEL	PREN	T2PRE			DCEN
r								rwv	rwv	rwv	rwv	rwv			rwv

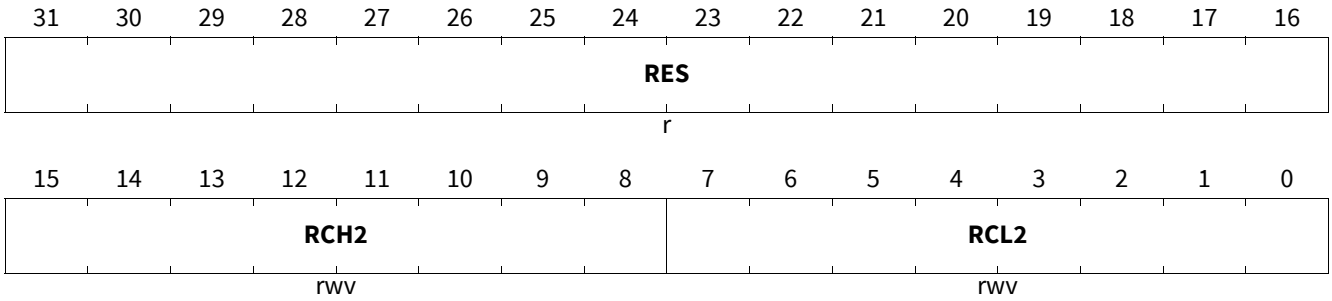
Field	Bits	Type	Description
DCEN	0	rwv	**Up/Down Counter Enable** 0_B **DISABLED**, Up/Down Counter function is disabled 1_B **ENABLED**, Up/Down Counter function is enabled and controlled by pin T2EX (Up=1, Down=0)
T2PRE	3:1	rwv	**Timer 2 Prescaler Bit** Selects the input clock for Timer 2 which is derived from the peripheral clock. 000_B **DIV1**, 001_B **DIV2**, 010_B **DIV4**, D 011_B **DIV8**, … 111_B **DIV128**,

Register description T20 and T21

Field	Bits	Type	Description
PREN	4	rwv	**Prescaler Enable** 0_B **DISABLED**, Prescaler is disabled and the 2 or 12 divider takes effect. 1_B **ENABLED**, Prescaler is enabled (see T2PRE bit) and the 2 or 12 divider is bypassed.
EDGESEL	5	rwv	**Edge Select in Capture Mode/Reload Mode/Baud-rate Generator Mode** 0_B **FALLING**, The falling edge at Pin T2EX is selected. 1_B **RISING**, The rising edge at Pin T2EX is selected.
T2RHEN	6	rwv	**Timer 2 External Start Enable** 0_B **DISABLED**, Timer 2 External Start is disabled. 1_B **ENABLED**, Timer 2 External Start is enabled.
T2REGS	7	rwv	**Edge Select for Timer 2 External Start** 0_B **FALLING**, The falling edge at Pin T2EX is selected. 1_B **RISING**, The rising edge at Pin T2EX is selected.
RES	31:8	r	**Reserved** Returns 0 if read; should be written with 0.

Timer 2 Reload/Capture Register

RC
Timer 2 Reload/Capture Register (0008$_H$) RESET_TYPE_5 Value: 0000 0000$_H$



Field	Bits	Type	Description
RCL2	7:0	rwv	**Reload/Capture Value** Note: Reload/Capture Value can be set by software (highest priority) and is updated by hardware during capture mode. These contents are loaded into the timer register upon an overflow condition, if CP_RL2=0.If CP_RL2=1, this register is loaded with the current timer count upon a negative/positive transition at pin T2EX when EXEN2=1.

Register description T20 and T21

Field	Bits	Type	Description
RCH2	15:8	rwv	**Reload/Capture Value** Note: Reload/Capture Value can be set by software (highest priority) and is updated by hardware during capture mode. These contents are loaded into the timer register upon an overflow condition, if CP_RL2=0.If CP_RL2=1, this register is loaded with the current timer count upon a negative/positive transition at pin T2EX when EXEN2=1.
RES	31:16	r	**Reserved** Always read as 0

Timer 2 Count Register

CNT
Timer 2 Count Register **(0010_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

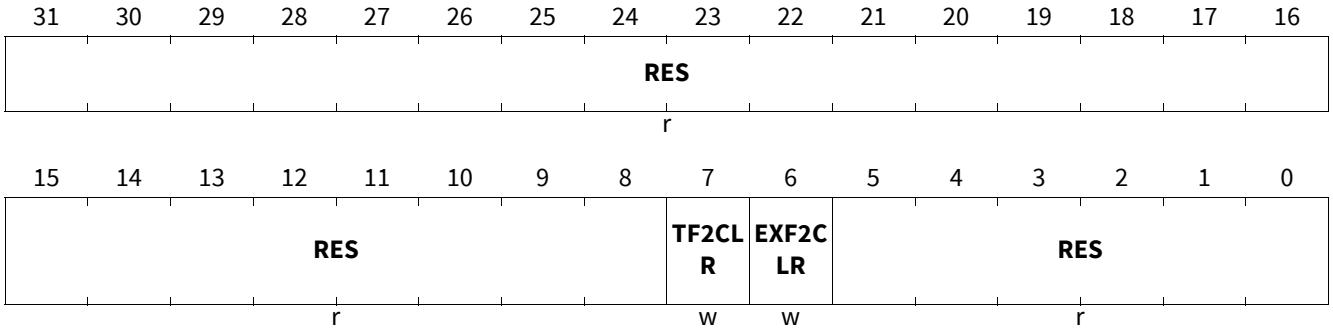
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T2H								**T2L**							
rwv								rwv							

Field	Bits	Type	Description
T2L	7:0	rwv	**Timer 2 Value** These bits indicate the current timer value T2[7:0]. Note: Timer 2 can be updated by software (highest priority) and is updated by hardware if T2R is set
T2H	15:8	rwv	**Timer 2 Value** These bits indicate the current timer value T2[15:8]. Note: Timer 2 can be updated by software (highest priority) and is updated by hardware if T2R is set
RES	31:16	r	**Reserved** Always read as 0

Register description T20 and T21

Timer 2 Interrupt Clear Register

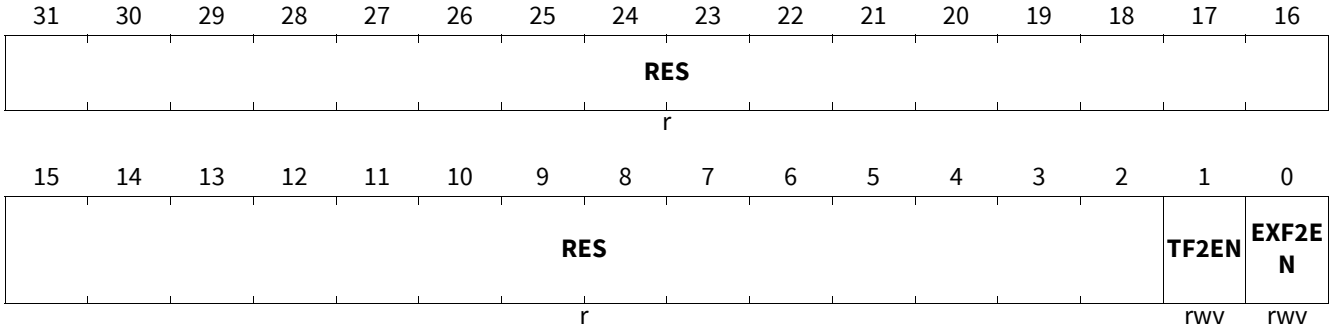
ICLR
Timer 2 Interrupt Clear Register **(0018_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
RES	5:0, 31:8	r	**Reserved** Always read as 0
EXF2CLR	6	w	**External Interrupt Clear Flag** 0_B **NA**, External interrupt is not cleared. 1_B **Clear**, External interrupt
TF2CLR	7	w	**Overflow/Underflow Interrupt Clear Flag** 0_B **NA**, Overflow/underflow interrupt is not cleared. 1_B **Clear**, Overflow/underflow interrupt

Timer 2 Control Register 1

CON1
Timer 2 Control Register 1 **($001C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
EXF2EN	0	rwv	**External Interrupt Enable** 0_B **DISABLE**, External interrupt. 1_B **ENABLE**, External interrupt
TF2EN	1	rwv	**Overflow/Underflow Interrupt Enable** 0_B **DISABLE**, Overflow/underflow interrupt. 1_B **ENABLE**, Overflow/underflow interrupt.

Register description T20 and T21

Field	Bits	Type	Description
RES	31:2	r	**Reserved** Always read as 0

25 Capture/Compare Unit 7 (CCU7)

25.1 Features overview

The CCU7 is a high-resolution 16-bit capture and compare unit with application-specific modes, mainly for AC drive control. Special operating modes support the control of Brushless DC-motors using Hall sensors or Back-EMF detection. Furthermore, block commutation and control mechanisms for multi-phase machines are supported.

This chapter gives an overview over the different building blocks and their main features.

The CCU7 provides following features:

- Timer T12 block features:
  - Six compare channels
  - Supports generation of three-phase PWM (six outputs, individual signals for high-side and low-side switches)
  - 16-bit resolution, maximum count frequency (peripheral clock)
  - Dead-time control for each channel to avoid short-circuits in the power stage
  - Concurrent update of T12 registers
  - Center-aligned and edge-aligned PWM can be generated, as well as rising edge and duration PWM pulses
  - Single-shot mode supported
  - Start can be controlled by external events
  - Capability of counting external events
- Timer T13 block features:
  - One independent compare channel with one output
  - 16-bit resolution, maximum count frequency (peripheral clock)
  - Concurrent update of T13 registers
  - Can be synchronized to T12
  - Event generation at period-match and compare-match
  - Single-shot mode supported
  - Start can be controlled by external events
  - Capability of counting external events
- Timer T14, T15 and T16 block features:
  - Each with one independent compare channel with one output
  - 16-bit resolution, maximum count frequency (module clock)
  - Dead-time control for each channel to avoid short-circuits in the power stage
  - Concurrent update of T14, T15 and T16 registers
  - Can be synchronized to T12
  - Event generation at period-match and compare-match
  - Single-shot mode supported
  - Start can be controlled by external events
  - Capability of counting external events

- Additional specific functions:
  - Block commutation support for brushless DC-drives with programmable state pattern, event-triggered next state switching and background speed capture
  - Programmable Hall-sensor pattern detection with noise filter
  - Integrated error handling
  - Fast emergency stop without CPU load via external signal ($\overline{\text{CTRAP}}$)
  - Control modes for multi-channel AC-drives
  - Output levels can be selected and adapted to the power stage

25.2 Block diagram

The CCU7 is comprised of a timer T12 block with six capture/compare channels, and a timer T13, T14, T15 and T16 block with one compare channel each. The T12 channels can independently generate PWM signals or accept capture triggers, or they can jointly generate control signal patterns to drive AC-motors or inverters.

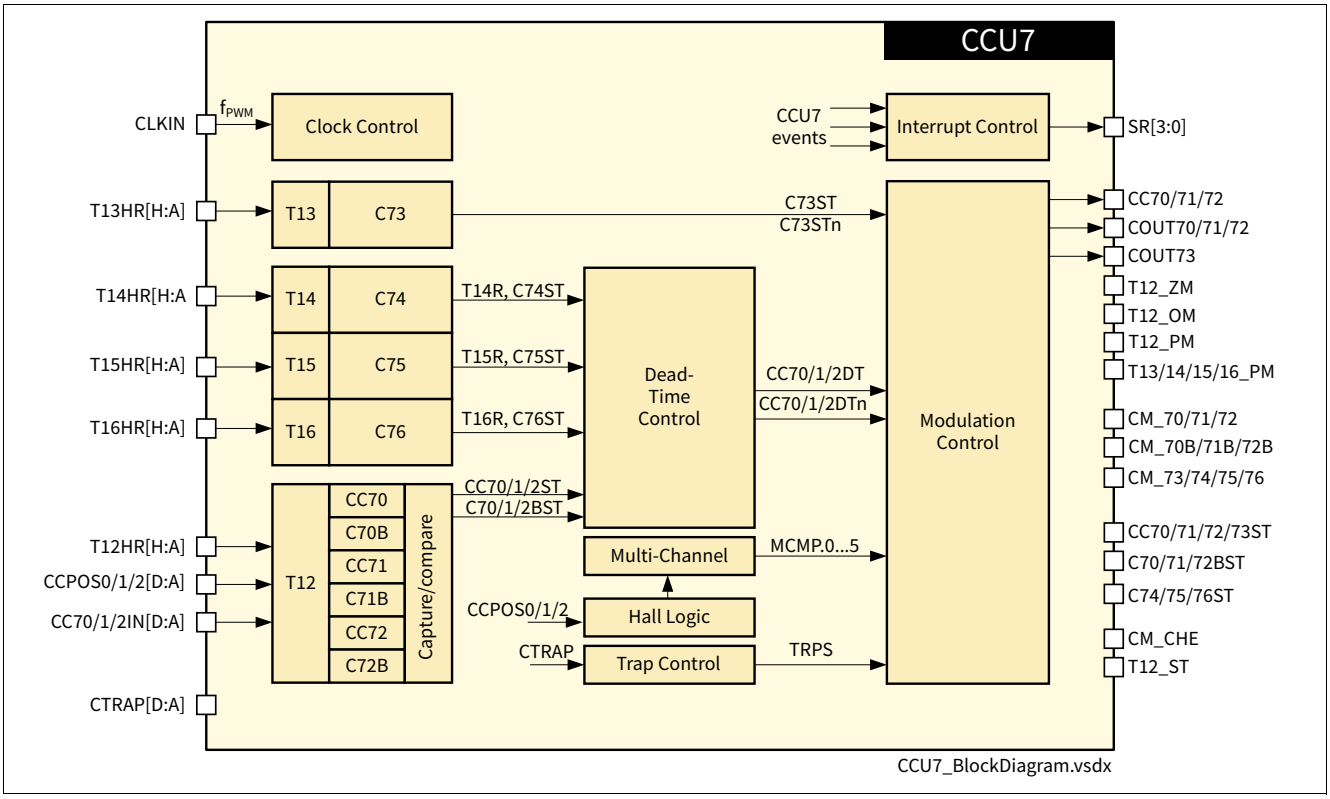


Figure 298 Block diagram CCU7

25.3 Interrupts

This chapter describes the interrupt handling of the CCU7 module.

Note: The actual interrupt functionality of the CCU7 is not used in the TLE989x/TLE988x. However, the status flags and the SRx (Service Request) lines are used to trigger actions in other modules of the TLE989x/TLE988x.

25.3.1 Interrupt structure

The HW interrupt event or the SW setting of the corresponding interrupt set bit (in register **ISS**/**ISS_2**) sets the event indication flags (in register **IS**/**IS_2**) and can trigger the interrupt generation. The interrupt pulse is generated independently from the interrupt status flag in registers IS/IS_2 (it is not necessary to clear the related status bit to be able to generate another interrupt). The interrupt flag can be cleared by SW by writing to the corresponding bit in register **ISR**/**ISR_2**.

If enabled by the related interrupt enable bit in register **IEN**/**IEN_2**, an interrupt pulse can be generated on one of the four service request outputs (SR0 to SR3) of the module. If more than one interrupt source is connected to the same interrupt node pointer (in register **INP**/**INP_2**), the requests are logically OR-combined to one common service request output.

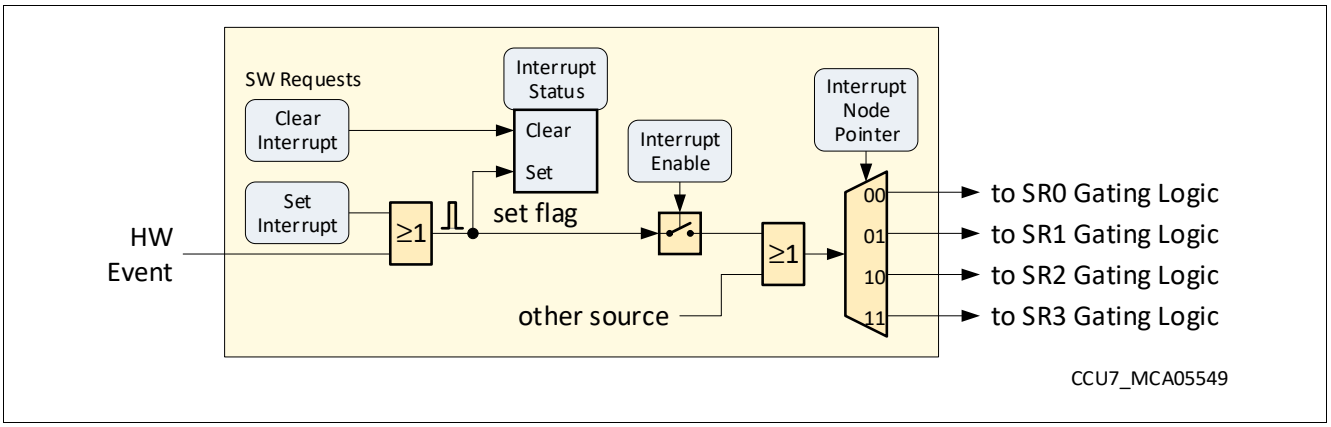


Figure 299 General interrupt structure

25.3.2 Interrupt sources

The following two figures illustrate the interrupt events of the CCU7, as well as their respective control and status bits, and how they are combined regarding the SRx lines.

Note that the lines SRx_[N:A] are ORed together onto one line, see **Figure 302** (see **Figure 361** for SRx_N).

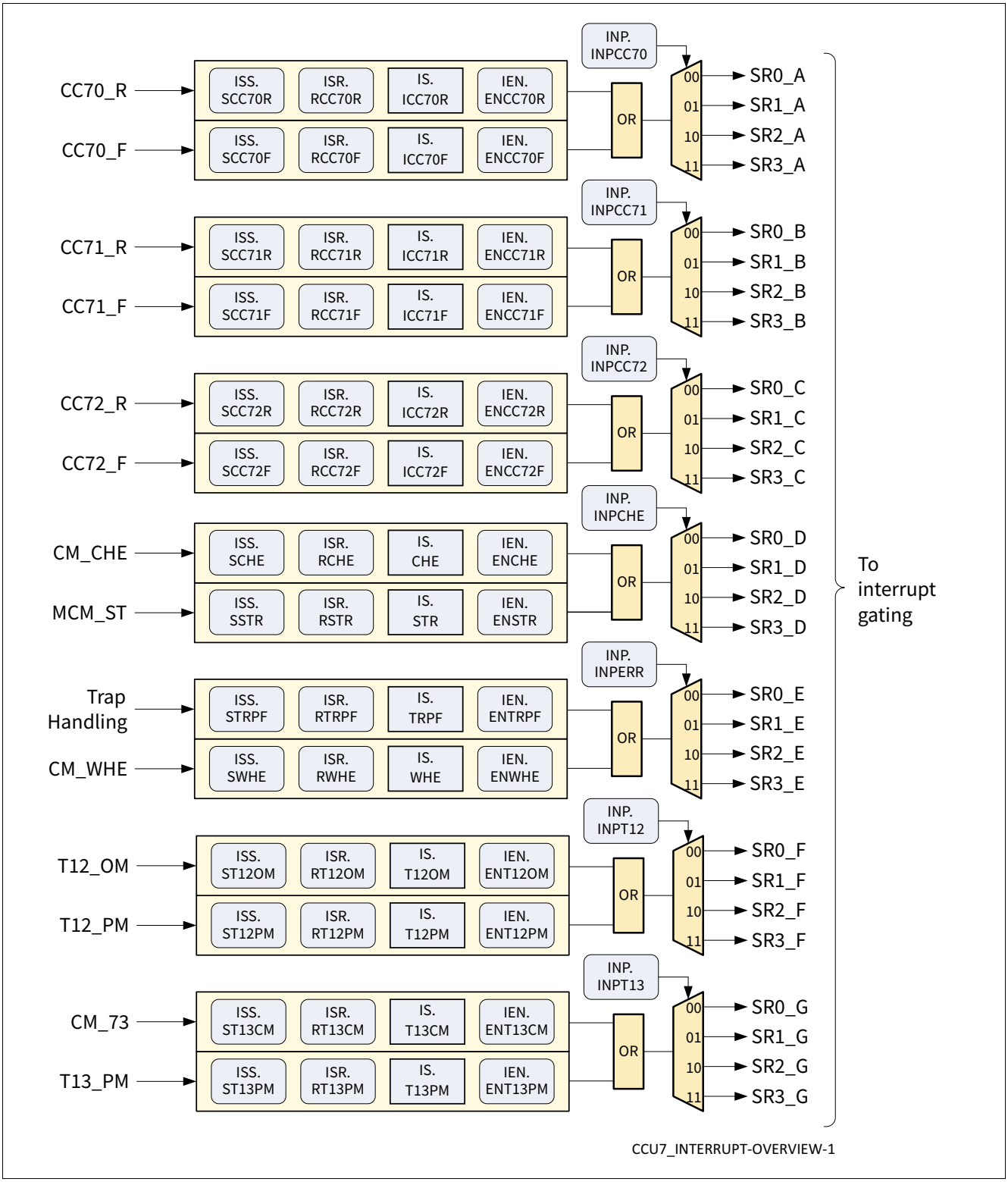


Figure 300 Interrupt sources and events, part 1

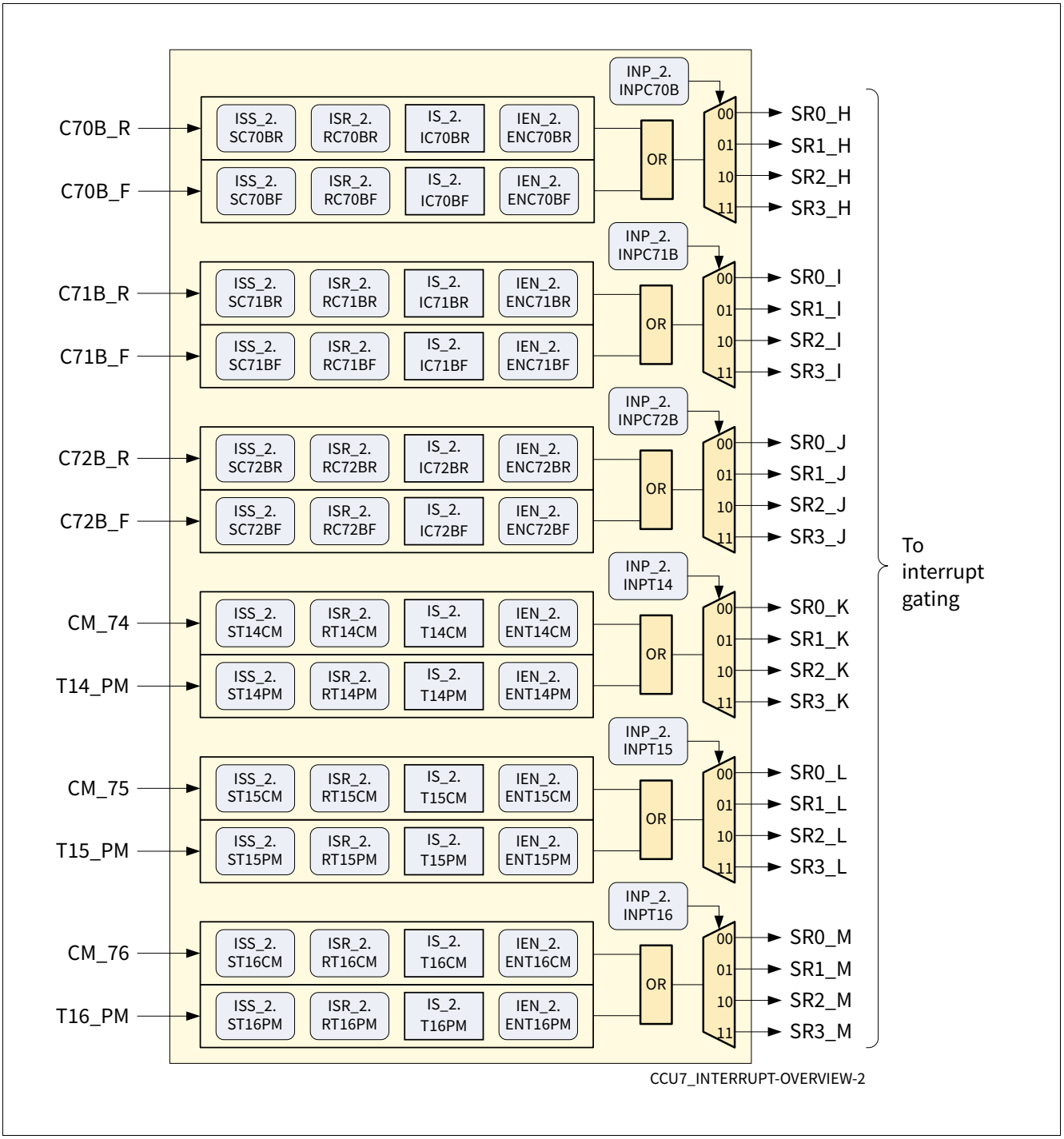


Figure 301 Interrupt sources and events, part 2

25.3.3 Interrupt gating

The interrupt gating allows to enable the activation of a service request line SRx only on certain conditions. The following options are available:

- Always: The activation of an SRx line is independent of other conditions
- CDIR = 1_B: The activation of an SRx line is only possible when timer T12 is counting down
- CDIR = 0_B: The activation of an SRx line is only possible when timer T12 is counting up
- External control: The activation of an SRx line is only possible when the respective input signal SR_GTx is at high level

These options are controlled via register **IGT**.

Note: In the CCU7 module inputs SR_GT0 through SR_GT3 are tied to 0.

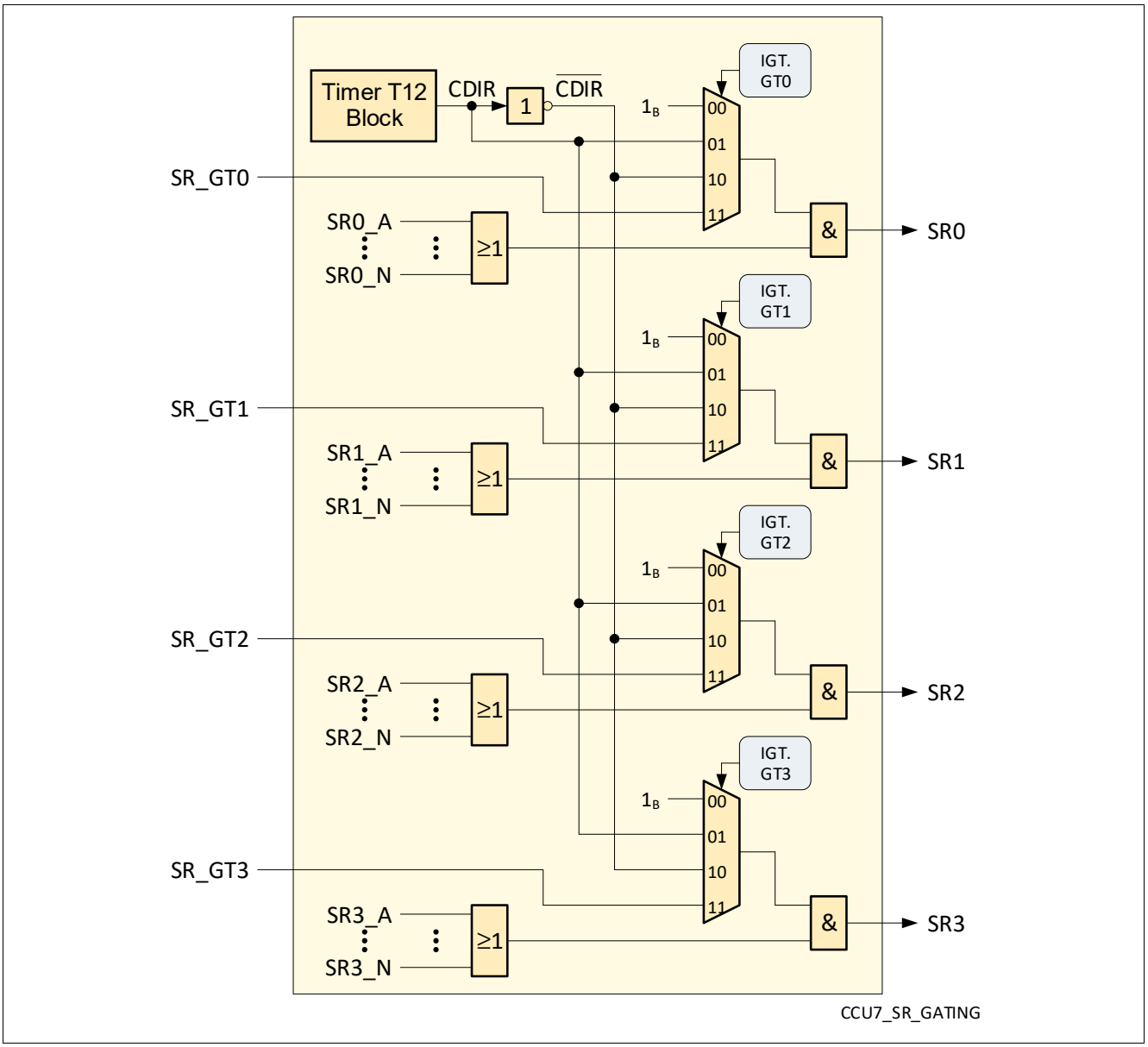


Figure 302 Interrupt gating structure

25.4 Clock control

The CCU7 is clocked with f_{PWM}. A fractional clock divider allows to divide the clock down with a very fine granularity. Two individual divider output frequencies are generated.

Each of the timers, as well as the dead-time generation block, can select one of the divider frequencies or the f_{PWM} as its input clock. Individual controls for this selection are available in register **T_CLK_CTRL**, as shown in **Figure 303**.

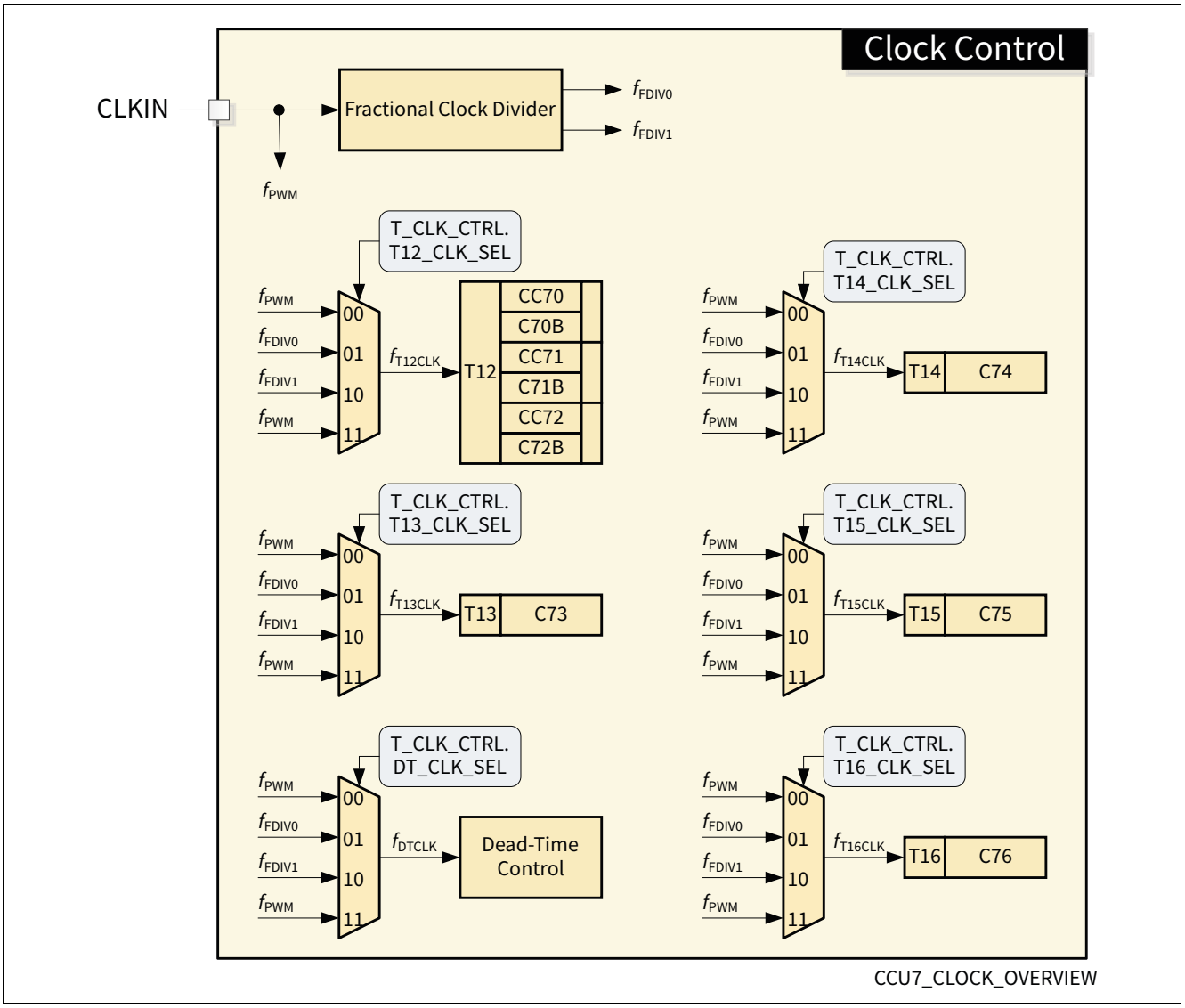


Figure 303 CCU7 clock overview

For more details please refer to **T12 clock selection**, **T13 clock selection**, **T1x clock selection**.

25.4.1 Fractional clock divider (FDIV0, FDIV1)

The fractional clock divider is shown in **Figure 304**. It provides two individual output frequencies, f_{DIV0} and f_{DIV1}. Each of these can be adjusted with a very fine granularity.

The fractional clock divider generates an output frequency based on two configurable values, P and Q:

$$f_{DIVx} = f_{PWM} * P / Q \quad \text{(for } P < Q\text{)} \tag{25.1}$$

For P ≥ Q, the output frequency f_{DIVx} is the same as the input frequency f_{PWM}.

Each of the two dividers has its own P and Q value pair, stored in registers **T_FDIVx (x=0-1)**, respectively. To allow a flexible selection of the P/Q value pair, there are two options for the bit width of P and Q:

- Both can either be 8-bit values, or P is a 6-bit value, while Q is a 10-bit value
- The selection of the bit width is done via bit fields FDIV0_SEL/FDIV1_SEL in register **T_CLK_CTRL**:
  - FDIVx_SEL = 0_B: P and Q are both 8-bit values. Bit field PQ in registers **T_FDIVx (x=0-1)** is interpreted such that P = bits[15:8], and Q = bits[7:0]
  - FDIVx_SEL = 1_B: P is a 6-bit value, while Q is a 10-bit value. Bit field PQ in registers **T_FDIVx (x=0-1)** is interpreted such that P = bits[15:10], and Q = bits[9:0]
- The divider settings can be chosen with registers **T_FDIVx (x=0-1)**
- Depending on bit **T_CLK_CTRL**.FDIV0_SEL / **T_CLK_CTRL**.FDIV1_SEL the values for P and Q of registers **T_FDIVx (x=0-1)** are interpreted as 8/8 value or 6/10 value
- If T_CLK_CTRL.FDIVx_SEL = 0: P = T_FDIVx[15:8] ; Q = T_FDIVx[7:0]
- If T_CLK_CTRL.FDIVx_SEL = 1: P = T_FDIVx[15:10] ; Q = T_FDIVx[9:0]

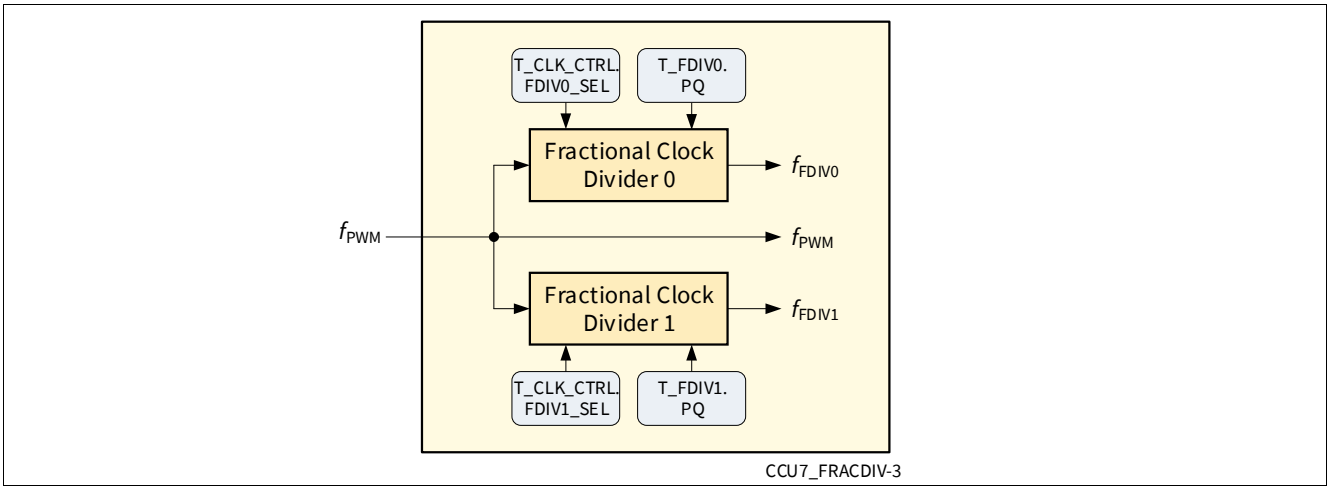


Figure 304 CCU7 fractional divider

Figure 305 illustrates the two options for the PQ bit field interpretation.

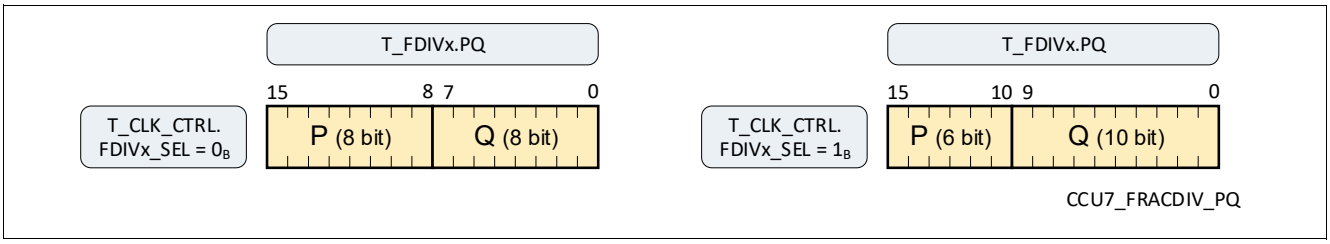


Figure 305 CCU7 fractional divider P/Q selection

25.5 Timer T12

The timer T12 block is the main unit to generate the 3-phase PWM signals. A 16-bit counter is connected to 6 channel registers via comparators, that generate a signal when the counter contents match one of the channel register contents. A variety of control functions facilitate the adaptation of the T12 structure to different application needs.

Besides the 3-phase PWM generation, the T12 block offers options for individual compare and capture functions, as well as dead-time control and hysteresis-like Compare mode.

This chapter provides information about:

- T12 overview (see **Chapter 25.5.1**)
- T12 counting scheme (see **Chapter 25.5.2**)
- T12 compare modes (see **Chapter 25.5.3**)
- T12 capture modes (see **Chapter 25.5.4**)
- T12 shadow transfer (see **Chapter 25.5.5**)
- T12 operating mode selection (see **Chapter 25.5.6**)

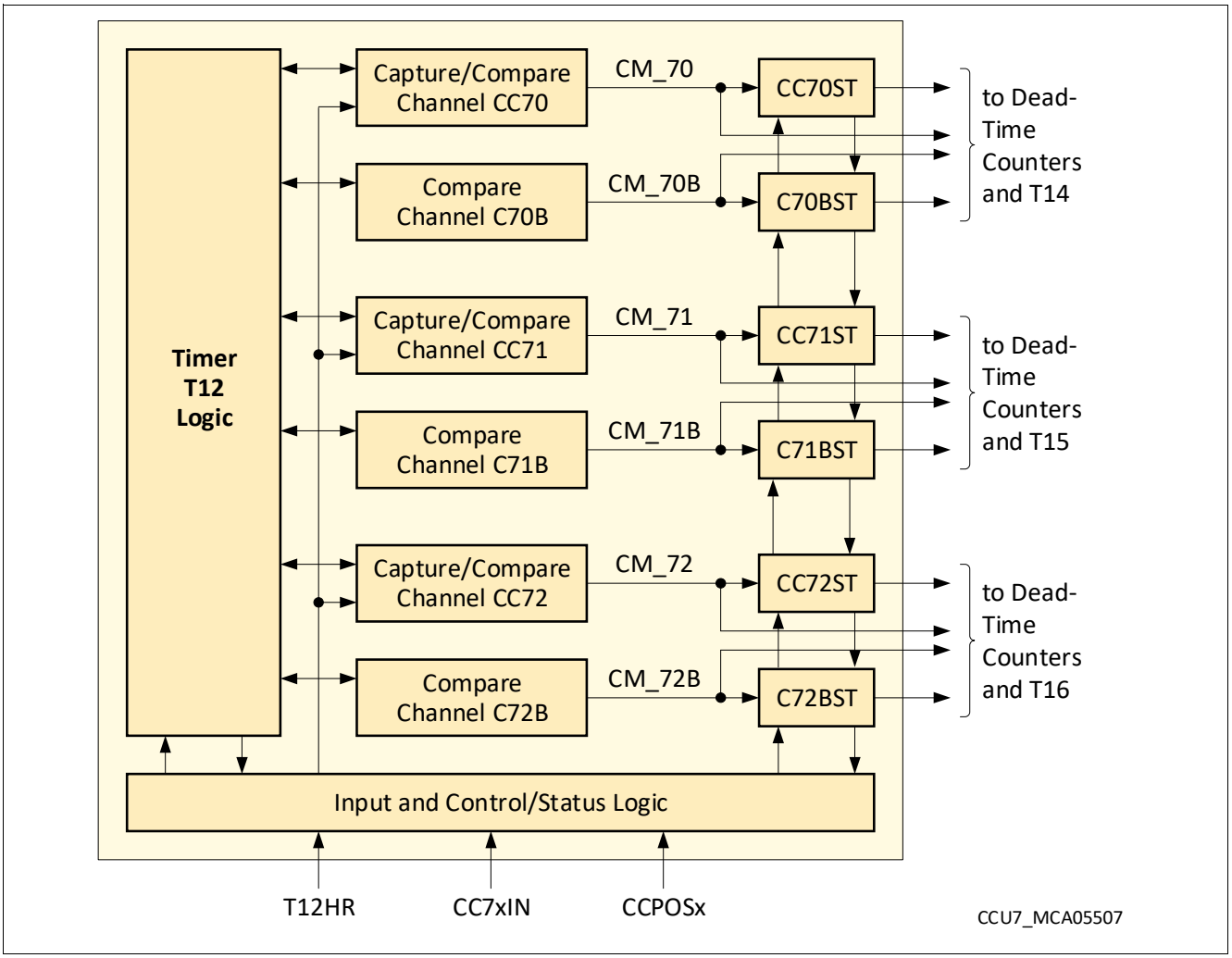


Figure 306 Overview diagram of the timer T12 block

25.5.1 T12 overview

Figure 307 shows a detailed block diagram of timer T12. The functions of the timer T12 block are controlled by bits in registers **TCTR0**, **TCTR2**, and **PISEL0**.

The clock for timer T12 can be chosen between f_{PWM}, f_{DIV0}, or f_{DIV1}, or it can count transitions at input signal T12HR. These options are controlled via bit fields **T_CLK_CTRL**.T12_CLK_SEL, **TCTR0**.T12PRE, and **PISEL0**.ISCNT12. T12 can count up or down, depending on the selected operation mode. A direction flag, CDIR, indicates the current counting direction.

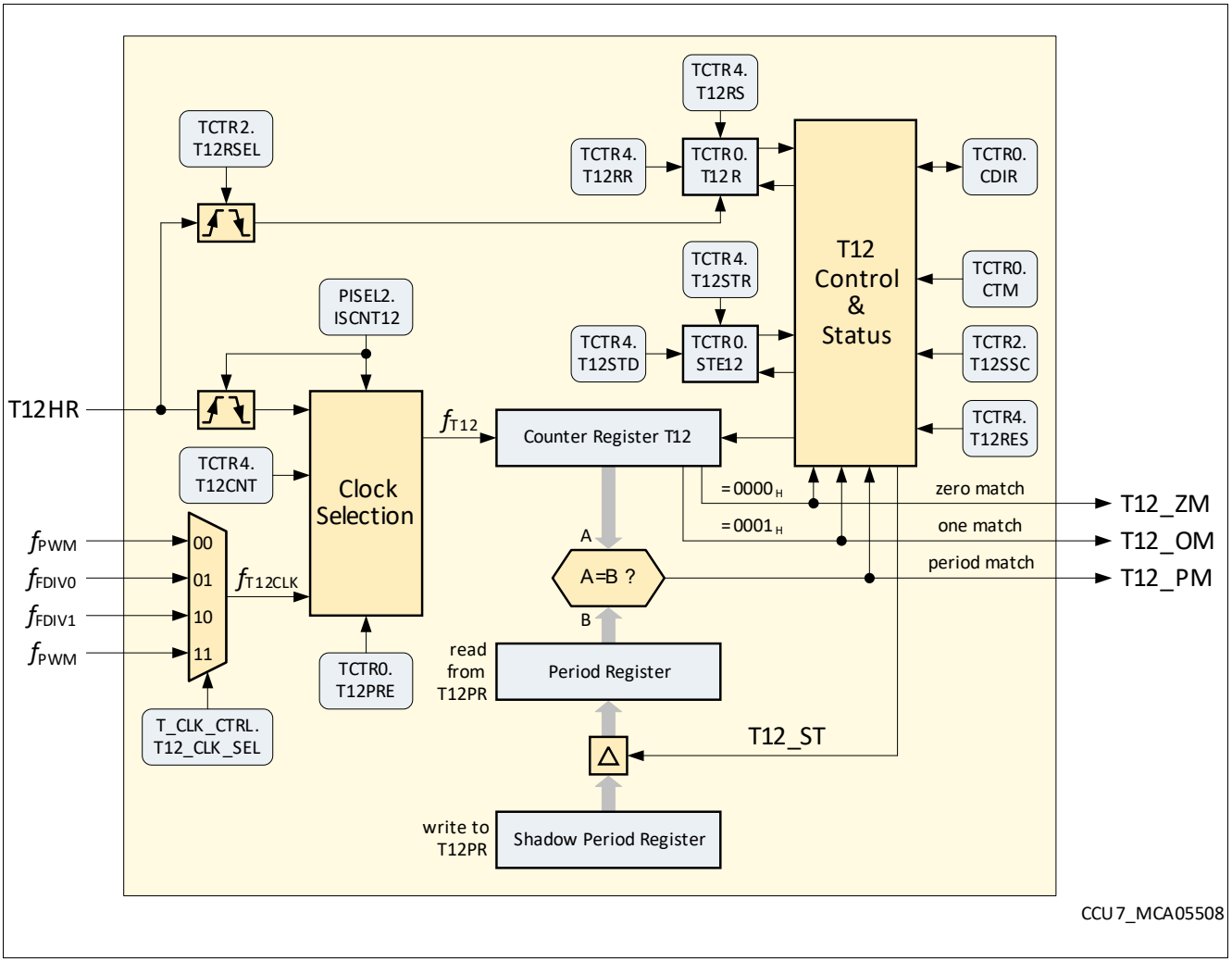


Figure 307 Timer T12 logic and period comparators

Via a comparator, the T12 counter register T12 is connected to a period register **T12PR**. This register determines the maximum count value for T12.

In Edge-aligned mode, T12 is cleared to 0000_H after it has reached the period value defined by T12PR. In Center-aligned mode, the count direction of T12 is set from 'up' to 'down' after it has reached the period value (please note that in this mode, T12 exceeds the period value by one before counting down). In both cases, signal T12_PM (T12 period-match) is generated. The period register receives a new period value from its shadow period register.

A read access to T12PR delivers the current period value at the comparator, whereas a write access targets the shadow period register to prepare another period value. The transfer of a new period value from the shadow period register into the period register (see **Chapter 25.5.5**) is controlled via the 'T12 shadow transfer' control signal, T12_ST. The generation of this signal depends on the operating mode and on the shadow transfer

enable bit STE12 in register **TCTR0**. Providing a shadow register for the period value as well as for other values related to the generation of the PWM signal allows a concurrent update by software for all relevant parameters.

Two further signals indicate whether the counter contents are equal to 0000_H (T12_ZM = zero-match) or 0001_H (T12_OM = one-match). These signals control the counting and switching behavior of T12.

The basic operating mode of T12, either Edge-aligned mode (**Figure 308**) or center-aligned mode (**Figure 309**), is selected via bit CTM in register **TCTR0**. A single-shot control bit, T12SSC, enables an automatic stop of the timer when the current counting period is finished (see **Figure 310** and **Figure 311**).

The start or stop of T12 is controlled by the run bit T12R, which can be modified by bits in register **TCTR4**. The run bit can be set/cleared by software via the associated set/clear bits T12RS or T12RR, it can also be set by a selected edge of the input signal T12HR (**TCTR2**.T12RSEL), or it is cleared by hardware according to preselected conditions.

The timer T12 run bit T12R must not be set while the applied T12 period value is zero.

Timer T12 can be cleared via control bit T12RES. Setting this write-only bit does only clear the timer contents, but has no further effects, for example, it does not stop the timer.

The generation of the T12 shadow transfer control signal, T12_ST, is enabled via bit STE12. This bit can be set or cleared by software indirectly through its associated set/clear control bits T12STR and T12STD in register **TCTR4**.

While timer T12 is running, write accesses to the count register T12 are not taken into account. If T12 is stopped and the Dead-Time counters are 0, write actions to register T12 are immediately taken into account.

25.5.2 T12 counting scheme

This chapter describes the clocking and counting capabilities of timer T12.

25.5.2.1 T12 clock selection

Timer mode

In Timer mode (**PISEL2**.ISCNT12 = 00_B), the input clock f_{T12CLK} of timer T12 can be selected with **T_CLK_CTRL**.T12_CLK_SEL. Possible sources are f_{PWM}, f_{FDIV0}, f_{FDIV1}. In addition, a 1/256 prescaler is available, which can be activated via bit T12PRE in register **TCTR0**.

Counter mode

In Counter mode, timer T12 counts one step:

- If a 1 is written to **TCTR4**.T12CNT and **PISEL2**.ISCNT12 = 01_B
- If a rising edge of input signal T12HR is detected and **PISEL2**.ISCNT12 = 10_B
- If a falling edge of input signal T12HR is detected and **PISEL2**.ISCNT12 = 11_B

25.5.2.2 Edge-aligned mode and Center-aligned mode

Edge-aligned mode

In Edge-aligned mode (**TCTR0**.CTM = 0), timer T12 is always counting upwards (**TCTR0**.CDIR = 0). When reaching the value given by the period register (period-match T12_PM), the value of T12 is cleared with the next counting step (saw tooth shape).

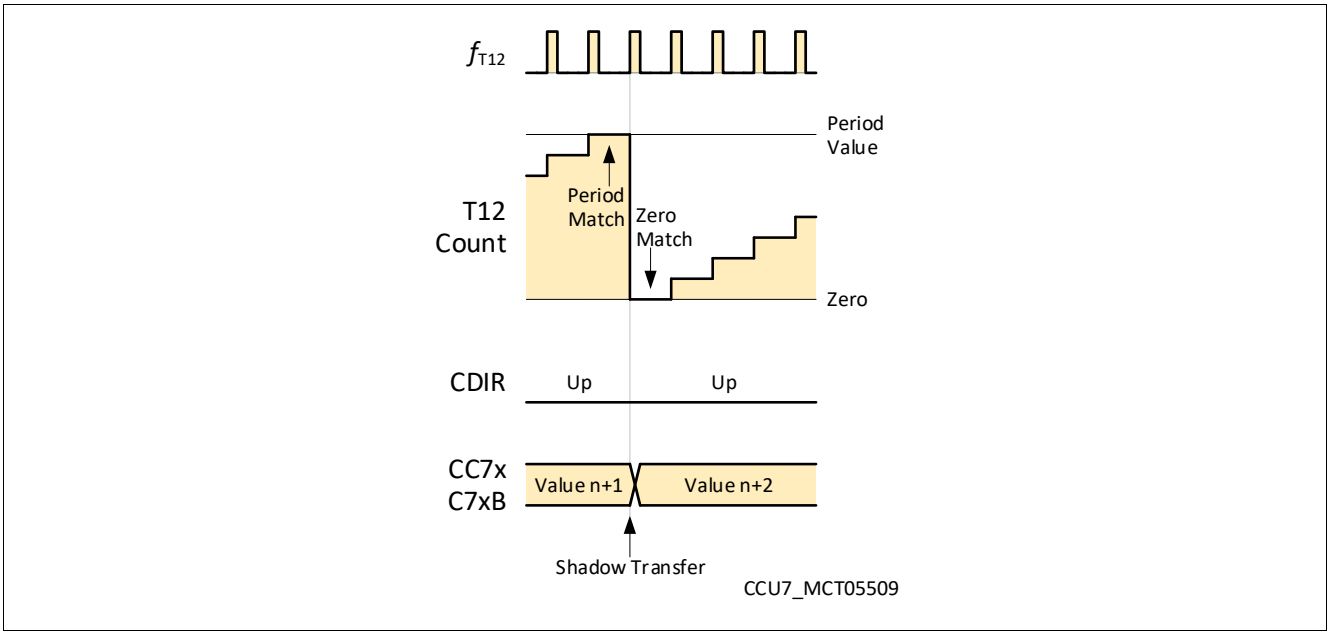


Figure 308 T12 operation in Edge-aligned mode

As a result, in Edge-aligned mode, the timer period is given by:

$T12_{PER}$ = <Period-Value> + 1; in T12 clocks (f_{T12}) (25.2)

Center-aligned mode

In Center-aligned mode (CTM = 1), timer T12 is counting upwards or downwards (triangular shape). When reaching the value given by the period register (period-match T12_PM) while counting upwards (CDIR = 0), the counting direction control bit CDIR is changed to downwards (CDIR = 1) with the next counting step.

When reaching the value 0001_H (one-match T12_OM) while counting downwards, the counting direction control bit CDIR is changed to upwards with the next counting step.

As a result, in Center-aligned mode, the timer period is given by:

$T12_{PER}$ = (<Period-Value> + 1) x 2; in T12 clocks (f_{T12}) (25.3)

- With the next clock event of f_{T12}, the count direction is set to counting up (CDIR = 0) when the counter reaches 0001_H while counting down
- With the next clock event of f_{T12}, the count direction is set to counting down (CDIR = 1) when the period-match is detected while counting up
- With the next clock event of f_{T12}, the counter counts up while CDIR = 0, and it counts down while CDIR = 1

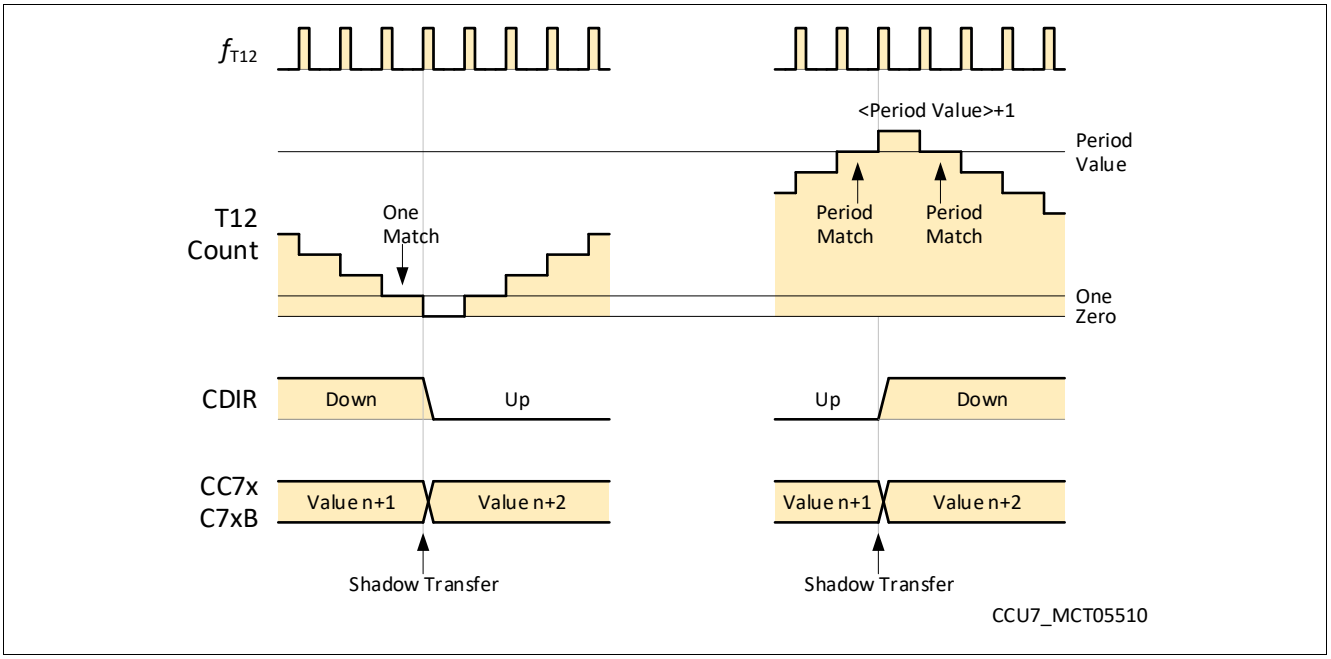


Figure 309 T12 operation in Center-aligned mode

*Note: Bit CDIR changes with the next timer clock event after the one-match or the period-match. Therefore, the timer continues counting in the previous direction for one cycle before actually changing its direction (see **Figure 309**).*

25.5.2.3 Single-shot mode

In Single-shot mode (bit T12SSC = 1), the timer T12 will stop when the current timer period is finished; the timer run bit T12R is automatically cleared by hardware in this case.

In edge-aligned operation, T12R is cleared when the timer becomes zero after having reached the period value (see **Figure 310**).

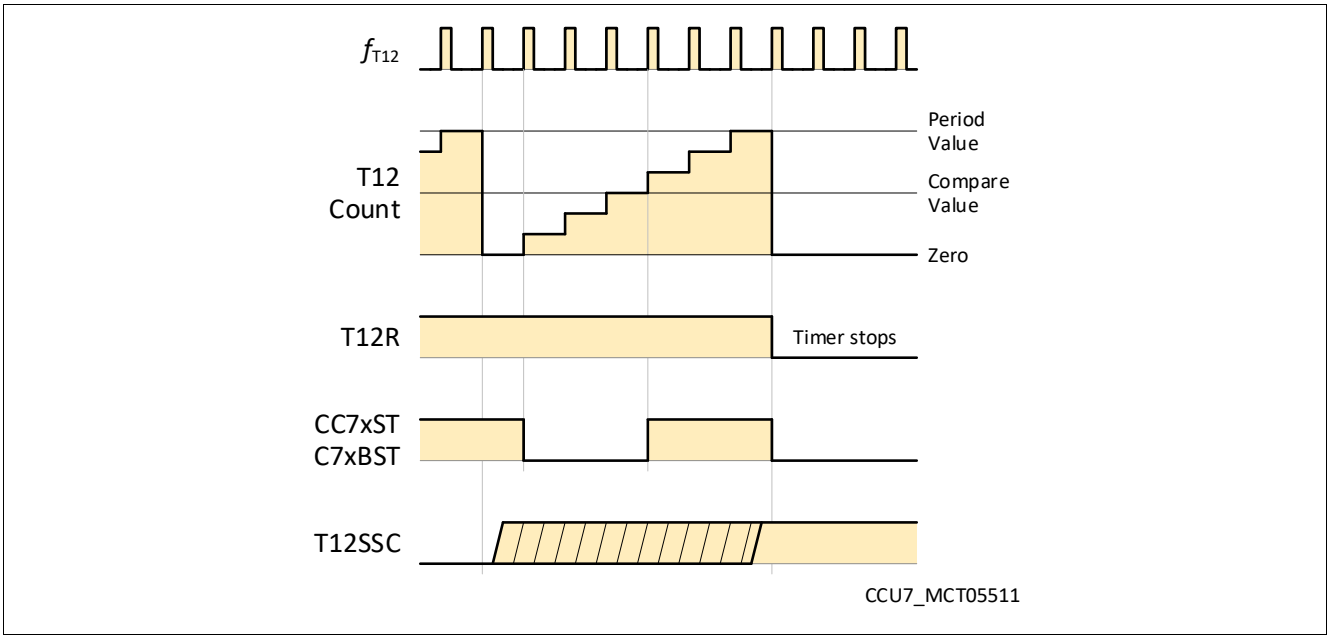


Figure 310 Single-shot operation in Edge-aligned mode

In Center-Aligned operation, the period is finished when the timer has counted down to zero (one clock cycle after the one-match while counting down, see **Figure 311**).

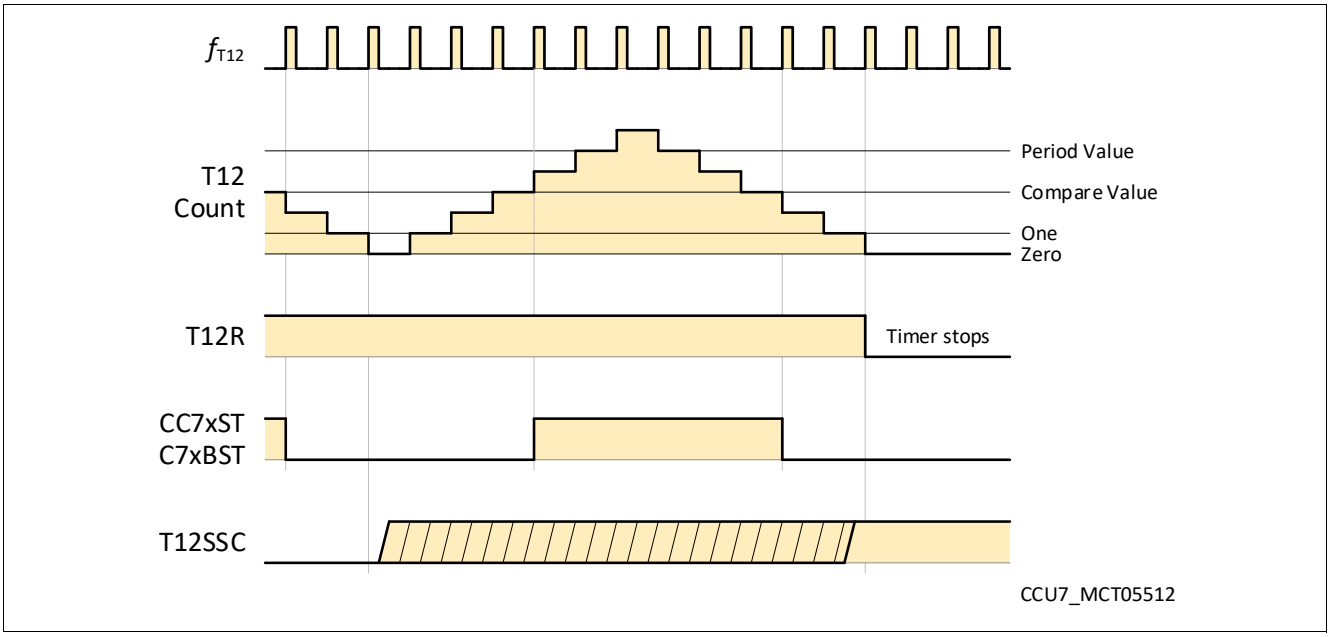


Figure 311 Single-shot operation in Center-aligned mode

25.5.3 T12 compare mode

Associated with timer T12 are six individual compare channels, which can perform compare operations with regard to the contents of the T12 counter.

Three of these channels can also perform capture operations, and they are named CC70, CC71, and CC72 (capture/compare channels CC7x). The capture functions are explained in **Chapter 25.5.4**.

The other three channels can perform compare functions in association with the CC7x channels. They are named C70B, C71B, and C72B (compare channels C7xB, also referred to as B-channels).

25.5.3.1 Compare channels

In Compare mode, the six compare channels CC7x and C7xB can generate complex three-phase PWM patterns. **Figure 312** shows a block diagram of the compare channels and their comparators.

Each compare channel is connected to the T12 counter register via its individual equal-to comparator, generating a match signal when the contents of the counter matches the contents of the associated compare register. Each channel consists of the comparator and a double register structure - the actual compare register CC7xR/C7xBR, feeding the comparator, and an associated shadow register CC7xSR/C7xBSR, that is preloaded by software and transferred into the compare register when the T12 Shadow Transfer signal, T12_ST, gets active.

Providing a shadow register for the compare value as well as for other values related to the generation of the PWM signal facilitates a concurrent update by software for all relevant parameters of a three-phase PWM.

The comparison rules are as follows:

- The compare-match CM_7x is true if the T12 count value is equal to CC7xR
- The compare-match CM_7xB is true if the T12 count value is equal to C7xBR

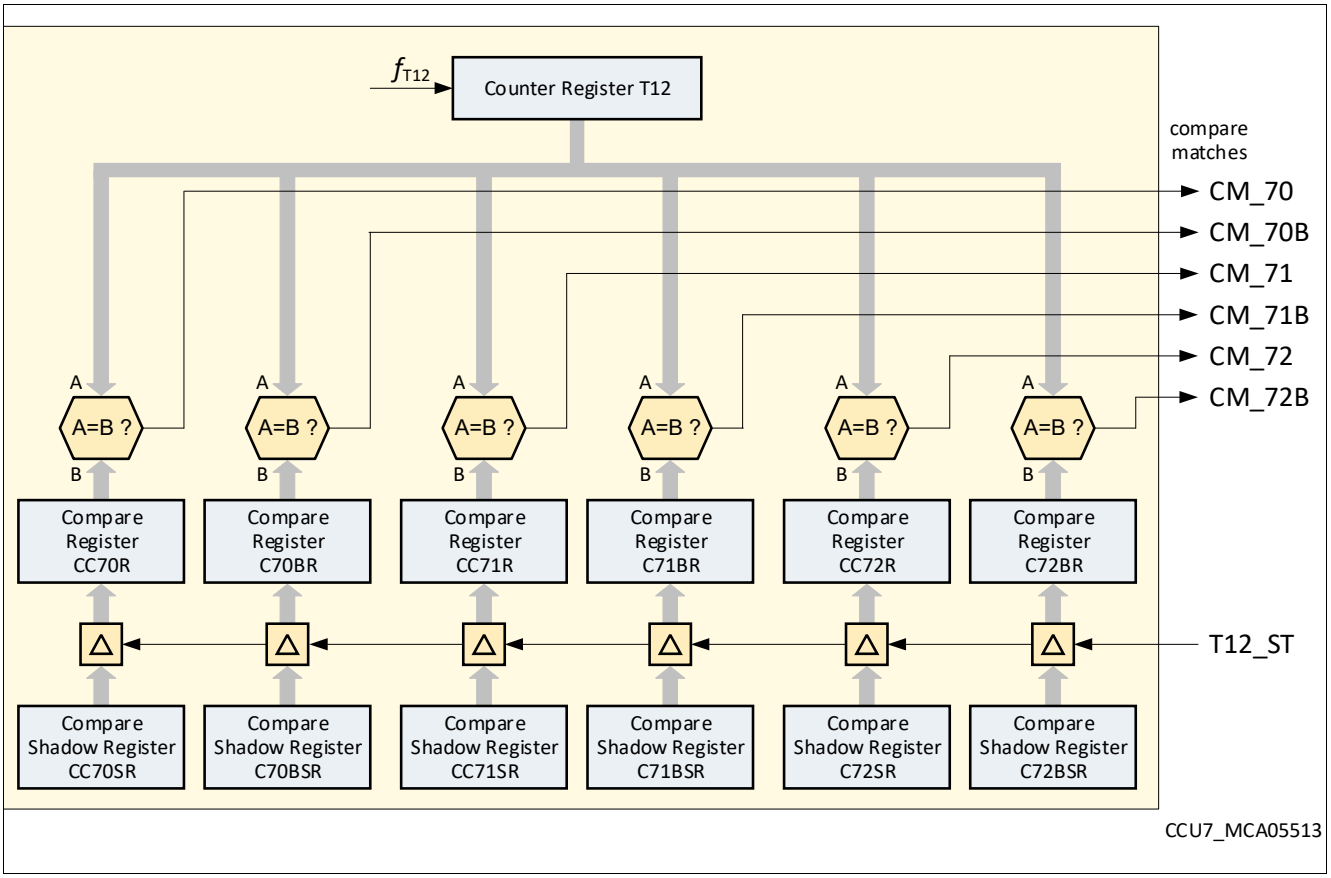


Figure 312 T12 channel comparators

25.5.3.2 Channel state bits

Associated with each (compare) channel is a state bit, **CMPSTAT**.CC7xST, **CMPSTAT_2**.C7xBST holding the status of the compare (or capture) operation (see **Figure 313** and **Figure 314**). In Compare mode, the state bits are modified according to a set of switching rules, depending on the current status of timer T12.

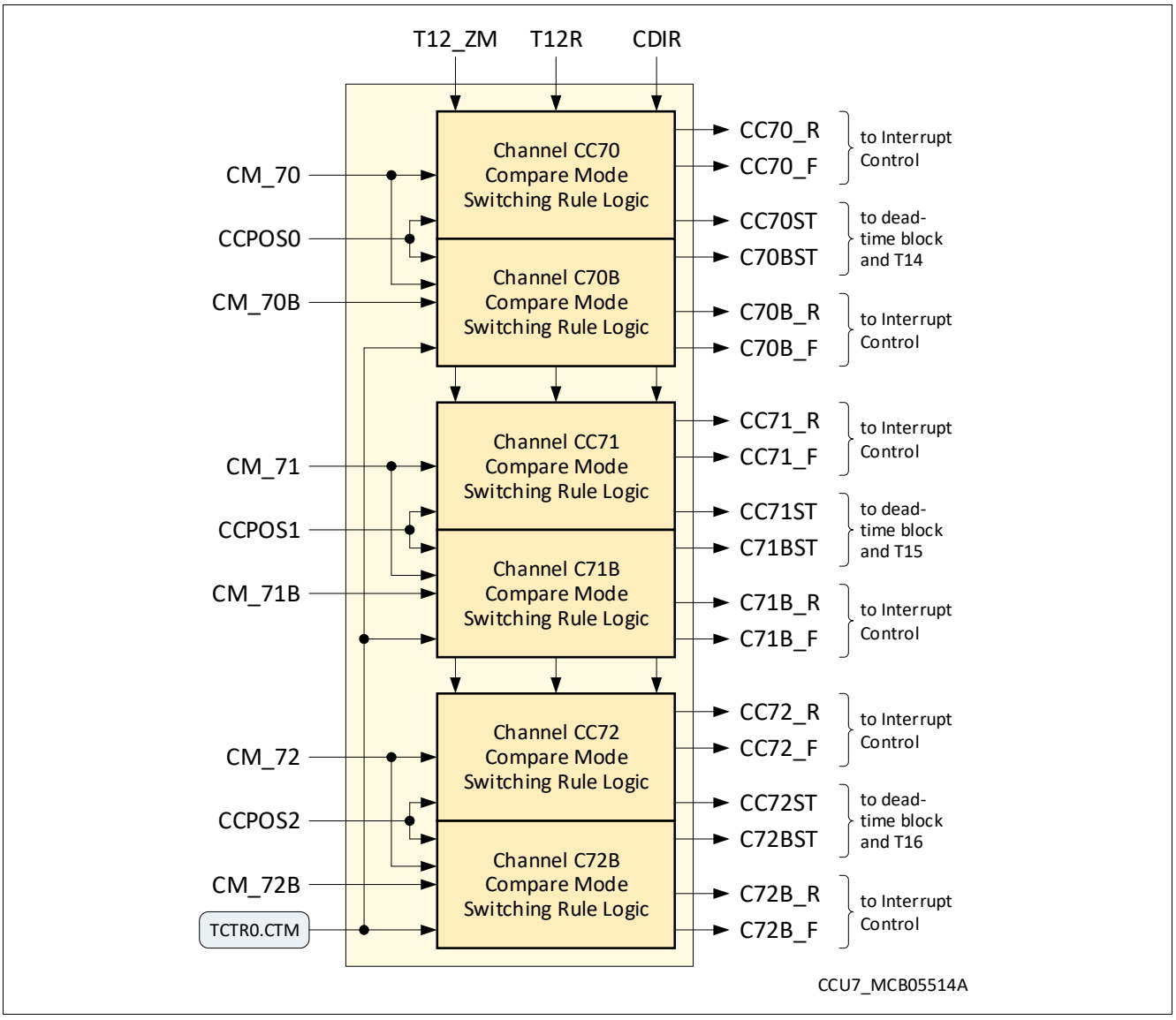


Figure 313 Compare mode state bits overview

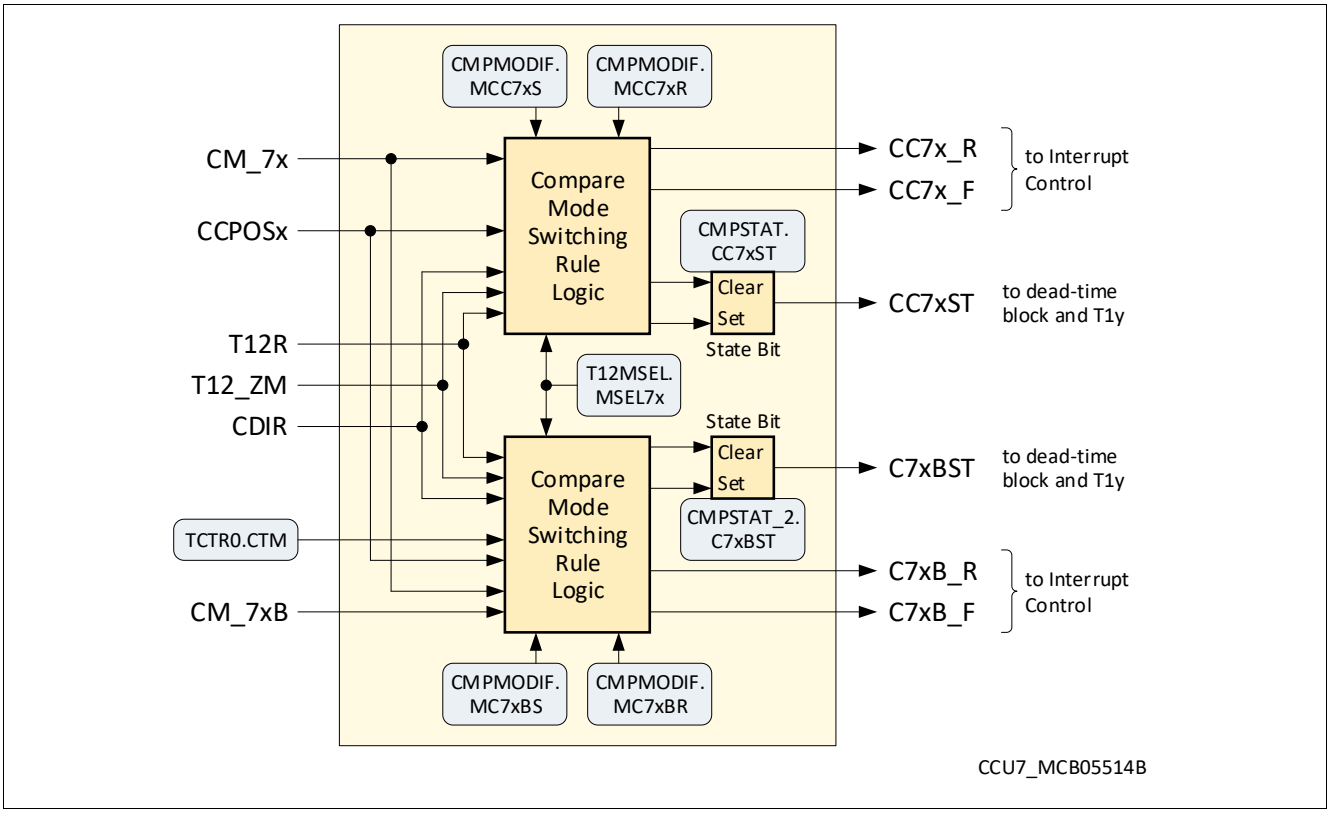


Figure 314 Compare mode state bits details

25.5.3.2.1 Channel state bits CC7xST

The inputs to the switching rule logic for the CC7xST/C7xBST state bits are the timer direction (CDIR), the timer run bit (T12R), the timer T12 zero-match signal (T12_ZM), and the actual individual compare-match signals CM_7x(B) as well as the mode control bits, **T12MSEL**.MSEL7x.

In addition, each state bit can be set or cleared by software via the appropriate set and clear bits in register **CMPMODIF** (MCC7xS and MCC7xR, resp. MC7xBS and MC7xBR).

The input signals CCPOSx are not relevant in normal Compare mode; these inputs are used in hysteresis-like Compare mode (see **Chapter 25.5.3.3**).

A compare interrupt event CC7x_R/C7xB_R is signaled when a compare-match is detected while counting upwards (CDIR = 0), whereas the compare interrupt event CC7x_F/C7xB_F is signaled when a compare-match is detected while counting down (CDIR = 1).

In the standard Compare modes (T12MSEL.MSEL7x = 0001_B, 0010_B, 0011_B), the state bits are set or cleared according to a specific set of rules, detailed in the following (in these modes, the inputs CCPOSx are not taken into account; see also **Chapter 25.5.3.3**).

Please note that, while compare-match signal CM_7xB only influences state bit C7xBST, the compare-match signal CM_7x influences both state bits, CC7xST and C7xBST.

The rules for modification of the state bit CC7xST by hardware are as follows:

- The hardware can only modify the state bit CC7xST while T12 is running, that is, while T12R = 1_B
- State bit CC7xST is set to 1_B with the next T12 clock (f_{T12}) when T12 is counting up (CDIR = 0_B):
  - After a CC7xR compare-match
  - After a T12 zero-match (T12_ZM = 1_B) AND a parallel CC7xR compare-match

- State bit CC7xST is cleared to 0_B with the next T12 clock (f_{T12}):

- After a CC7xR compare-match when T12 is counting down (CDIR = 1_B)
- After a T12 zero-match (T12_ZM = 1_B) AND NO parallel CC7xR compare-match when T12 is counting up (CDIR = 0_B)

For the state bits C7xBST of the B-channels, different rules for modification apply, depending also on the operating mode of timer T12. The rules for modification of the state bit C7xBST by hardware are as follows:

- The hardware can only modify the state bit C7xBST while T12 is running, that is, while T12R = 1_B
- When timer T12 is operating in Edge-aligned mode (T12 is always counting up, CDIR = 0_B):
  - State bit C7xBST is set to 1_B with the next T12 clock (f_{T12}) after a CC7xR compare-match
  - State bit C7xBST is cleared to 0_B with the next T12 clock (f_{T12}) after a C7xBR compare-match
  - State bit C7xBST is cleared to 0_B with the next T12 clock (f_{T12}) after a T12 zero-match and NO parallel CC7xR compare-match
- When timer T12 is operating in Center-aligned mode:
  - State bit C7xBST is set to 1_B with the next T12 clock (f_{T12}) after a CC7xR compare-match when T12 is counting up (CDIR = 0_B)
  - State bit C7xBST is cleared to 0_B with the next T12 clock (f_{T12}) after a C7xBR compare-match when T12 is counting down (CDIR = 1_B)
  - State bit C7xBST is cleared to 0_B with the next T12 clock (f_{T12}) after a T12 zero-match and NO parallel CC7xR compare-match
    (This rule only applies if the compare value in C7xBR is higher than the T12 period value, and if the compare value in CC7xR is > 0)

Figure 315 illustrates the compare operation and state bit setting for the Edge-aligned mode:

- Example a) shows the case in which the value in CC_7xR is less than the one in C7xBR
- Example b) shows the case in which the value in CC_7xR is equal to the one in C7xBR
- Example c) shows the case in which the value in CC_7xR is greater than the one in C7xBR

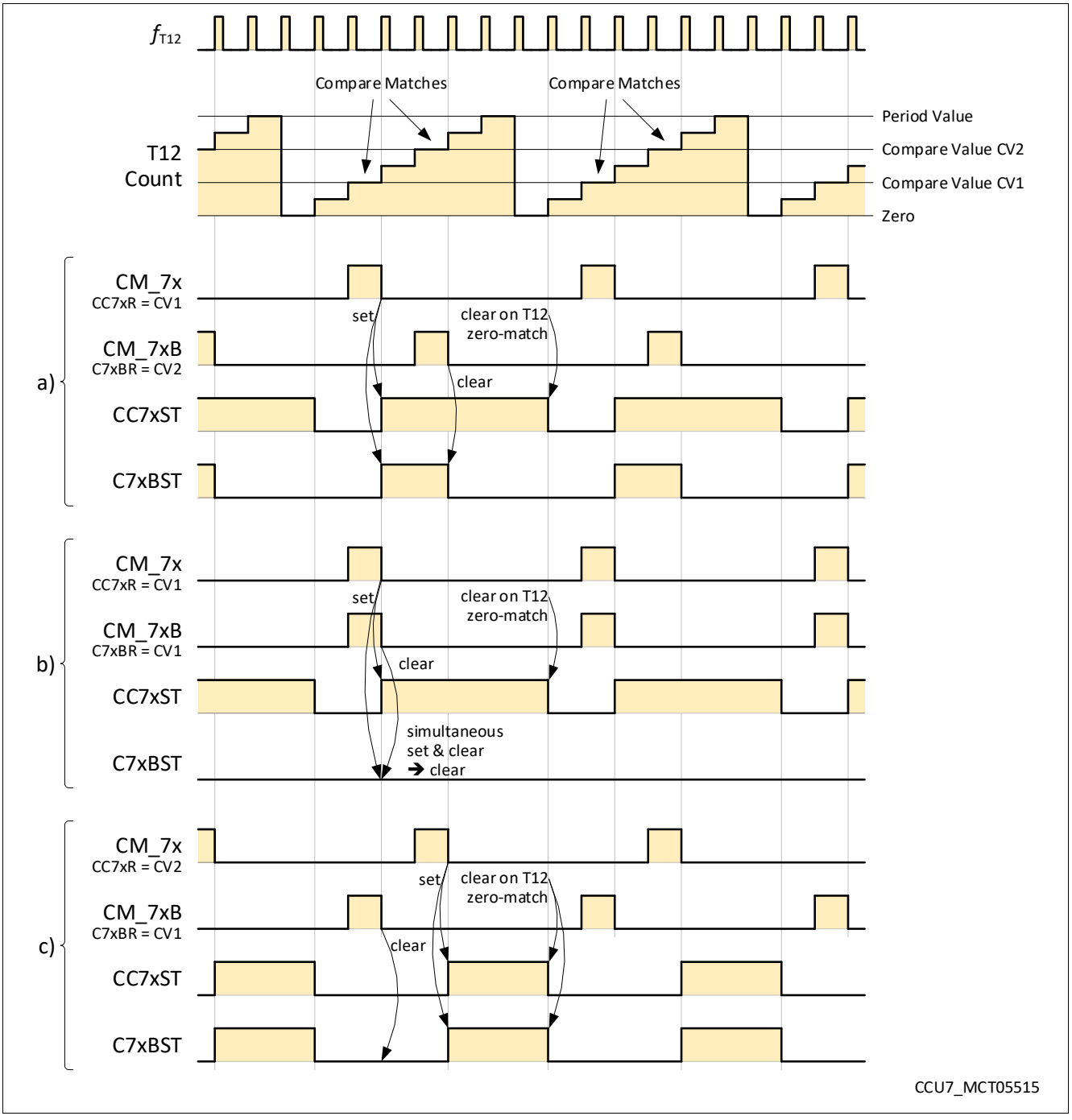


Figure 315 Compare operation, Edge-aligned mode

Figure 316 illustrates the compare operation and state bit setting for the Center-aligned mode:

- Example a) shows the case in which the value in CC_7xR is less than the one in C7xBR
- Example b) shows the case in which the value in CC_7xR is equal to the one in C7xBR
- Example c) shows the case in which the value in CC_7xR is greater than the one in C7xBR

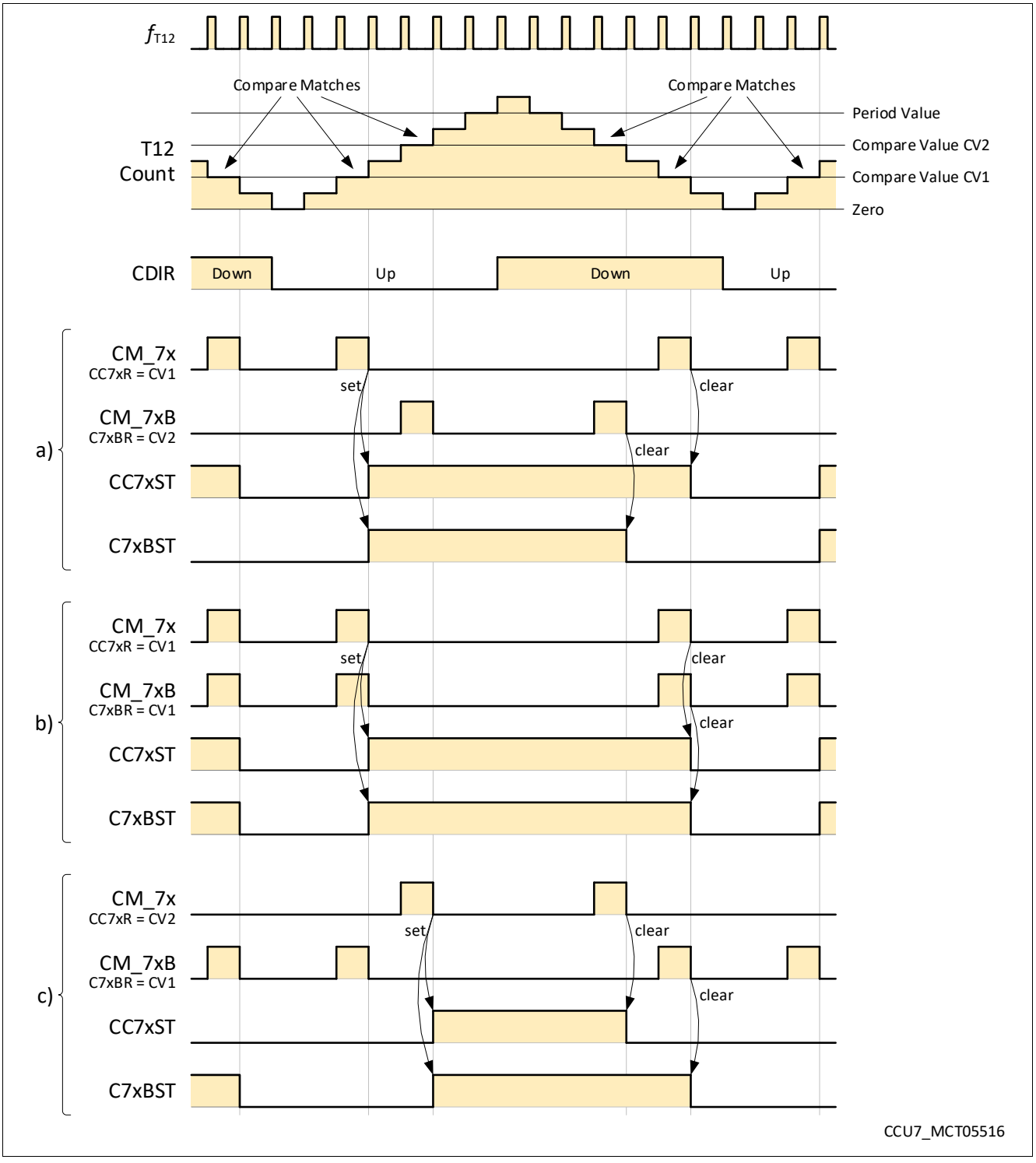


Figure 316 Compare operation, Center-aligned mode

Figure 317 illustrates some more examples for compare waveforms. It is important to note that in these examples, it is assumed that some of the compare values are changed while the timer is running. This change is performed via a software preload of the shadow register, CC7xSR/C7xBSR. The value is transferred to the actual compare register CC7xR/C7xBR with the T12 shadow transfer signal, T12_ST, that is assumed to be enabled.

Some specific examples shown in **Figure 317** are explained in more detail below:

- Example b):
  This example illustrates the transition to a duty cycle of 100% for state bit CC7xST. First, a compare value of 0001_H is used, then changed to 0000_H. Please note that a low pulse with the length of one T12 clock is still produced in the cycle where the new value 0000_H is in effect; this pulse originates from the previous value 0001_H. In the following timer cycles, the state bit CC7xST remains at 1, producing a 100% duty cycle signal. In this case, the compare rule ‘zero-match AND compare-match’ is in effect.
  One can see that the compare-match CM_7x is still active, setting state bit C7xBST each time it occurs.
- Example f):
  This example shows the transition to a duty cycle of 0%. The new compare value CC7xR is set to <Period-Value> + 1, and the state bit CC7ST remains cleared.
  Since no compare-match CM_7x occurs in this case, also the state bit C7xBST is not set to 1.
- Example g):
  In this example, the compare value C7xBR is set to a value higher than the T12 period value. Thus, no compare-match signal CM_7xB will be generated. In this case, the state bit C7xBST is cleared after the T12 zero-match signal (see the third rule for C7xBST in Center-aligned mode).

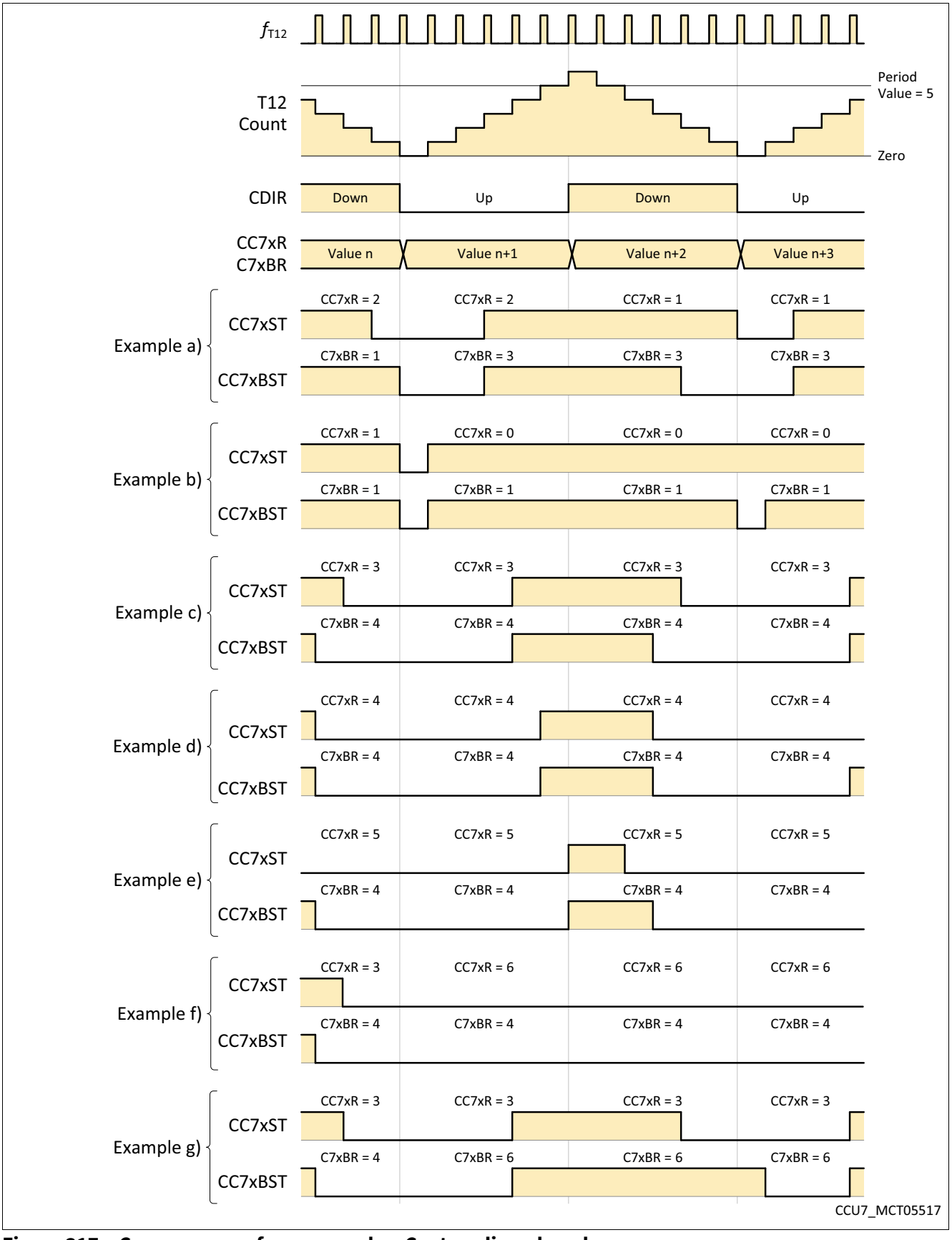


Figure 317 Compare waveform examples, Center-aligned mode

25.5.3.3 Hysteresis-like control mode

The hysteresis-like control mode (**T12MSEL**.MSEL7x = 1001_B) offers the possibility to switch off the PWM output if the input CCPOSx becomes 0.

There is following switching rule:

- CC7xST and C7xBST is reset when the level at CCPOSx input is low (x = 0 to 3)

The CCPOSx inputs are evaluated with f_{PWM}.

This mode can be used to introduce a timing-related behavior to a hysteresis controller. A standard hysteresis controller detects if a value exceeds a limit and switches its output according to the compare result. Depending on the operating conditions, the switching frequency and the duty cycle are not fixed, but change permanently.

See example in **Figure 318**.

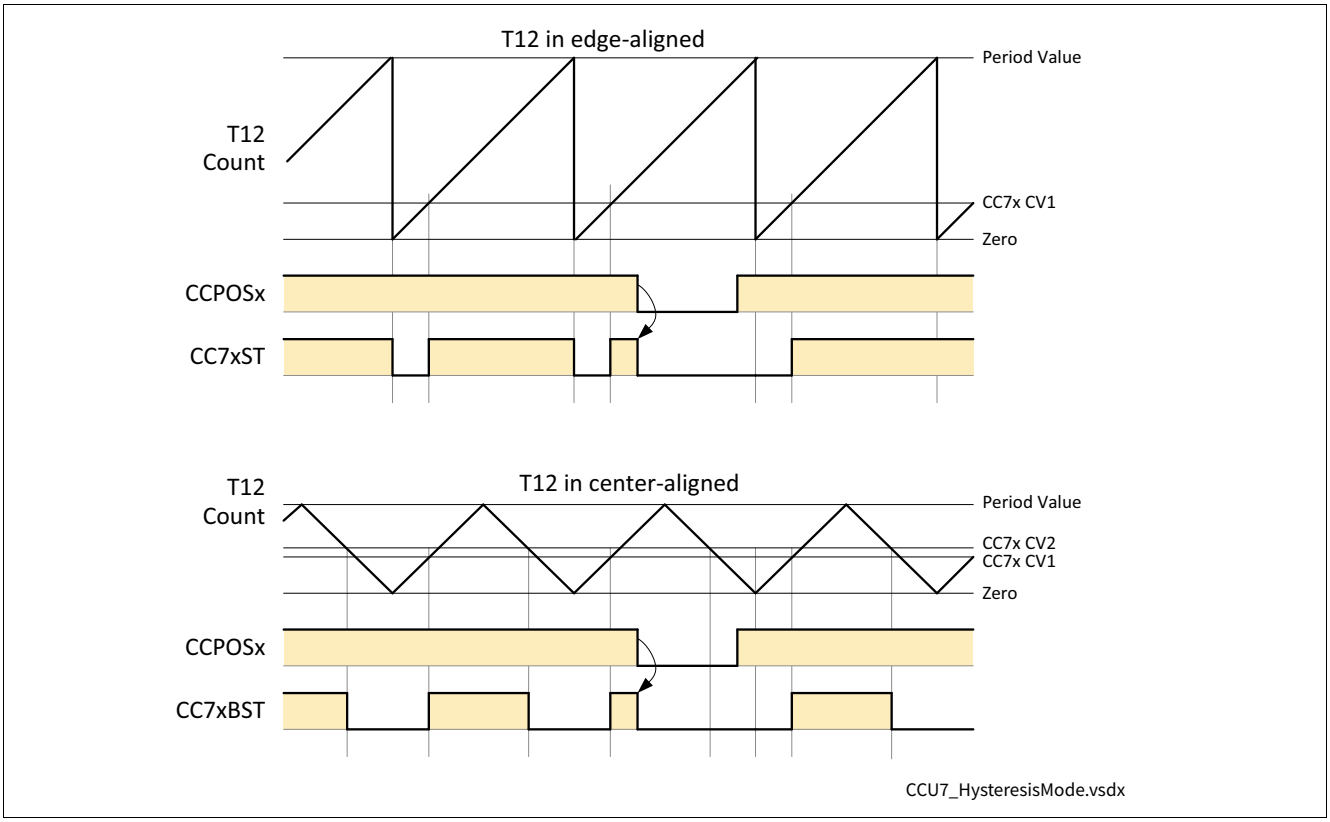


Figure 318 Hysteresis mode waveform example

25.5.4 T12 capture modes

Each of the three channels CC70, CC71, and CC72 of the T12 block can also be used to capture T12 time information in response to an external signal CC7xIN.

In Capture mode, the interrupt event CC7x_R is activated when a rising edge is detected at the input CC7xIN, whereas the interrupt event CC7x_F is activated when a falling edge is detected.

When a selected capture event is detected, the associated state bit CC7xST in register **CMPSTAT** is set to 1. The state bit is not cleared by hardware, but can be cleared by software.

There are a number of different modes for capture operation. In all modes, both of the registers of a channel are used. The selection of the capture modes is done via the **T12MSEL**.MSEL7x bit fields and can be selected individually for each of the channels.

Table 297 Capture modes overview

MSEL7x	Mode	Signal	Active Edge	CC7xSR stored in	T12 stored in
0100_B	1	CC7xIN	Rising	–	CC7xR
		CC7xIN	Falling	–	CC7xSR
0101_B	2	CC7xIN	Rising	CC7xR	CC7xSR
0110_B	3	CC7xIN	Falling	CC7xR	CC7xSR
0111_B	4	CC7xIN	Any	CC7xR	CC7xSR

Capture mode 1

Figure 319 illustrates Capture mode 1. When a rising edge (0-to-1 transition) is detected at the corresponding input signal CC7xIN, the current contents of timer T12 are captured into register CC7xR. When a falling edge (1-to-0 transition) is detected at the input signal CC7xIN, the contents of timer T12 are captured into register CC7xSR.

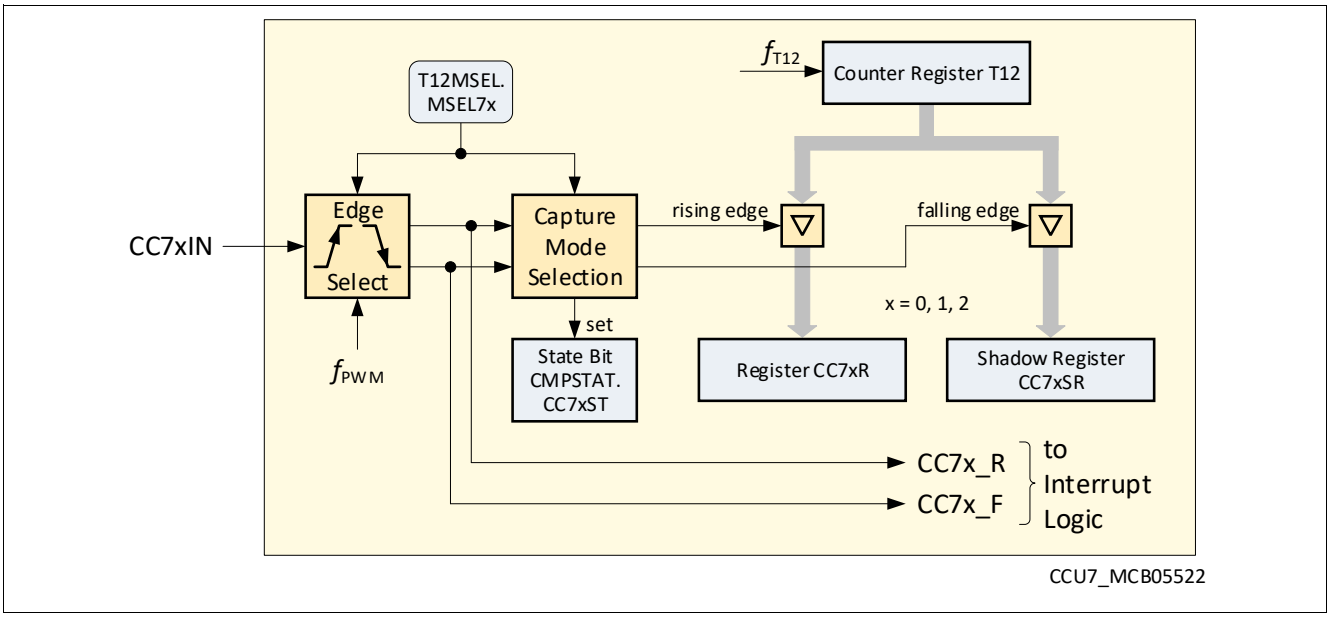


Figure 319 Capture mode 1 block diagram

Capture modes 2, 3 and 4

Capture modes 2, 3 and 4 are shown in **Figure 320**. They differ only in the active edge causing the capture operation. In each of the three modes, when the selected edge is detected at the corresponding input signal CC7xIN, the current contents of the shadow register CC7xSR are transferred into register CC7xR, and the current timer T12 contents are captured in register CC7xSR (simultaneous transfer). The active edge is a rising edge of CC7xIN for Capture mode 2, a falling edge for mode 3, and both, a rising or a falling edge for Capture mode 4, as shown in **Table 297**. These capture modes are very useful in cases where there is little time between two consecutive edges of the input signal.

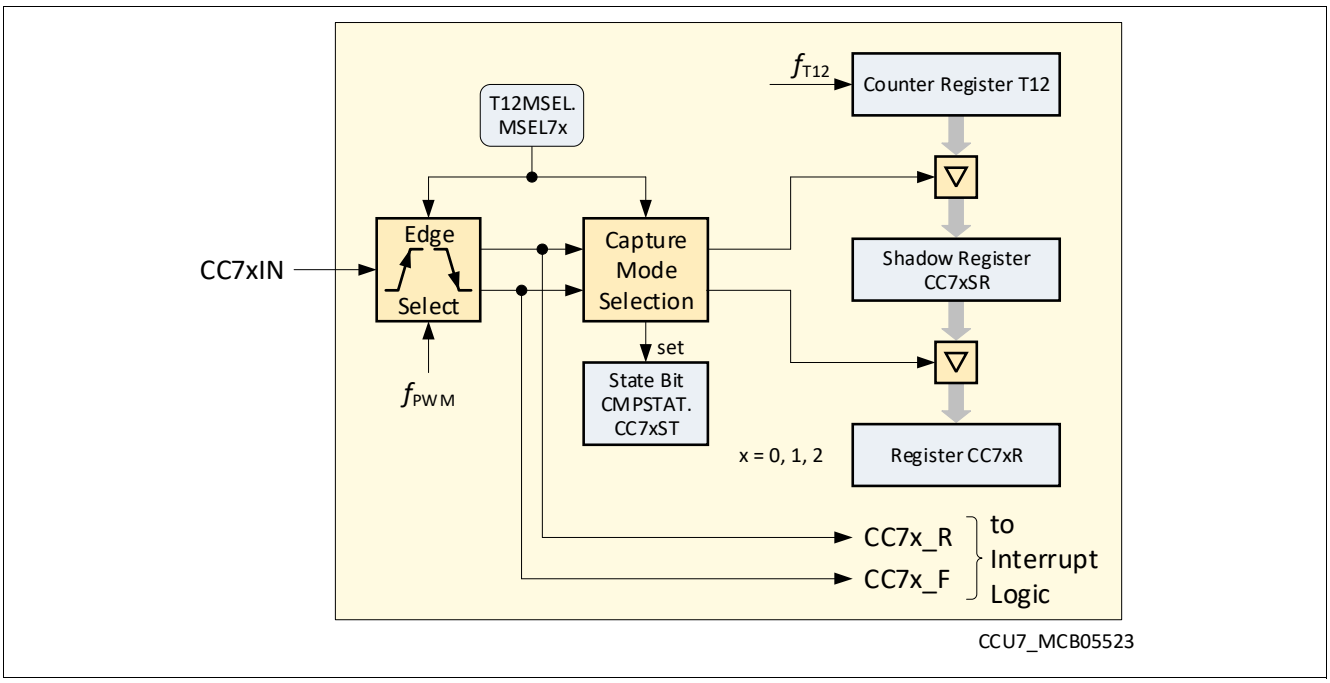


Figure 320 Capture modes 2, 3 and 4 block diagram

Multi-input capture modes

Five further capture modes are called multi-input capture modes, as they use two different external inputs, signal CC7xIN and signal CCPOSx.

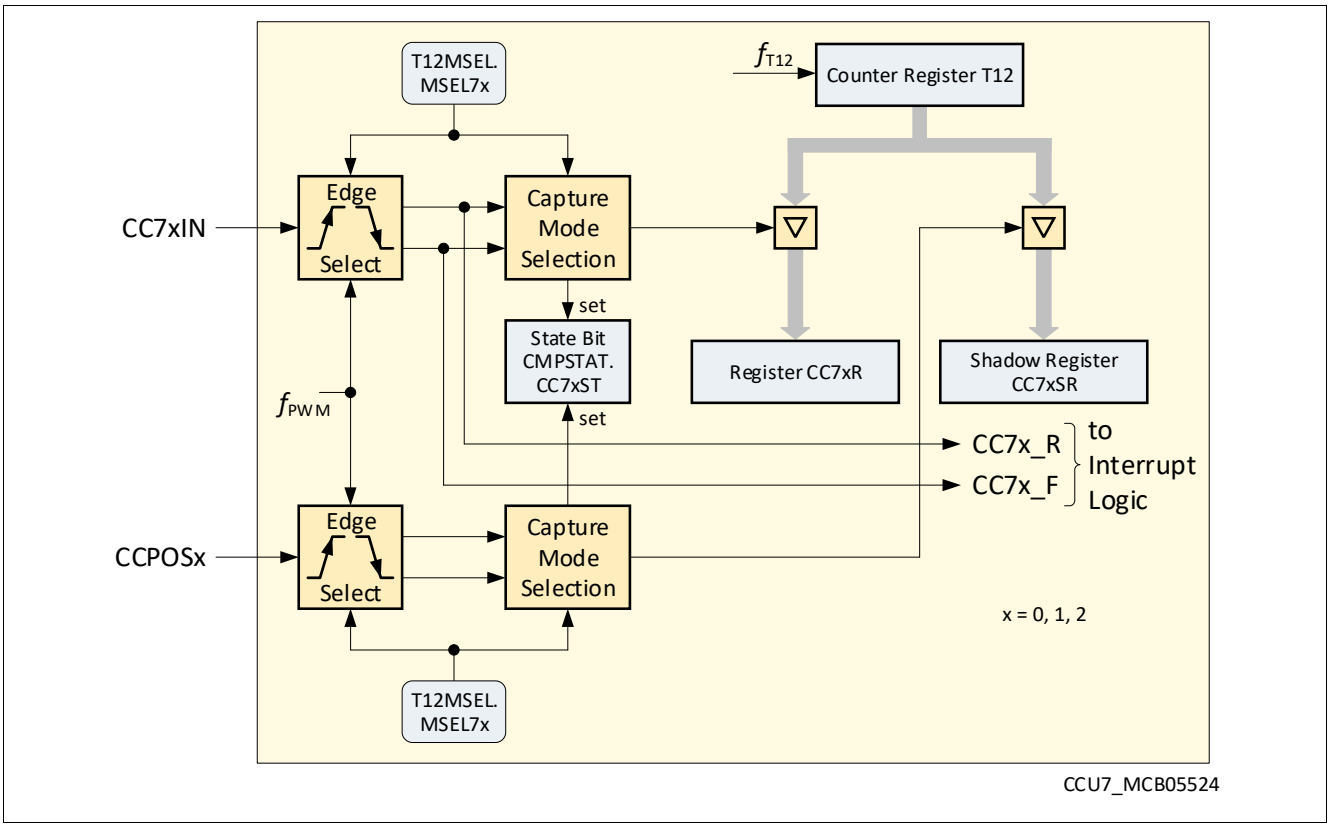


Figure 321 Multi-input capture modes block diagram

In each of these modes, the current T12 contents are captured in register CC7xR in response to a selected event at input signal CC7xIN, and in register CC7xSR in response to a selected event at input signal CCPOSx. The possible events can be opposite input transitions, or the same transitions, or any transition at the two inputs. The different options are detailed in **Table 298**.

In each of the various capture modes, the associated state bit CC7xST in register **CMPSTAT** is set to 1 when the selected capture trigger event at signal CC7xIN or CCPOSx has occurred. The state bit is not cleared by hardware, but can be cleared by software.

In addition, appropriate signal lines to the interrupt logic are activated, that can generate an interrupt request to the CPU. Regardless of the selected active edge, all edges detected at signal CC7xIN can lead to the activation of the appropriate interrupt request line (see also **Chapter 25.3**).

Table 298 Multi-input capture modes overview

MSEL7x	Mode	Signal	Active edge	T12 stored in
1010_B	5	CC7xIN	Rising	CC7xR
		CCPOSx	Falling	CC7xSR
1011_B	6	CC7xIN	Falling	CC7xR
		CCPOSx	Rising	CC7xSR
1100_B	7	CC7xIN	Rising	CC7xR
		CCPOSx	Rising	CC7xSR

Table 298 Multi-input capture modes overview (cont'd)

MSEL7x	Mode	Signal	Active edge	T12 stored in
1101_B	8	CC7xIN	Falling	CC7xR
		CCPOSx	Falling	CC7xSR
1110_B	9	CC7xIN	Any	CC7xR
		CCPOSx	Any	CC7xSR
1111_B	–	Reserved (no capture or compare action)		

25.5.5 T12 shadow transfer control

A special shadow transfer signal (T12_ST) can be generated to facilitate updating the period and compare values of the compare channels CC7x and C7xB synchronously to the operation of T12. Providing a shadow register for values defining one PWM period facilitates a concurrent update by software for all relevant parameters. The next PWM period can run with a new set of parameters.

Figure 322 shows the shadow register structure and the shadow transfer signals, as well as the read/write accessibility of the various registers.

Following registers have a shadow mechanism which is triggered by T12_ST:

- T12 period register T12PR
- T12 compare register CC7xR via CC7xSR
- T12 compare register CCx7xBR via CC7xBSR
- Passive state level register PSLR
- Compare state bits CMPSTAT.CC7xPS/COUT7xPS
- Compare modification bits CMPMODIF_2.CC7xINV/CC7xBINV

Following registers have a shadow mechanism which is triggered by T12_ST or via capture and hall events:

- T12 compare register CC7xR via CC7xSR

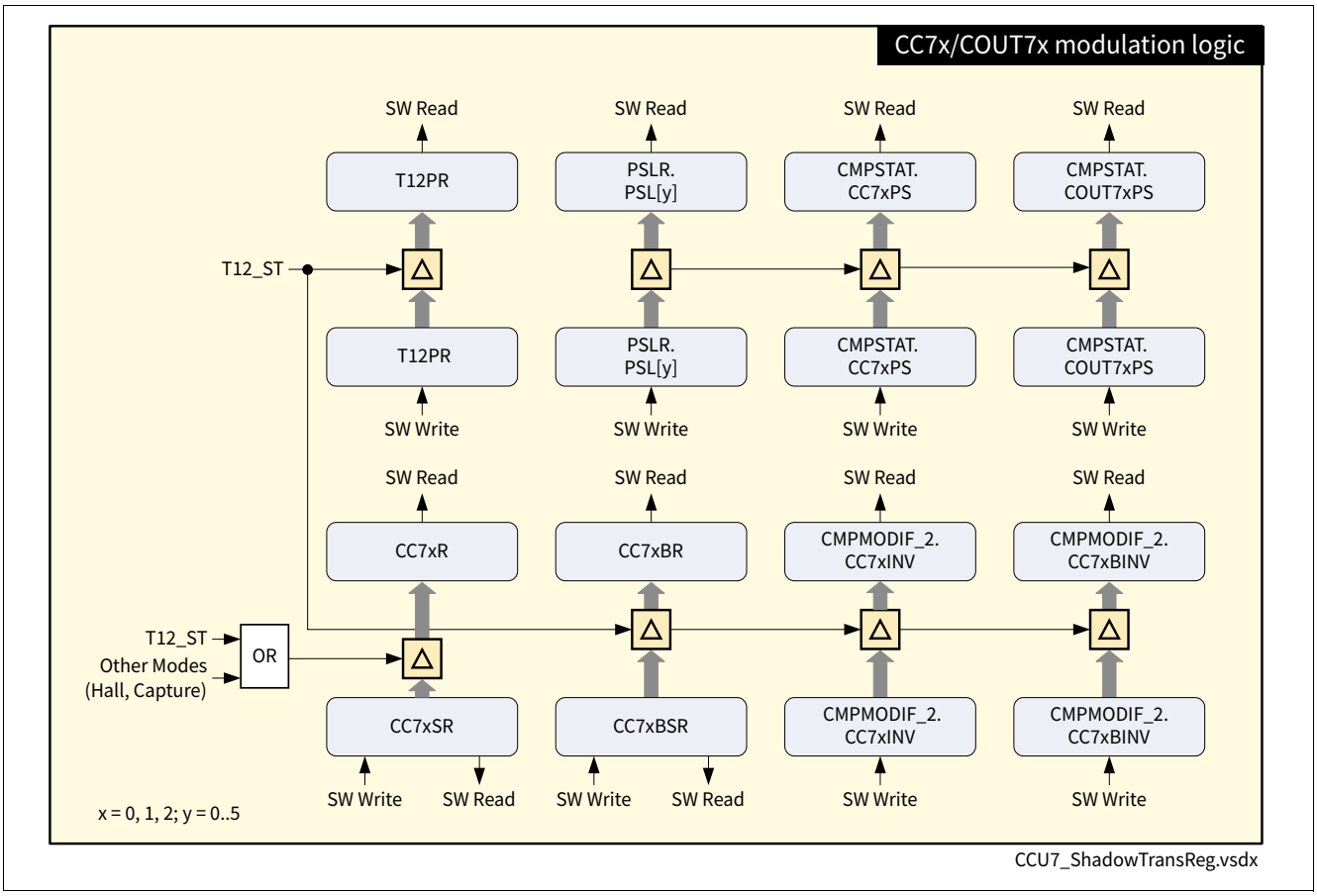


Figure 322 T12 shadow transfer control overview

25.5.5.1 Compare mode shadow transfer

The generation of this signal is enabled by software via bit **TCTR0**.STE12 (set by writing 1 to the write-only bit **TCTR4**.T12STR, cleared by writing 1 to the write-only bit **TCTR4**.T12STD).

A T12 shadow register transfer takes place when T12_ST gets active. There are two options:

Option A: software requests the shadow transfer via setting TCTR4.T12STR

- T12 is stopped (T12R = 0)
  - STE12 = 1
- T12 is running (T12R = 1)
  - STE12 = 1 and a period-match (T12_PM = 1) is detected while counting up, or
  - STE12 = 1 and a one-match (T12_OM = 1) is detected while counting down

Note: STE12 is cleared by hardware once the shadow transfer has happened.

Option B: software requests the shadow transfer via setting MCMOUTS.STRMCM

- The Multi-channel mode is enabled (MODCTR.MCMEN = 1) and
- The MCM reminder flag MCMOUT.R is set and
  - T12_PM is set while T12 is up-counting and MCMCTR.STE12U is set or
  - T12_OM is set while T12 is down-counting and MCMCTR.STE12D is set

Note: MCMOUT.R is cleared by hardware once the MCM_ST triggered shadow transfer has happened.

Option B offers the possibility to distinguish between a T12_PM and a T12_OM event, hence user can select which event is used for the shadow transfer. Therefore the Multi-channel mode has to be enabled (MCMEN = 1) and the MCM's switching selection and synchronization (MCMCTR.SWSEL/SWSYN) has to be set accordingly.

When signal T12_ST is active, a shadow register transfer is triggered with the next cycle of the T12 clock see also **Figure 308** and **Figure 309**.

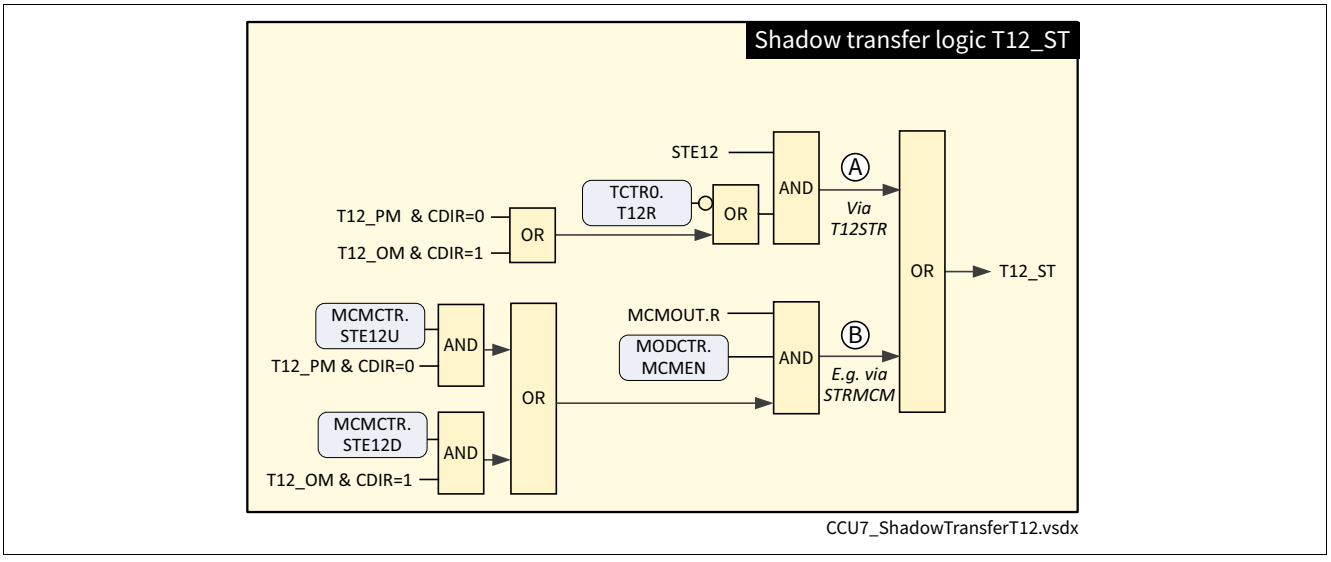


Figure 323 T12 shadow transfer logic

25.5.5.2 Capture and Hall mode shadow transfer

In Capture and Hall mode there are also events which trigger a shadow transfer to T12 compare register CC7xR via CC7xSR.

Following shadow mechanism are available:

- Capture modes: rising/falling edges at CC7xIN and/or CCPOSx inputs trigger shadow transfers to CC7xSR/CC7xR (see **Chapter 25.5.4**)
- Hall sensor mode: event CM_CHE triggers transfers from T12 period register to CC70R, CC71SR to CC70R and CC72SR to CC72R (see **Chapter 25.10.4**)

25.5.6 Timer T12 operating mode selection

The operating mode for the T12 channels are defined by the bit fields **T12MSEL**.MSEL7x.

Table 299 T12 capture/compare modes overview

MSEL7x	Selected operating mode	Available on channels CC7x	Available on channels C7xB
0000_B, 1111_B	Capture/compare modes switched off	x	x
0001_B, 0010_B, 0011_B	Compare mode, see **Chapter 25.5.3** same behavior for all three codings	x	x
$01XX_B$	Double-register capture modes, see **Chapter 25.5.4**	x	–
1000_B	Hall sensor mode, see **Chapter 25.10** In order to properly enable this mode, all three MSEL7x fields have to be programmed to Hall sensor mode.	x	–
1001_B	Hysteresis-like compare mode, see **Chapter 25.5.3.3**	x	x
1010_B 1011_B, 1100_B, 1101_B, 1110_B	Multi-input capture modes, see **Chapter 25.5.4**	x	–

The clocking and counting scheme of the timers are controlled by the timer control registers **TCTR0** and **TCTR2**. Specific actions are triggered by write operations to register **TCTR4**.

25.6 Timer T13

Note: Although the timers T13, T14, T15, and T16 are very similar, the description for these have been split into separate sections, one for timer T13, and one for timers T14, T15, and T16, because of some important differences regarding relevant registers and specific functions.

Timer T13 is implemented similarly to timer T12, but only with one channel in Compare mode. A 16-bit up-counter is connected to a channel register via a comparator, that generates a signal when the counter contents match the contents of the channel register. A variety of control functions facilitate the adaptation of the T13 structure to different application needs. In addition, T13 can be started synchronously to timer T12 events.

This chapter provides information about:

- T13 overview (see **Chapter 25.6.1**)
- Counting scheme (see **Chapter 25.6.2**)
- Compare mode (see **Chapter 25.6.3**))
- Shadow register transfer (see **Chapter 25.6.4**)

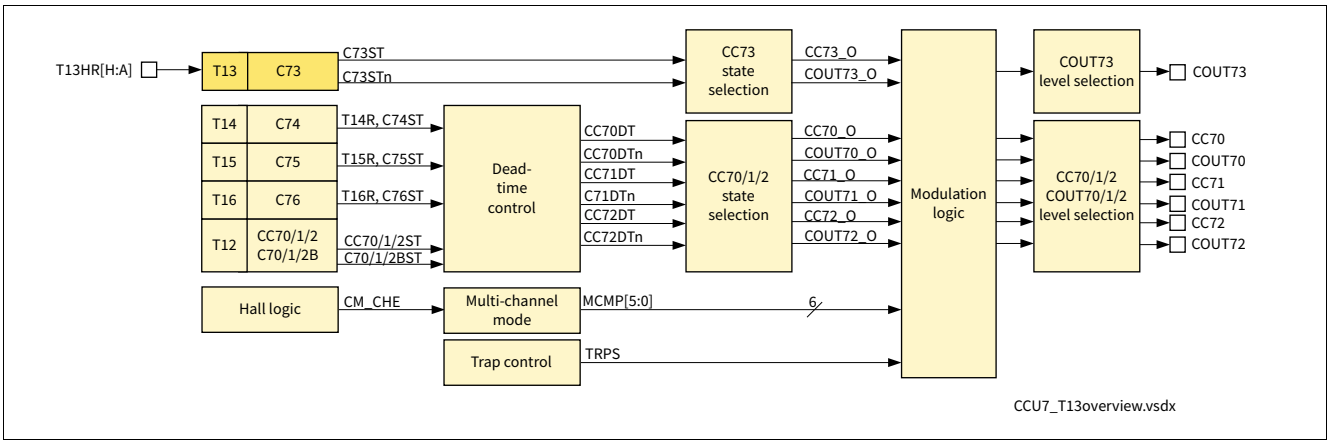


Figure 324 T13 overview

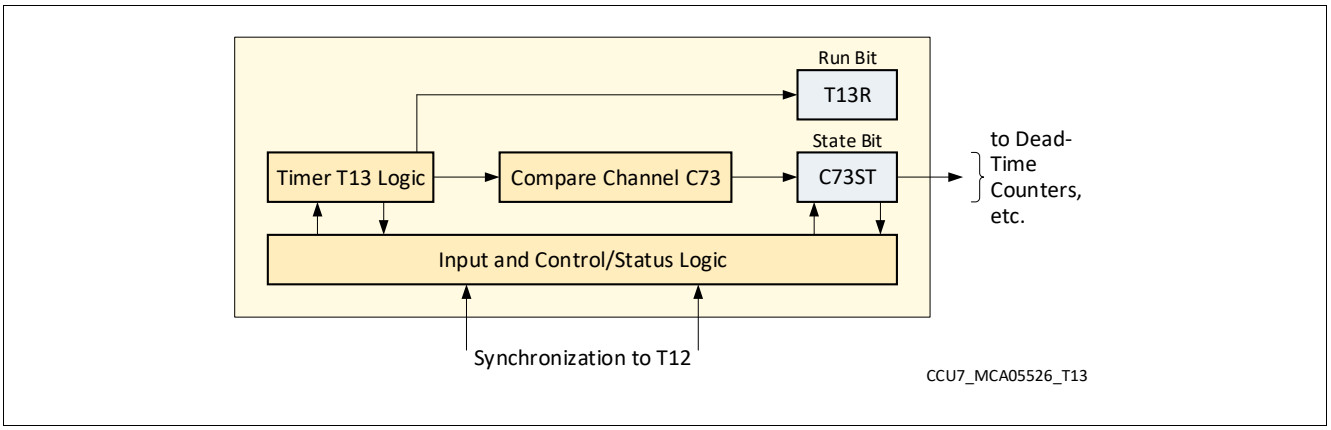


Figure 325 Overview diagram of the timer T13 block

25.6.1 T13 overview

Figure 326 shows a detailed block diagram of timer T13. The functions of the timer are controlled by bits in registers **TCTR1**, **TCTR2**, **TCTR4**, and **PISEL2**.

Timer T13 receives its input clock, f_{T13}, from the module clock f_{PWM} via a programmable prescaler and an optional 1/256 divider or from an input signal T13HR. Timer T13 can only count up (similar to the Edge-aligned mode of T12).

Via a comparator, the timer register T13 is connected to the period register **T13PR** (see **T1xPR (x=3-6)**). This register determines the maximum count value for T13. When T13 reaches the period value, signal T13_PM (T13 period-match) is generated, and T13 is cleared to 0000_H with the next T13 clock edge. The period register receives a new period value from its shadow period register, which is loaded via software. A read access to T13PR delivers the current period value at the comparator, whereas a write access targets the shadow period register to prepare another period value. The transfer of a new period value from the shadow period register into the period register (see **Chapter 25.6.4**) is controlled via the 'T13 Shadow Transfer' control signal, T13_ST. The generation of this signal depends on the operating mode and on the shadow transfer enable bit STE13 in register **TCTR0**.

Providing a shadow register for the period value as well as for other values related to the generation of the PWM signal allows a concurrent update by software for all relevant parameters.

A further signal indicates whether the counter contents are equal to 0000_H (T13_ZM = zero-match).

A Single-Shot control bit, T13SSC, enables an automatic stop of the timer when the current counting period is finished (see **Figure 329**).

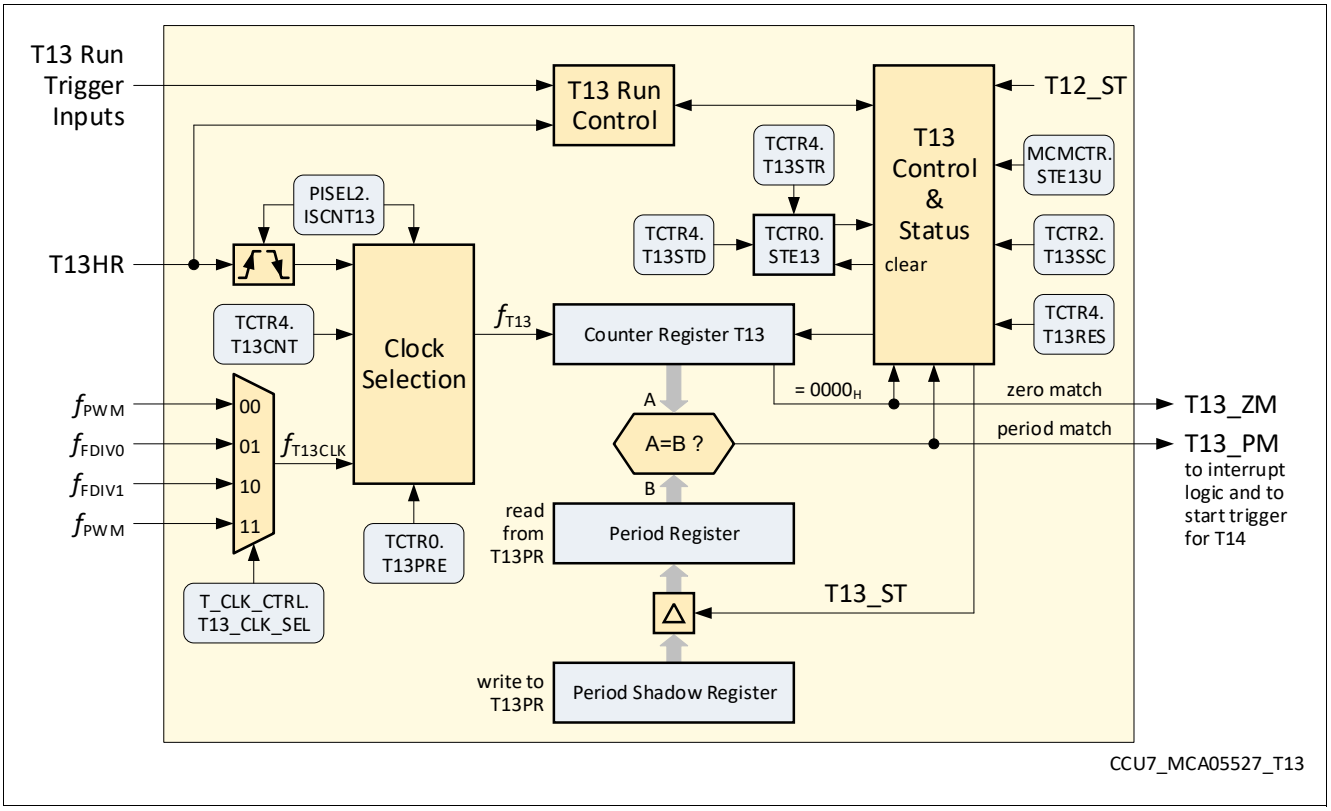


Figure 326 T13 counter logic and period comparator

The start or stop of T13 is controlled by the run bit, T13R. This control bit can be set by software via the associated set/clear bits T13RS or T13RR in register **TCTR4**, or it is cleared by hardware according to preselected conditions (Single-shot mode).

The timer T13 run bit T13R must not be set while the applied T13 period value is zero.

Bit T13R can be set automatically if an event of T12 is detected to synchronize T13 timings to T12 events, e.g. to generate a programmable delay via T13 after an edge of a T12 compare channel before triggering an AD conversion. See **Figure 327** for the T13 run control logic.

Two bit fields, T13TEC and T13TED, control the synchronization of T13 to timer T12 events. T13TEC selects the trigger event, while T13TED determines for which T12 count direction the trigger should be active. See **Table 300** and **Table 301**.

Timer T13 can be cleared to 0000_H via control bit T13RES. Setting this write-only bit only clears the timer contents, but has no further effects, e.g., it does not stop the timer.

The generation of the T13 shadow transfer control signal, T13_ST, is enabled via bit STE13. This bit can be set or cleared by software indirectly through its associated set/reset control bits T13STR and T13STD in register **TCTR4**.

While timer T13 is running, write accesses to the count register T13 are not taken into account. If T13 is stopped, write actions to register T13 are immediately taken into account.

Note: The T13 period register and its associated shadow register are located at the same physical address. A write access to this address targets the shadow register, while a read access reads from the actual period register.

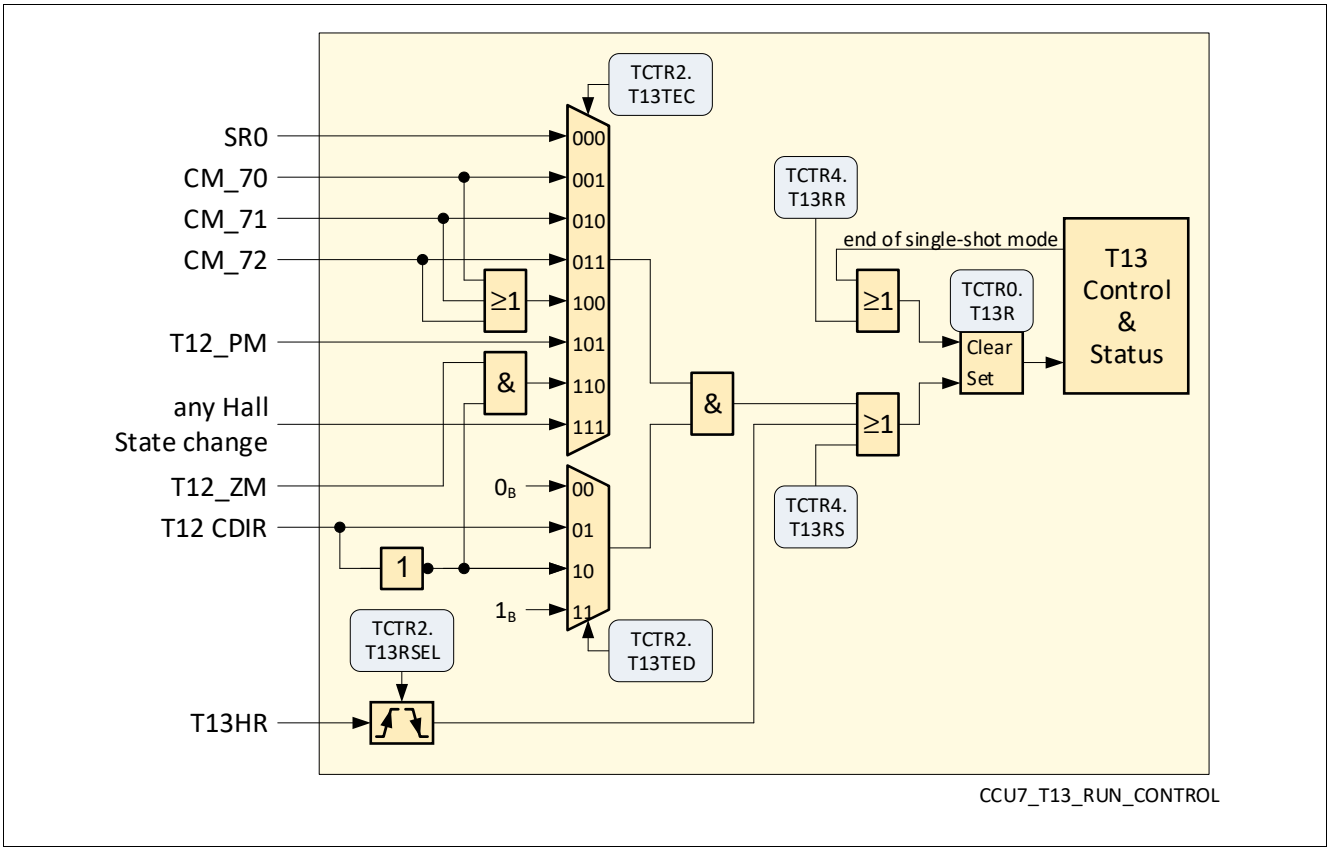


Figure 327 T13 run control

25.6.2 T13 counting scheme

This chapter describes the clocking and the counting capabilities of timer T13.

25.6.2.1 T13 clock selection

Timer mode

In Timer mode (**PISEL2**.ISCNT13 = 00_B), the input clock f_{T13CLK} of timer T13 can be selected with **T_CLK_CTRL**.T13_CLK_SEL. Possible sources are f_{PWM}, f_{FDIV0}, f_{FDIV1}. In addition, a 1/256 prescaler is available, which can be activated via bit T13PRE in register **TCTR0**.

Counter mode

In Counter mode, timer T13 counts one step:

- If a 1 is written to **TCTR4**.T13CNT and **PISEL2**.ISCNT13 = 01_B
- If a rising edge of input signal T13HR is detected and **PISEL2**.ISCNT13 = 10_B
- If a falling edge of input signal T13HR is detected and **PISEL2**.ISCNT13 = 11_B

25.6.2.2 T13 counting

The period of the timer is determined by the value in the period register T13PR according to the following formula:

$T13_{PER}$ = <Period-Value> + 1; in T13 clocks (f_{T13}). (25.4)

Timer T13 can only count up, comparable to the Edge-aligned mode of T12. This leads to very simple ‘counting rule’ for the T13 counter:

- The counter is cleared with the next T13 clock edge if a period-match is detected. The counting direction is always upwards.

The behavior of T13 is illustrated in **Figure 328**.

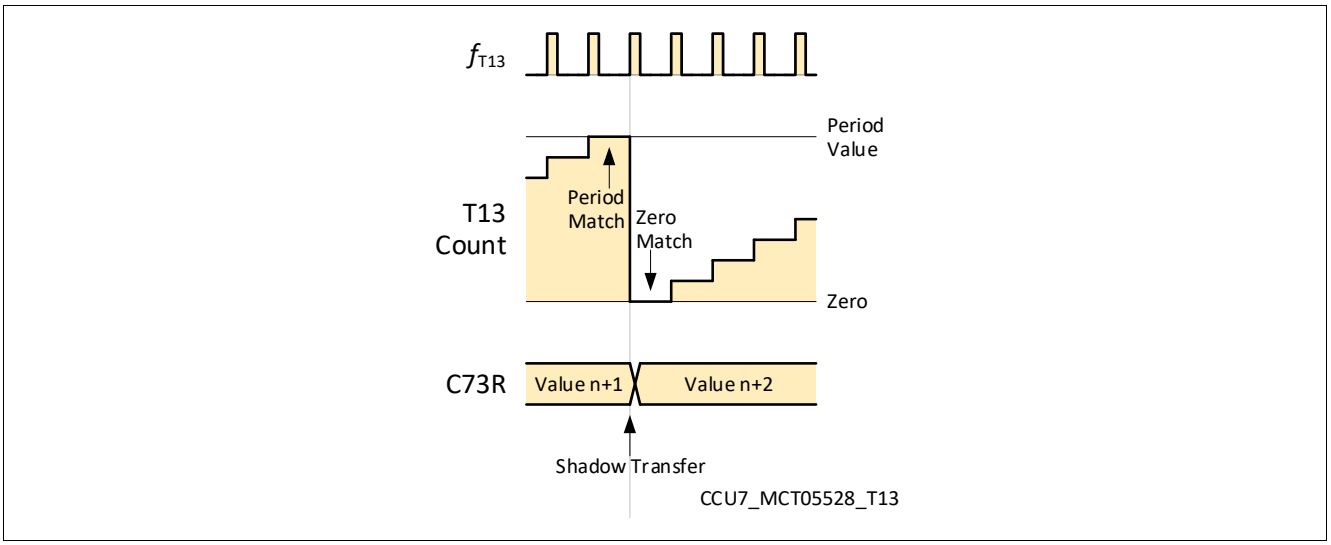


Figure 328 T13 counting sequence

25.6.2.3 Single-shot mode

In Single-shot mode, the timer run bit T13R is cleared by hardware. If bit T13SSC = 1, the timer T13 will stop when the current timer period is finished.

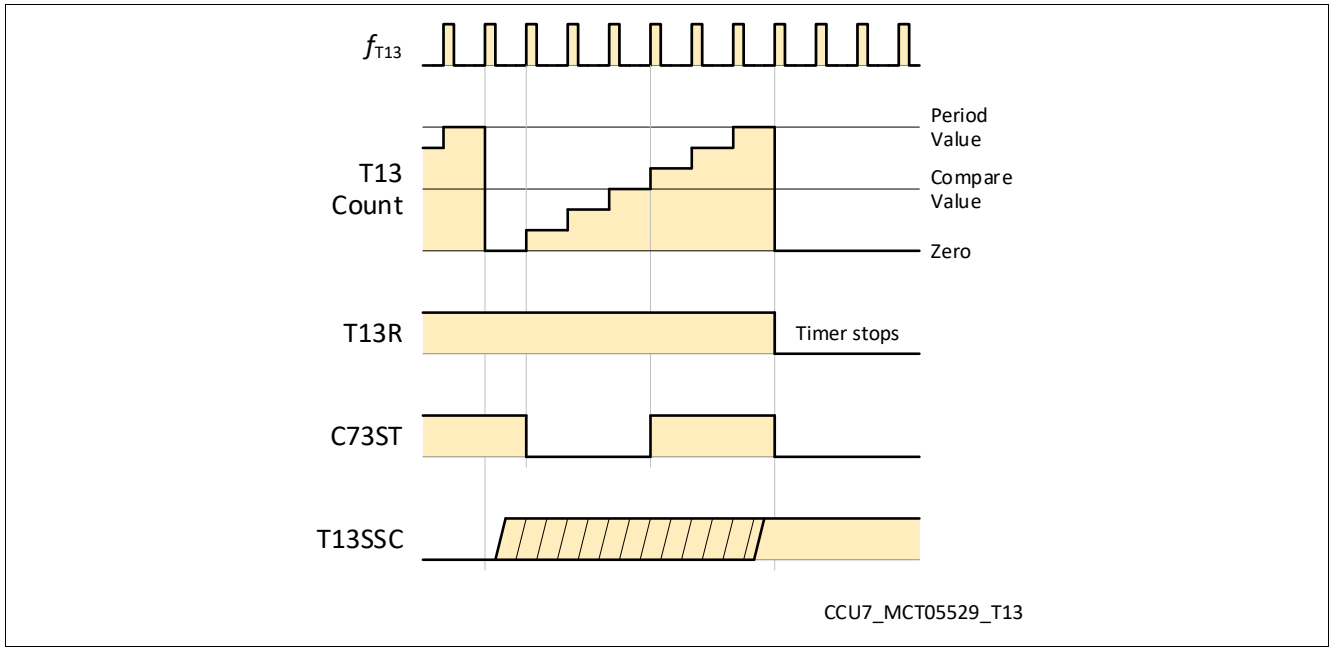


Figure 329 Single-shot operation of timer T13

25.6.2.4 Synchronization to T12

Timer T13 can be synchronized to a T12 event. Bit fields T13TEC and T13TED select the event that is used to start timer T13. The selected event sets bit T13R via HW, and T13 starts counting. Combined with the Single-shot mode, this feature can be used to generate a programmable delay after a T12 event.

Figure 330 shows an example for the synchronization of T13 to a T12 event. Here, the selected event is a compare-match (compare value = 2) while counting up. The clocks of T12 and T13 can be different (other prescaler factor); the figure shows an example in which T13 is clocked with half the frequency of T12.

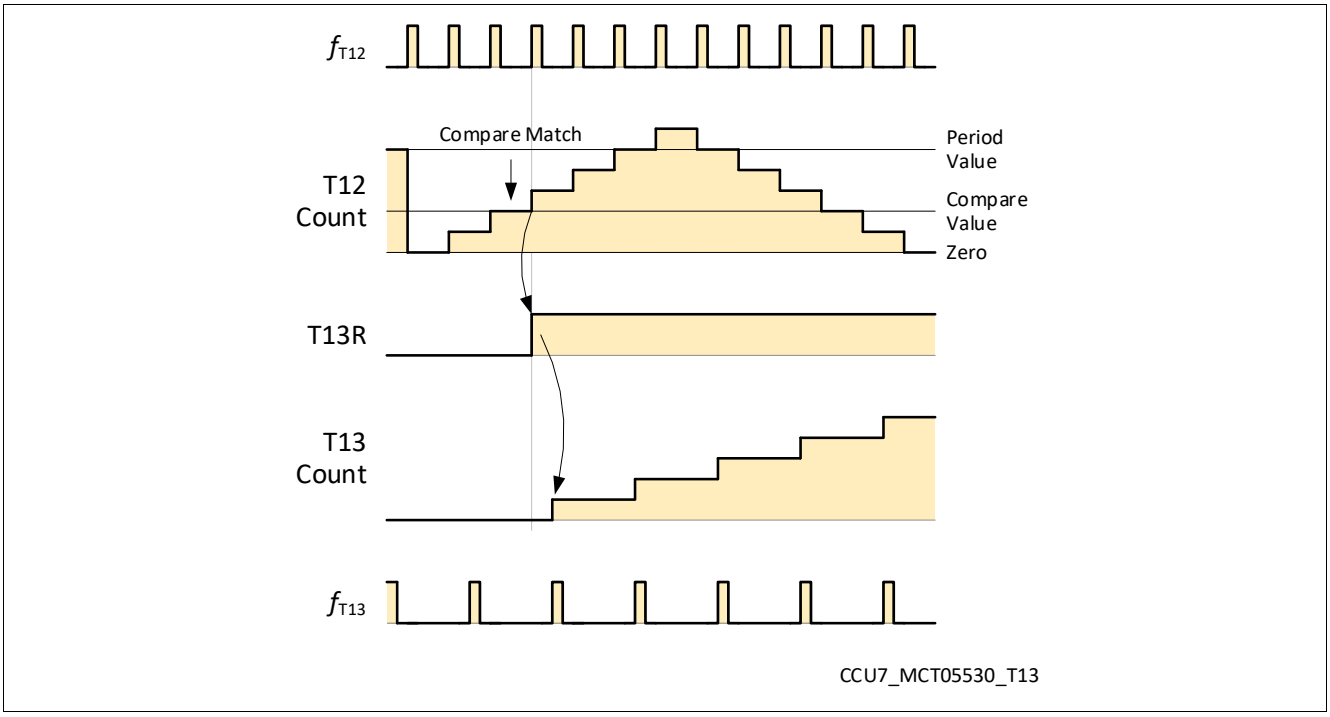


Figure 330 Synchronization of T13 to T12 compare-match

Bit field T13TEC selects the trigger event to start T13 (automatic set of T13R for synchronization to T12 compare signals) according to the combinations shown in **Table 300**. Bit field T13TED additionally specifies for which count direction of T12 the selected trigger event should be regarded (see **Table 301**).

Table 300 T13 trigger event selection

T13TEC	Timer 13 selected event
000_B	SR0
001_B	T12 CM_70
010_B	T12 CM_71
011_B	T12 CM_72
100_B	Any compare event (T12 CM_7x)
101_B	T12_PM
110_B	T12_ZM AND CDIR = 0
111_B	Any Hall state change

Table 301 T12 trigger event additional specifier

T13TED	Timer T13 selected event specifier
00_B	Reserved, no triggering
01_B	Selected event is active while T12 is counting up (CDIR = 0)
10_B	Selected event is active while T12 is counting down (CDIR = 1)
11_B	Selected event is active independently of the count direction of T12

25.6.3 T13 compare mode

Associated with timer T13 is one compare channel C73, that can perform compare operations with regard to the contents of the T13 counter.

Figure 325 gives an overview on the T13 channel in Compare mode. The channel is connected to the T13 counter register via an equal-to comparator, generating a compare-match signal when the contents of the counter matches the contents of the compare register.

The channel consists of the comparator and a double register structure - the actual compare register, **C73R** (see **C7xR (x=3-6)**), feeding the comparator, and an associated shadow register, **C73SR** (see **C7xSR (x=3-6)**), that is preloaded by software and transferred into the compare register when signal T13 shadow transfer, T13_ST, gets active. Providing a shadow register for the compare value as well as for other values related to the generation of the PWM signal facilitates a concurrent update by software for all relevant parameters.

Associated with the channel is a state bit, **CMPSTAT**.C73ST, holding the status of the compare operation. **Figure 331** gives an overview on the logic for the compare operation and the state bit.

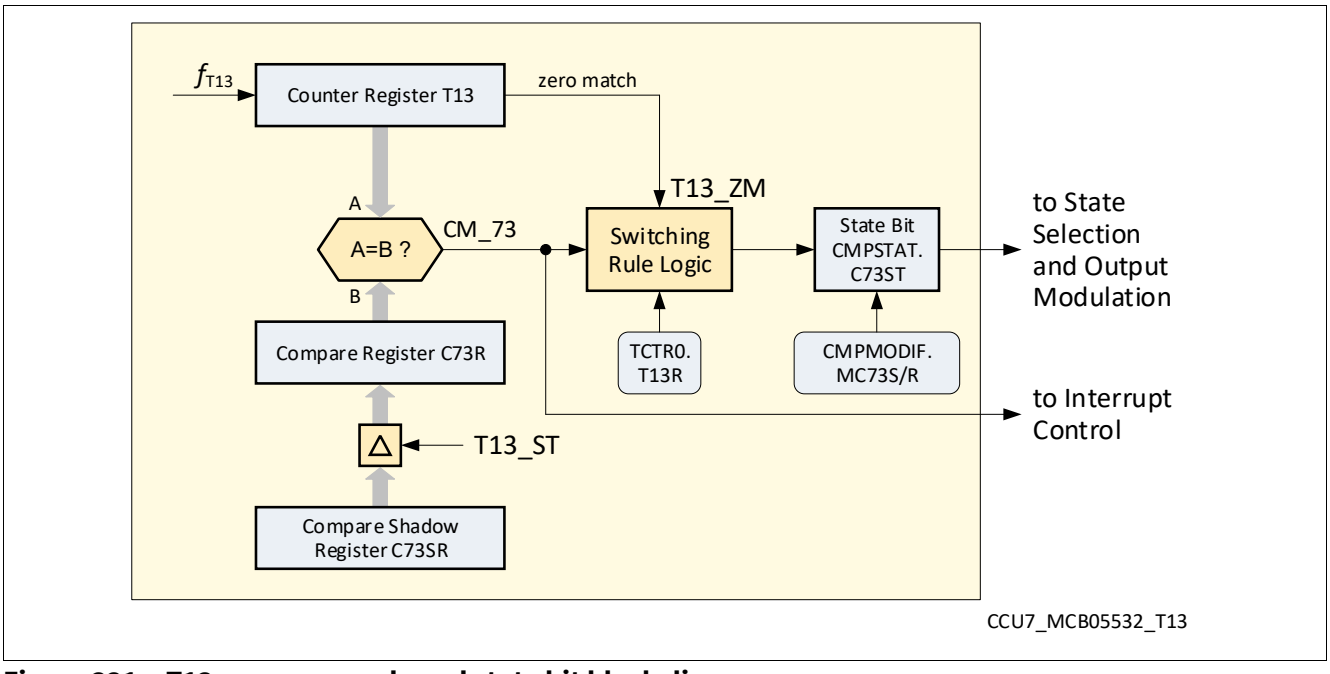


Figure 331 T13 compare mode and state bit block diagram

A compare interrupt event CM_73 is signaled when a compare-match is detected. The actual setting of a state bit has no influence on the interrupt generation.

The inputs to the switching rule logic for the C73ST bit are the timer run bit T13R, the timer zero-match signal T13_ZM, and the actual individual compare-match signal CM_73. In addition, the state bit can be set or cleared by software via bits MC73S and MC73R in register **CMPMODIF**.

An indication of the state bit C73ST by hardware is only possible while timer T13 is running (T13R = 1). If this is the case, the following switching rules apply for setting and resetting the state bit in Compare mode:

- State bit C73ST is set to 1:
  - With the next T13 clock (f_{T13}) after a compare-match (i.e., when the counter is incremented above the compare value)
  - With the next T13 clock (f_{T13}) after a zero-match AND a parallel compare-match
- State bit C73ST is cleared to 0:
  - With the next T13 clock (f_{T13}) after a zero-match AND NO parallel compare-match

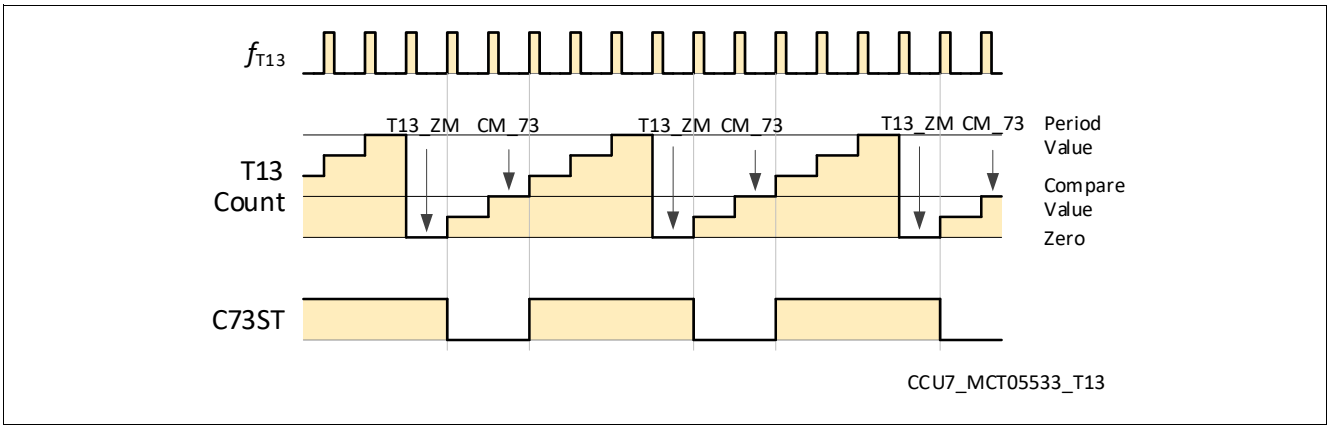


Figure 332 T13 compare operation

25.6.4 T13 shadow transfer

A special shadow transfer signal (T13_ST) can be generated to facilitate updating the period and compare values of the compare channel C73 synchronously to the operation of T13. Providing a shadow register for values defining one PWM period facilitates a concurrent update by software for all relevant parameters. The next PWM period can run with a new set of parameters. The generation of this signal is enabled by software via bit **TCTR0**.STE13 (set by writing 1 to the write-only bit **TCTR4**.T1xSTR, cleared by writing 1 to the write-only bit **TCTR4**.T1xSTD).

When signal T13_ST is active, a shadow register transfer is triggered with the next cycle of the T13 clock. Bit STE13 is automatically cleared with the shadow register transfer.

A T13 shadow register transfer takes place (T13_ST active):

- While timer T13 is not running (T13R = 0), or
- STE13 = 1 and a period-match is detected while T13R = 1

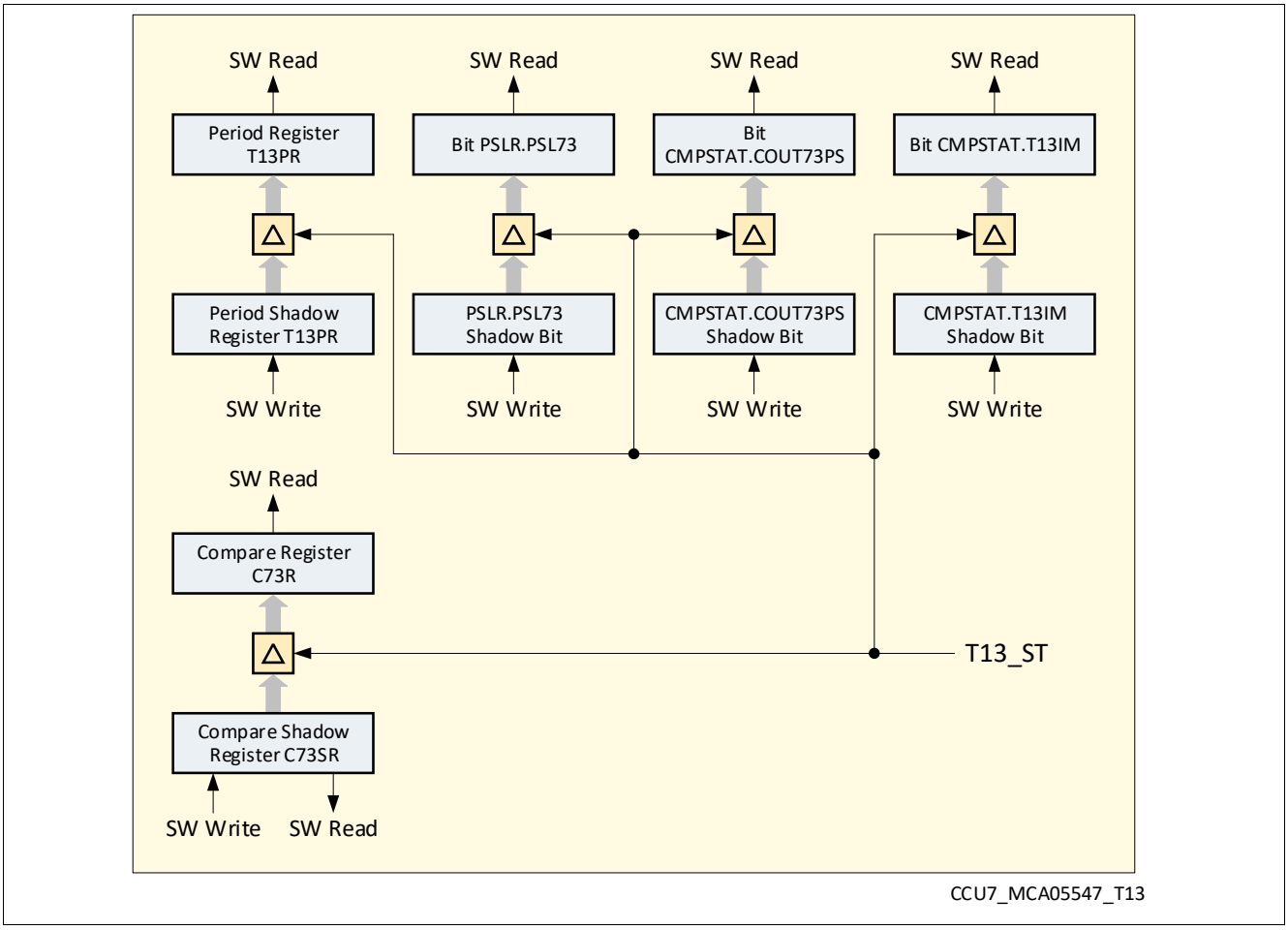


Figure 333 T13 shadow transfer register overview

25.7 Timers T14, T15, T16

Note: Although the timers T13, T14, T15, and T16 are very similar, the description for these have been split into separate sections, one for timer T13, and one for timers T14, T15, and T16, because of some important differences regarding relevant registers and specific functions.

Timers T14, T15, and T16 are implemented similarly to timer T13, with one channel in Compare mode. A 16-bit up-counter is connected to a channel register via a comparator, that generates a signal when the counter contents match the contents of the channel register. A variety of control functions facilitate the adaptation of the timers to different application needs. In addition, the timers can be started synchronously to timer T12 events.

In the following, the timers T14, T15, and T16 are referred to as T1x, with x = 4, 5, or 6.

This chapter provides information about:

- T1x overview (see **Chapter 25.7.1**)
- T1x counting scheme (see **Chapter 25.7.2**)
- T1x compare mode (see **Chapter 25.7.3**)
- T1x shadow transfer (see **Chapter 25.7.4**)

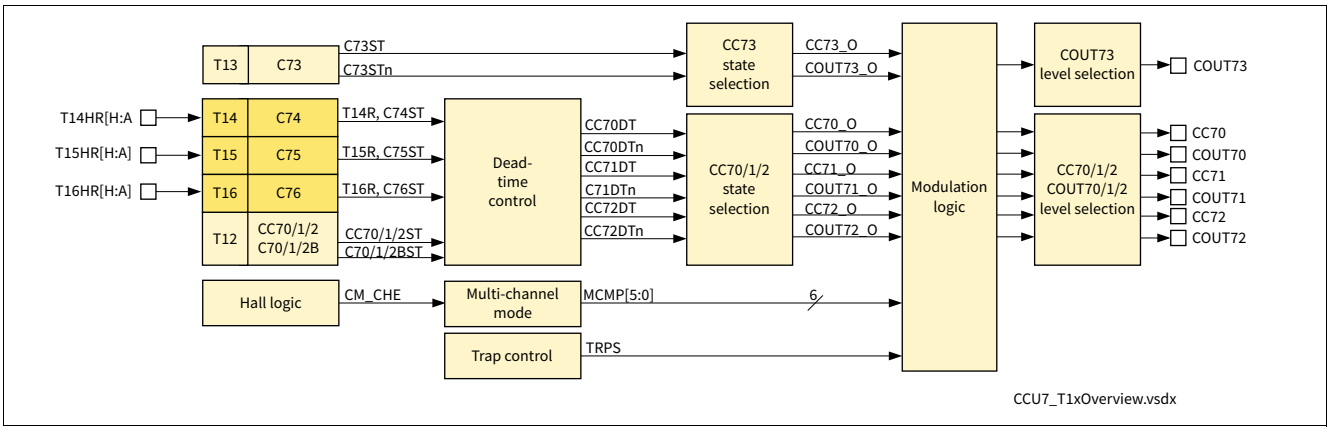


Figure 334 T14, T15, T16 overview

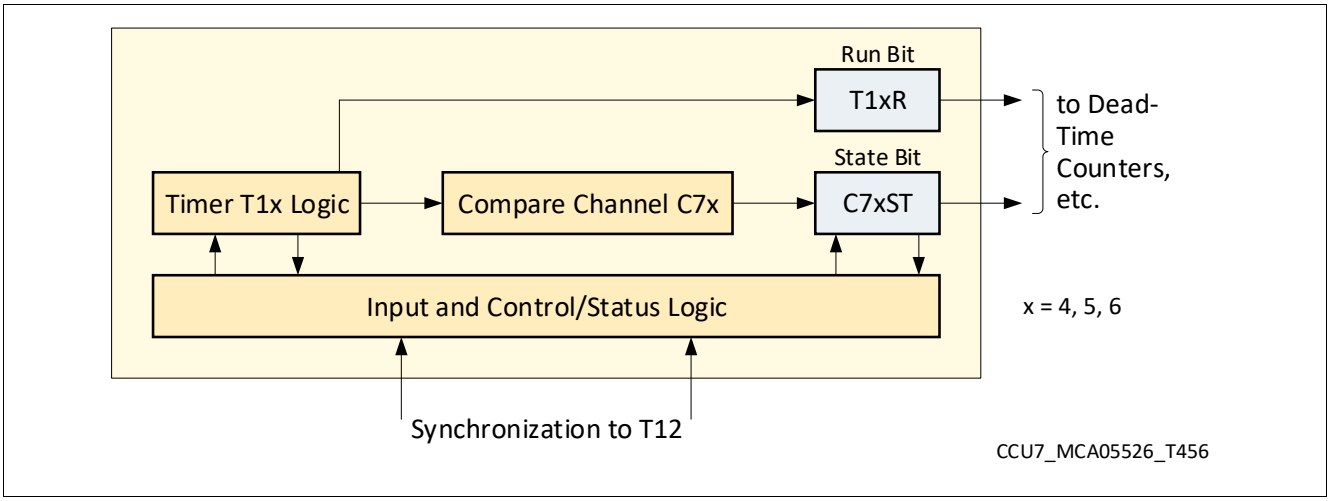


Figure 335 Overview diagram of the timer T1x block

25.7.1 T14, T15, T16 overview

Figure 336 shows a detailed block diagram of a timer T1x. The functions of the timer block are controlled by bits in registers **TCTR1**, and:

- **TCTR2x (x=4-6)** and **PISEL2x (x=4-6)**, x = 4 for timer T14
- **TCTR2x (x=4-6)** and **PISEL2x (x=4-6)**, x = 5 for timer T15
- **TCTR2x (x=4-6)** and **PISEL2x (x=4-6)**, x = 6 for timer T16

Timer T1x receives its input clock, f_{T1x}, from the module clock f_{PWM} via a programmable prescaler and an optional 1/256 divider or from an input signal T1xHR. Timer T1x can only count up (similar to the Edge-aligned mode of T12).

Via a comparator, the timer T1x counter register T1x is connected to the period register T1xPR (see **T1xPR (x=3-6)**). This register determines the maximum count value for T1x. When T1x reaches the period value, signal T1x_PM (T1x period-match) is generated and T1x is cleared to 0000_H with the next T1x clock edge. The period register receives a new period value from its shadow period register, T1xPS, that is loaded via software. The transfer of a new period value from the shadow register into T1xPR is controlled via the 'T1x Shadow Transfer' control signal, T1x_ST. The generation of this signal depends on the associated control bit STE1x.

Providing a shadow register for the period value as well as for other values related to the generation of the PWM signal facilitates a concurrent update by software for all relevant parameters.

Another signal indicates whether the counter contents are equal to 0000_H (T1x_ZM).

A single-shot control bit, T1xSSC, enables an automatic stop of the timer when the current counting period is finished (see **Figure 341**).

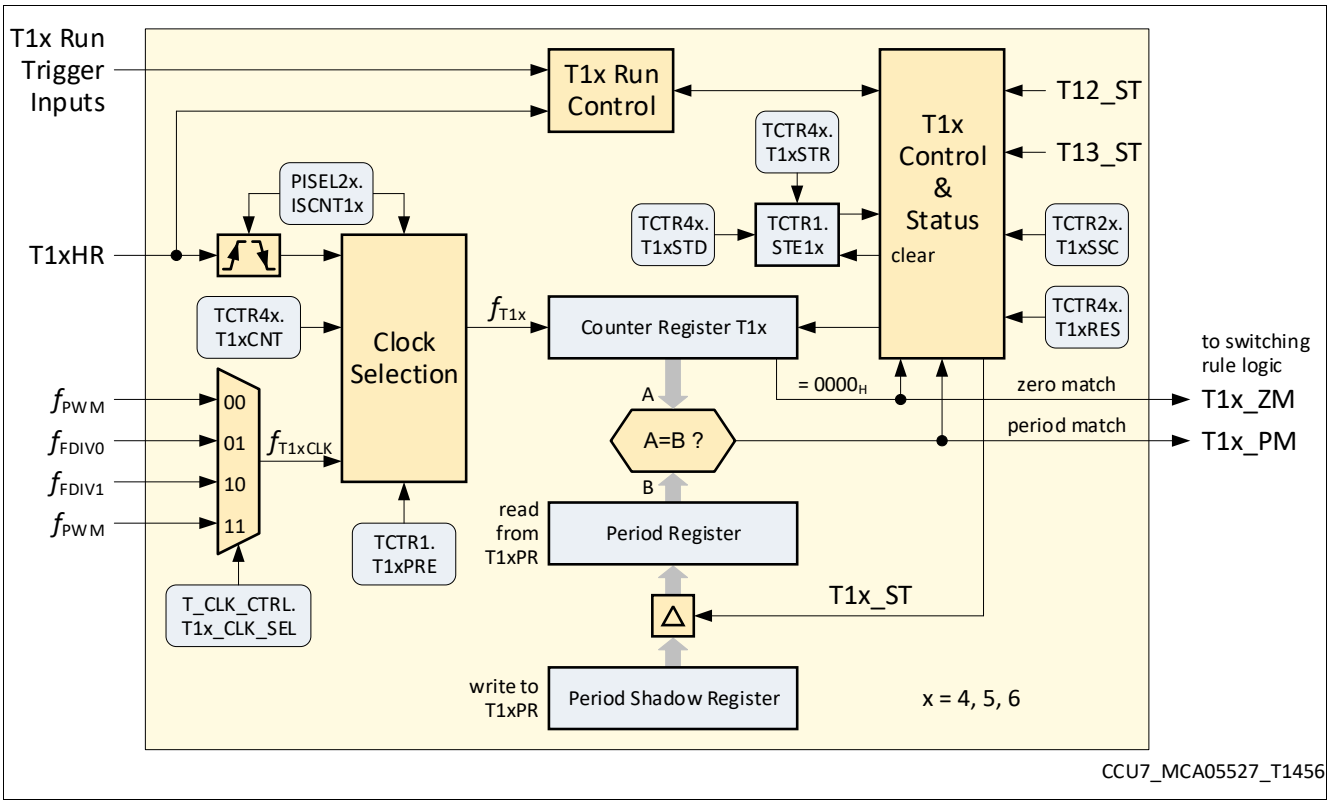


Figure 336 T1x counter logic and period comparator

The start or stop of T1x is controlled by the Run bit, T1xR. This control bit can be set by software via the associated set/clear bits T1xRS or T1xRR in register TCTR4x, or it is cleared by hardware according to preselected conditions (Single-shot mode).

The timer T1x run bit T1xR must not be set while the applied T1x period value is zero.

Bit T1xR can be set automatically if an event of T12 is detected to synchronize T1x timings to T12 events, e.g. to generate a programmable delay via T1x after an edge of a T12 compare channel before triggering an AD conversion. Since the three timers have different input selections for the run bit synchronization, their run control logic is shown in separate figures (see **Figure 337**, **Figure 338**, and **Figure 339**).

Two bit fields, T1xTEC and T1xTED, control the synchronization of T1x to timer T12 events. T1xTEC selects the trigger event, while T1xTED determines for which T12 count direction the trigger should be active. See **Table 302** and **Table 303**.

Timer T1x can be cleared to 0000_H via control bit T1xRES. Setting this write-only bit only clears the timer contents, but has no further effects, e.g., it does not stop the timer.

The generation of the T1x shadow transfer control signal, T1x_ST, is enabled via bit STE1x in register **TCTR1**. This bit can be set or cleared by software indirectly through its associated set/reset control bits T1xSTR and T1xSTD.

While timer T1x is running, write accesses to the count register T1x are not taken into account. If T1x is stopped, write actions to register T1x are immediately taken into account.

Note: *The T1x period register and its associated shadow register are located at the same physical address. A write access to this address targets the shadow register, while a read access reads from the actual period register.*

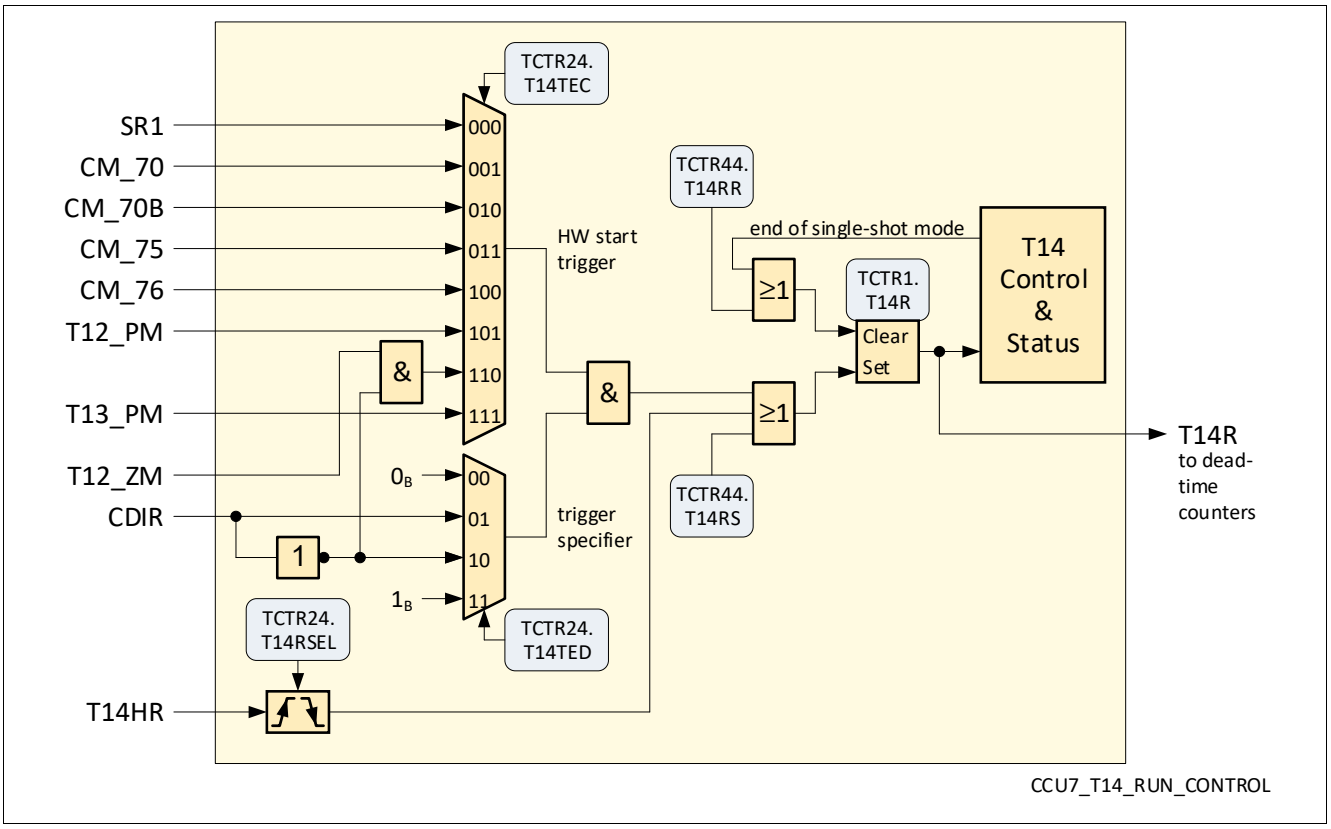


Figure 337 T14 run control

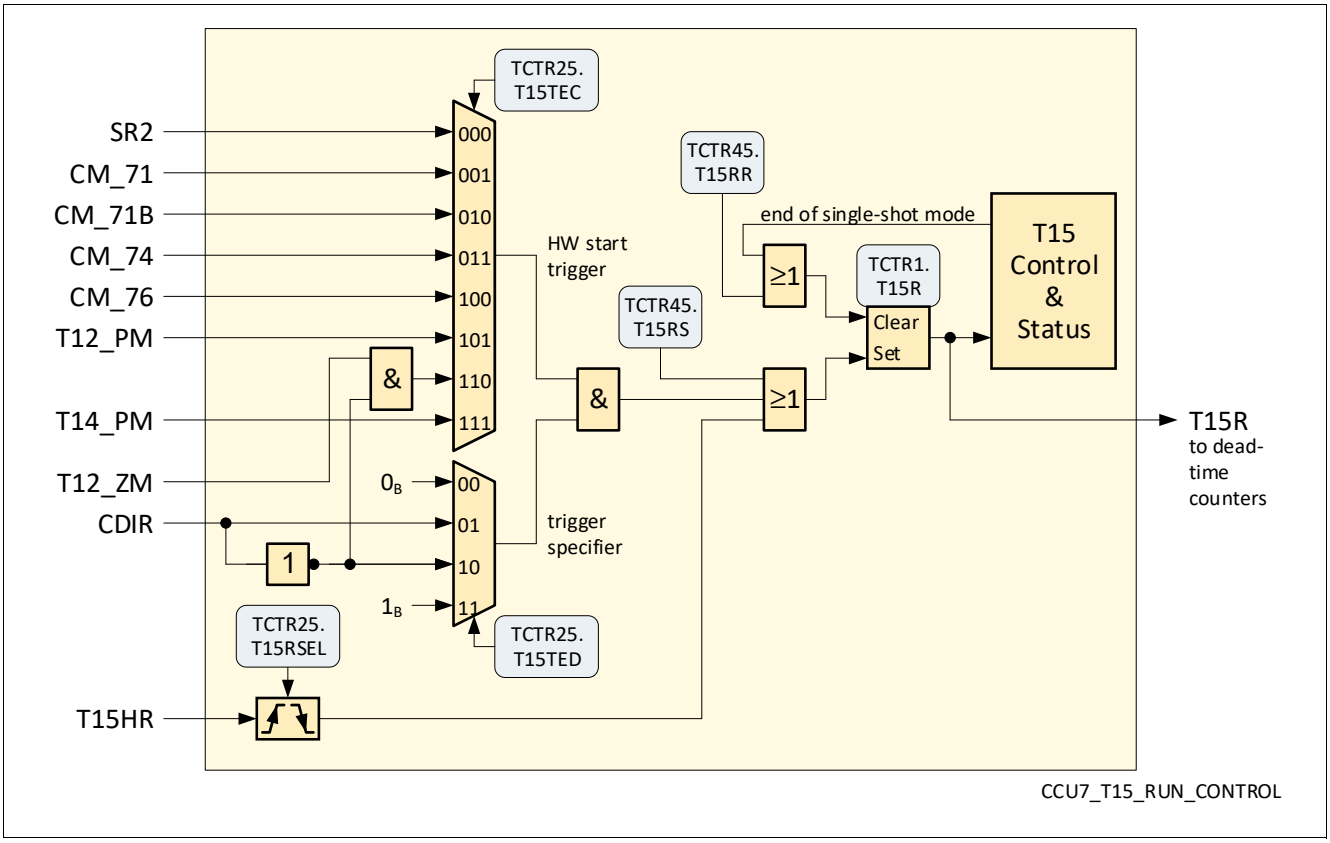


Figure 338 T15 run control

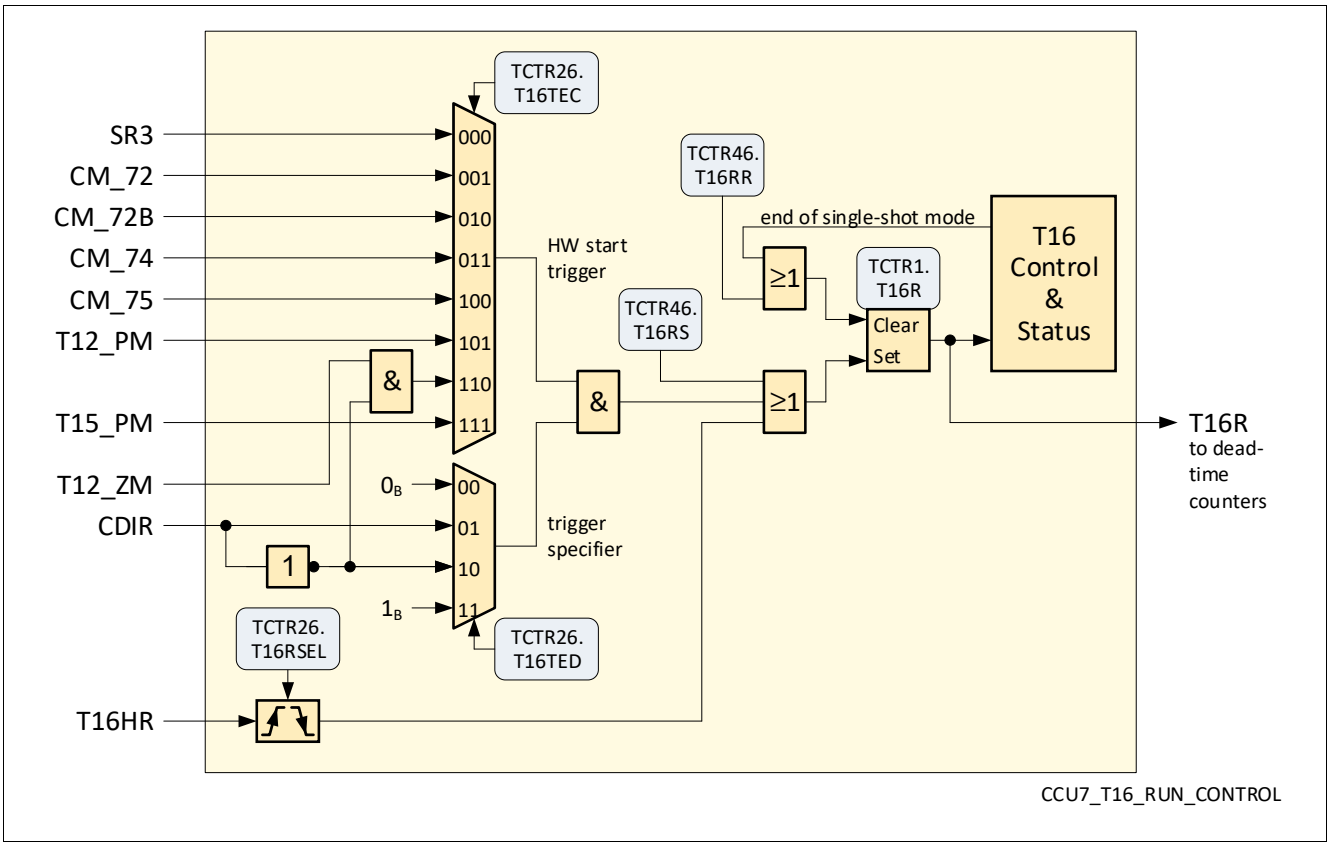


Figure 339 T16 run control

25.7.2 T1x counting scheme

This chapter describes the clocking and the counting capabilities of T1x.

25.7.2.1 T1x clock selection

Timer mode

In Timer mode (PISEL2x.ISCNT1x = 00_B), the input clock f_{T1x} of timer T1x can be selected with **T_CLK_CTRL**.T1x_CLK_SEL. Possible sources are f_{PWM}, f_{FDIV0}, f_{FDIV1}. In addition, a 1/256 prescaler is available, which can be activated via bit TxPRE in register **TCTR1**. Note that this prescaler selection is not an individual control, but applies to all three timers T14, T15, and T16.

Counter mode

In Counter mode, timer T1x counts one step:

- If a 1 is written to TCTR4x.T1xCNT and PISEL2x.ISCNT1x = 01_B
- If a rising edge of input signal T1xHR is detected and PISEL2x.ISCNT1x = 10_B
- If a falling edge of input signal T1xHR is detected and PISEL2x.ISCNT1x = 11_B

25.7.2.2 T1x counting

The period of the timer is determined by the value in the period register T1xPR according to the following formula:

$T1x_{PER}$ = <Period-Value> + 1; in T1x clocks (f_{T1x}) (25.5)

Timer T1x can only count up, comparable to the Edge-aligned mode of T12. This leads to very simple ‘counting rule’ for the T1x counter:

- The counter is cleared with the next T1x clock edge if a period-match is detected. The counting direction is always upwards

The behavior of T1x is illustrated in **Figure 340**.

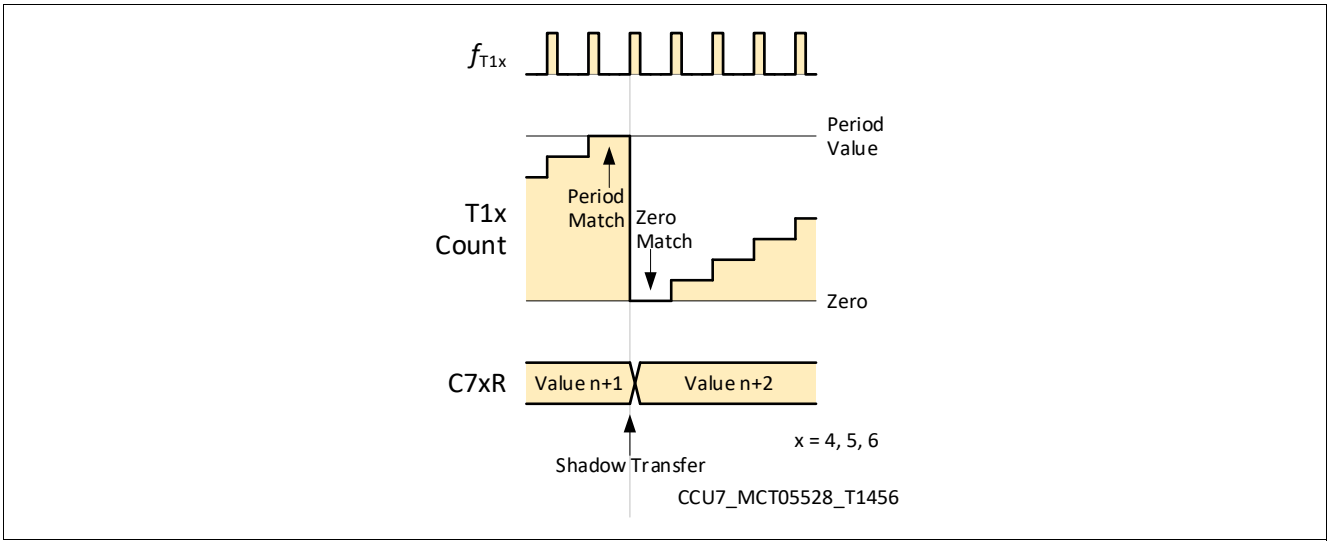


Figure 340 T1x counting sequence

25.7.2.3 Single-shot mode

In Single-shot mode, the timer run bit T1xR is cleared by hardware. If bit T1xSSC = 1, the timer T1x will stop when the current timer period is finished.

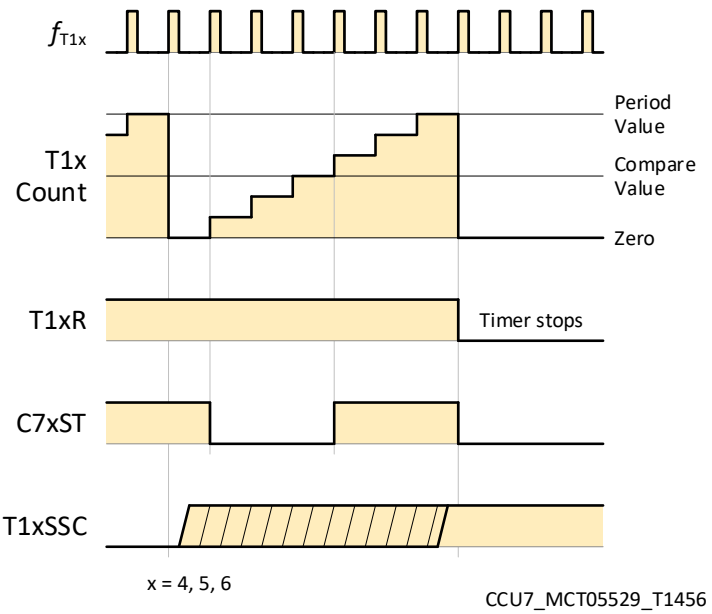


Figure 341 Single-shot operation of timer T1x

25.7.2.4 Synchronization to T12

Timer T1x can be synchronized to a T12 event. Bit fields T1xTEC and T1xTED select the event that is used to start timer T1x. The selected event sets bit T1xR via HW, and T1x starts counting. Combined with the Single-shot mode, this feature can be used to generate a programmable delay after a T12 event.

Figure 342 shows an example for the synchronization of T1x to a T12 event. Here, the selected event is a compare-match (compare value = 2) while counting up. The clocks of T12 and T1x can be different (other prescaler factor); the figure shows an example in which T1x is clocked with half the frequency of T12.

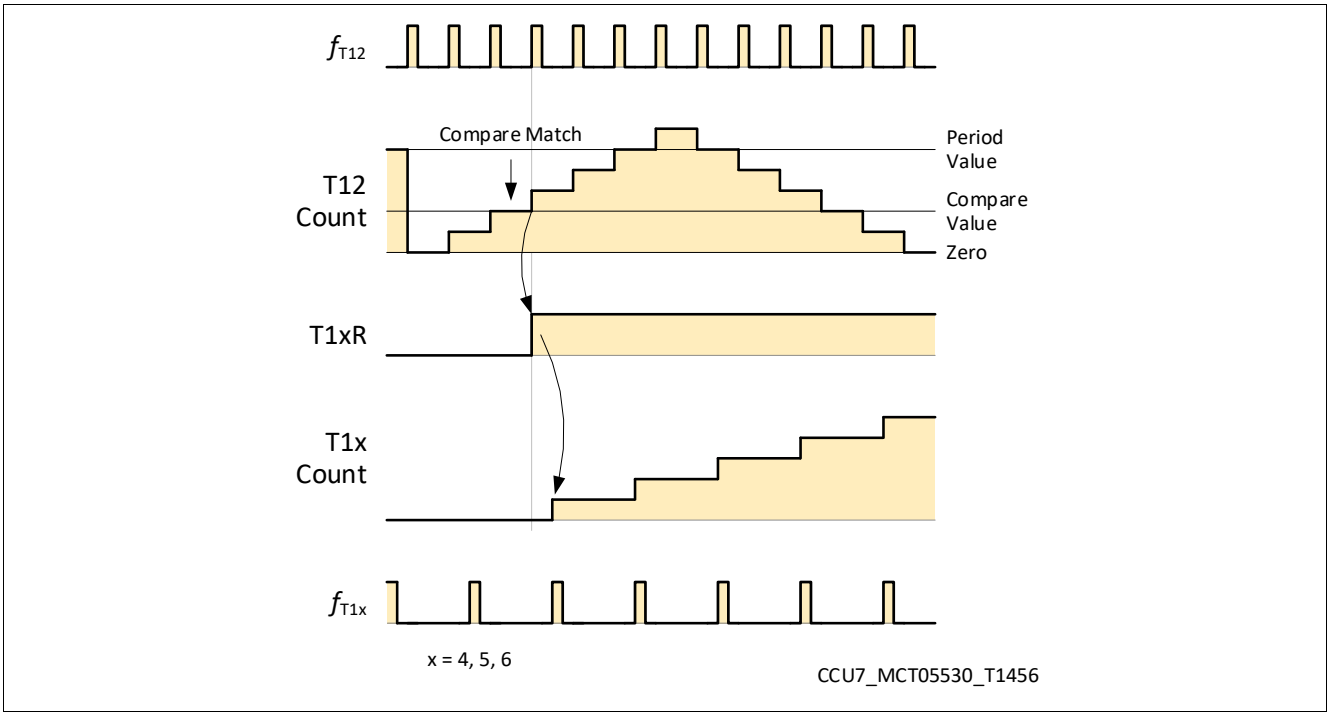


Figure 342 Synchronization of T1x to T12 compare-match

Bit field T1xTEC selects the trigger event to start T1x (automatic set of T1xR for synchronization to T12 compare signals) according to the combinations shown in **Table 302**. Bit field T1xTED additionally specifies for which count direction of T12 the selected trigger event should be regarded (see **Table 303**).

Table 302 T14, T15, T16 trigger event selection

T14TEC T15TEC T16TEC	Timer 14 Selected event	Timer 15 Selected event	Timer 16 Selected event
000_B	SR1	SR2	SR3
001_B	T12 CM_70	T12 CM_71	T12 CM_72
010_B	T12 CM_70B	T12 CM_71B	T12 CM_72B
011_B	T15 CM_75	T14 CM_74	T14 CM_74
100_B	T16 CM_76	T16 CM_76	T15 CM_75
101_B	T12_PM	T12_PM	T12_PM
110_B	T12_ZM AND CDIR = 0	T12_ZM AND CDIR = 0	T12_ZM AND CDIR = 0
111_B	T13_PM	T14_PM	T15_PM

Table 303 T12 trigger event additional specifier

T14TED T15TED T16TED	Timer T14, T15, T16 Selected event specifier
00_B	Reserved, no triggering
01_B	Selected event is active while T12 is counting up (CDIR = 0)
10_B	Selected event is active while T12 is counting down (CDIR = 1)
11_B	Selected event is active independently of the count direction of T12

25.7.3 T1x compare mode

Associated with timer T1x is one compare channel, that can perform compare operations with regard to the contents of the T1x counter.

Figure 343 gives an overview on the T1x channel in Compare mode. The channel is connected to the T1x counter register via an equal-to comparator, generating a compare-match signal when the contents of the counter matches the contents of the compare register.

The channel consists of the comparator and a double register structure - the actual compare register, C7xR (see **C7xR (x=3-6)**), feeding the comparator, and an associated shadow register, C7xSR (see **C7xSR (x=3-6)**), that is preloaded by software and transferred into the compare register when signal T1x shadow transfer, T1x_ST, gets active. Providing a shadow register for the compare value as well as for other values related to the generation of the PWM signal facilitates a concurrent update by software for all relevant parameters.

Associated with the channel is a state bit, **CMPSTAT**.C7xST, holding the status of the compare operation. **Figure 343** gives an overview on the logic for the compare operation and the state bit.

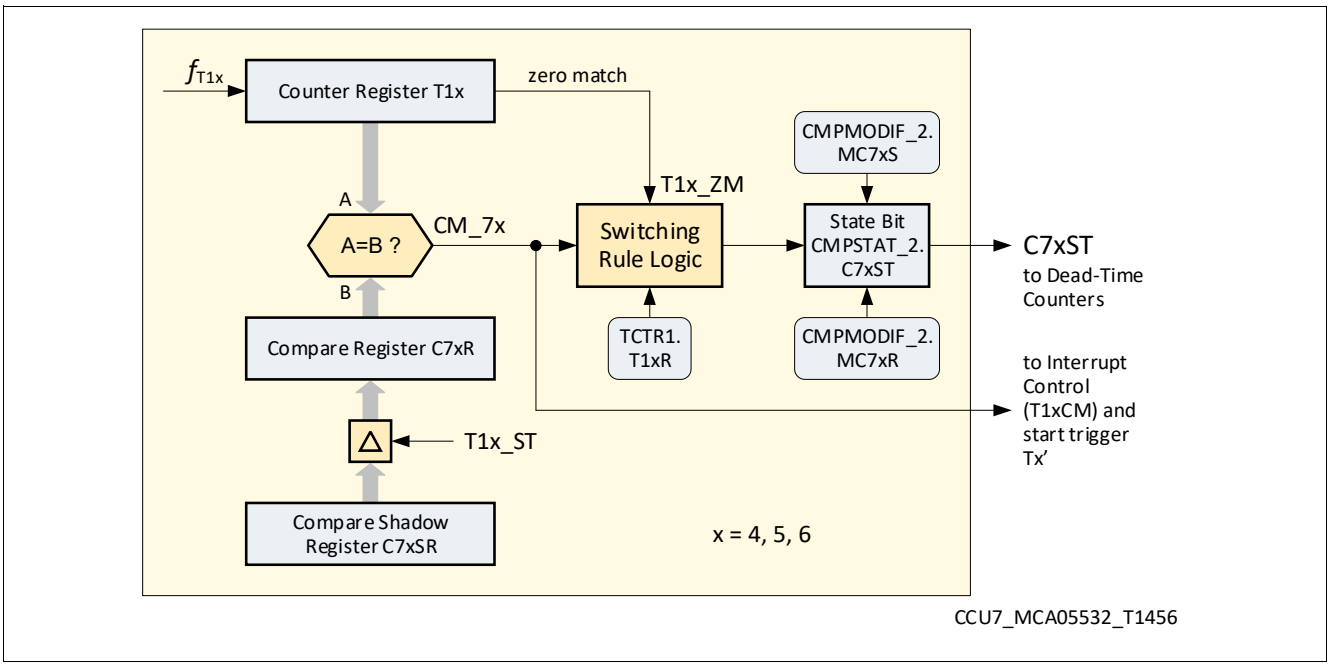


Figure 343 T1x compare mode and state bit block diagram

A compare interrupt event CM_7x is signaled when a compare-match is detected. The actual setting of a state bit has no influence on the interrupt generation.

The inputs to the switching rule logic for the C7xST bit are the timer run bit (T1xR), the timer zero-match signal (T1x_ZM), and the actual individual compare-match signal CM_7x. In addition, the state bit can be set or cleared by software via bits MC73S and MC73R in register **CMPMODIF_2**.

A modification of the state bit C7xST by hardware is only possible while timer T1x is running (T1xR = 1). If this is the case, the following switching rules apply for setting and resetting the state bit in Compare mode:

- State bit C7xST is set to 1:
  - With the next T1x clock (f_{T1x}) after a compare-match (i.e., when the counter is incremented above the compare value)
  - With the next T1x clock (f_{T1x}) after a zero-match AND a parallel compare-match
- State bit C7xST is cleared to 0:
  - With the next T1x clock (f_{T1x}) after a zero-match AND NO parallel compare-match

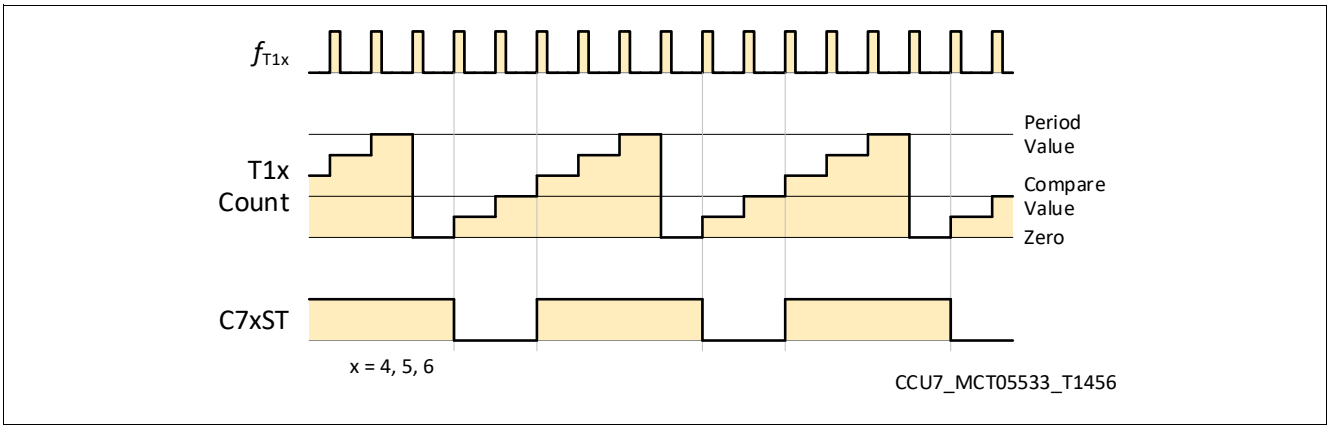


Figure 344 T1x compare operation

25.7.4 T1x shadow transfer

A special shadow transfer signal (T1x_ST) can be generated to facilitate updating the period and compare values of the compare channels C7x synchronously to the operation of T1x. Providing a shadow register for values defining one PWM period facilitates a concurrent update by software for all relevant parameters. The next PWM period can run with a new set of parameters. The generation of this signal is enabled by software via bit **TCTR1**.STE1x (set by writing 1 to the write-only bit **TCTR4x (x=4-6)**.T1xSTR, cleared by writing 1 to the write-only bit **TCTR4x (x=4-6)**.T1xSTD).

When signal T1x_ST is active, a shadow register transfer is triggered with the next cycle of the T1x clock. Bit STE1x is automatically cleared with the shadow register transfer.

A T1x shadow register transfer takes place (T1x_ST active):

- While timer T1x is not running (T1xR = 0), or
- STE1x = 1 and a period-match is detected while T1xR = 1

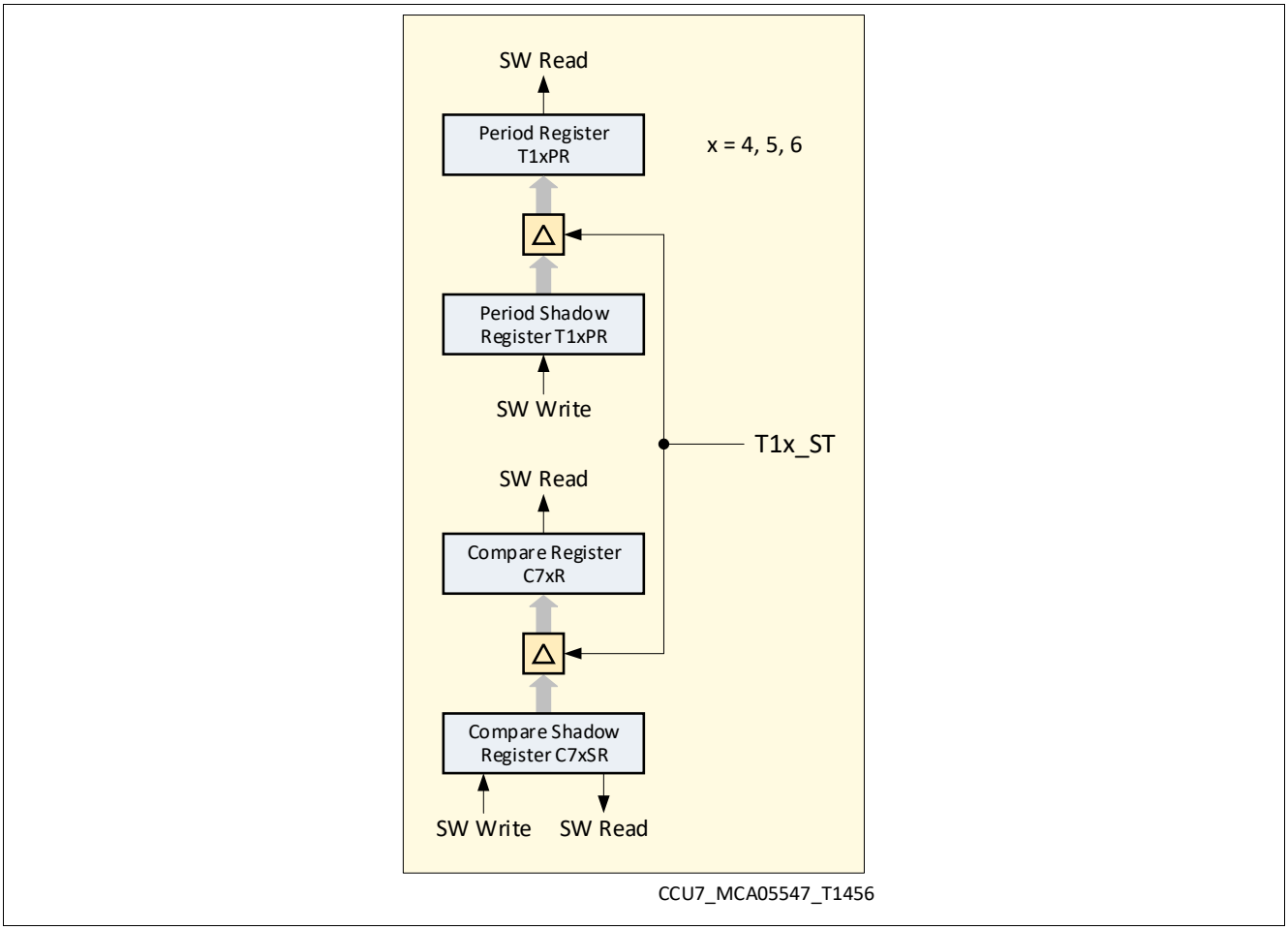


Figure 345 T1x shadow register overview

25.8 Trap control

The trap functionality permits the PWM outputs to react on the state of the input signal $\overline{\text{CTRAP}}$. This functionality can be used to switch off the power devices if the trap input becomes active (e.g. to perform an emergency stop). The trap handling and the effect on the output modulation are controlled by the bits in the trap control register **TRPCTR**. The trap flags TRPF and TRPS are located in register **IS** and can be set/cleared by SW by writing to registers **ISS** and **ISR**.

Figure 346 gives an overview on the trap function.

The trap flag TRPF monitors the trap input and initiates the entry into the trap state. The trap state bit TRPS determines the effect on the outputs and controls the exit of the trap state.

When a trap condition is detected ($\overline{\text{CTRAP}}$ = 0) and the input is enabled (TRPPEN = 1), both, the trap flag TRPF and the trap state bit TRPS, are set to 1 (trap state active).

The output of the trap state bit TRPS leads to the output modulation block and can there deactivate the outputs (set them to the passive state). Individual enable control bits for each of the six T12-related outputs and the T13-related output facilitate a flexible adaptation to the application needs.

There are a number of different ways to exit the trap state. This offers SW the option to select the best operation for the application. Exiting the trap state can be done either immediately when the trap condition is removed ($\overline{\text{CTRAP}}$ = 1), or under software control, or synchronously to the PWM generated by either timer T12 or timer T13.

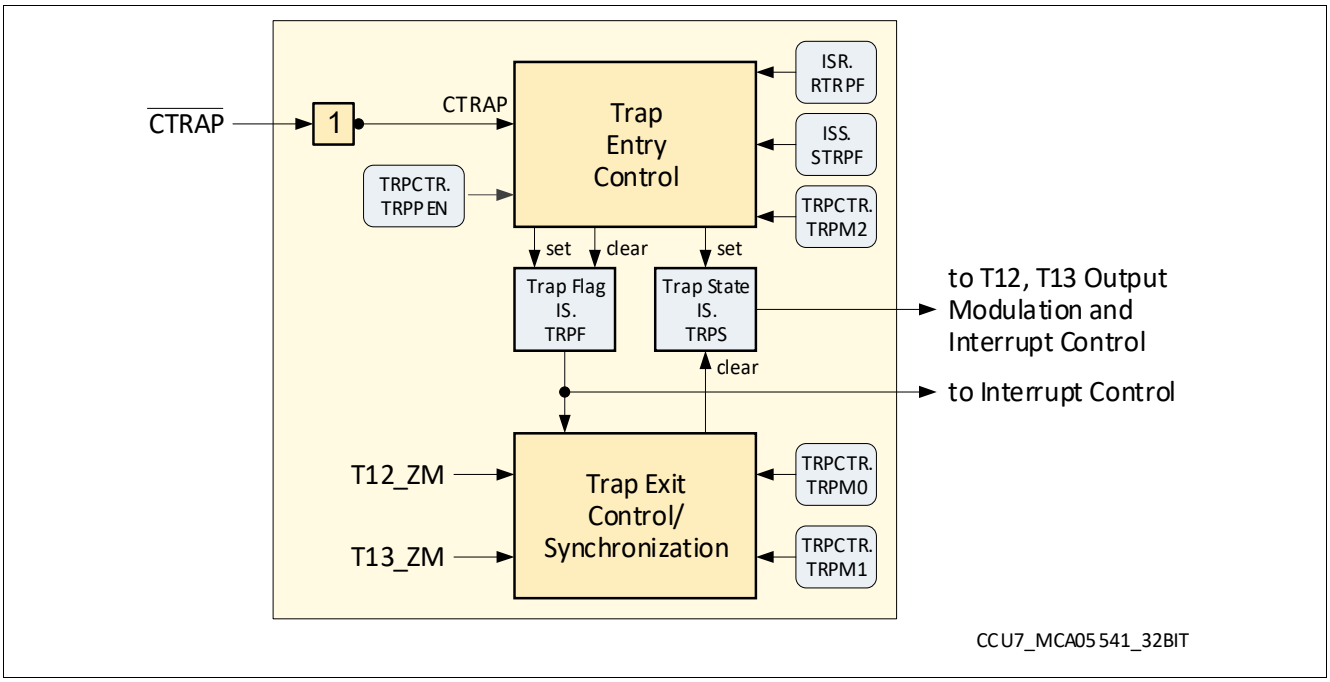


Figure 346 Trap logic block diagram

Clearing of TRPF is controlled by the mode control bit TRPM2 in register **TRPCTR**. If TRPM2 = 0, TRPF is automatically cleared by HW when $\overline{\text{CTRAP}}$ returns to the inactive level ($\overline{\text{CTRAP}}$ = 1). When TRPM2 = 1, TRPF must be reset by SW after $\overline{\text{CTRAP}}$ has become inactive.

Clearing of TRPS is controlled by the mode control bits TRPM1 and TRPM0 in register **TRPCTR**. A reset of TRPS terminates the trap state and returns to normal operation. There are three options selected by TRPM1 and TRPM0. One is that the trap state is left immediately when the trap flag TRPF is cleared, without any synchronization to timers T12 or T13. The other two options facilitate the synchronization of the termination of the trap state to the count periods of either timer T12 or timer T13. **Figure 347** gives an overview on the associated operation.

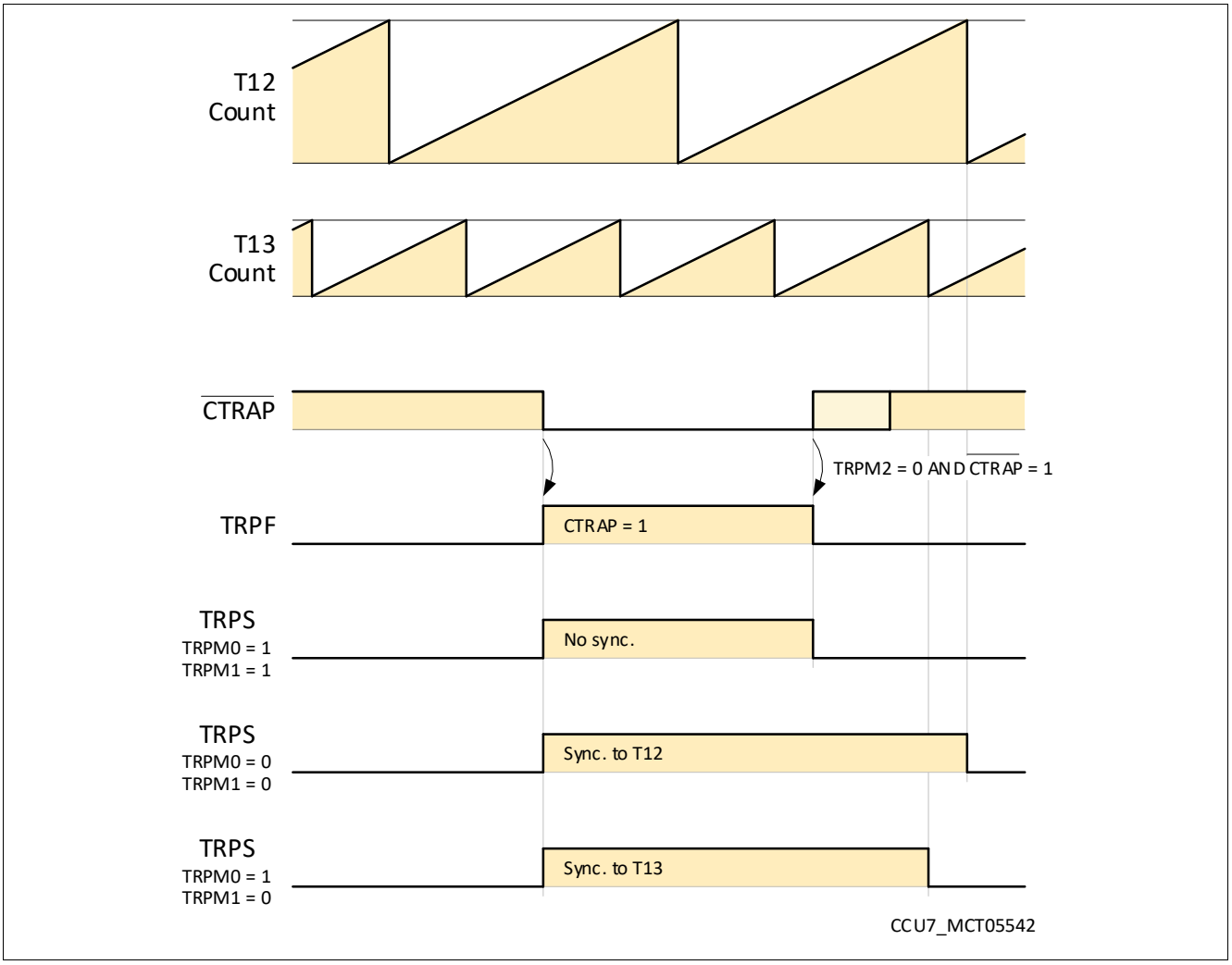


Figure 347 Trap state synchronization (with TRPM2 = 0)

25.9 Multi-channel mode

The Multi-channel mode offers the possibility to modulate the six compare channels CC7x/COUT7x individually with six software defined bits. Therefore each bit in bit field **MCMOUT**.MCMP[5:0] is assigned to one compare channel. If Multi-channel mode is enabled (bit **MODCTR**.MCMEN = 1), the MCMP[5:0] bits can actively gate (via zero value) or allow modulation (via one value) of the corresponding compare channel (see also **Figure 359**).

This bit field has its own shadow bit field **MCMOUTS**.MCMPS, that can be written by software. The transfer of the new value in MCMPS to the bit field MCMP can be triggered by, and synchronized to, T12 or T13 events. This structure permits the software to write the new value, that is then taken into account by the hardware at a well-defined moment and synchronized to a PWM signal. This avoids unintended pulses due to unsynchronized modulation sources.

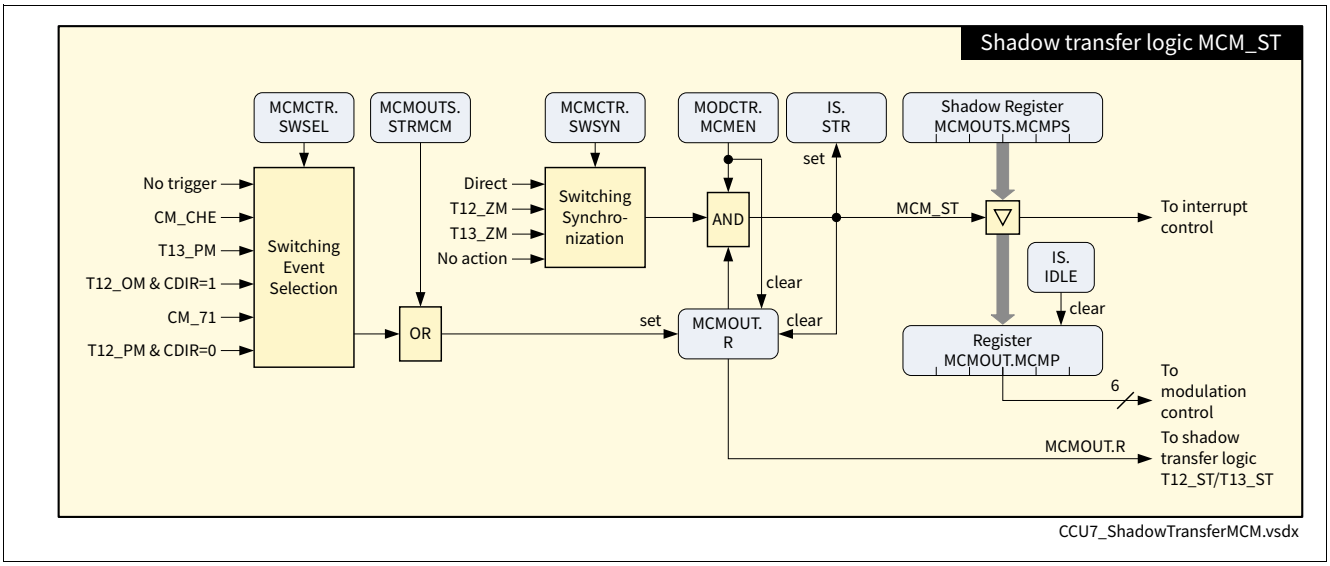


Figure 348 Multi-channel mode block diagram

Figure 348 shows the functional blocks for the multi-channel operation. The event that triggers the update of bit field MCMP is chosen by SWSEL. In order to synchronize the update of MCMP to a PWM generated by T12 or T13, bit field SWSYN allows the selection of the synchronization event leading to the transfer from MCMPS to MCMP. Due to this structure, an update takes place with a new PWM period. A reminder flag R is set when the selected switching event occurs (the event is not necessarily synchronous to the modulating PWM), and is cleared when the transfer takes place (MCM_ST) or when MCMEN=0. This flag can be monitored by software to check for the status of this logic block. If the shadow transfer from MCMPS to MCMP takes place, bit **IS**.STR becomes set and an interrupt can be generated.

In addition to the multi-channel shadow transfer event MCM_ST, the shadow transfers for T12 (T12_ST) and T13 (T13_ST) can be generated to allow concurrent updates of applied duty cycles for T12 and/or T13 modulation and multi-channel patterns.

If it is explicitly desired, the update takes place immediately with the occurrence of the selected event when the direct synchronization mode is selected (SWSYN = 0). The update can also be requested by software by writing to bit field MCMPS with the shadow transfer request bit STRMCM = 1. The option to trigger an update by SW is possible for all settings of SWSEL.

By using the direct mode and bit STRMCM = 1, the update takes place completely under software control.

Table 304 Multi-channel mode switching event selection

SWSEL	Selected event (see register MCMCTR)
000_B	No automatic event detection
001_B	Correct Hall event (CM_CHE) detected at input signals CCPOSx without additional delay
010_B	T13 period-match (T13_PM)
011_B	T12 one-match while counting down (T12_OM and CDIR = 1)
100_B	T12 compare channel 71 event while counting up (CM_71 and CDIR = 0) to support the phase delay function by CC71 for block commutation mode
101_B	T12 period-match while counting up (T12_PM and CDIR = 0)
110_B, 111_B	Reserved, no action

Table 305 Multi-channel mode switching synchronization

SWSYN	Synchronization event (see register MCMCTR)
00_B	Direct mode: the trigger event directly causes the shadow transfer
01_B	T13 zero-match (T13_ZM), the MCM shadow transfer is synchronized to a T13 PWM
10_B	T12 zero-match (T12_ZM), the MCM shadow transfer is synchronized to a T12 PWM
11_B	Reserved, no action

25.10 Hall sensor mode

For Brushless DC-motors in block commutation mode, the Multi-channel mode has been introduced to provide efficient means for switching pattern generation. These patterns need to be output in relation to the angular position of the motor. For this, usually Hall sensors or Back-EMF sensing are used to determine the angular rotor position. The CCU7 provides three inputs, CCPOS0, CCPOS1, and CCPOS2, that can be used as inputs for the Hall sensors or the Back-EMF detection signals.

There is a strong correlation between the motor position and the output modulation pattern. When a certain position of the motor has been reached, indicated by the sampled Hall sensor inputs (the Hall pattern), the next, pre-determined Multi-Channel Modulation pattern has to be output. Because of different machine types, the modulation pattern for driving the motor can vary. Therefore, it is wishful to have a wide flexibility in defining the correlation between the Hall pattern and the corresponding Modulation pattern. Furthermore, a hardware mechanism significantly reduces the CPU load for block-commutation.

The CCU7 offers the flexibility by having a register containing the currently assumed Hall pattern (CURH), the next expected Hall pattern (EXPH) and the corresponding output pattern (MCMP). A new Modulation pattern is output when the sampled Hall inputs match the expected ones (EXPH). To detect the next rotation phase (segment for block commutation), the CCU7 monitors the Hall inputs for changes. When the next expected Hall pattern is detected, the next corresponding Modulation pattern is output.

To increase for noise immunity (to a certain extend), the CCU7 offers the possibility to introduce a sampling delay for the Hall inputs. Some changes of the Hall inputs are not leading to the expected Hall pattern, because they are only short spikes due to noise. The Hall pattern compare logic compares the Hall inputs to the next expected pattern and also to the currently assumed pattern to filter out spikes.

For the Hall and Modulation output patterns, a double-register structure is implemented. While register **MCMOUT** holds the actually used values, its shadow register **MCMOUTS** can be loaded by software from a pre-defined table, holding the appropriate Hall and Modulation patterns for the given motor control.
A transfer from the shadow register into register MCMOUT can take place when a correct Hall pattern change is detected. Software can then load the next values into register MCMOUTS. It is also possible by software to force a transfer from MCMOUTS into MCMOUT.

Note: The Hall input signals CCPOSx and the CURH and EXPH bit fields are arranged in the following order:
CCPOS0 corresponds to CURH.0 (LSB) and EXPH.0 (LSB).
CCPOS1 corresponds to CURH.1 and EXPH.1.
CCPOS2 corresponds to CURH.2 (MSB) and EXPH.2 (MSB).

25.10.1 Hall pattern evaluation

The Hall sensor inputs CCPOSx can be permanently monitored via an edge detection block (with the module clock f_{PWM}). In order to suppress spikes on the Hall inputs due to noise in rugged inverter environment, two optional noise filtering methods are supported by the Hall logic (both methods can be combined).

- Noise filtering with delay: For this function, the mode control bit fields MSEL7x for all T12 compare channels must be programmed to 1000_B and DBYP = 0. The selected event triggers Dead-Time Counter 0 to generate a programmable delay (defined by bit field **T12DT0_VAL**.DTM_RISE, see **T12DTx_VAL (x=0-2)**). When the delay has elapsed, the evaluation signal HCRDY becomes activated. Output modulation with T12 PWM signals is not possible in this mode.
- Noise filtering by synchronization to PWM: The Hall inputs are not permanently monitored by the edge detection block, but samples are taken only at defined points in time during a PWM period. This can be used to sample the Hall inputs when the switching noise (due to PWM) does not disturb the Hall input signals.

If neither the delay function of Dead-Time Counter 0 is used for the Hall pattern evaluation nor the Hall mode for Brushless DC-drive control is enabled, the timer T12 block is available for PWM generation and output modulation.

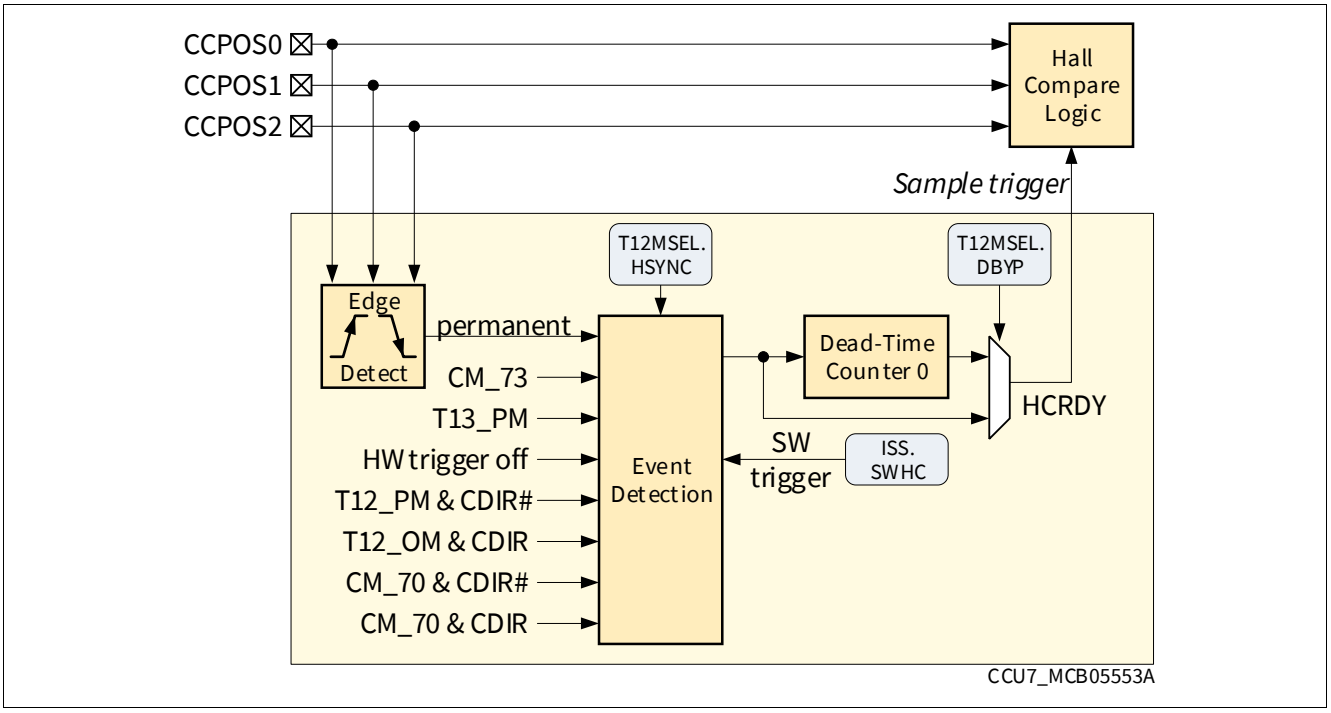


Figure 349 Hall pattern evaluation

If the evaluation signal HCRDY (Hall Compare Ready, see **Figure 350**) becomes activated, the Hall inputs are sampled and the Hall compare logic starts the evaluation of the Hall inputs.

Figure 349 illustrates the events for Hall pattern evaluation and the noise filter logic, **Table 306** summarizes the selectable trigger input signals.

Table 306 Hall sensor mode trigger event selection

HSYNC	Selected event (see register T12MSEL)
000_B	Any edge at any of the inputs CCPOSx, independent from any PWM signal (permanent check)
001_B	A T13 compare-match (CM_73)
010_B	A T13 period-match (T13_PM)
011_B	Hall sampling triggered by HW sources is switched off
100_B	A T12 period-match while counting up (T12_PM and CDIR = 0)
101_B	A T12 one-match while counting down (T12_OM and CDIR = 1)
110_B	A T12 compare-match of compare channel CC71 while counting up (CM_71 and CDIR = 0)
111_B	A T12 compare-match of compare channel CC71 while counting down (CM_71 and CDIR = 1)

25.10.2 Hall pattern compare logic

Figure 350 gives an overview on the double-register structure and the pattern compare logic. Software writes the next modulation pattern (MCMPS) and the corresponding current (CURHS) and expected (EXPHS) Hall patterns into the shadow register MCMOUTS. Register MCMOUT holds the actually used values CURH and EXPH. The modulation pattern MCMP is provided to the output modulation block. The current (CURH) and expected (EXPH) Hall patterns are compared to the sampled Hall sensor inputs CPOSx (visible in register **CMPSTAT**). Sampling of the inputs and the evaluation of the comparator outputs is triggered by the evaluation signal HCRDY (Hall Compare Ready), that is detailed in the next section.

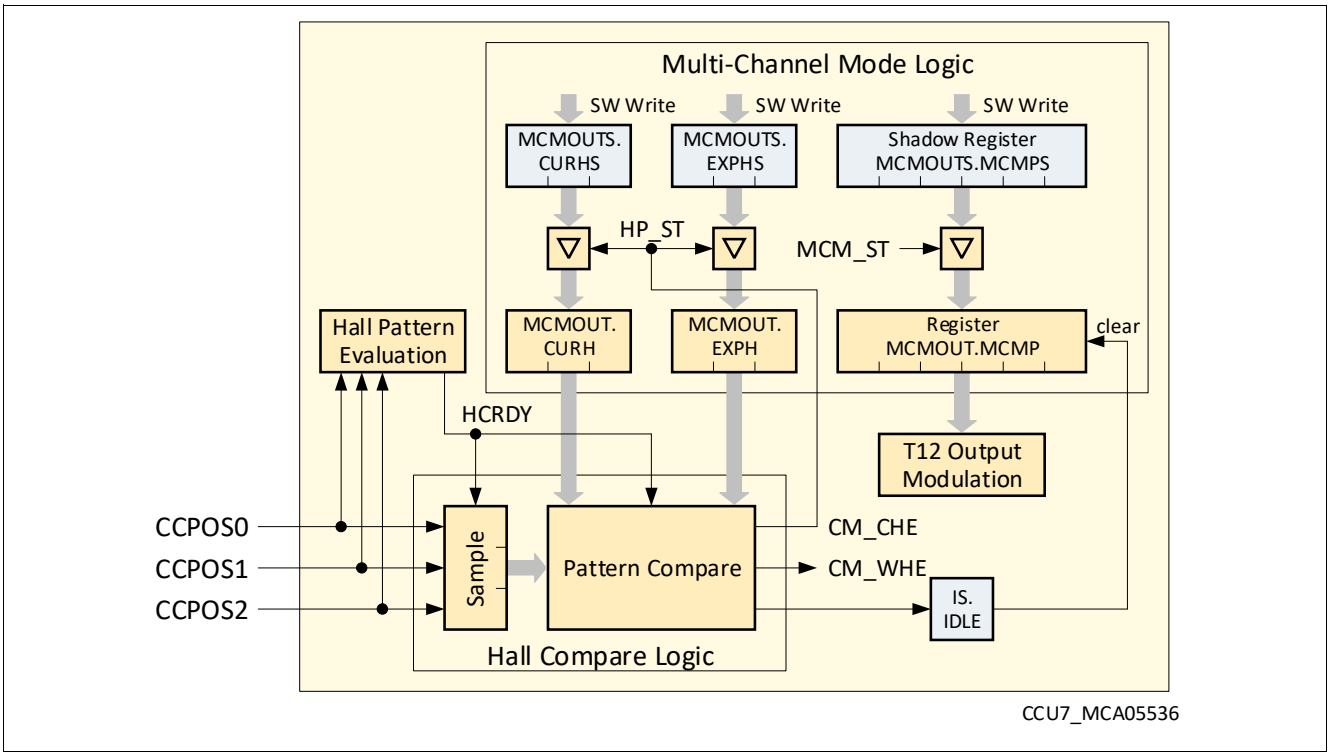


Figure 350 Hall pattern compare logic

- If the sampled Hall pattern matches the value programmed in CURH, the detected transition was a spike (no Hall event) and no further actions are necessary
- If the sampled Hall pattern matches the value programmed in EXPH, the detected transition was the expected event (correct Hall event CM_CHE) and the MCMP value has to change
- If the sampled Hall pattern matches neither CURH nor EXPH, the transition was due to a major error (wrong Hall event CM_CWE) and can lead to an emergency shut down (IDLE)

At every correct Hall event (CM_CHE), the next Hall patterns are transferred from the shadow register MCMOUTS into MCMOUT (Hall pattern shadow transfer HP_ST), and a new Hall pattern with its corresponding output pattern can be loaded (e.g. from a predefined table in memory) by software into MCMOUTS. For the Modulation patterns, signal MCM_ST is used to trigger the transfer.

Loading this register can also be done by writing MCMOUTS.STRHP = 1 (for EXPH and CURH) or MCMOUTS.STRMCMP = 1 (for MCMP).

Note: If, in a corner case, a hardware event occurs simultaneously with a software write where MCMOUTS.STRHP = 1 or MCMOUTS.STRMCMP = 1, the current contents of MCMOUTS is copied to the corresponding bit fields of MCMOUT. The new value written to MCMOUTS will be loaded upon the next event.

25.10.3 Hall mode flags

Depending on the Hall pattern compare operation, a number of flags are set in order to indicate the status of the module and to trigger further actions and interrupt requests.

- Flag **IS**.CHE (Correct Hall Event) is set by signal CM_CHE when the sampled Hall pattern matches the expected one (EXPH). This flag can also be set by SW by setting bit **IS**.SCHE = 1. If enabled by bit **IEN**.ENCHE = 1, the set signal for CHE can also generate an interrupt request to the CPU. Bit field **INP**.INPCHE defines which service request output becomes activated in case of an interrupt request. To clear flag CHE, SW needs to write **ISR**.RCHE = 1.
- Flag IS.WHE indicates a Wrong Hall Event. Its handling for flag setting and resetting as well as interrupt request generation are similar to the mechanism for flag CHE.
- The implementation of flag STR is done in the same way as for CHE and WHE. This flag is set by HW by the shadow transfer signal MCM_ST (see also **Figure 348**).

Please note that for flags CHE, WHE, and STR, the interrupt request generation is triggered by the set signal for the flag. That means, a request can be generated even if the flag is already set. There is no need to clear the flag in order to enable further interrupt requests.

The implementation for the IDLE flag is different. It is set by HW through signal CM_WHE if enabled by bit ENIDLE. Software can also set the flag via bit SIDLE. As long as bit IDLE is set, the modulation pattern field MCMP is cleared to force the outputs to the passive state. Flag IDLE must be cleared by software by writing RIDLE = 1 in order to return to normal operation. To fully restart from IDLE mode, the transfer requests for the bit fields in register MCMOUTS to register MCMOUT have to be initiated by software via bits STRMCM and STRHP in register MCMOUTS. In this way, the release from IDLE mode is under software control, but can be performed synchronously to the PWM signal.

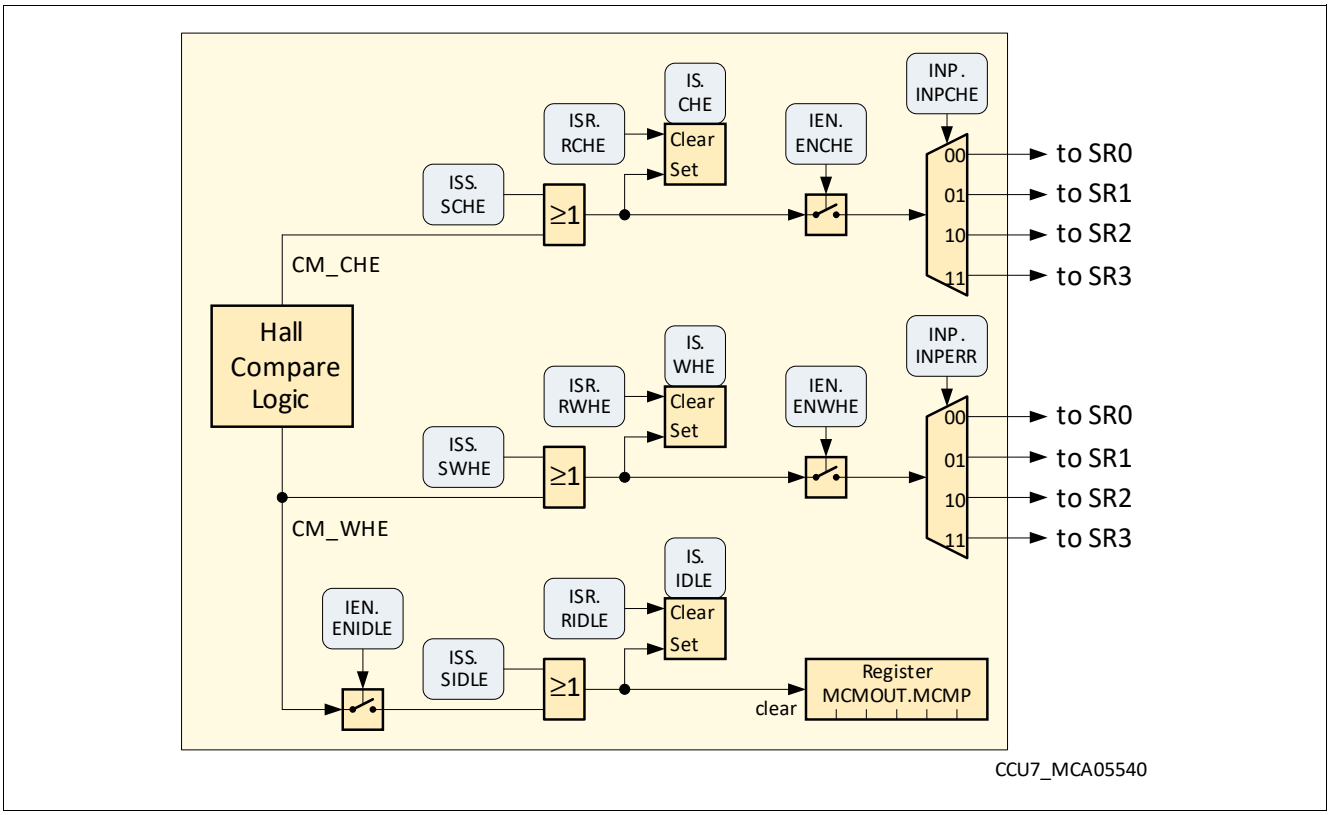


Figure 351 Hall mode flags

25.10.4 Hall mode for Brushless DC-motor control

The CCU7 provides a mode for the timer T12 block especially targeted for convenient control of block commutation patterns for brushless DC-motors. This mode is selected by setting all **T12MSEL**.MSEL7x bit fields of the T12 channels to 1000_B.

In this mode, illustrated in **Figure 352**, channel CC70 is placed in Capture mode to measure the time elapsed between the last two correct Hall events, channel CC71 is in Compare mode to provide a programmable phase delay between the Hall event and the application of a new PWM output pattern, and channel CC72 is also in Compare mode as first time-out criterion. A second time-out criterion can be built by the T12 period-match event.

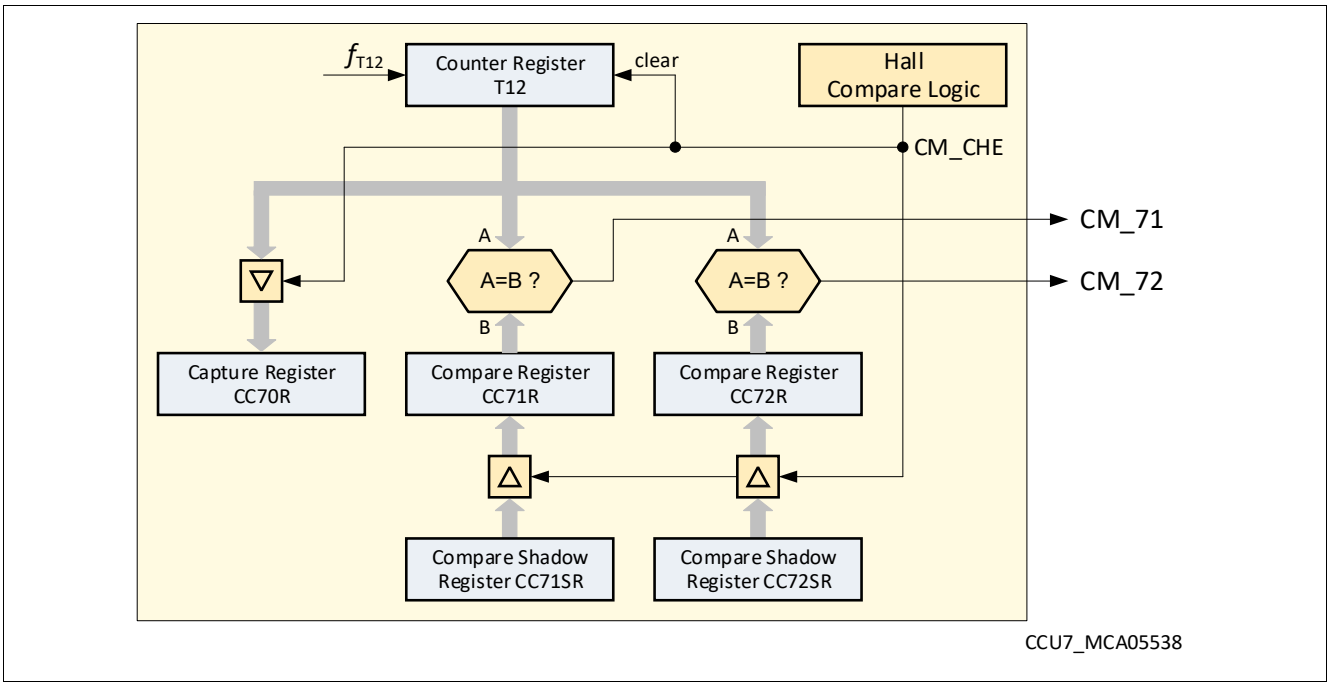


Figure 352 T12 block in Hall sensor mode

The signal CM_CHE from the Hall compare logic is used to transfer the new compare values from the shadow registers CC7xSR into the actual compare registers CC7xR, performs the shadow transfer for the T12 period register, to capture the current T12 contents into register CC70R, and to clear T12.

Note: In this mode, the shadow transfer signal T12_ST is not generated. Not all shadow bits, such as the PSLy bits, will be transferred to their main registers. To program the main registers, SW needs to write to these registers while timer T12 is stopped. In this case, a SW write actualizes both registers.

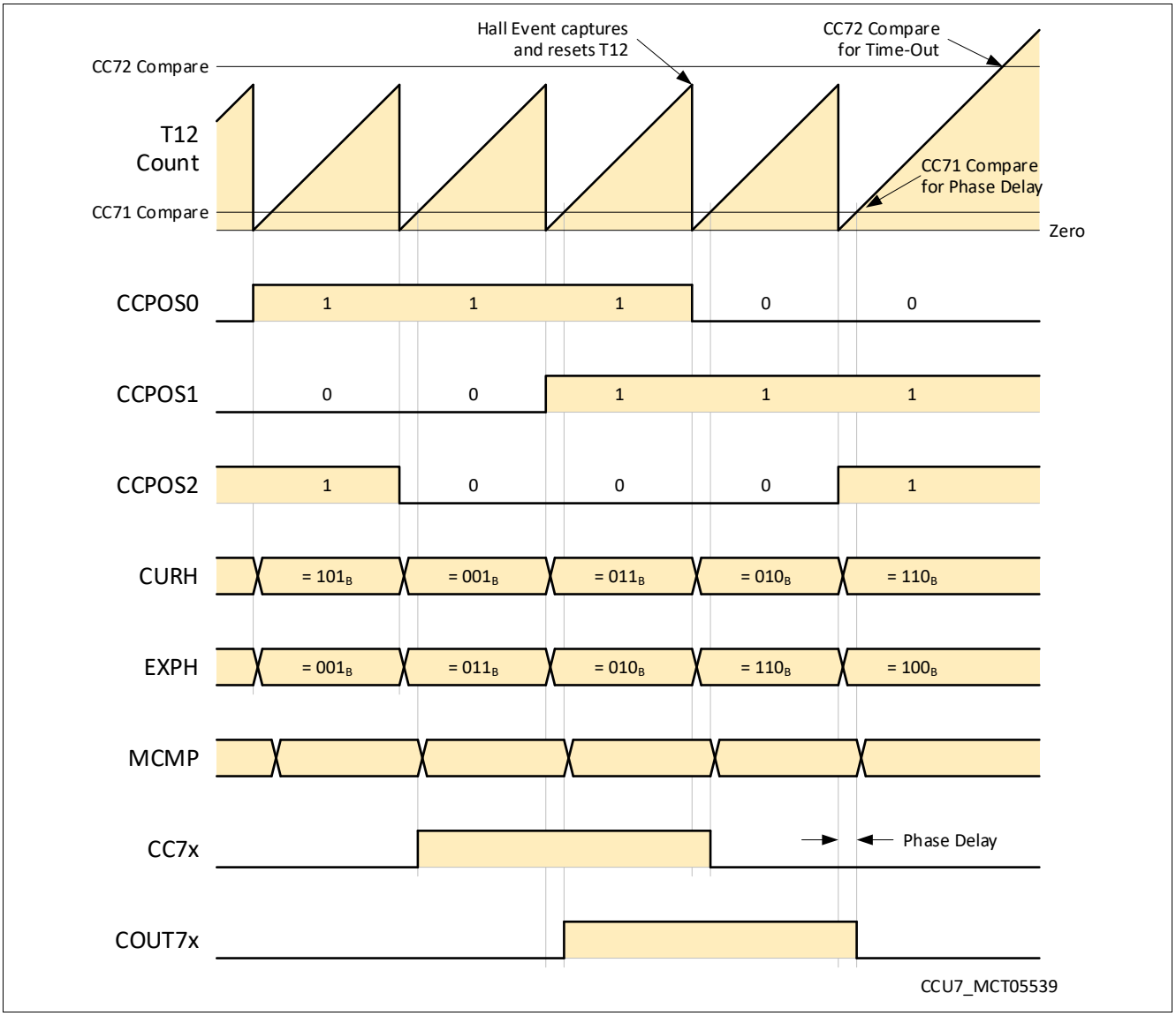


Figure 353 Brushless DC-motor control example (all MSEL7x = 1000_B)

After the detection of an expected Hall pattern (CM_CHE active), the T12 count value is captured into channel CC70 (representing the actual rotor speed by measuring the elapsed time between the last two correct Hall events), and T12 is reset. When the timer reaches the compare value in channel CC71, the next multi-channel state is switched by triggering the shadow transfer of bit field MCMP (if enabled in bit field **MCMCTR**.SWSYN). This trigger event can be combined with the synchronization of the next multi-channel state to the PWM source (to avoid spikes on the output lines, see **Chapter 25.9**). This compare function of channel CC71 can be used as a phase delay from the position sensor input signals to the switching of the output signals, that is necessary if a sensorless back-EMF technique or Hall sensors are used. The compare value in channel CC72 can be used as a time-out trigger (interrupt), indicating that the actual motor speed is far below the desired destination value. An abnormal load change can be detected with this feature and PWM generation can be disabled.

25.11 Dead-time control

When controlling a half-bridge, the generation of complementary control signals for the high-side and the low-side is necessary. Due to the switching time of the power switches a so-called dead-time between the OFF-command and the ON-command is necessary.

In most cases, the switching behavior of the connected power switches is not symmetrical concerning the switch-on and switch-off times. Therefore separately programmable dead-times for ON/OFF-delays (DTM_RISE and DTM_FALL) are available.

The dead-time control has following features:

- Three independent dead-time counters (DTC0/1/2) with each generating a dead-time between two complementary signals based on one trigger input signal
- Trigger inputs from T12 status bits (CC7xST and CC7xSTn, CC7xBST and CC7xBSTn, x = 0 to 3, n = inverted)
- Trigger inputs from T14, T15, T16 run and status bits (T14/15/16R and CC74/75/67ST)
- Independently programmable dead-time for rising and falling edge based on f_{DTCLK} with 8-bit value

The block diagram for the dead-time control is shown in **Figure 354**.

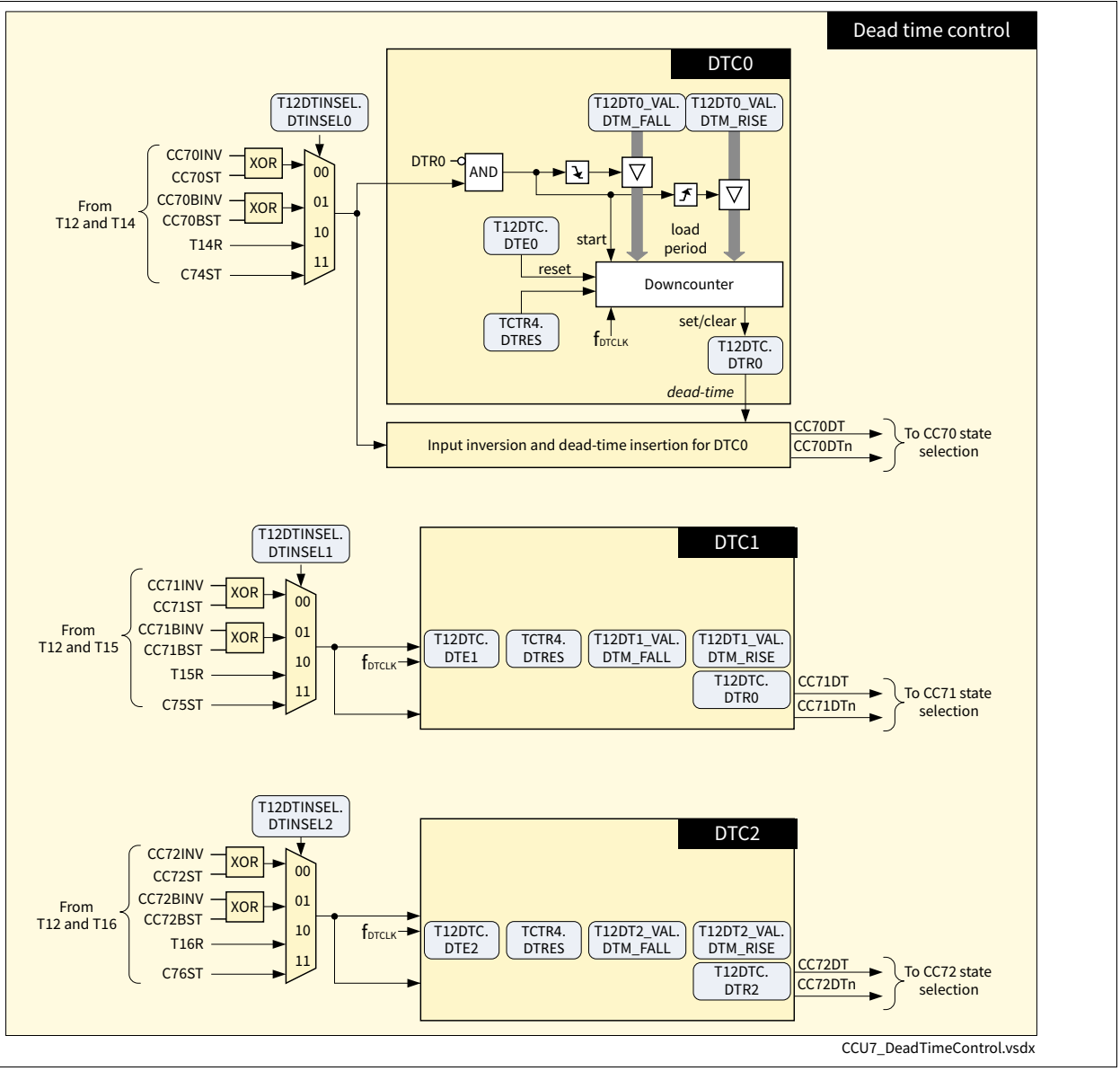


Figure 354 Dead-time control block diagram

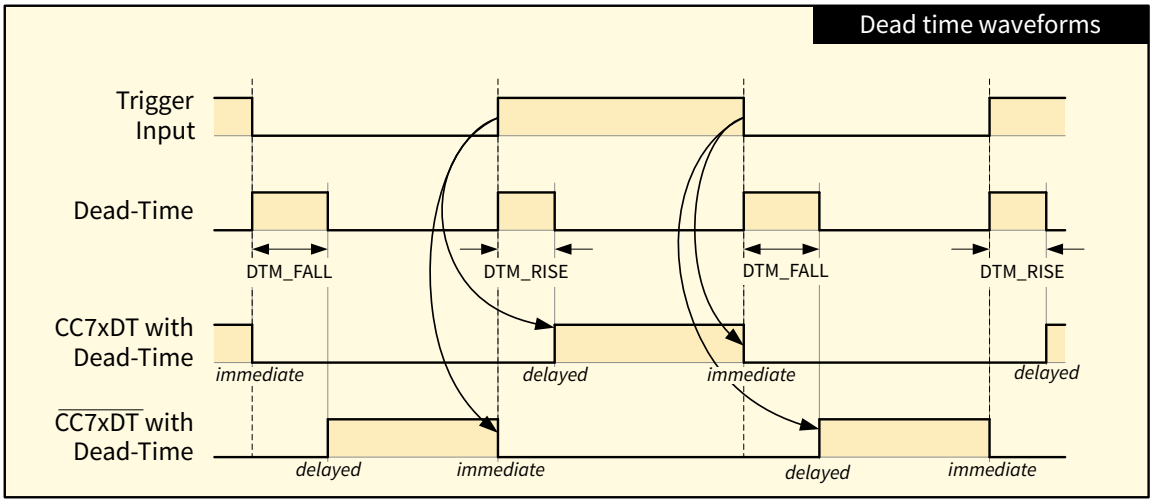


Figure 355 Dead-time generation waveforms

25.11.1 Dead-time generation

The dead-time generation consists of three independent channels, see **Figure 354**. The dead-time generation waveform is shown in **Figure 355**.

Each channel can be individual enabled via **T12DTC**.DTEx.

The state of each dead-time counter can be seen in **T12DTC**.DTRx.

For each dead-time channel, one of four inputs can be selected via **T12DTINSEL**.DTINSELx (see **Chapter 25.11.2**). If the DTCx is enabled, an edge of its input triggers the corresponding DTM_RISE/FALL value to be loaded to its period and starts the counter (DTRx = 1). The counter counts down to zero and stops at zero (DTRx = 0). Once started, a re-trigger is ignored.

Writing **TCTR4**.DTRES = 1 resets the counters and DTRx.

The length of the dead-time (i.e. counting period of the DTCx, DTRx = 1) is applied in the following way:

- If DTCx trigger input signal has a rising edge: T12DTx_VAL.DTM_RISE is loaded to the period
- If DTCx trigger input signal has a falling edge: T12DTx_VAL.DTM_FALL loaded to the period

Each DTCx channel assembles two output signals in the following way:

- CC7xDT and CC7xDTn are complementary signals
- Based on the transition of the trigger signal input a dead-time (DTM_FALL/DTM_RISE) is inserted. The dead-time generation follows following rule:
  - The low to high edge of the control signal (CC7xDT/CC7xDTn) is delayed
  - The high to low edge of the control signal (CC7xDT/CC7xDTn) is not delayed

Due to the dead-time insertion the resulting duty cycle is modified and may be considered for software compensation.

25.11.2 Dead-time input selection

The dead-time input selection defines the input which will be used for the dead-time generation. With **T12DTINSEL**.DTINSELx, the input source for the dead-time generation can be chosen according to **Table 307**. Each dead-time channel has its own input selection.

The state bits CC7xST and CC7xBST can be optionally inverted via setting bits CC7xINV/CC7xBINV. As these bits have a shadow mechanism, it is possible to change the state level synchronous to a hardware event.

Table 307 Dead-time input selection

DTINSEL0 DTINSEL1 DTINSEL2	Selected input for dead-time channel 0	Selected Input for dead-time channel 1	Selected Input for dead-time channel 2
00_B	T12 state bit CC70ST	T12 state bit CC71ST	T12 state bit CC72ST
01_B	T12 state bit C70BST	T12 state bit C71BST	T12 state bit C72BST
10_B	T14 run bit T14R	T15 run bit T15R	T16 run bit T16R
11_B	T14 state bit C74ST	T15 state bit C75ST	T16 state bit C76ST

25.12 Modulation control

The modulation control allows to modulate several CCU7 related signals at a central point. **Figure 356** gives an overview on the signal path from its source to the CCU7 outputs. The output signals CC70/1/2, COUT70/1/2/3 can be modulated by following inputs:

- T12 related channel state bits CC70/1/2ST and C70/1/2BST
- T13 state bit
- T14, T15, T16 related state and run bits
- Hall logic bit CM_CHE
- Multi-channel bits MCMP[5:0]
- Trap control bit TRPS

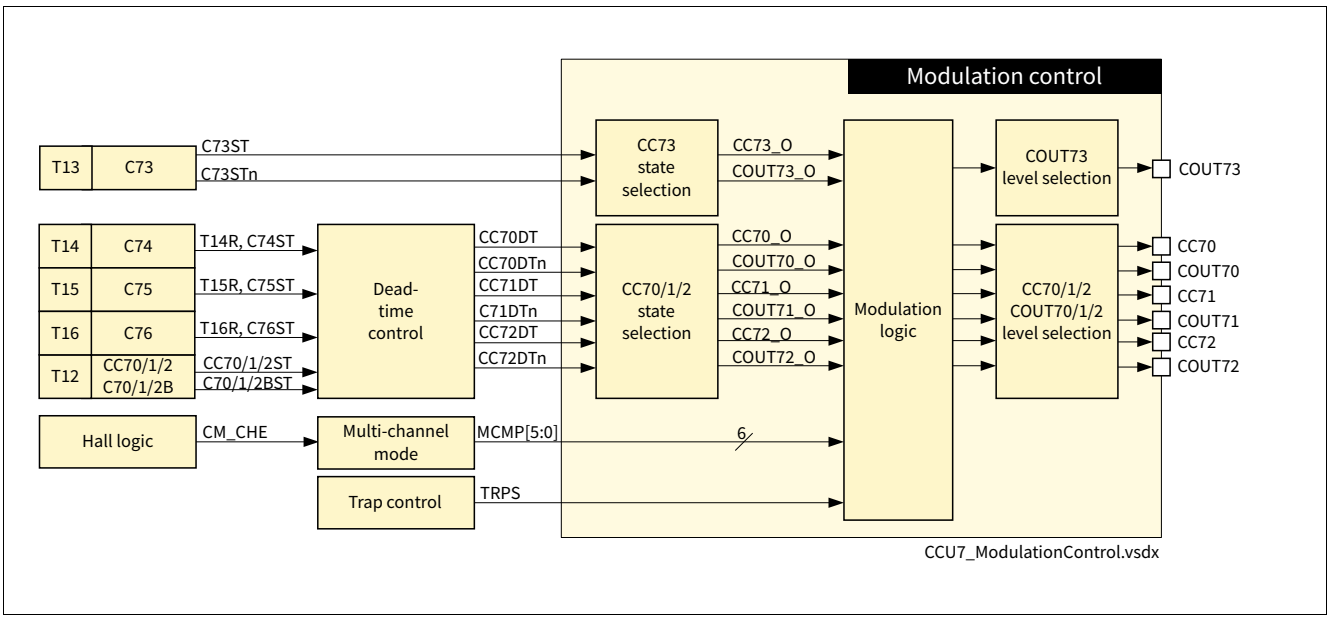


Figure 356 Modulation control overview

The output modulation consists of three blocks:

- State selection
- Modulation logic
- Level selection

25.12.1 State selection for CC70/1/2

The state selection for CC70/1/2 is based on the signals CC7xDT and $\overline{\text{CC7xDT}}$ delivered by the dead-time control (see **Figure 357**). Both signals are never high at the same time, but can be low at the same time. This happens during the dead-time of each compare channel after a change of the corresponding input.

The user can select for each output signal CC7x_O and COUT7x_O, weather the CC7xDT signal or the inverted one (CC7xDTn) is used in the output modulation. The selection is done via **CMPSTAT**.CC7xPS/COUT7xPS.

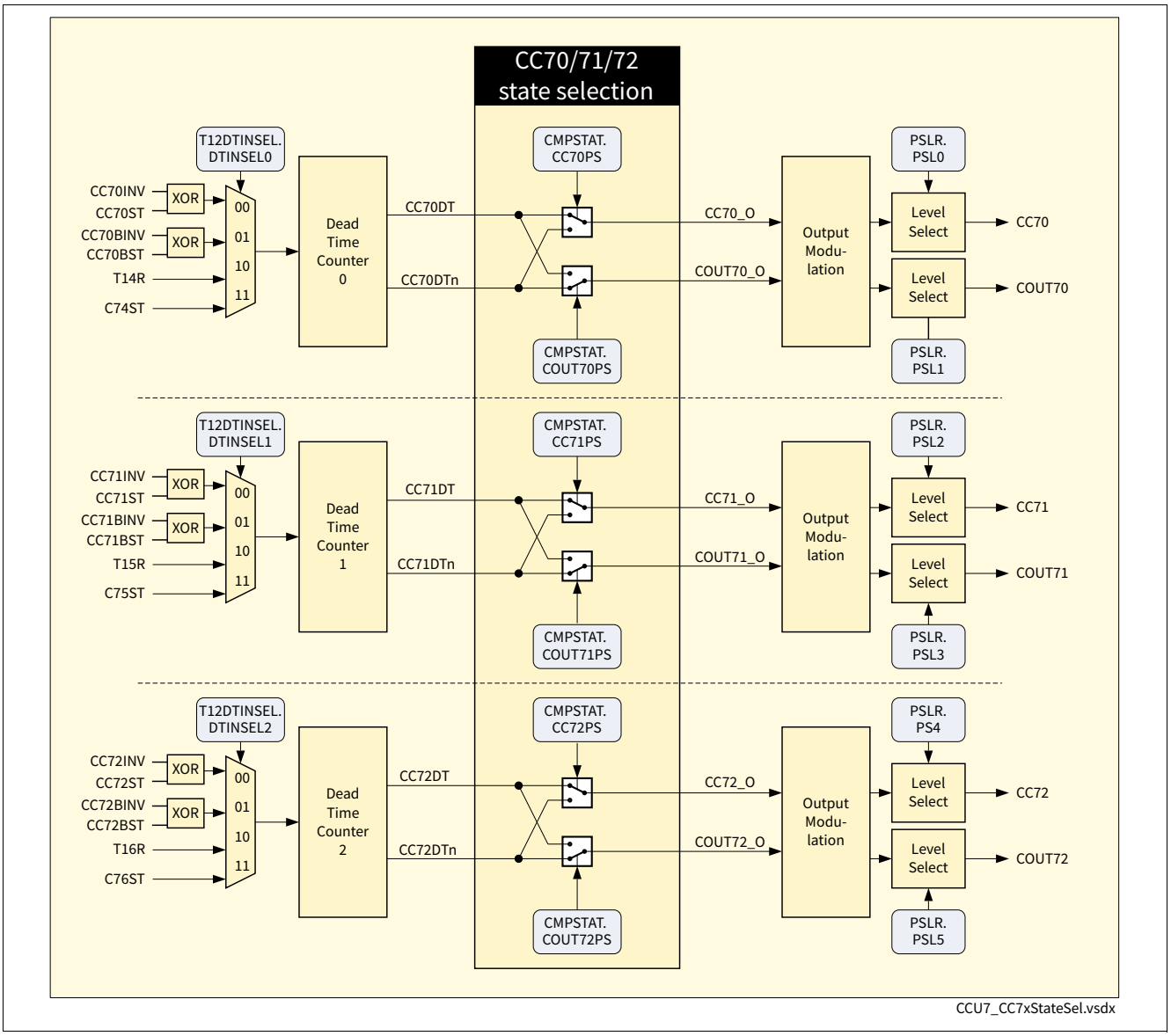


Figure 357 State selection for CC70/1/2_O and COUT70/1/2_O

25.12.2 State selection for CC73

The user can select for each output signal CC73_O and COUT73_O, weather the CC73ST signal or the inverted one (CC73STn) is used in the output modulation. The selection is done via **CMPSTAT**.T13IM/COUT73PS, see **Figure 358**.

The state selection for CC73 is based on T13 state bit **CMPSTAT**.CC73.

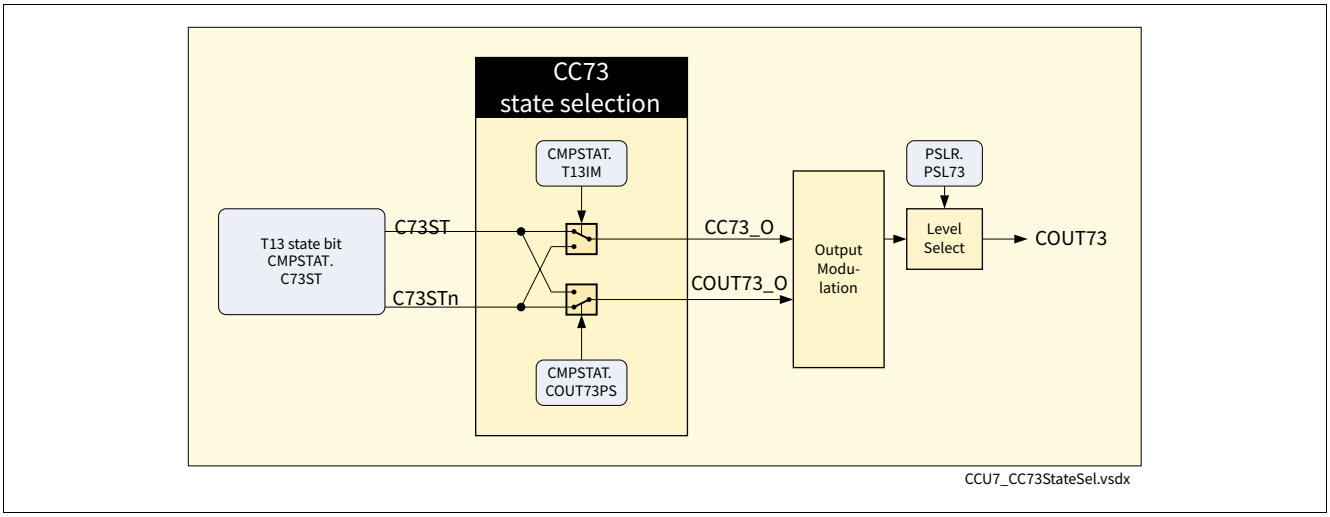


Figure 358 State selection for CC73_O/COUT73_O

25.12.3 Modulation logic

The modulation logic consists of six identical blocks for the channels CC70/1/2 and COUT70/1/2 and a similar block for the channel COUT73. The core of the modulation is a central AND gate for each channel, see **Figure 359**.

The inputs for the modulation logic are following signals:

- CC7x_O signals from the state selection (x = 0 to 3)
- COUT7x_O signals from the state selection (x = 0 to 3)
- MCMP[i] signals from the multi-channel (i = 0 to 5)
- Trap state signal TRPS

Each signal has an individual enable for the modulation.

The output signal of each channel of the modulation logic is connected to the level selection.

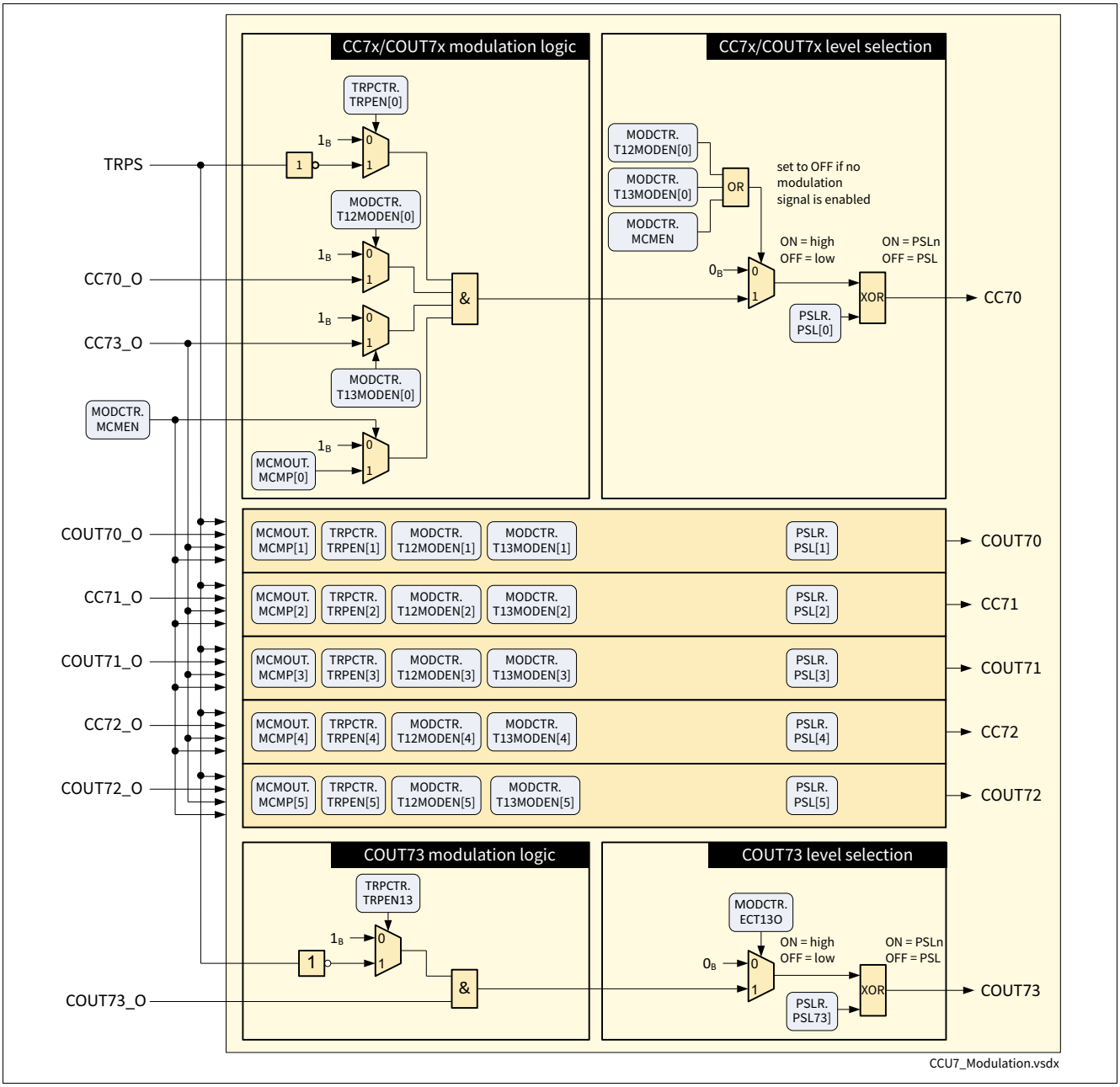


Figure 359 Modulation logic and level selection

25.12.4 Output level selection

It is possible to adapt the CCU7 outputs to the driver stage polarity for a power switch. Therefore the passive level of the CC70/1/2 and COUT70/1/2/3 can be defined. The signals before the modulation logic are handled with positive logic (low = inactive, high = active). A connected driver stage may have high-active or low-active inputs for each individual high-side or low-side switch. The user can define the passive level for each CCU7 output which is connected to the driver stage. This is done via an via individual bit in register **PSLR**. The definition is like following:

- Passive level is the OFF-level of the driver stage
- Active level is the ON-level of the driver stage

The PSL[5:0] and PSL73 bits have shadow registers and are updated with the T12 shadow transfer signal (T12_ST). This allows synchronization of software writes and hardware events and avoids unwanted transitions at the output. A read action to **PSLR** returns the actually used values, whereas a write action targets the shadow bits.

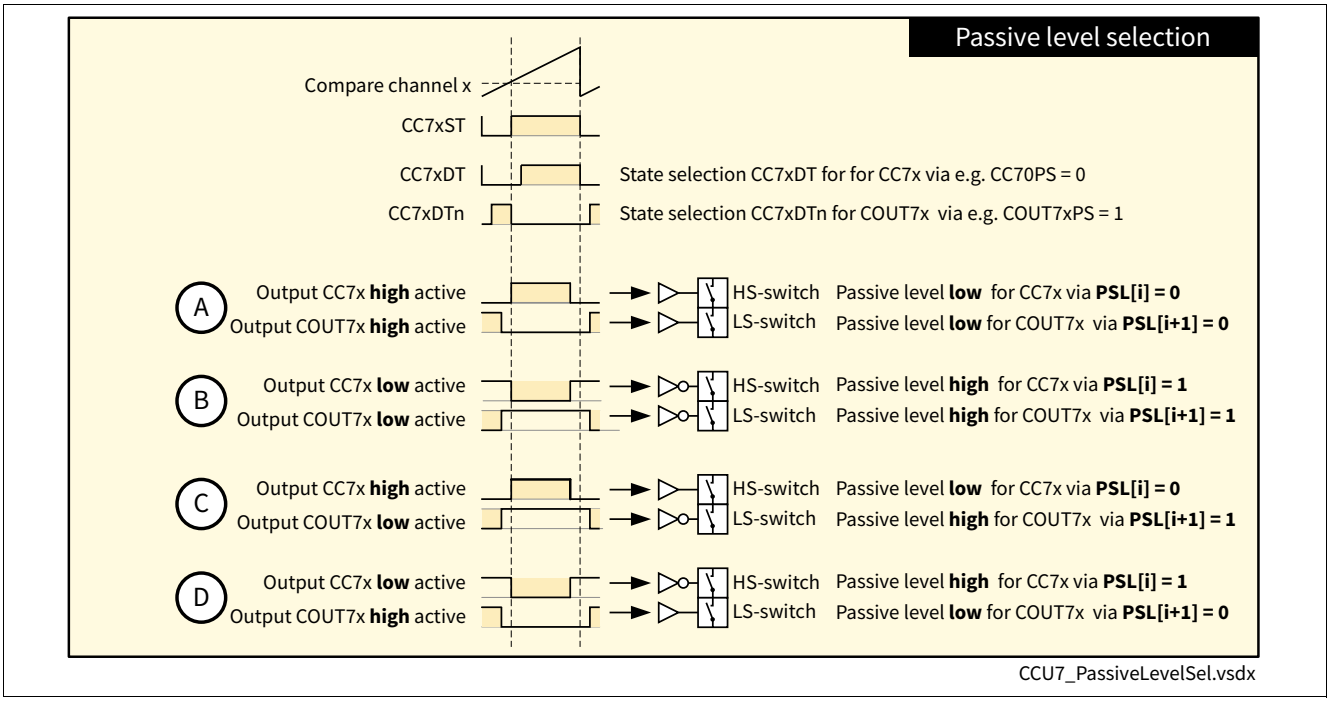


Figure 360 Passive level selection examples

25.13 Input monitoring

A selected event which occurs at a CCU7 input signal can be monitored through **IMON**.x. Each input signal can be included for the detection of a lost bit event (**IMON**.LBE), if enabled through its individual lost indicator enable bits (**LI**.yEN). The lost bit event occurs if a selected event occurs again with the previous event captured (IMON.x remains set) and its lost indicator is enabled for at least one of the monitored input signals. The lost bit event can be enabled (**LI**.LBEEN) for an interrupt to be generated at one of the SRx line, selected through **LI**.INPLBE. The LBE output signal of the kernel can be connected to a capture input to indicate when the lost bit event happens.

The lost bit event can be used as a kind of interrupt or event watchdog to monitor if an action related to an event has been processed before a second event of the same type occurs. Like this, if a certain event is treated by an interrupt that should be monitored, the related indication flag has to be cleared by SW. If the SW has not yet cleared the flag and the event occurs again, the event is considered as being lost and another interrupt can be generated to inform the system about the loss. This can be also used to indicate that input events occur too often and the main task has not enough time to treat them.

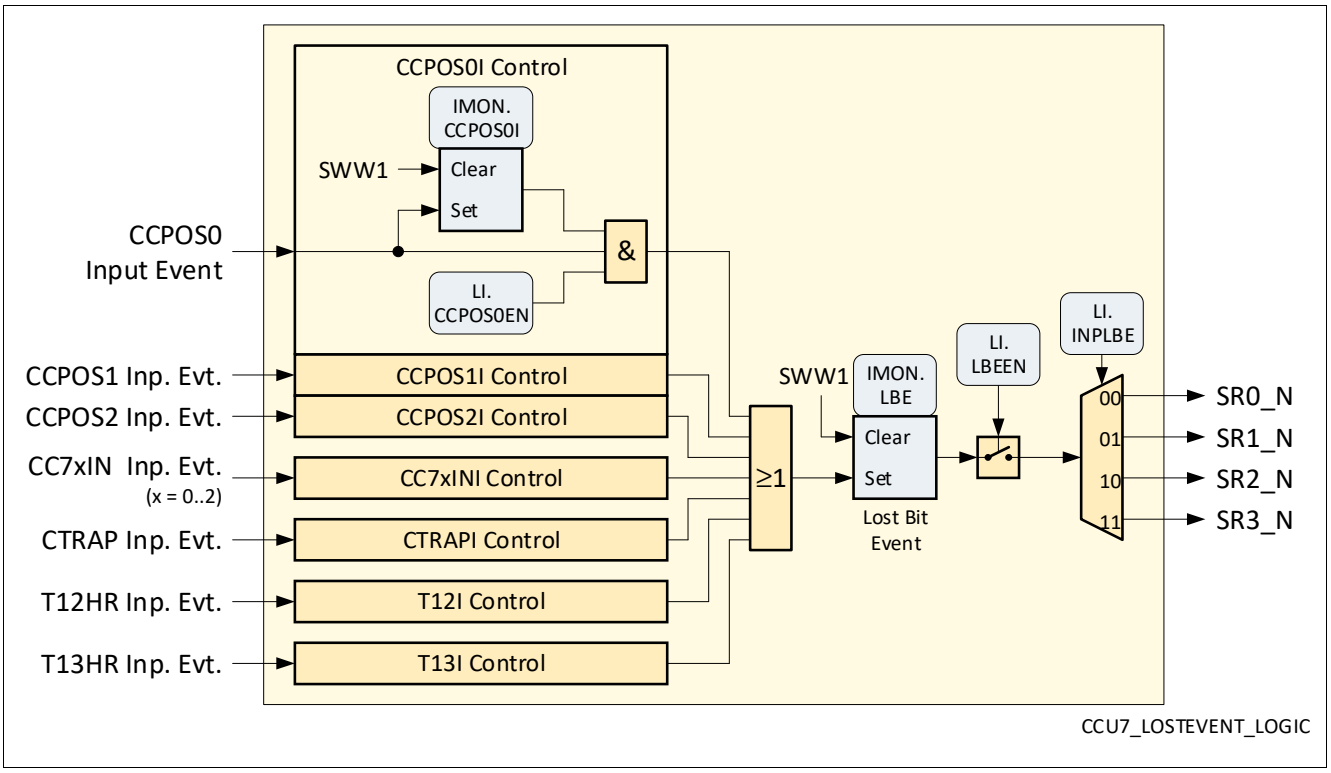


Figure 361 Lost event logic

25.14 Debug support

The behavior of CCU7 upon an On-chip-debug-support (OCDS) suspend request is controlled by the **OCS** register. CCU7 supports both Hard suspend mode and Soft suspend mode.

Hard suspend mode

In Hard suspend mode the CCU7 kernel clock is switched off immediately. Reading and writing of registers is possible but will enable the kernel clock for a few cycles.

Attention: ***Register accesses with clocking in Hard suspend mode can have unintended side effects like signals becoming and staying active. This can affect also other modules, so a CCU7 kernel reset might not be sufficient to bring the system into a defined state.***

Soft suspend mode

In Soft suspend mode CCU7 may finalize specific actions before it enters the suspended state with **OCS**.SUSSTA set. This can be advantageous for applications (e.g. motor control) where critical system states could occur if the kernel clock would be switched off immediately upon an OCDS suspend request.

Soft suspend mode does not influence the kernel clock. Reading and writing of registers is possible without the side effects that may occur when reading/writing registers in Hard suspend mode.

For CCU7, two basic soft suspend options (**Stop mode 0**, **Stop mode 1**) are available, selected via bit field **OCS**.SUS.

In addition, the internal functional blocks (T12, T13, T14, T15, T16, Hall logic, Trap logic) may individually be programmed via their Sensitivity Bits in register **KSCSR** to accept or ignore a suspend request. If the request sensitivity is disabled, the block continues normal operation. If the request sensitivity is enabled, the block operates as specified for the selected stop mode. The mapping of the CCU7 functional blocks to their Sensitivity Bits is shown in **Table 308**.

Stop mode 0

In Stop mode 0, if selected to be stopped, the Hall and Trap logic stops immediately, while timer T12 and/or T13 continues normal operation (if running) until it reaches the end of the PWM period; then it stops (same stop condition as in Single-shot mode). When timer T12 stops, the capture inputs CC7xIN are frozen.

Stop mode 1

In Stop mode 1, if selected to be stopped, the internal functional blocks (T12, T13, T14, T15, T16, Hall logic, Trap logic) will stop immediately upon a suspend request.

If the sensitivity bits for timer T12 or T13 are set, the corresponding output lines enabled for the trap condition are set to their passive values (similar to a trap condition).

Table 308 summarizes the reaction of the CCU7 functional blocks to OCDS suspend requests.

Table 308 CCU7 functional blocks

Block	Reaction to OCDS suspend request	Sensitivity bit
0	Timer T12: if bit SB0 = 1_B: • Stop mode 0: timer T12 continues (if running) until the end of the PWM period. Then it stops and the CCxIN input stages are frozen. • Stop mode 1: timer T12 stops immediately and the CC7xIN input stages are frozen. Output lines CC7x, COUT7x with enabled trap functionality are set to their passive states.	KSCSR.SB0
1	Timer T13: if bit SB1 = 1_B: • Stop mode 0: timer T13 continues (if running) until the end of the PWM period. Then it stops. • Stop mode 1: timer T13 stops immediately. If trap functionality for COUT63 is enabled, it is set to the passive state.	KSCSR.SB1
2	Timer T14: if bit SB2 = 1_B: • Stop mode 0: timer T14 continues (if running) until the end of the PWM period. Then it stops. • Stop mode 1: timer T14 stops immediately.	KSCSR.SB2
3	Timer T15: if bit SB3 = 1_B: • Stop mode 0: timer T15 continues (if running) until the end of the PWM period. Then it stops. • Stop mode 1: timer T15 stops immediately.	KSCSR.SB3
4	Timer T16: if bit SB4 = 1_B: • Stop mode 0: timer T16 continues (if running) until the end of the PWM period. Then it stops. • Stop mode 1: timer T16 stops immediately.	KSCSR.SB4
5	Hall logic: if bit SB5 = 1_B • Hall logic is stopped immediately and the CCPOSx input stage is frozen (same behavior for Stop mode 0 and 1).	KSCSR.SB5
6	Trap logic: if bit SB6 = 1_B • Trap logic is stopped immediately and the CTRAP input stage is frozen (same behavior for Stop mode 0 and 1).	KSCSR.SB6

25.15 Programmer's guide

25.15.1 Initialization sequence (for example "Asymmetrical SVM using T12, T13")

The example in **Chapter 25.15.3** uses the initialization sequence in **Table 309**.

Table 309 Initialization sequence for example "Asymmetrical SVM using T12, T13"

Step	Description	Option	Selection/criteria	Configuration example (pseudo code)
1	CCU7 inputs	optional	CCx, CCPOSx, CTRAP, T12HR config T13HR, T12/3CNT, T12/13EXT T1xHR, T1xCNT, T1xEXT	PISEL0 = 0x40; /* use CTRAPB (P2.6) */ PISEL2 = 0; PISEL24/25/26 = 0;
2	CCU7 clock	optional	clock config	T_FDIV0 = 0; T_FDIV1 = 0; T_CLK_CTRL = 0; /* default clock for all timers fpwm = fsys0 */
3	Timer control register	manda-tory	T12 config T13 config T14/5/6 config multi-channel n. u. capture/compare trap config	TCTR0 = 0; /* T12 in center-aligned */ TCTR2 = 0x7A; /* T13 single shot on T12_ZM*/ TCTR24/25/26 = 0; /* not used*/ MCMCTR = 0; /* not used */ T12MSEL = 0x333; /* standard compare mode */ TRPCTR = 0xBF00; /* trap for CC7x/COUT7x enabled, trap exit with TRPF=0 and T12_ZM and CTRAP=1*/ MODCTR = 0x3F; /* modulation enabled for CC7x/COUT7x */
4	State and level selection	manda-tory	adapt to half-bridge driver inputs	CMPSTAT = 0x6A00; /* complementary states: CC7x=0, COUT7x=1 */ PSLR = 0; /* passive level is low for all channels*/
5	Timer period	manda-tory	T12 period T13 period T14 period T15 period T16 period	T12PR = 0x05DB; /* 3000 ticks*/ T13PR = 0x07BB; /* 1979 ticks*/ T14PR = 0; /* not used*/ T15PR = 0; /* not used*/ T16PR = 0; /* not used*/
6	Compare values	manda-tory	compare duty cycles in CC7x/CC7Bx/C7y shadow register	CC70SR = 0x02EE; /* set start duty cycle to 50% */ CC71SR = 0x02EE; /* set start duty cycle to 50% */ CC72SR = 0x02EE; /* set start duty cycle to 50% */ CC70BSR = 0x02EE; /* set start duty cycle to 50% */ CC71BSR = 0x02EE; /* set start duty cycle to 50% */ CC72BSR = 0x02EE; /* set start duty cycle to 50%*/ C73SR = 0x01E0; /* 480 ticks*/ C74/75/76SR = 0; /* not used*/
7	Initial shadow transfer	manda-tory		TCTR3 =0x3; /* STR12 =STR13 =1 */

Table 309 Initialization sequence for example "Asymmetrical SVM using T12, T13" (cont'd)

Step	Description	Option	Selection/criteria	Configuration example (pseudo code)
8	Dead-time	optional	dead-time config	T12DTC = 0x7; /* all channels enabled*/ T12DT0_VAL = 0x5A3C; /* fall=90, rise = 60 ticks*/ T12DT1_VAL = 0x5A3C; /* fall=90, rise = 60 ticks*/ T12DT2_VAL = 0x5A3C; /* fall=90, rise = 60 ticks*/ T12DTINSEL = 0x0111; /* use C7xBST for DT gen*/
9	Lost indicator	optional	lost indicator config	LI = 0; /* not used */
10	Interrupts	optional	status flags interrupt gating node pointer interrupt enable	ISR = 0xF7FF; /* clear all flags if any */ ISR_2 = 0x3F3F; /*clear all flags if any */ IGT = 0; /* no interrupt gating */ INP = 0; INP_2 = 0; IEN = 0; /* here or after device init */ IEN_2 = 0;
11	Debug behavior	optional	stop behavior of CCU7 after debug request	OCS = 0; /* hall and trap immediate, timers at period*/ KSCSR = 0; /* behavior according to OCS setting*/

Additionally the CCU7 module interconnects have to be configured accordingly.

25.15.1.1 CCU7 input configuration

CTRAPB is mapped to GPIO P2.6, hence GPIO setting must be input with e.g. pull-up enabled:

- P2_PUD = 0x0040 0040; /* P2.6 is input by default, pull-up enabled */

25.15.1.2 CCU7 output configuration

CC70/71/72 and COUT70/71/72 are used and must be connected to e.g. GPIOs:

- P0_OMR &= ~0x0308; /* write passive level */
- P0_DIR = 0x0308; /* COUT70 - P0.3, CC72 - P0.9, COUT72 - P0.8 */
- P0_ALTSEL0 |= 0x3000; /* select alternate output function */
- P0_ALTSEL1 |= 0x0011; /* select alternate output function */
- P1_OMR = 0x0007 0000; /* write passive level */
- P1_DIR |= 0x07; /* CC70 - P1.2, CC71 - P1.0, CC72 - P1.1 */
- P1_ALTSEL0 |= 0x0222; /* select alternate output function */

25.15.1.3 CCU7 interrupt configuration

Enable CCU7 interrupts in CCU7 registers:

- IEN |= 0x40; /* T12_OM enabled */

NVIC configuration:

- NVIC_ISER |= 0x04; /* CCU7_INP0 --> NVIC_IRQ2*/

25.15.1.4 Start timer in CCU7 registers

Start timer T12:

- CCU7 registers: TCTR4 |= 0x2; /* start T12 via T12RS =1*/

25.15.1.5 Interrupt service routine

- CC70SR = varCompR0; /* update duty cycle CC70 */
- CC71SR = varCompR1; /* update duty cycle CC71 */
- CC72SR = varCompR2; /* update duty cycle CC72 */
- CC70BSR = varCompF0; /* update duty cycle CC70 */
- CC71BSR = varCompF1; /* update duty cycle CC71 */
- CC72BSR = varCompF2; /* update duty cycle CC72 %*/
- C73SR = varADC1st; /* update ADC single shunt first trigger */
- T13PR = varADC2nd; /* update ADC single shunt second trigger */
- TCTR3 =0x3; /* request shadow transfer STR12 =STR13 =1 */

25.15.2 Symmetrical space vector modulation example using T12, T13

The example shown in **Figure 362** is a 3-phase waveform which is symmetrical to its center point (here T12_PM). It uses T12 compare channels CC70/1/2 for the PWM generation and T13 compare CM_73 and period-match T13_PM for ADC triggers. This example is a typical configuration for sinusoidal PWM modulation for BLDC motors with a single shunt current measurement, aka sensorless space vector modulation (SVM).

The CCU7 configuration be taken from **Table 310**.

Table 310 Symmetrical PWM using T12, T13

Used timer	Timer mode	Timer start via	Active time via	ADC trigger via	Shadow transfer via	Interrupt e.g.
T12	center-aligned	SW, T12RS=1	CC70ST CC71ST CC72ST	–	T12_PM and T12_OM	SR[0] = T12_ZM
T13	single-shot	HW trigger, SR[0] = T12_ZM	–	CM_73 T13_PM	–	–

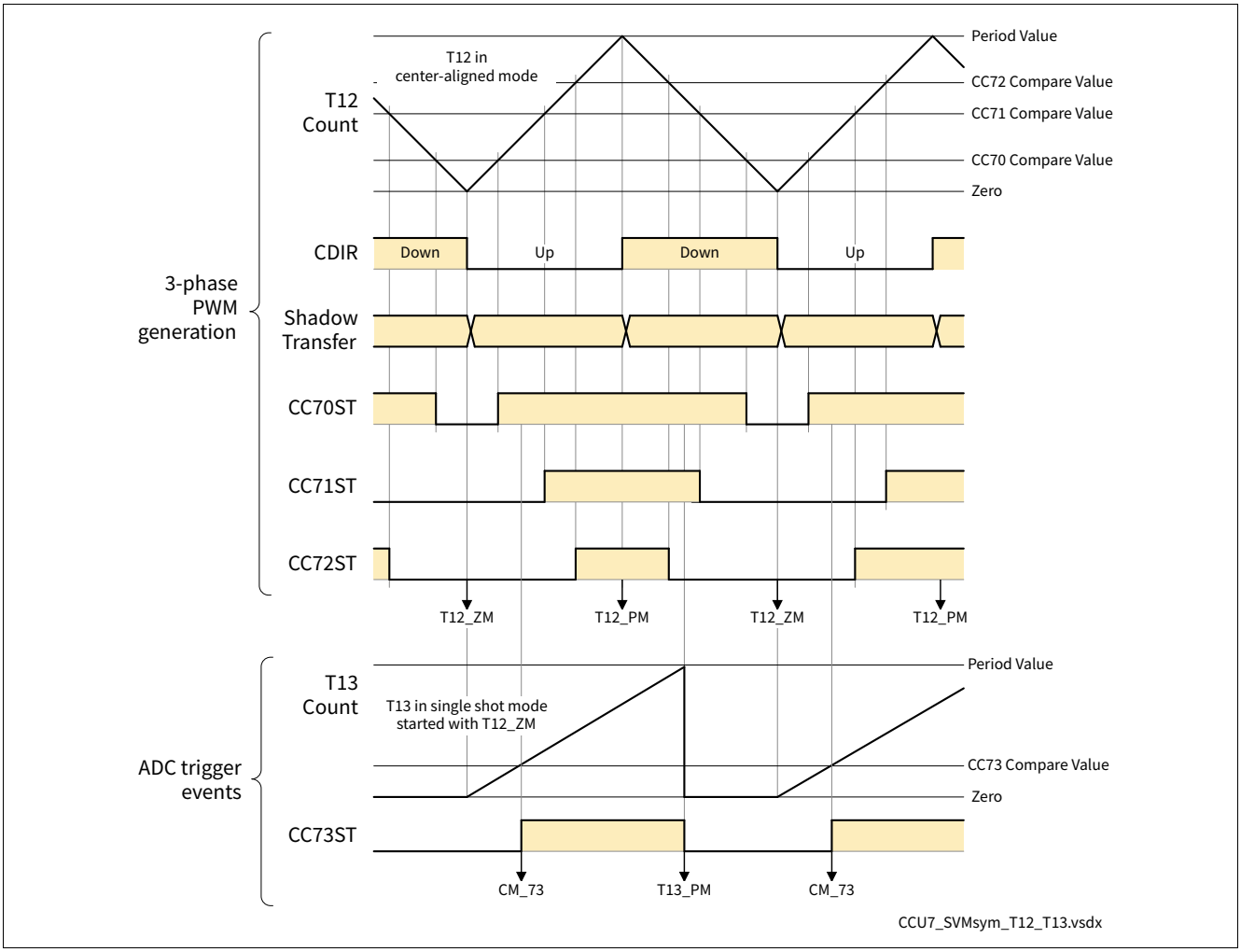


Figure 362 Symmetrical space vector modulation for single shunt applications with T12, T13

25.15.3 Asymmetrical space vector modulation example using T12, T13

The example shown in **Figure 363** is a 3-phase waveform which is asymmetrical to its center point (here T12_PM). It uses T12 compare channels CC70/1/2 and C70B/71B/72B for the PWM generation and T13 compare CM_73 and period-match T13_PM for ADC triggers. This example is a typical configuration for sinusoidal PWM modulation for BLDC motors with a single shunt current measurement, aka sensorless space vector modulation (SVM).

Using an asymmetrical pattern it is possible to reduce the dead-zone when the single-shunt current cannot be measured because the time of the current pulse is too short, i.e. two compare values are very close together (space vector angle close to the 60° segment). In the example the C72BST has the same starting point as C71BST but ends earlier, hence it is shifted towards left. This makes the single shunt current pulse longer (none overlapping time of C71BST and C72BST).

The CCU7 configuration be taken from **Table 310**.

Table 311 Symmetrical PWM using T12, T13

Used timer	Timer mode	Timer start via	Active time via	ADC trigger via	Shadow transfer via	Interrupt, e.g.
T12	center-aligned	SW, T12RS=1	CC70BST CC71BST CC72BST	–	T12_PM and T12_OM	SR[0] =T12_ZM
T13	single-shot	HW trigger, T12_ZM	–	CM_73 T13_PM	–	–

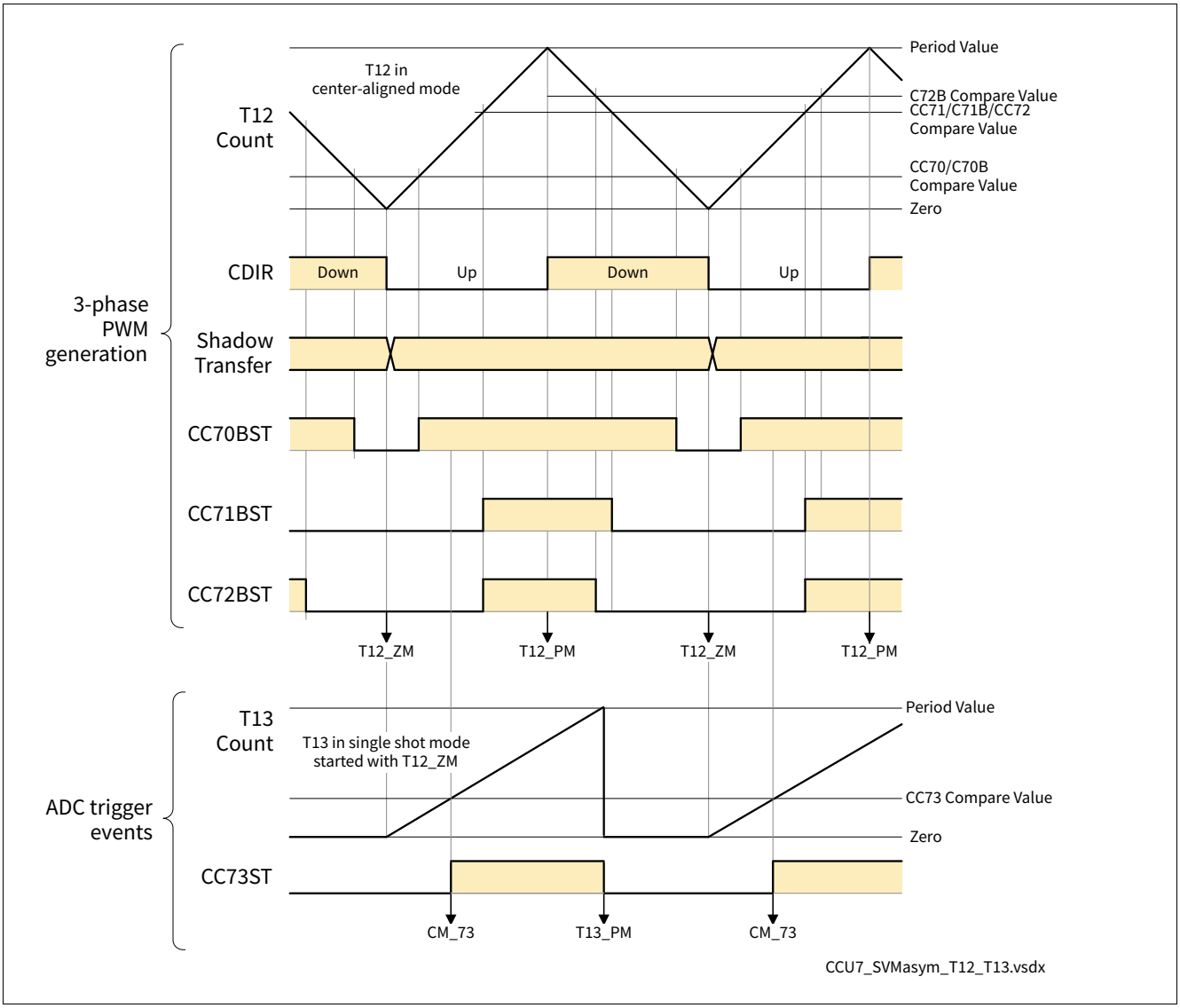


Figure 363 Asymmetrical space vector modulation for single shunt application with T12, T13

25.15.4 Asymmetrical space vector modulation example using T12, T14, T15, T16

The example shown in **Figure 364** is a 3-phase waveform which is asymmetrical to its center point (here T12_PM). It uses T12 compare channels CC70/1/2 and the run bits of T14, T15 and T16 for the PWM generation and compare events CM_74/75/76 for ADC triggers.

This example is a typical configuration for sinusoidal PWM modulation for BLDC motors with a single shunt current measurement, aka sensorless space vector modulation (SVM). Using the run bits of T14, T15 and T16 it is possible to create arbitrary pulses within the T12 period, even none-overlapping pulses are possible.

Using such an asymmetrical pattern it is possible to eliminate the dead-zone when the single-shunt current cannot be measured because the time of the current pulse is too short, i.e. two compare values are very close together (space vector angle close to the 60° segment corner and/or its amplitude is low). In the example the second generated pulse of T16 is within the up-counting T12 phase, i.e. far left shifted.

The T12_ZM is used as interrupt request event, which gives the software control loop a full T12 period for its calculation. The hardware shadow transfer is done at T12_OM event.

The CCU7 configuration be taken from **Table 312**.

Table 312 Symmetrical PWM using T12, T14, T15, T16

Used timer	Timer mode	Timer start via	Active time via	ADC trigger via	Shadow transfer via	Interrupt, e.g.
T12	center-aligned	SW, T12RS=1	–	–	T12_OM	SR[0] = T12_ZM
T14	single-shot	HW trigger, SR[0] = T12_ZM	T14R	CM_74	–	–
T15	single-shot	HW trigger, SR[0] = T12_ZM	T15R	CM_75	–	–
T16	single-shot	HW trigger, SR[0] = T12_ZM	T16R	CM_76	–	–

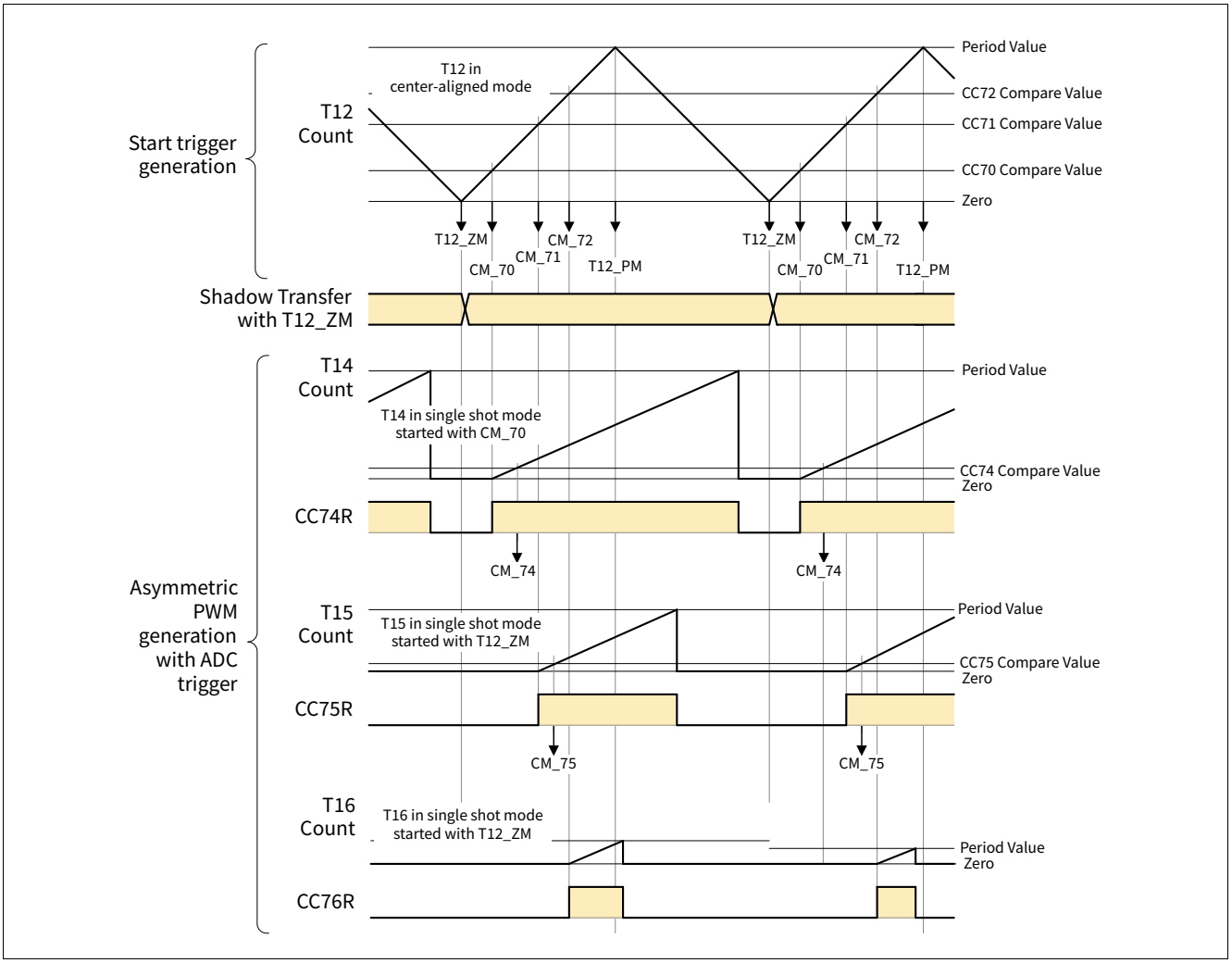


Figure 364 Asymmetrical space vector modulation for single shunt application with T12, T14, T15, T16

25.16 Register description CCU7

25.16.1 CCU7 Address Maps

Table 313 Register Address Space - CCU7

Module	Base Address	End Address	Note
CCU7	40008000_H	$4000BFFF_H$	

Table 314 Register Overview - CCU7 (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CLC	Clock Control Register	0000_H	1034
MCFG	Module Configuration Register	0004_H	1034
ID	Module Identification Register	0008_H	1035
PISEL0	Port Input Select Register 0	$000C_H$	1036
PISEL2	Port Input Select Register 2	0010_H	1038
PISEL2x	Port Input Select Register 2x	0014_H+(x-4)*4	1039
KSCSR	Kernel State Control Sensitivity Register	0020_H	1040
T12	Timer T12 Counter Register	0024_H	1041
T12PR	Timer T12 Period Register	0028_H	1041
T12DTC	Dead-Time Control Register for Timer T12 Low	$002C_H$	1042
T12DTx_VAL	Dead-Time value Register CC7x	0030_H+x*4	1044
T12DTINSEL	Dead-Time Input Selection Register (Demo in Windows)	$003C_H$	1045
CC7xR	Capture/Compare Register for Channel CC7x	0040_H+x*4	1045
CC7xSR	Capture/Compare Shadow Reg. for Channel CC7x	$004C_H$+x*4	1046
CC7xBR	Compare Register for Channel C7xB	0058_H+x*4	1047
CC7xBSR	Compare Shadow Reg. for Channel C7xBSR	0064_H+x*4	1047
T1xR	Timer T1x Counter Register	0070_H+(x-3)*4	1048
T1xPR	Timer T1x Period Register	0080_H+(x-3)*4	1048
C7xR	Compare Register for Channel C7x	0090_H+(x-3)*4	1049
C7xSR	Compare Shadow Reg. for Channel CC7x	$00A0_H$+(x-3)*4	1049
CMPSTAT	Compare State Register	$00B0_H$	1050
CMPSTAT_2	Compare State Register 2	$00B4_H$	1054
CMPMODIF	Compare State Modification Register	$00B8_H$	1056
CMPMODIF_2	Compare State Modification Register 2	$00BC_H$	1061
T12MSEL	T12 Capture/Compare Mode Select Register	$00C0_H$	1064
TCTR0	Timer Control Register 0	$00C4_H$	1065
TCTR1	Timer Control Register 1	$00C8_H$	1066

Table 314 Register Overview - CCU7 (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Page Number
TCTR2	Timer Control Register 2	$00CC_H$	1068
TCTR2x	Timer Control Register 2x	$00D0_H$+(x-4)*4	1070
TCTR4	Timer Control Register 4	$00DC_H$	1071
TCTR4x	Timer Control Register 4x	$00E0_H$+(x-4)*4	1072
MODCTR	Modulation Control Register	$00EC_H$	1073
TRPCTR	Trap Control Register	$00F0_H$	1074
PSLR	Passive State Level Register	$00F4_H$	1076
MCMOUTS	Multi-Channel Mode Output Shadow Register	$00F8_H$	1077
MCMOUT	Multi-Channel Mode Output Register	$00FC_H$	1078
MCMCTR	Multi-Channel Mode Control Register	0100_H	1080
IMON	Input Monitoring Register	0104_H	1082
LI	Lost Indicator Register	0108_H	1083
IS	Capture/Compare Interrupt Status Register 1	$010C_H$	1085
ISS	Capture/Compare Interrupt Status Set Register 1	0110_H	1088
ISR	Capture/Compare Interrupt Status Reset Register 1	0114_H	1089
INP	Capture/Compare Interrupt Node Pointer Register 1	0118_H	1091
IEN	Capture/Compare Interrupt Enable Register 1	$011C_H$	1092
IS_2	Capture/Compare Interrupt Status Register 2	0120_H	1095
ISS_2	Capture/Compare Interrupt Status Set Register 2	0124_H	1097
ISR_2	Capture/Compare Interrupt Status Reset Register 2	0128_H	1098
INP_2	Capture/Compare Interrupt Node Pointer Register 2	$012C_H$	1099
IEN_2	Capture/Compare Interrupt Enable Register 2	0130_H	1101
OCS	OCDS Control and Status Register	0134_H	1103
T_FDIVx	Fractional Dividerx	0138_H+x*4	1104
T_CLK_CTRL	Timer Clock Selection	0140_H	1104
TCTR3	Timer Control Register 3	0144_H	1105
IGT	Interupt gating Register	0148_H	1107

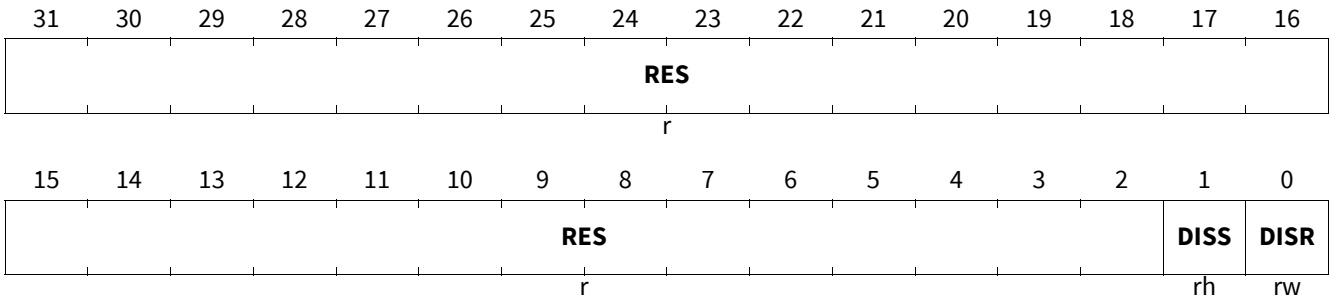
25.16.2 CCU7 Registers

Clock Control Register

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. Register CLC controls the module clock signal and the reactivity to the sleep signal.

Note: Upon an accepted Sleep Mode request (with EDIS = '1'), orupon a disable request (DISR = '1'), the CCU7 kernel clock is switched offimmediately. Therefore, software should ensure that the system controlled by the CCU7 kernelhas reached a safe state before triggering a Sleep Mode or module disable request.

CLC
Clock Control Register **(0000_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. 0_B Module disable is not requested. 1_B Module disable is requested.
DISS	1	rh	**Module Disable Status Bit** Bit indicates the current status of the module. 0_B Module is enabled. 1_B Module is disabled.
RES	15:2, 31:16	r	**Reserved - 0** Read as 0; should be written with 0.

Module Configuration Register

The module configuration register contains bits describing the functionality that is available in the CCU7 module.

MCFG

Module Configuration Register **(0004_H)** **RESET_TYPE_5 Value: $0000\ 8077_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WREN	RES								T16	T15	T14	RES	MCM	T13	T12
rw	r								rw	rw	rw	r	rw	rw	rw

Field	Bits	Type	Description
T12	0	rw	**T12 Available** This bit indicates if the T12 block is available. 0_B The T12 block is not available. A write access to T12PR is ignored. 1_B The T12 block is available. A write access to T12PR is executed.
T13	1	rw	**T13 Available** This bit indicates if the T13 block is available. 0_B The T13 block is not available. A write access to T13PR is ignored. 1_B The T13 block is available. A write access to T13PR is executed.
MCM	2	rw	**Multi-Channel Mode Available** This bit indicates if the multi-channel mode functionality is available. 0_B The multi-channel mode functionality is not available. A write access to MCMOUTS is ignored. 1_B The multi-channel mode functionality is available. A write access to MCMOUTS is executed.
RES	3, 14:7, 31:16	r	**Reserved; - 0** read as 0; should be written with 0.
T14	4	rw	**T14 Available -T14** 0_B The T14 block is not available. A write access to T14PR is ignored. 1_B The T14 block is available. Awrite access to T14PR is executed.
T15	5	rw	**T15 Available** 0_B The T15 block is not available. Awrite access to T15PR is ignored. 1_B The T15 block is available. Awrite access to T15PR is executed.
T16	6	rw	**T16 Available** 0_B The T16 block is not available. Awrite access to T16PR is ignored. 1_B The T16 block is available. Awrite access to T16PR is executed.
WREN	15	rw	**Write Enable** This bit enables/disabled the write capability of this register. 0_B Write accesses to this register are ignored. 1_B This register can be written (default).

Module Identification Register

The CCU7 Module Identification Register ID containsread-only information about the module identification number and its revision.

ID

Module Identification Register **(0008_H)** **RESET_TYPE_5 Value: $0000\ 0701_H$**

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

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

Field	Bits	Type	Description
MODREV	7:0	r	**Module Revision Number** MODREV defines the module revision number. The value of a module revision starts with 01_H (first revision), 02_H, 03_H,…up to FF_H.
MODNUM	15:8	r	**Module Number Value** This bit field defines the module identification number for the CCU7:07_H
RES	31:16	r	**Reserved - 0** Read as 0.

Port Input Select Register 0

PISEL0

Port Input Select Register 0 **($000C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IST12HR		ISPOS2		ISPOS1		ISPOS0		ISTRP		ISCC72		ISCC71		ISCC70	
rw		rw		rw		rw		rw		rw		rw		rw	

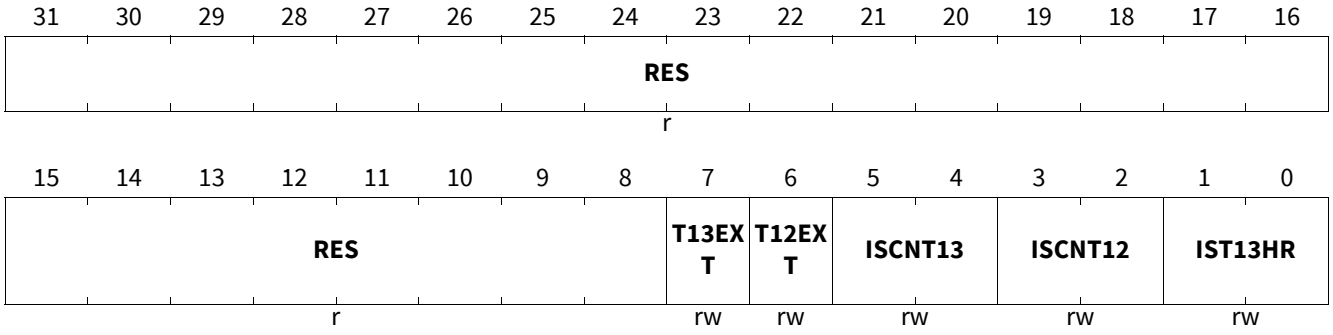
Field	Bits	Type	Description
ISCC70	1:0	rw	**Input Select for CC70** This bit field defines the port pin that is used for the CC70 capture input signal. 00_B **CC70_0**, The input CC70INA is selected 01_B **CC70_1**, The input CC70INB is selected 10_B **CC70_2**, The input CC70INC is selected 11_B **CC70_3**, The input CC70IND is selected

Field	Bits	Type	Description
ISCC71	3:2	rw	**Input Select for CC71** This bit field defines the port pin that is usedfor the CC71 capture input signal. 00_B **CC71_0**, The input CC71INA is selected 01_B **CC71_1**, The input CC71INB is selected 10_B **CC71_2**, The input CC71INC is selected 11_B **CC71_3**, The input CC71IND is selected
ISCC72	5:4	rw	**Input Select for CC72** This bit field defines the port pin that is usedfor the CC72 capture input signal. 00_B **CC72_0**, The input CC72INA is selected 01_B **CC72_1**, The input CC72INB is selected 10_B **CC72_2**, The input CC72INC is selected 11_B **CC72_3**, The input CC72IND is selected
ISTRP	7:6	rw	**Input Select for CTRAP** This bit field defines the port pin that is usedfor the $\overline{\text{CTRAP}}$ input signal. 00_B **CTRAP_0**, The input CTRAPA is selected 01_B **CTRAP_1**, The input CTRAPB is selected 10_B **CTRAP_2**, The input CTRAPC is selected 11_B **CTRAP_3**, The input CTRAPD is selected
ISPOS0	9:8	rw	**Input Select for CCPOS0** This bit field defines the port pin that is usedfor the CCPOS0 input signal. 00_B **CCPOS0_0**, The input pin for CCPOS0_0. 01_B **CCPOS0_1**, The input pin for CCPOS0_1. 10_B **CCPOS0_2**, The input pin for CCPOS0_2. 11_B **CCPOS0_3**, The input pin for CCPOS0_3.
ISPOS1	11:10	rw	**Input Select for CCPOS1** This bit field defines the port pin that is usedfor the CCPOS1 input signal. 00_B **CCPOS1_0**, The input pin for CCPOS1_0. 01_B **CCPOS1_1**, The input pin for CCPOS1_1. 10_B **CCPOS1_2**, The input pin for CCPOS1_2. 11_B **CCPOS1_3**, The input pin for CCPOS1_3.
ISPOS2	13:12	rw	**Input Select for CCPOS2** This bit field defines the port pin that is usedfor the CCPOS2 input signal. 00_B **CCPOS2_0**, The input pin for CCPOS2_0. 01_B **CCPOS2_1**, The input pin for CCPOS2_1. 10_B **CCPOS2_2**, The input pin for CCPOS2_2. 11_B **CCPOS2_3**, The input pin for CCPOS2_3.

Field	Bits	Type	Description
IST12HR	15:14	rw	**Input Select for T12HR** This bit field defines the input signal used asT12HR input. 00_B **T12HRA**, Either signal T12HRA (if T12EXT = 0) or T12HRE (if T12EXT = 1) is selected. 01_B **T12HRB**, Either signal T12HRB (if T12EXT = 0) or T12HRF (if T12EXT = 1) is selected. 10_B **T12HRC**, Either signal T12HRC (if T12EXT = 0) or T12HRG (if T12EXT = 1) is selected. 11_B **T12HRD**, Either signal T12HRD (if T12EXT = 0) or T12HRH (if T12EXT = 1) is selected.
RES	31:16	r	**Reserved** Returns 0 if read.

Port Input Select Register 2

PISEL2
Port Input Select Register 2 **(0010_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
IST13HR	1:0	rw	**Input Select for T13HR** This bit field defines the input signal used asT13HR input. 00_B **T13HRA**, Either signal T13HRA (if T13EXT = 0) or T13HRE (if T13EXT = 1) is selected. 01_B **T13HRB**, Either signal T13HRB (if T13EXT = 0) or T13HRF (if T13EXT = 1) is selected. 10_B **T13HRC**, Either signal T13HRC (if T13EXT = 0) or T13HRG (if T13EXT = 1) is selected. 11_B **T13HRD**, Either signal T13HRD (if T13EXT = 0) or T13HRH (if T13EXT = 1) is selected.

Field	Bits	Type	Description
ISCNT12	3:2	rw	**Input Select for T12 Counting Input** This bit field defines the input event leading to a counting action of T12. 00_B **T12prescaler**, The T12 prescaler generates the counting events. Bit TCTR4.T12CNT is not taken into account. 01_B **TCTR4T12CNT**, Bit TCTR4.T12CNT written with 1 is a counting event. The T12 prescaler is not taken into account. 10_B **Risingedge**, The timer T12 is counting each rising edge detected in the selected T12HR signal. 11_B **Fallingedge**, The timer T12 is counting each falling edge detected in the selected T12HR signal.
ISCNT13	5:4	rw	**Input Select for T13 Counting Input** This bit field defines the input event leadingto a counting action of T13. 00_B **T13prescaler**, The T13 prescaler generates the counting events. Bit TCTR4.T13CNT is not taken into account. 01_B **TCTR4T13CNT**, Bit TCTR4.T13CNT written with 1 is a counting event. The T13 prescaler is not taken into account. 10_B **Risingedge**, The timer T13 is counting each rising edge detected in the selected T13HR signal. 11_B **FallingEdge**, The timer T13 is counting each falling edge detected in the selected T13HR signal.
T12EXT	6	rw	**Extension for T12HR Inputs** This bit extends the 2-bit field IST12HR. 0_B **Ext_0**, One of the signals T12HR[D:A] is selected. 1_B **Ext_1**, One of the signals T12HR[H:E] is selected.
T13EXT	7	rw	**Extension for T13HR Inputs** This bit extends the 2-bit field IST13HR. 0_B **Ext_0**, One of the signals T13HR[D:A] is selected. 1_B **Ext_1**, One of the signals T13HR[H:E] is selected.
RES	15:8, 31:16	r	**Reserved**

Port Input Select Register 2x

PISEL2x (x=4-6)
Port Input Select Register 2x **(0014_H+(x-4)*4)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES											T1xEXT	ISCNT1x		IST1xHR	
r											rw	rw		rw	

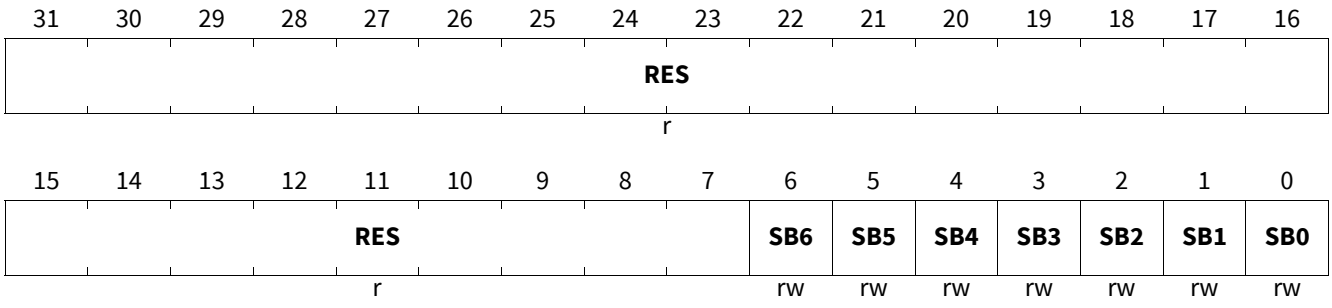
Field	Bits	Type	Description
IST1xHR	1:0	rw	**Input Select for T1xHR** This bit field defines the input signal used asT1xHR input. 00_B **T1xHRA**, Either signal T1xHRA (if T1xEXT = 0) or T1xHRE (if T1xEXT = 1) is selected. 01_B **T1xHRB**, Either signal T1xHRB (if T1xEXT = 0) or T1xHRF (if T1xEXT = 1) is selected. 10_B **T1xHRC**, Either signal T1xHRC (if T1xEXT = 0) or T1xHRG (if T1xEXT = 1) is selected. 11_B **T1xHRD**, Either signal T1xHRD (if T1xEXT = 0) or T1xHRH (if T1xEXT = 1) is selected.
ISCNT1x	3:2	rw	**Input Select for T1x Counting Input** This bit field defines the input event leadingto a counting action of T1x. 00_B **T1xprescaler**, The T1x prescaler generates the counting events. Bit TCTR4x.T1xCNT is not taken into account. 01_B **TCTR4xT1xCNT**, Bit TCTR4x.T1xCNT written with 1 is a counting event. The T1x prescaler is not taken into account. 10_B **Risingedge**, The timer T1x is counting each rising edge detected in the selected T1xHR signal. 11_B **FallingEdge**, The timer T1x is counting each falling edge detected in the selected T1xHR signal.
T1xEXT	4	rw	**Extension for T1xHR Inputs** This bit extends the 2-bit field IST1xHR. 0_B **Ext_0**, One of the signals T1xHR[D:A] is selected. 1_B **Ext_1**, One of the signals T1xHR[H:E] is selected.
RES	15:5, 31:16	r	**Reserved**

Kernel State Control Sensitivity Register

The kernel state control sensitivity register bits define which internal block is affected by Stop Modes 0 and 1, as described in section.

KSCSR

Kernel State Control Sensitivity Register (0020_H) RESET_TYPE_5 Value: $0000\ 0000_H$



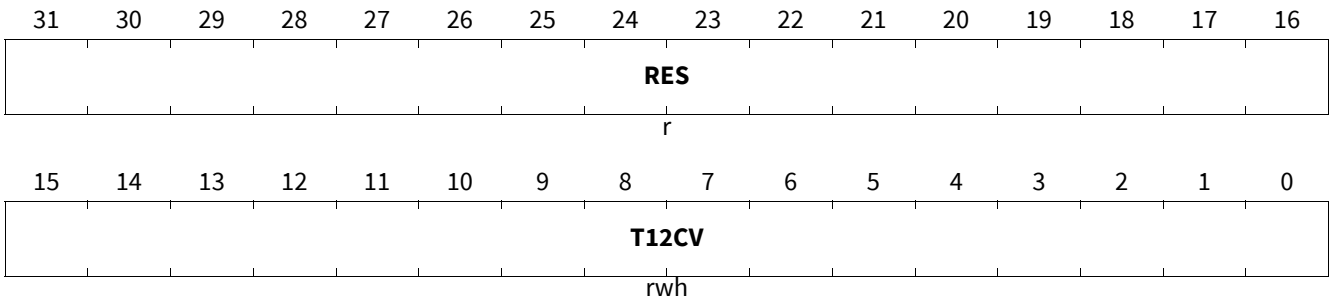
Field	Bits	Type	Description
SBx (x=0-6)	x	rw	**Sensitivity Block x SBx (x=0,1,2,...)** This bit defines if block x of the CCU7 kernel is sensitive to StopMode 0 or Stop Mode 1. The functional definition of the blocks is given in . 0_B Block x is not sensitive to Stop Mode 0 or Stop Mode 1. It continues normal operation without respecting the defined stop condition. 1_B Block x is sensitive to Stop Mode 0 or Stop Mode 1. It is respecting the defined stop condition.
RES	31:7	r	**Reserved; - 0** Returns 0 if read; should be written with 0.

Timer T12 Counter Register

Register T12 represents the counting value of timer T12. It can only be written while thetimer T12 is stopped. Write actions while T12 is running are not taken into account.Register T12 can always be read by software.

In edge-aligned mode, T12 only counts up, whereas in center-aligned mode, T12 can count up anddown.

T12
Timer T12 Counter Register **(0024_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
T12CV	15:0	rwh	**Timer T12 Counter Value** This register represents the 16-bit countervalue of timer T12.
RES	31:16	r	**Reserved** Returns 0 if read.

Timer T12 Period Register

Register T12PR contains the period value for timer T12. The period value is compared to the actual counter value of T12 and the resulting counter actions depend on the defined counting rules. This register has a shadow register and the shadow transfer is controlled by bit STE12. A read action by software delivers the value which is currently used for the compare action, whereas the write action targets a shadow register. The shadow register structure allows a concurrent update of all T12-related values.

T12PR

Timer T12 Period Register **(0028$_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T12PV															
rwh															

Field	Bits	Type	Description
T12PV	15:0	rwh	**T12 Period Value** The value T12PV defines the counter value for T12,which leads to a period-match. On reaching this value, the timerT12 is set to zero (edge-aligned mode) or changes its count direction to down counting (center-aligned mode).
RES	31:16	r	**Reserved** Returns 0 if read.

Dead-Time Control Register for Timer T12 Low

Register T12DTC controls the dead-time generation for the timer T12 compare channels. Eachchannel can be independently enabled/disabled for dead-time generation. If enabled, thetransition from passive state to active state is delayed by the value defined by bit fieldDTM. The dead-time counter can only be reloaded while it is zero.

The dead time countersare clocked with the same frequency as T12. This structure allows symmetrical dead-timegeneration in center-aligned and in edge-aligned PWM mode. A duty cycle of 50% leads toCC7x, COUT7x switched on for: 0.5 * period - dead time.

T12DTC

Dead-Time Control Register for Timer T12 Low **(002C$_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					DTR2	DTR1	DTR0	RES					DTE2	DTE1	DTE0
r					rh	rh	rh	r					rw	rw	rw

Field	Bits	Type	Description
DTE0	0	rw	**Dead-Time Enable Bit 0** Bit DTE0 enables and disables the dead-time generation for compare channel 0 of timer T12. 0_B **Disabled**, Dead-time generation is disabled. The corresponding outputs switch from the passive state to the active state (according to the actual compare status) without any delay. 1_B **Enabled**, Dead-time generation is enabled. The corresponding outputs switch from the passive state to the active state (according to the compare status) with the delay programmed in bit field DTM_RISE/FALL.
DTE1	1	rw	**Dead-Time Enable Bit 1** Bit DTE1 enables and disables the dead-time generation for compare channel 1 of timer T12. 0_B **Disabled**, Dead-time generation is disabled. The corresponding outputs switch from the passive state to the active state (according to the actual compare status) without any delay. 1_B **Enabled**, Dead-time generation is enabled. The corresponding outputs switch from the passive state to the active state (according to the compare status) with the delay programmed in bit field DTM_RISE/FALL.
DTE2	2	rw	**Dead-Time Enable Bit 2** Bit DTE2 enables and disables the dead-time generationfor compare channel 2 of timer T12. 0_B **Disabled**, Dead-time generation is disabled. The corresponding outputs switch from the passive state to the active state (according to the actual compare status) without any delay. 1_B **Enabled**, Dead-time generation is enabled. The corresponding outputs switch from the passive state to the active state (according to the compare status) with the delay programmed in bit field DTM_RISE/FALL.
RES	7:3, 15:11, 31:16	r	**Reserved**
DTR0	8	rh	**Dead-Time Run Indication Bit 0** Bit DTR0 indicate the status of the dead-time generationfor compare channel 0 of timer T12. 0_B **Passive**, Dead-Time Counter 0 is currently in the passive state. 1_B **Active**, Dead-Time Counter 0 is currently in the active state.
DTR1	9	rh	**Dead-Time Run Indication Bit 1** Bit DTR1 indicates the status of the dead-time generation for compare channel 1 of timer T12. 0_B **Passive**, Dead-Time Counter 1 is currently in the passive state. 1_B **Active**, Dead-Time Counter 1 is currently in the active state.

Field	Bits	Type	Description
DTR2	10	rh	**Dead-Time Run Indication Bit 2** Bit DTR2 indicates the status of the dead-time generation for compare channel 2 of timer T12. 0_B **Passive**, Dead-Time Counter 2 is currently in the passive state. 1_B **Active**, Dead-Time Counter 2 is currently in the active state.

Dead-Time value Register CC7x

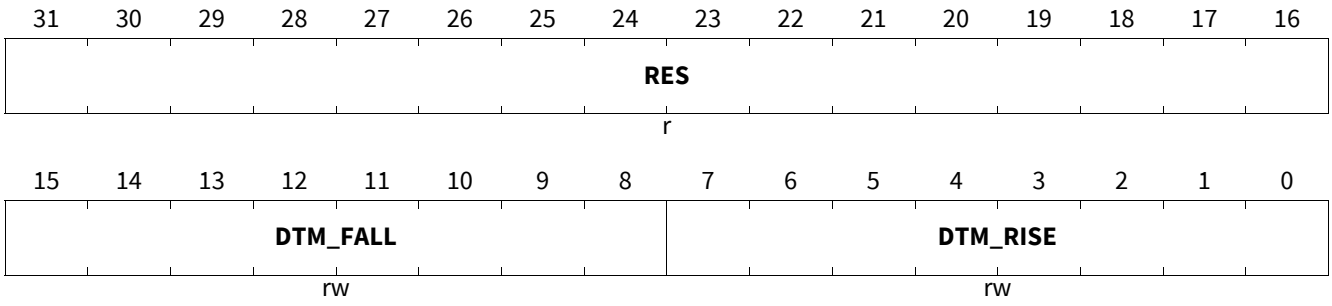
Register T12DTC controls the dead-time generation for the timer T12 compare channels. Eachchannel can be independently enabled/disabled for dead-time generation. If enabled, thetransition from passive state to active state is delayed by the value defined by bit fieldDTM. The dead-time counter can only be reloaded while it is zero.

The dead time counters are clocked with the same frequency as T12. This structure allowssymmetrical dead-time generation in center-aligned and in edge-aligned PWM mode. A dutycycle of 50% leads to CC7x, COUT7x

switched on for: 0.5 * period - DTM rise.

switched off for: 0.5 * period - DTM fall

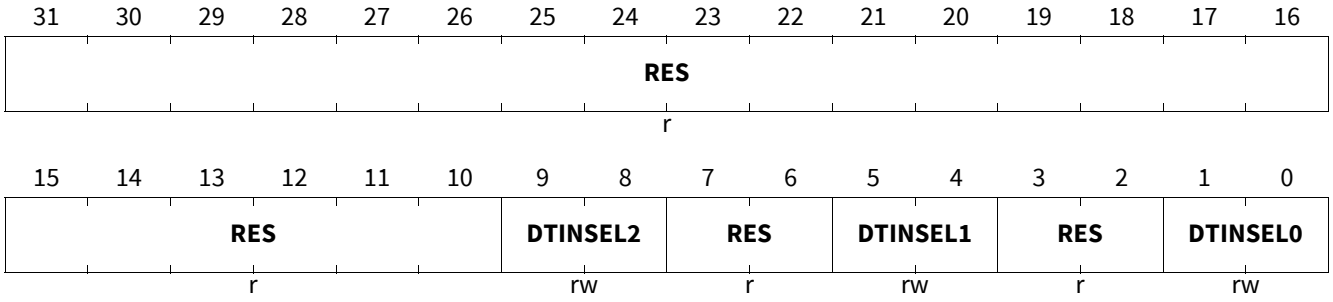
T12DTx_VAL (x=0-2)
Dead-Time value Register CC7x **(0030_H+x*4)** **RESET_TYPE_5 Value: 0000 0000_H**



Field	Bits	Type	Description
DTM_RISE	7:0	rw	**Dead-Time rise** Bit field DTM determines the programmable delay between switching from the passive state to the active state of the selected outputs. The switching from the active state to the passive state is not delayed.
DTM_FALL	15:8	rw	**Dead-Time fall** Bit field DTM determines the programmable delay between switching from the passive state to the active state of the selected outputs. The switching from the active state to the passive state is not delayed.
RES	31:16	r	**Reserved** Returns 0 if read.

Dead-Time Input Selection Register (Demo in Windows)

T12DTINSEL
Dead-Time Input Selection Register [Demo in Windows](003C$_H$) **RESET_TYPE_5 Value: 0000 0000$_H$**



Field	Bits	Type	Description
DTINSEL0	1:0	rw	**Deadtime Input selection 0** 00_B **CC70ST**, State Bit of Compare Channel 70 01_B **CC70BST**, State Bit of Compare Channel 70 B 10_B **T14R**, Run Bit from T14 Timer 11_B **T14ST**, State Bit of Timer14
RES	3:2, 7:6, 15:10, 31:16	r	**Reserved** Returns 0 if read.
DTINSEL1	5:4	rw	**Deadtime Input selection 1** 00_B **CC71ST**, State Bit of Compare Channel 71 01_B **CC71BST**, State Bit of Compare Channel 71 B 10_B **T15R**, Run Bit from T15 Timer 11_B **T15ST**, State Bit of Timer15
DTINSEL2	9:8	rw	**Deadtime Input selection 2** 00_B **CC72ST**, State Bit of Compare Channel 72 01_B **CC72BST**, State Bit of Compare Channel 72 B 10_B **T16R**, Run Bit from T16 Timer 11_B **T16ST**, State Bit of Timer16

Capture/Compare Register for Channel CC7x

In compare mode, the registers CC7xR (x = 0 - 2) are the actual compare registers for T12. The values stored in CC7xR are compared (all three channels in parallel) to the counter value of T12. In capture mode, the current value of the T12 counter register is captured by registers CC7xR if the corresponding capture event is detected.

CC7xR (x=0-2)

Capture/Compare Register for Channel CC7x (0040$_H$+x*4) **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCV															
rh															

Field	Bits	Type	Description
CCV	15:0	rh	**Capture/Compare Value** In compare mode, the bit fields CCV contain the values, that are compared to the T12 counter value. In capture mode, the captured value of T12 can be read from these registers.
RES	31:16	r	**Reserved; - 0** Returns 0 if read; should be written with 0.

Capture/Compare Shadow Reg. for Channel CC7x

The registers CC7xR can only be read by SW, the modification of the value is done by a shadow register transfer from register CC7xSR. The corresponding shadow registers CC7xSR can be read and written by SW. In capture mode, the value of the T12 counter register can also be captured by registers CC7xSR if the selected capture event is detected (depending on the selected capture mode).

Note: The shadow registers can also be written by SW in capture mode. In this case, the HW capture event wins over the SW write if both happen in the same cycle (the SW write is discarded).

CC7xSR (x=0-2)

Capture/Compare Shadow Reg. for Channel CC7x(004C$_H$+x*4) **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCS															
rwh															

Field	Bits	Type	Description
CCS	15:0	rwh	**Shadow Register for Channel x Capture/Compare Value** In compare mode, the bit fields contents of CCS are transferred to the bit fields CCV for the corresponding channel during a shadow transfer. In capture mode, the captured value of T12 can be read from these registers.

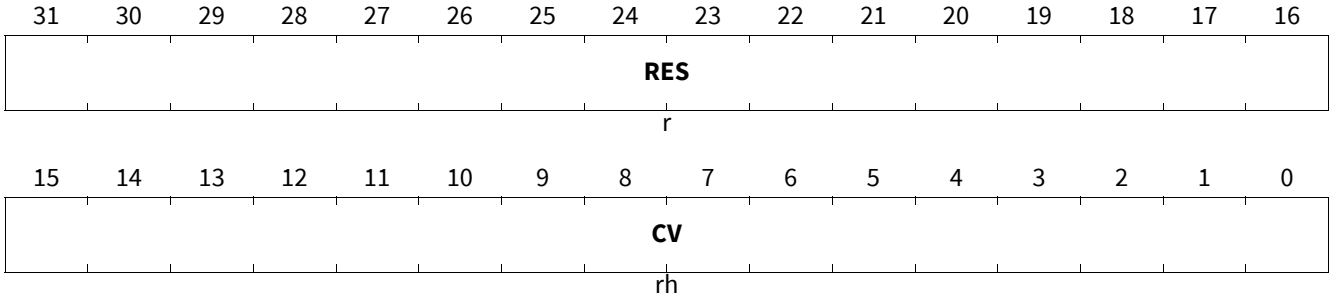
Field	Bits	Type	Description
RES	31:16	r	**Reserved; - 0** Returns 0 if read; should be written with 0.

Compare Register for Channel C7xB

The registers C7xBR (x = 0 - 2) are the actual compare registers forT12. The values stored in C7xBR are compared (all three channels in parallel) to the countervalue of T12.

CC7xBR (x=0-2)

Compare Register for Channel C7xB (0058$_H$+x*4) RESET_TYPE_5 Value: 0000 0000$_H$



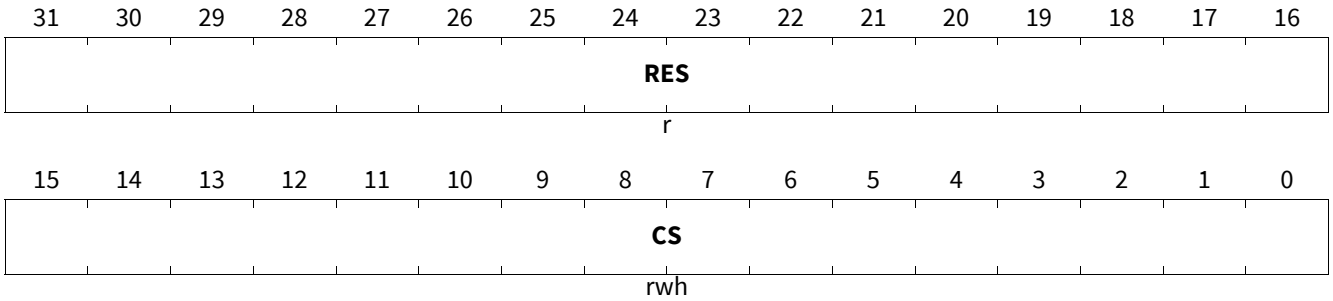
Field	Bits	Type	Description
CV	15:0	rh	**Compare Value** In compare mode, the bit fields CCV contain the values, that are compared to the T12 counter value. In capture mode, the captured value of T12 can be read from these registers.
RES	31:16	r	**Reserved; - 0** Returns 0 if read; should be written with 0.

Compare Shadow Reg. for Channel C7xBSR

The registers C7xBR can only be read by SW, the modification of thevalue is done by a shadow register transfer from register C7xBSR. The corresponding shadowregisters C7xBSR can be read and written by SW.

CC7xBSR (x=0-2)

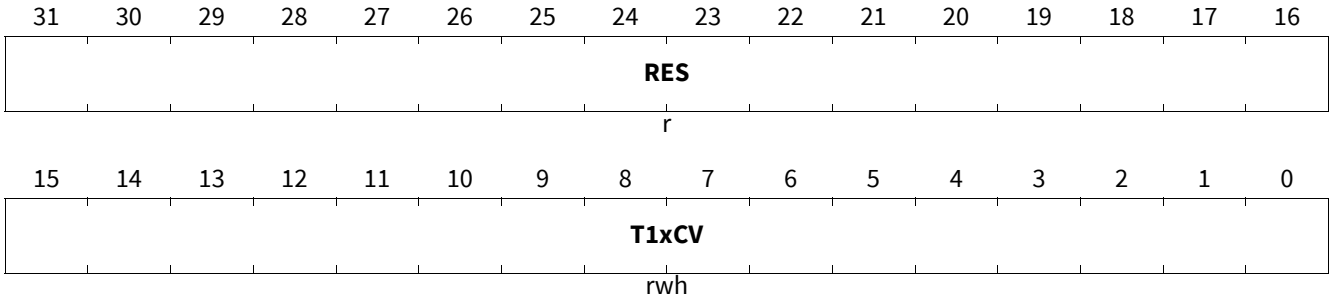
Compare Shadow Reg. for Channel C7xBSR (0064$_H$+x*4) RESET_TYPE_5 Value: 0000 0000$_H$



Field	Bits	Type	Description
CS	15:0	rwh	**Shadow Register for Channel x Compare Value** In compare mode, the bit fields contents of CCS are transferred to the bit fields CCV for the corresponding channel during a shadow transfer. In capture mode, the captured value of T12 can be read from these registers.
RES	31:16	r	**Reserved; - 0** Returns 0 if read; should be written with 0.

Timer T1x Counter Register

T1xR (x=3-6)
Timer T1x Counter Register **(0070_H+(x-3)*4)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
T1xCV	15:0	rwh	**Timer T1x Counter Value** This register represents the countervalue of timer T1x.
RES	31:16	r	**Reserved** Returns 0 if read.

Timer T1x Period Register

The generation of the patterns for a single channel pulse widthmodulation (PWM) is based on timer T1x. The registers related to timer T1x can be concurrentlyupdated (with well-defined conditions) in order to ensure consistency of the PWM signal. T1xcan be synchronized to several timer T12 events. Timer T1x only supports compare mode on itscompare channel CC7x. Register T1x represents the counting value of timer T1x. It can only bewritten while the timer T1x is stopped. Write actions while T1x is running are not taken intoaccount. Register T1x can always be read by SW. Timer T1x only supports edge-aligned mode(counting up).

Note: While timer T1x is stopped, the internal clock divider is reset in orderto ensure reproducible timings and delays.

T1xPR (x=3-6)

Timer T1x Period Register **(0080_H+(x-3)*4)** **RESET_TYPE_5 Value: 0000 0000_H**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T1xPV															
rwh															

Field	Bits	Type	Description
T1xPV	15:0	rwh	**T1x Period Value** The value T1xPV defines the counter value for Tx3,which leads to a period-match. On reaching this value, the timer T1x is set to zero.
RES	31:16	r	**Reserved** Returns 0 if read.

Compare Register for Channel C7x

Register C7xR is the actual compare register for T1x. The value storedin C7xR is compared to the counter value of T1x. The state bit C7xST is located inCMPSTAT/CMPSTAT_2

C7xR (x=3-6)

Compare Register for Channel C7x **(0090_H+(x-3)*4)** **RESET_TYPE_5 Value: 0000 0000_H**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCV															
rh															

Field	Bits	Type	Description
CCV	15:0	rh	**Channel C7x Compare Value** The bit filed CCV contains the value compared to T1x countervalue
RES	31:16	r	**Reserved; - 0** Returns 0 if read; should be written with 0.

Compare Shadow Reg. for Channel CC7x

The registers CC7xR can only be read by SW, the modification of thevalue is done by a shadow register transfer from register CC7xSR. The corresponding shadowregisters CC7xSR can be read and written by SW.

C7xSR (x=3-6)

Compare Shadow Reg. for Channel CC7x (00A0$_H$+(x-3)*4) RESET_TYPE_5 Value: 0000 0000$_H$

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCS															
rwh															

Field	Bits	Type	Description
CCS	15:0	rwh	**Shadow Register for Channel CC7x Compare Value** The contents of bit field CCS are transferred to the bit filed CCVduring shadow transfer
RES	31:16	r	**Reserved; - 0** Returns 0 if read; should be written with 0.

Compare State Register

The Compare State Register CMPSTAT contains status bits monitoring the current capture and compare state, and control bits defining the active/passive state of the compare channels.

CMPSTAT

Compare State Register (00B0$_H$) RESET_TYPE_5 Value: 0000 0000$_H$

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T13IM	COUT 73PS	COUT 72PS	CC72PS	COUT 71PS	CC71PS	COUT 70PS	CC70PS	RES	CC73ST	CCPOS2	CCPOS1	CCPOS0	CC72ST	CC71ST	CC70ST
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	r	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
CC70ST	0	rh	**Capture/Compare State Bits** Bits CC7xST monitor the state of the capture/compare channels. Bits CC7xST are related to T12; bit CC73ST is related to T13. These bits are set and reset according to the T12and T13 switching rules. 0_B **Less**, In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by software the last time. 1_B **Greater**, In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.

Field	Bits	Type	Description
CC71ST	1	rh	**Capture/Compare State Bits** Bits CC7xST monitor the state of the capture/compare channels. Bits CC7xST are related to T12; bit CC73ST is related to T13. These bits are set and reset according to the T12and T13 switching rules. 0_B **Less**, In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by software the last time. 1_B **Greater**, In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.
CC72ST	2	rh	**Capture/Compare State Bits** Bits CC7xST monitor the state of the capture/comparechannels. Bits CC7xST are related to T12; bit CC73ST is relatedto T13. These bits are set and reset according to the T12and T13 switching rules. 0_B **Less**, In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by software the last time. 1_B **Greater**, In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.
CCPOS0	3	rh	**Sampled Hall Pattern Bit 0** Bit CCPOS0 indicate the value of the input Hall pattern that has been compared to the current and expected value. The value is sampled when the event hcrdy (Hall compare ready) occurs. 0_B **Zero**, The input CCPOS0 has been sampled as 0. 1_B **One**, The input CCPOS0 has been sampled as 1.
CCPOS1	4	rh	**Sampled Hall Pattern Bit 1** Bit CCPOS1 indicate the value of the input Hallpattern that has been compared to the current and expected value.The value is sampled when the event hcrdy (Hall compare ready) occurs. 0_B **Zero**, The input CCPOS1 has been sampled as 0. 1_B **One**, The input CCPOS1 has been sampled as 1.
CCPOS2	5	rh	**Sampled Hall Pattern Bit 2** Bit CCPOS2 indicate the value of the input Hallpattern that has been compared to the current and expected value.The value is sampled when the event hcrdy (Hall compare ready) occurs. 0_B **Zero**, The input CCPOS2 has been sampled as 0. 1_B **One**, The input CCPOS2 has been sampled as 1.
CC73ST	6	rh	**Compare State Bits** Bit CC73ST is related to T13. These bits are set and reset according to the T12and T13 switching rules. 0_B **Less**, The counter value is less than the compare value. 1_B **Greater**, The counter value is greater than or equal to the compare value.
RES	7, 31:16	r	**Reserved** Returns 0 if read.

Field	Bits	Type	Description
CC70PS	8	rwh	**Passive State Select for Compare Outputs** Bits CC7xPS select the state of the corresponding compare channel, which is considered to be the passive state. During the passive state, the passive level (defined in register PSLR) is driven by the output pin. Bits CC7xPS are related to T12. In capture mode, these bits are not used. 0_B **Zero**, The corresponding compare output drives passive level while CC7xST is 0. 1_B **One**, The corresponding compare output drives passive level while CC7xST is 1.
COUT70PS	9	rwh	**Passive State Select for Compare Outputs** Bits COUT7xPS select the state of the correspondingcompare channel, which is considered to be thepassive state. Duringthe passive state, the passive level (defined in register PSLR)isdriven by the output pin. Bits COUT7xPS (x = 0, 1, 2) are relatedto T12 In capture mode, these bits are not used. 0_B **Zero**, The corresponding compare output drives passive level while CC7xST is 0. 1_B **One**, The corresponding compare output drives passive level while CC7xST is 1.
CC71PS	10	rwh	**Passive State Select for Compare Outputs** Bits CC7xPS select the state of the correspondingcompare channel, which is considered to be thepassive state. Duringthe passive state, the passive level (defined in register PSLR)isdriven by the output pin. Bits CC7xPS are related to T12. In capture mode, these bits are not used. 0_B **Zero**, The corresponding compare output drives passive level while CC7xST is 0. 1_B **One**, The corresponding compare output drives passive level while CC7xST is 1.
COUT71PS	11	rwh	**Passive State Select for Compare Outputs** Bits COUT7xPS select the state of the correspondingcompare channel, which is considered to be thepassive state. Duringthe passive state, the passive level (defined in register PSLR)isdriven by the output pin. Bits COUT7xPS (x = 0, 1, 2) are relatedto T12. In capture mode, these bits are not used. 0_B **Zero**, The corresponding compare output drives passive level while CC7xST is 0. 1_B **One**, The corresponding compare output drives passive level while CC7xST is 1.

Field	Bits	Type	Description
CC72PS	12	rwh	**Passive State Select for Compare Outputs** Bits CC7xPS select the state of the correspondingcompare channel, which is considered to be thepassive state. Duringthe passive state, the passive level (defined in register PSLR)isdriven by the output pin. Bits CC7xPS are related to T12. In capture mode, these bits are not used. 0_B **Zero**, The corresponding compare output drives passive level while CC7xST is 0. 1_B **One**, The corresponding compare output drives passive level while CC7xST is 1.
COUT72PS	13	rwh	**Passive State Select for Compare Outputs** COUT7xPS select the state of the correspondingcompare channel, which is considered to be thepassive state. Duringthe passive state, the passive level (defined in register PSLR)isdriven by the output pin. Bits COUT7xPS (x = 0, 1, 2) are relatedto T12. In capture mode, these bits are not used. 0_B **Zero**, The corresponding compare output drives passive level while CC7xST is 0. 1_B **One**, The corresponding compare output drives passive level while CC7xST is 1.
COUT73PS	14	rwh	**Passive State Select for Compare Outputs** Bits COUT7xPS select the state of the correspondingcompare channel, which is considered to be thepassive state. Duringthe passive state, the passive level (defined in register PSLR)isdriven by the output pin. Bits COUT7xPS (x = 0, 1, 2) are relatedto T12. In capture mode, these bits are not used. 0_B **Zero**, The corresponding compare output drives passive level while CC7xST is 0. 1_B **One**, The corresponding compare output drives passive level while CC7xST is 1.
T13IM	15	rwh	**T13 Inverted Modulation** Bit T13IM inverts the T13 signal for the modulation of the CC7x and COUT7x (x = 0, 1, 2) signals. 0_B **Notinverted**, T13 output is not inverted. 1_B **Inverted**, T13 output is inverted for further modulation.

Compare State Register 2

CMPSTAT_2
Compare State Register 2 **($00B4_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					CC72ST	CC71ST	CC70ST	RES	C76ST	C75ST	C74ST	RES	C72BST	C71BST	C70BST
r					rh	rh	rh	r	rh	rh	rh	r	rh	rh	rh

Field	Bits	Type	Description
C70BST	0	rh	**Compare State Bits B** Bits C70BST monitor the state of the comparechannels. Bit C70BST is related to T12. These bits are set and reset according to the T12 switching rules. 0_B **Less**, In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by software the last time. 1_B **Greater**, In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.
C71BST	1	rh	**Compare State Bits B** Bits C71BST monitor the state of the comparechannels. Bit C71BST is related to T12. These bits are set and reset according to the T12 switching rules. 0_B **Less**, In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by software the last time. 1_B **Greater**, In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.
C72BST	2	rh	**Compare State Bits B** Bits C72BST monitor the state of the comparechannels. Bit C72BST is related to T12. These bits are set and reset according to the T12 switching rules. 0_B **Less**, In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by software the last time. 1_B **Greater**, In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.

Field	Bits	Type	Description
RES	3, 7, 15:11, 31:16	r	**Reserved** Returns 0 if read.
C74ST	4	rh	**Compare State Bits** Bits C74ST monitor the state of the compare channel. Bits C74ST are related to T14 These bit is set and reset according to the T14 rules. 0_B **Less**, In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by software the last time. 1_B **Greater**, In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.
C75ST	5	rh	**Compare State Bits** Bits C75ST monitor the state of the compare channel. Bits C75ST are related to T15 These bit is set and reset according to the T15 rules. 0_B **Less**, In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by software the last time. 1_B **Greater**, In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.
C76ST	6	rh	**Compare State Bits** Bits C76ST monitor the state of the compare channel. Bits C76ST are related to T16 These bit is set and reset according to the T16 rules. 0_B **Less**, In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by software the last time. 1_B **Greater**, In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.
CC70ST	8	rh	**Capture/Compare State Bits** This Bit is an read copy of CMPSTAT.CC70ST 0_B **Less**, In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by software the last time. 1_B **Greater**, In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.

Field	Bits	Type	Description
CC71ST	9	rh	**Capture/Compare State Bits** This Bit is an read copy of CMPSTAT.CC71ST 0_B **Less**, In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by software the last time. 1_B **Greater**, In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.
CC72ST	10	rh	**Capture/Compare State Bits** This Bit is an read copy of CMPSTAT.CC72ST 0_B **Less**, In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by software the last time. 1_B **Greater**, In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.

Compare State Modification Register

The Compare Status Modification Register CMPMODIF provides software-control (independent set and clear conditions) for the channel state bits CC7xST. This feature enables the user to individually change the status of the output lines by software, for example when the corresponding compare timer is stopped.

CMPMODIF
Compare State Modification Register **($00B8_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	MCC73R	MCC72BR	MCC71BR	MCC70BR	MCC72R	MCC71R	MCC70R	RES	MCC73S	MCC72BS	MCC71BS	MCC70BS	MCC72S	MCC71S	MCC70S
r	w	w	w	w	w	w	w	r	w	w	w	w	w	w	w

Field	Bits	Type	Description
MCC70S	0	w	**Capture/Compare Status Modification Bit 0 (Set)** This bit is used to set the corresponding CC70ST bit by software. This feature allows the user to individually change the status of the output lines by software, e.g. when the corresponding compare timer is stopped. This allows a bit manipulation of CC70ST-bits by a single data write action. The following functionality of a write access to bits concerning the same capture/compare state bit is provided: (MCC70R, MCC70S) = 00: Bit CC70ST is not changed 01: Bit CC70ST is set 10: Bit CC70ST is cleared 11: Bit CC70ST is toggled
MCC71S	1	w	**Capture/Compare Status Modification Bit 1 (Set)** This bit is used to set the corresponding CC71ST bit by software. This feature allows the user to individually changethe status of the output lines by software, e.g. when the correspondingcompare timer is stopped. This allows a bit manipulation of CC71ST-bitsby a single data write action. The following functionality of a write access tobits concerning the same capture/compare state bit is provided: MCC71R, MCC71S = 00: Bit CC71ST is not changed 01: Bit CC71ST is set 10: Bit CC71ST is cleared 11: Bit CC71ST is toggled
MCC72S	2	w	**Capture/Compare Status Modification Bit 2 (Set)** This bit is used to set the corresponding CC72ST bit by software. This feature allows the user to individually changethe status of the output lines by software, e.g. when the correspondingcompare timer is stopped. This allows a bit manipulation of CC72ST-bitsby a single data write action. The following functionality of a write access tobits concerning the same capture/compare state bit is provided: MCC72R, MCC72S = 00: Bit CC72ST is not changed 01: Bit CC72ST is set 10: Bit CC72ST is cleared 11: Bit CC72ST is toggled

Field	Bits	Type	Description
MCC70BS	3	w	**Compare B Status Modification Bit 0 (Set)** This bit is used to set the corresponding C70BST bit by software. This feature allows the user to individually changethe status of the output lines by software,e.g. when the correspondingcompare timer is stopped. This allows a bit manipulation ofC70BST-bitsby a single data write action. The following functionality of a write access tobits concerning the same capture/compare state bit is provided: MCC70BR, MCC70BS = 00: Bit C70BST is not changed 01: Bit C70BST is set 10: Bit C70BST is cleared 11: Bit C70BST is toggled
MCC71BS	4	w	**Compare B Status Modification Bit 1 (Set)** This bit is used to set the corresponding C71BST bit by software. This feature allows the user to individually changethe status of the output lines by software,e.g. when the correspondingcompare timer is stopped. This allows a bit manipulation ofC71BST-bitsby a single data write action. The following functionality of a write access tobits concerning the same capture/compare state bit is provided: MCC71BR, MCC71BS = 00: Bit C71BST is not changed 01: Bit C71BST is set 10: Bit C71BST is cleared 11: Bit C71BST is toggled
MCC72BS	5	w	**Compare B Status Modification Bit 2 (Set)** This bit is used to set the corresponding C72 BST bit by software. This feature allows the user to individually change the status of the output lines by software, e.g. when the corresponding compare timer is stopped. This allows a bit manipulation of C72BST-bitsby a single data write action. The following functionality of a write access tobits concerning the same capture/compare state bit is provided: MCC72BR, MCC72BS = 00: Bit C72BST is not changed 01: Bit C72BST is set 10: Bit C72BST is cleared 11: Bit C72BST is toggled

Field	Bits	Type	Description
MCC73S	6	w	**Capture/Compare Status Modification Bits (Set)** This bit is used to set the corresponding CC73ST bit by software. This feature allows the user to individually changethe status of the output lines by software, e.g. when the correspondingcompare timer is stopped. This allows a bit manipulation of CC73ST-bitsby a single data write action. The following functionality of a write access tobits concerning the same capture/compare state bit is provided: MCC73R, MCC73S = 00: Bit CC73ST is not changed 01: Bit CC73ST is set 10: Bit CC73ST is cleared 11: Bit CC73ST is toggled 0_B **Notchanged**, Bit CC73ST is not changed. 1_B **Set**, Bit CC73ST is set.
RES	7, 15, 31:16	r	**Reserved**
MCC70R	8	w	**Capture/Compare Status Modification Bit 0(Reset)** This bit is used to reset the corresponding CC70ST bit by software. This feature allows the user to individually changethe status of the output lines by software, e.g. when the correspondingcompare timer is stopped. This allows a bit manipulation of CC70T-bits by a single data write action. The following functionality of a write access to bit concerning the same capture/compare state bit is provided: MCC70R, MCC70S = 00: Bit CC70ST is not changed 01: Bit CC70ST is set 10: Bit CC70ST is cleared 11: Bit CC70ST is toggled 0_B **Notchanged**, Bit CC70ST is not changed. 1_B **Set**, Bit CC70ST is set.
MCC71R	9	w	**Capture/Compare Status Modification Bit 1(Reset)** This bit is used to reset the corresponding CC71ST bit by software. This feature allows the user to individually changethe status of the output lines by software, e.g. when the correspondingcompare timer is stopped. This allows a bit manipulation of CC71ST-bitsby a single data write action. The following functionality of a write access tobit concerning the same capture/compare state bit is provided: MCC71R, MCC71S = 00: Bit CC71ST is not changed 01: Bit CC71ST is set 10: Bit CC71ST is cleared 11: Bit CC71ST is toggled 0_B **Notchanged**, Bit CC71ST is not changed. 1_B **Set**, Bit CC71ST is set.

Field	Bits	Type	Description
MCC72R	10	w	**Capture/Compare Status Modification Bit 2(Reset)** This bit is used to reset the corresponding CC72ST bit by software. This feature allows the user to individually changethe status of the output lines by software, e.g. when the correspondingcompare timer is stopped. This allows a bit manipulation of CC72ST-bitsby a single data write action. The following functionality of a write access tobit concerning the same capture/compare state bit is provided: MCC72R, MCC72S = 00: Bit CC72ST is not changed 01: Bit CC72ST is set 10: Bit CC72ST is cleared 11: Bit CC72ST is toggled 0_B **Notchanged**, Bit CC72ST is not changed. 1_B **Set**, Bit CC72ST is set.
MCC70BR	11	w	**Compare B Status Modification Bit 0(Reset)** This bit is used to reset the corresponding C70 BST bit by software. This feature allows the user to individually changethe status of the output lines by software,e.g. when the correspondingcompare timer is stopped. This allows a bit manipulation ofC70BST-bitsby a single data write action. The following functionality of a write access tobit concerning the same capture/compare state bit is provided: MCC70BR, MCC70BS = 00: Bit C70BST is not changed 01: Bit C70BST is set 10: Bit C70BST is cleared 11: Bit C70BST is toggled 0_B **Notchanged**, Bit CC70ST is not changed. 1_B **Set**, Bit CC70ST is set.
MCC71BR	12	w	**Compare B Status Modification Bit 1(Reset)** This bit is used to reset the corresponding C71 BST bit by software. This feature allows the user to individually changethe status of the output lines by software,e.g. when the correspondingcompare timer is stopped. This allows a bit manipulation ofC71BST-bitsby a single data write action. The following functionality of a write access tobit concerning the same capture/compare state bit is provided: MCC71BR, MCC71BS = 00: Bit C71BST is not changed 01: Bit C71BST is set 10: Bit C71BST is cleared 11: Bit C71BST is toggled 0_B **Notchanged**, Bit CC70ST is not changed. 1_B **Set**, Bit CC70ST is set.

Field	Bits	Type	Description
MCC72BR	13	w	**Compare B Status Modification Bit 2(Reset)** This bit is used to reset the corresponding C72 BST bit by software. This feature allows the user to individually changethe status of the output lines by software,e.g. when the correspondingcompare timer is stopped. This allows a bit manipulation ofC72BST-bits by a single data write action. The following functionality of a write access tobit concerning the same capture/compare state bit is provided: MCC72BR, MCC72BS = 00: Bit C72BST is not changed 01: Bit C72BST is set 10: Bit C72BST is cleared 11: Bit C72BST is toggled 0_B **Notchanged**, Bit CC70ST is not changed. 1_B **Set**, Bit CC70ST is set.
MCC73R	14	w	**Capture/Compare Status Modification Bits (Reset)** These bits are used to reset the correspondingCC73ST bit by software. This feature allows the user to individually changethe status of the output lines by software, e.g. when the correspondingcompare timer is stopped. This allows a bit manipulation of CC73ST-bitsby a single data write action. The following functionality of a write access tobits concerning the same capture/compare state bit is provided: MCC73R, MCC73S = 0_B **Notchanged**, Bit CC73ST is not changed. 1_B **Set**, Bit CC73ST is set.

Compare State Modification Register 2

CMPMODIF_2
Compare State Modification Register 2 **($00BC_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	C72BINV	C71BINV	C70BINV	RES	MC76R	MC75R	MC74R	RES	CC72INV	CC71INV	CC70INV	RES	MC76S	MC75S	MC74S
r	rwh	rwh	rwh	r	w	w	w	r	rwh	rwh	rwh	r	w	w	w

Field	Bits	Type	Description
MC74S	0	w	**Compare Status Modification Bit 4 (Set)** This bit is used to set the corresponding C74ST bit by software. This feature allows the user to individually changethe status of the output lines by software, e.g. when the corresponding compare timer is stopped. This allows a bit manipulation of C74ST-bits by a single data write action. (MCC74R, MCC74S)= 00: Bit C74ST is not changed 01: Bit C74ST is set 10: Bit C74ST is cleared 11: Bit C74ST is toggled 0_B **Notchanged**, Bit CC74ST is not changed. 1_B **Set**, Bit CC74ST is set.
MC75S	1	w	**Compare Status Modification Bit 5 (Set)** This bit is used to set the corresponding C75ST bit by software. This feature allows the user to individually changethe status of the output lines by software,e.g. when the correspondingcompare timer is stopped. This allowsa bit manipulation of C75ST-bitsby a single data writeaction. (MCC75R,MCC75S)= 00: Bit C75ST is not changed 01: Bit C75ST is set 10: Bit C75ST is cleared 11: Bit C75ST is toggled 0_B **Notchanged**, Bit CC75ST is not changed. 1_B **Set**, Bit CC75ST is set.
MC76S	2	w	**Compare Status Modification Bit 6 (Set)** This bit is used to set the corresponding C76ST bit by software. This feature allows the user to individually changethe status of the output lines by software,e.g. when the correspondingcompare timer is stopped. This allowsa bit manipulation of C76ST-bitsby a single data writeaction. (MCC76R,MCC76S)= 00: Bit C76ST is not changed 01: Bit C76ST is set 10: Bit C76ST is cleared 11: Bit C76ST is toggled 0_B **Notchanged**, Bit CC76ST is not changed. 1_B **Set**, Bit CC76ST is set.
RES	3, 7, 11, 31:15	r	**Reserved**
CC70INV	4	rwh	**Output channel CC70ST Inversion Enable** 0_B **Disabled**, Output Channel CC70ST not inverted 1_B **Enabled**, Output Channel CC70ST inverted
CC71INV	5	rwh	**Output channel CC71ST Inversion Enable** 0_B **Disabled**, Output Channel CC71ST not inverted 1_B **Enabled**, Output Channel CC71ST inverted

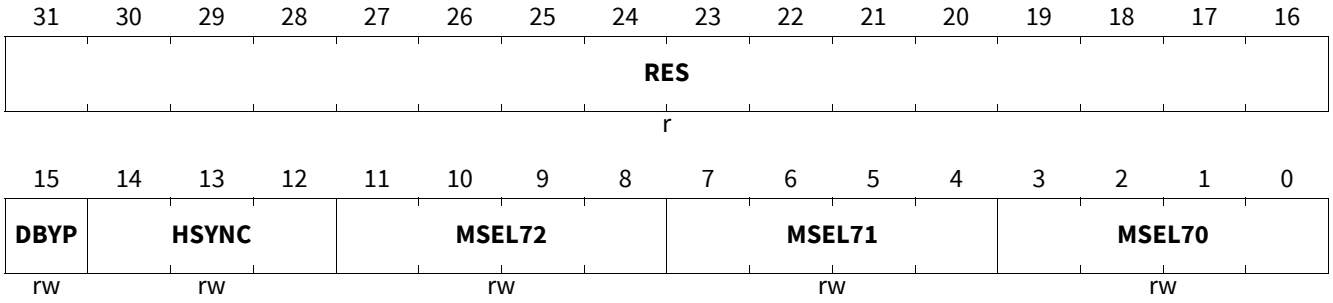
Field	Bits	Type	Description
CC72INV	6	rwh	**Output channel CC72ST Inversion Enable** 0_B **Disabled**, Output Channel CC72ST not inverted 1_B **Enabled**, Output Channel CC72ST inverted
MC74R	8	w	**Compare Status Modification Bit 4(Reset)** This bit is used to reset the corresponding C74ST bit by software. This feature allows the user to individually changethe status of the output lines by software,e.g. when the correspondingcompare timer is stopped. This allowsa bit manipulation of C74ST-bitsby a single data writeaction. (MCC74R,MCC74S)= 00: Bit C74ST is not changed 01: Bit C74ST is set 10: Bit C74ST is cleared 11: Bit C74ST is toggled 0_B **Notchanged**, Bit CC74ST is not changed. 1_B **Set**, Bit CC74ST is set.
MC75R	9	w	**Compare Status Modification Bit 5(Reset)** This bit is used to reset the corresponding C75ST bit by software. This feature allows the user to individually changethe status of the output lines by software,e.g. when the correspondingcompare timer is stopped. This allowsa bit manipulation of C75ST-bitsby a single data writeaction. (MCC75R,MCC75S)= 00: Bit C75ST is not changed 01: Bit C75ST is set 10: Bit C75ST is cleared 11: Bit C75ST is toggled 0_B **Notchanged**, Bit CC75ST is not changed. 1_B **Set**, Bit CC76ST is set.
MC76R	10	w	**Compare Status Modification Bit 6(Reset)** This bit is used to reset the corresponding C76ST bit by software. This feature allows the user to individually changethe status of the output lines by software,e.g. when the correspondingcompare timer is stopped. This allowsa bit manipulation of C76ST-bitsby a single data writeaction. (MCC76R,MCC76S)= 00: Bit C76ST is not changed 01: Bit C76ST is set 10: Bit C76ST is cleared 11: Bit C76ST is toggled 0_B **Notchanged**, Bit CC76ST is not changed. 1_B **Set**, Bit CC76ST is set
C70BINV	12	rwh	**Output channel C70BST Inversion Enable** 0_B **Disabled**, Output Channel C70BST not inverted 1_B **Enabled**, Output Channel C70BST inverted
C71BINV	13	rwh	**Output channel C71BST Inversion Enable** 0_B **Disabled**, Output Channel C71BST not inverted 1_B **Enabled**, Output Channel C71BST inverted

Field	Bits	Type	Description
C72BINV	14	rwh	**Output channel C72BST Inversion Enable** 0_B **Disabled**, Output Channel C72BST not inverted 1_B **Enabled**, Output Channel C72BST inverted

T12 Capture/Compare Mode Select Register

T12MSEL

T12 Capture/Compare Mode Select Register (00C0$_H$) RESET_TYPE_5 Value: 0000 0000$_H$



Field	Bits	Type	Description
MSEL7x (x=0-2)	4*x+3:4*x	rw	**Capture/Compare Mode Selection MSEL7x (x=0,1,2)** These bit fields select the operating mode of the three T12 capture/compare channels. Each channel (x=0, 1, 2) can be programmed individually for one of these modes (except for Hall Sensor Mode)
HSYNC	14:12	rw	**Hall Synchronization** Bit field HSYNC defines the source for the sampling of the Hall input pattern and the comparison to the current and the expected Hall pattern bit fields. 000_B **Any**, CCPOSx edge 001_B **T13comparematch**, T13 compare match 010_B **T13periodmatch**, T13 period match 011_B **Hall**, Hall sampling off. 100_B **T12periodmatch**, T12 period match UP 101_B **T12onematch**, T12 one match DOWN 110_B **T12comparematchUP**, A T12 compare match 1 UP 111_B **T12comparematchDOWN**, A T12 compare match 1 DOWN
DBYP	15	rw	**Delay Bypass** Bit DBYP defines if the source signal for the sampling of the Hall input pattern (selected by HSYNC) uses the dead-time counter DTC0 of timer T12 as additional delay or if the delay is bypassed. 0_B **Notactive**, The delay bypass is not active. The dead-time counter DTC0 is generating a delay after the source signal becomes active. 1_B **Active**, The delay bypass is active. The dead-time counter DTC0 is not used by the sampling of the Hall pattern.
RES	31:16	r	**Reserved** Returns 0 if read.

Timer Control Register 0

Register TCTR0 controls the basic functionality of both timers T12 and T13.

TCTR0
Timer Control Register 0 **($00C4_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		STE13	T13R	T13PRE	RES			CTM	CDIR	STE12	T12R	T12PRE	RES		
r		rh	rh	rw	r			rw	rh	rh	rh	rw	r		

Field	Bits	Type	Description
RES	2:0, 10:8, 15:14, 31:16	r	**Reserved**
T12PRE	3	rw	**Timer T12 Prescaler Bit** In order to support higher clock frequencies, an additional prescaler factor of 1/256 can be enabled for the prescaler for T12. 0_B **Disabled**, The additional prescaler for T12 is disabled. 1_B **Enabled**, The additional prescaler for T12 is enabled.
T12R	4	rh	**Timer T12 Run Bit** T12R starts and stops timer T12. It is set/cleared by SW by setting bits T12RR or T12RS or it iscleared by HW according to the function defined by bit field T12SSC. 0_B **Stop**, Timer T12 is stopped. 1_B **Run**, Timer T12 is running.
STE12	5	rh	**Timer T12 Shadow Transfer Enable** Bit STE12 enables or disables the shadow transfer of the T12 period value, the compare values and passive state select bits and levels from their shadow registers to the actual registersif a T12 shadow transfer event is detected. Bit STE12 is clearedby hardware after the shadow transfer. A T12 shadow transfer event is a period-match while counting up or a one-match while counting down. 0_B **Disabled**, The shadow register transfer is disabled. 1_B **Enabled**, The shadow register transfer is enabled.
CDIR	6	rh	**Count Direction of Timer T12** This bit is set/reset according to the countingrules of T12. 0_B **UP**, T12 counts up. 1_B **DOWN**, T12 counts down.

Field	Bits	Type	Description
CTM	7	rw	**T12 Operating Mode** 0_B **EdgealignedMode**, T12 always counts up and continues counting from zero after reaching the period value. 1_B **CenteralignedMode**, T12 counts down after detecting a period-match and counts up after detecting a one-match.
T13PRE	11	rw	**Timer T13 Prescaler Bit** In order to support higher clock frequencies, anadditional prescaler factor of 1/256 can be enabled for the prescalerfor T13. 0_B **Disabled**, The additional prescaler for T13 is disabled. 1_B **Enabled**, The additional prescaler for T13 is enabled.
T13R	12	rh	**Timer T13 Run Bit** T13R starts and stops timer T13. It is set/cleared by SW by setting bits T13RR or T13RS or it is set/cleared by HW according to the function defined by bit fields T13SSC, T13TEC and T13TED. 0_B **Stop**, Timer T13 is stopped. 1_B **Run**, Timer T13 is running.
STE13	13	rh	**Timer T13 Shadow Transfer Enable** Bit STE13 enables or disables the shadow transferof the T13 period value, the compare value and passive state select bit and level from their shadow registers to the actual registersif a T13 shadow transfer event is detected. Bit STE13 is clearedby hardware after the shadow transfer. A T13 shadow transfer event is a period-match. 0_B **Disabled**, The shadow register transfer is disabled. 1_B **Enabled**, The shadow register transfer is enabled.

Timer Control Register 1

TCTR1
Timer Control Register 1 **($00C8_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STE13	STE12	STE16	T16R	RES		STE15	T15R	RES		STE14	T14R	T1xPRE	RES		
rh	rh	rh	rh	r		rh	rh	r		rh	rh	rw	r		

Field	Bits	Type	Description
RES	2:0, 7:6, 31:16	r	**Reserved** Returns 0 if read.

Field	Bits	Type	Description
T1xPRE	3	rw	**Timer T14 / 15 /16 Prescaler Bit** In order to support higher clock frequencies, anadditional prescaler factor of 1/256 can be enabled for the prescalerfor T12. 0_B **Disabled**, The additional prescaler for T12 is disabled. 1_B **Enabled**, The additional prescaler for T12 is enabled.
T14R	4	rh	**Timer T14 Run Bit** T14R starts and stops timer T14. It is set/reset by software by setting bits T14RS or T14RR or it is set/reset by hardware according to the function defined by bit fields T14SSC,T14TEC and T14TED. 0_B **Stop**, Timer T14 is stopped. 1_B **Run**, Timer T14 is running.
STE14	5	rh	**Timer T14 Shadow Transfer Enable** Bit STE14 enables or disables the shadow transferof the T14 period value, the compare value and passive state select bit and level from their shadow registers to the actual registers if a T14 shadow transfer event is detected. Bit STE14 is cleared by hardware after the shadow transfer. A T14 shadow transfer event is a period-match. 0_B **Disabled**, The shadow register transfer is disabled. 1_B **Enabled**, The shadow register transfer is enabled.
T15R	8	rh	**Timer T15 Run Bit** T15R starts and stops timer T15. It is set/resetby software by settingbits T15RS or T15RR or it is set/reset byhardware according tothe function defined by bit fields T15SSC,T15TEC and T15TED. 0_B **Stop**, Timer T15 is stopped. 1_B **Run**, Timer T15 is running.
STE15	9	rh	**Timer T15 Shadow Transfer Enable** Bit STE15 enables or disables the shadow transferof the T15 period value, the compare value and passive state select bit and level from their shadow registers to the actual registers if a T15 shadow transfer event is detected. Bit STE15 is cleared by hardware after the shadow transfer. A T15 shadow transfer event is a period-match. 0_B **Disabled**, The shadow register transfer is disabled. 1_B **Enabled**, The shadow register transfer is enabled.
RES	11:10	r	**Reserved** Returns 0 if read.
T16R	12	rh	**Timer T16 Run Bit** T16R starts and stops timer T16. It is set/resetby software by setting bits T16RS orT16RR or it is set/reset byhardware according to the function defined by bit fieldsT16SSC,T16TEC and T16TED. 0_B **Stop**, Timer T16 is stopped. 1_B **Run**, Timer T16 is running.

Field	Bits	Type	Description
STE16	13	rh	**Timer T16 Shadow Transfer Enable** Bit STE16 enables or disables the shadow transferof the T16 period value, the comparevalue and passive state selectbit and level from their shadow registers to the actualregistersif a T16 shadow transfer event is detected. Bit STE16 is clearedby hardwareafter the shadow transfer. A T16 shadow transfer event is a period-match. 0_B **Disabled**, The shadow register transfer is disabled. 1_B **Enabled**, The shadow register transfer is enabled.
STE12	14	rh	**Timer T12 Shadow Transfer Enable** Bit STE12 enables or disables the shadow transferof the T12 period value, the compare values and passive state selectbits and levels from their shadow registers to the actual registersif a T12 shadow transfer event is detected. Bit STE12 is clearedby hardware after the shadow transfer. A T12 shadow transfer event is a period-match whilecounting up or a one-match while counting down. 0_B **Disabled**, The shadow register transfer is disabled. 1_B **Enabled**, The shadow register transfer is enabled.
STE13	15	rh	**Timer T13 Shadow Transfer Enable** Bit STE13 enables or disables the shadow transferof the T13 period value, the compare value and passive state selectbit and level from their shadow registers to the actual registersif a T13 shadow transfer event is detected. Bit STE13 is clearedby hardware after the shadow transfer. A T13 shadow transfer event is a period-match. 0_B **Disabled**, The shadow register transfer is disabled. 1_B **Enabled**, The shadow register transfer is enabled.

Timer Control Register 2

Register TCTR2 controls the single-shot and the synchronization functionality of both timers T12 and T13. Both timers can run in single-shot mode. In this mode, they stop their counting sequence automatically after one counting period with a count value of zero. The single-shot mode and the synchronization feature of T13 to T12 allow the generation of events with a programmable delay after well-defined PWM actions of T12. For example, this feature can be used to trigger AD conversions, after a specified delay (to avoid problems due to switching noise), synchronously to a PWM event.

TCTR2
Timer Control Register 2 **($00CC_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

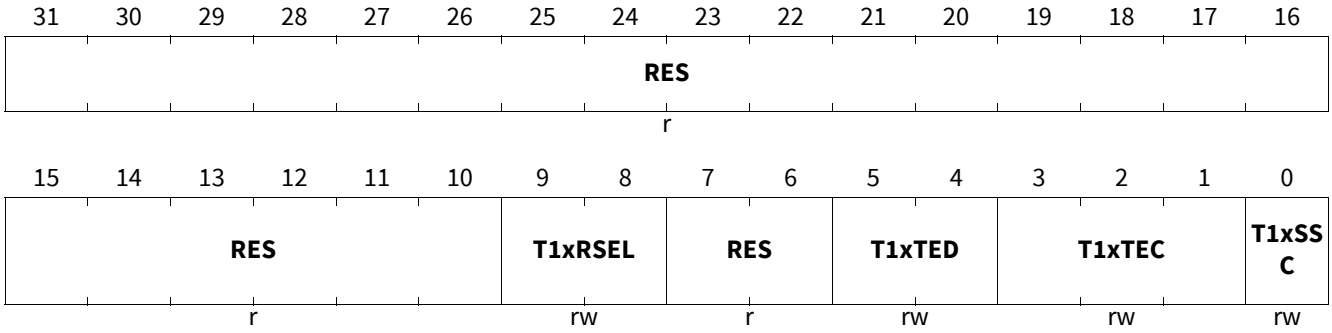
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				T13RSEL		T12RSEL		RES	T13TED		T13TEC			T13SSC	T12SSC
r				rw		rw		r	rw		rw			rw	rw

Field	Bits	Type	Description
T12SSC	0	rw	**Timer T12 Single Shot Control** This bit controls the single shot-mode of T12. 0_B **Disabled**, The single-shot mode is disabled, no hardware action on T12R. 1_B **Enabled**, The single shot mode is enabled, the bit T12R is reset by hardware if: T12 reaches its period value in edge-aligned mode T12 reaches the value 1 while down counting in center-aligned mode. In parallel to the reset action of bit T12R, the bits CC7xST (x = 0, 1, 2) are reset.
T13SSC	1	rw	**Timer T13 Single Shot Control** This bit controls the single shot-mode of T13. 0_B **Noaction**, No hardware action on T13R 1_B **Enabled**, The single-shot mode is enabled, the bit T13R is reset by hardware if T13 reaches its period value. In parallel to the reset action of bit T13R, the bit CC73ST is reset.
T13TEC	4:2	rw	**T13 Trigger Event Control** Bit field T13TEC selects the trigger event to startT13 (automatic set of T13R for synchronization to T12 compare signals) according to following combinations: refer to Table-6
T13TED	6:5	rw	**Timer T13 Trigger Event Direction** Bit field T13TED delivers additional information to control the automatic set of bit T13R in the case that the trigger action defined by T13TEC is detected. Refer to Table-7
RES	7, 15:12, 31:16	r	**Reserved** Returns 0 if read.
T12RSEL	9:8	rw	**Timer T12 External Run Selection** Bit field T12RSEL defines the event of signal T12HRthat can set the run bit T12R by hardware. 00_B **Disabled**, The external setting of T12R is disabled. 01_B **Risingedge**, Bit T12R is set if a rising edge of signal T12HR is detected. 10_B **Fallingedge**, Bit T12R is set if a falling edge of signal T12HR is detected. 11_B **Edge**, Bit T12R is set if an edge of signal T12HR is detected.
T13RSEL	11:10	rw	**Timer T13 External Run Selection** Bit field T13RSEL defines the event of signal T13HRthat can set the run bit T13R by hardware. 00_B **Disabled**, The external setting of T13R is disabled. 01_B **Risingedge**, Bit T13R is set if a rising edge of signal T13HR is detected. 10_B **Fallingedge**, Bit T13R is set if a falling edge of signal T13HR is detected. 11_B **Edge**, Bit T13R is set if an edge of signal T13HR is detected.

Register description CCU7

Timer Control Register 2x

TCTR2x (x=4-6)
Timer Control Register 2x **($00D0_H$+(x-4)*4)** **RESET_TYPE_5 Value: 0000 0000_H**

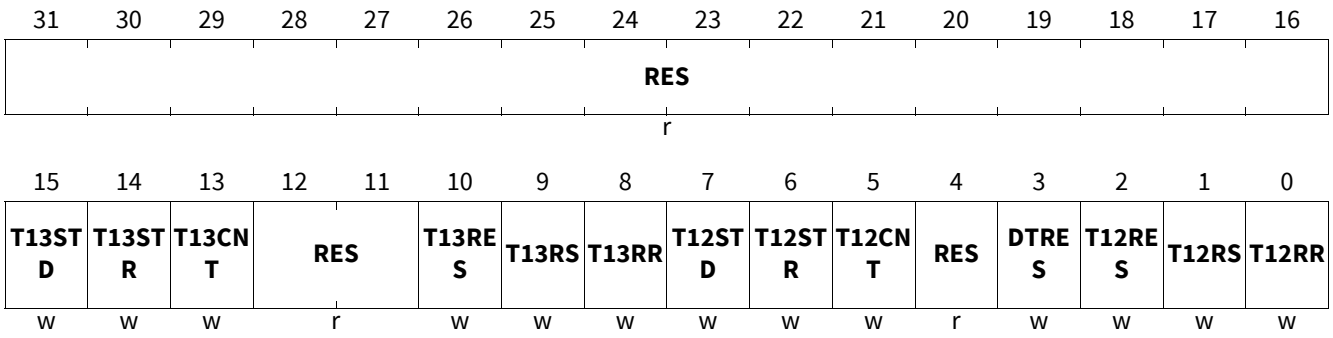


Field	Bits	Type	Description
T1xSSC	0	rw	**Timer T1x Single Shot Control** This bit controls the single shot-mode of T1x. 0_B **Disabled**, The single-shot mode is disabled, no hardware action on T12R. 1_B **Enabled**, The single shot mode is enabled, the bit T12R is reset by hardware if: T12 reaches its period value in edge-aligned mode T12 reaches the value 1 while down counting in center-aligned mode. In parallel to the reset action of bit T12R, the bits CC7xST (x = 0, 1, 2) are reset.
T1xTEC	3:1	rw	**T1x Trigger Event Control** Bit field T1xTEC selects the trigger event to startT1x (automatic set of T1xR for synchronization to T12 compare signals) according to following combinations: refer to Table-6
T1xTED	5:4	rw	**Timer T1x Trigger Event Direction** Bit field T1xTED delivers additional information to control the automatic set of bit T1xR in the case that the triggeraction defined by T1xTEC is detected. Refer to Table-7
RES	7:6, 15:10, 31:16	r	**Reserved** Returns 0 if read.
T1xRSEL	9:8	rw	**Timer T1x External Run Selection** Bit field T1xRSEL defines the event of signal T1xHRthat can set the run bit T1xR by hardware. 00_B **Disabled**, The external setting of T1xR is disabled. 01_B **Risingedge**, Bit T1xR is set if a rising edge of signal T1xHR is detected. 10_B **Fallingedge**, Bit T1xR is set if a falling edge of signal T1xHR is detected. 11_B **Edge**, Bit T1xR is set if an edge of signal T1xHR is detected.

Timer Control Register 4

Register TCTR4 provides software-control (independent set and clear conditions) for the run bits T12R and T13R. Furthermore, the timers can be reset (while running) and bits STE12 and STE13 can be controlled by software. Reading these bits always returns 0.

TCTR4
Timer Control Register 4 **($00DC_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
T12RR	0	w	**Timer T12 Run Reset** Setting this bit resets the T12R bit. 0_B **No_Change**, T12R is not changed. 1_B **T12R_cleared**, T12R is cleared, T12 stops counting.
T12RS	1	w	**Timer T12 Run Set** Setting this bit sets the T12R bit. 0_B **No_change**, T12R is not changed. 1_B **T12R_set**, T12R is set, T12 counts.
T12RES	2	w	**Timer T12 Reset** 0_B **No_change**, No change on T12. 1_B **Zero**, The T12 counter register is reset to zero. The switching of the output signals is according to the switching rules. Setting of T12RES has no impact on bit T12R.
DTRES	3	w	**Dead-Time Counter Reset** 0_B **No_change**, No effect on the dead-time counters. 1_B **Zero**, The three dead-time counter channels are reset to zero.
RES	4	r	**Reserved** Returns 0 if read.
T12CNT	5	w	**Timer T12 Count Event** 0_B **No_change**, No effect on T12 1_B **Count**, If enabled (PISEL2), timer T12 counts one step.
T12STR	6	w	**Timer T12 Shadow Transfer Request** 0_B **No_change**, no change to STE12 1_B **STE12_set**, STE12 is set, enabling the shadow transfer.
T12STD	7	w	**Timer T12 Shadow Transfer Disable** 0_B **No_change**, no change to STE12 1_B **STE12reset**, STE12 is reset without triggering the shadow transfer.

Field	Bits	Type	Description
T13RR	8	w	**Timer T13 Run Reset** Setting this bit resets the T13R bit. 0_B **No_change**, T13R is not changed. 1_B **T13R_cleared**, T13R is cleared, T13 stops counting.
T13RS	9	w	**Timer T13 Run Set** Setting this bit sets the T13R bit. 0_B **No_change**, T13R is not changed. 1_B **T13R_set**, T13R is set, T13 counts.
T13RES	10	w	**Timer T13 Reset** 0_B **No_change**, No effect on T13. 1_B **Zero**, The T13 counter register is reset to zero. The switching of the output signals is according to the switching rules. Setting of T13RES has no impact on bit T13R.
RES	12:11, 31:16	r	**Reserved** Returns 0 if read.
T13CNT	13	w	**Timer T13 Count Event** 0_B **No_change**, No effect on T13 1_B **Count**, If enabled (PISEL2), timer T13 counts one step.
T13STR	14	w	**Timer T13 Shadow Transfer Request** 0_B **No_change**, No effect on STE13 1_B **STE13set**, STE13 is set, enabling the shadow transfer.
T13STD	15	w	**Timer T13 Shadow Transfer Disable** 0_B **No_change**, No effect on STE13 1_B **STE13reset**, STE13 is reset without triggering the shadow transfer.

Timer Control Register 4x

TCTR4x (x=4-6)
Timer Control Register 4x **($00E0_H$+(x-4)*4)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES									T1xSTD	T1xSTR	T1xCNT	RES	T1xRES	T1xRS	T1xRR
r									w	w	w	r	w	w	w

Field	Bits	Type	Description
T1xRR	0	w	**Timer T1x Run Reset** Setting this bit resets the T1xR bit. 0_B **Noinfluence**, T1xR is not influenced. 1_B **T1xRcleared**, T1xR is cleared, T1x stops counting.

Field	Bits	Type	Description
T1xRS	1	w	**Timer T1x Run Set** Setting this bit sets the T1xR bit. 0_B **Noinfluence**, T1xR is not influenced. 1_B **T1xRset**, T1xR is set, T1x counts.
T1xRES	2	w	**Timer T1x Reset** 0_B **Noeffect**, No effect on T1x. 1_B **Zero**, The T1x counter register is reset to zero. The switching of the output signals is according to the switching rules. Setting of T1xRES has no impact on bit T1xR.
RES	3, 15:7, 31:16	r	**Reserved** Returns 0 if read.
T1xCNT	4	w	**Timer T1x Count Event** 0_B **Noaction**, 1_B **Count**, If enabled (PISEL2x), timer T1x counts one step.
T1xSTR	5	w	**Timer T1x Shadow Transfer Request** 0_B **Noaction**, 1_B **STE1xset**, STE1x is set, enabling the shadow transfer.
T1xSTD	6	w	**Timer T1x Shadow Transfer Disable** 0_B **Noaction**, 1_B **STE1xreset**, STE1x is reset without triggering the shadow transfer.

Modulation Control Register

MODCTR
Modulation Control Register **($00EC_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECT13O	RES	T13MODEN						MCMEN	RES	T12MODEN					
rw	r	rw						rw	r	rw					

Field	Bits	Type	Description
T12MODEN	5:0	rw	**T12 Modulation Enable** Setting these bits enables the modulation of the corresponding compare channel by a PWM pattern generated by timerT12. The bit positions are corresponding to the following output signals: Bit 0: modulation of CC70 Bit 1: modulation of COUT70 Bit 2: modulation of CC71 Bit 3: modulation of COUT71 Bit 4: modulation of CC72 Bit 5: modulation of COUT72 The enable feature of the modulation is defined as follows: 00_H **Disabled**, The modulation of the corresponding output signal by a T12 PWM pattern is disabled. 01_H **Enabled**, The modulation of the corresponding output signal by a T12 PWM pattern is enabled.
RES	6, 14, 31:16	r	**Reserved** Returns 0 if read.
MCMEN	7	rw	**Multi-Channel Mode Enable** 0_B **Disabled**, The modulation of the corresponding output signal by a multi-channel pattern according to bit field MCMOUT is disabled. 1_B **Enabled**, The modulation of the corresponding output signal by a multi-channel pattern according to bit field MCMOUT is enabled.
T13MODEN	13:8	rw	**T13 Modulation Enable** Setting these bits enables the modulation of the corresponding compare channel by a PWM pattern generated by timer T13. The bit positions are corresponding to the following output signals: Bit 0: modulation of CC70 Bit 1: modulation of COUT70 Bit 2: modulation of CC71 Bit 3: modulation of COUT71 Bit 4: modulation of CC72 Bit 5: modulation of COUT72 The enable feature of the modulation is defined as follows: 00_H **Disabled**, The modulation of the corresponding output signal by a T13 PWM pattern is disabled. 01_H **Enabled**, The modulation of the corresponding output signal by a T13 PWM pattern is enabled.
ECT13O	15	rw	**Enable Compare Timer T13 Output** 0_B **Disabled**, The alternate output function COUT73 is disabled. 1_B **Enabled**, The alternate output function COUT73 is enabled for the PWM signal generated by T13.

Trap Control Register

The register TRPCTR controls the trap functionality. It contains independent enable bits for each output signal and control bits to select the behavior in case of a trap condition. The trap condition is a low level on

the CTRAP input pin, that is monitored (inverted level) by bit IS. TRPF. While TRPF=1 (trap input active), the trap state bit IS.TRPS is set to 1.

TRPCTR
Trap Control Register **($00F0_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRPPEN	TRPEN13	TRPEN						RES					TRPM2	TRPM1	TRPM0
rw	rw	rw						r					rw	rw	rw

Field	Bits	Type	Description
TRPM0	0	rw	**Trap Mode Control Bit 0** Together with bit TRPM1, these two bits define the behavior of theselected outputs when leaving the trap state after the trap condition has become inactiveagain. A synchronization to the timer driving the PWM pattern avoids unintended pulses whenleaving the trap state. The behavior resulting from the combination [TRPM1, TRPM0]: 00: The trap state is left (return to normal operation) after TRPF has become 0 againwhen a zero-match of T12 (while counting up) is detected (synchronization to T12). 01: The trap state is left (return to normal operation) after TRPF has become 0 againwhen a zero-match of T13 is detected (synchronization to T13). 10: Reserved 11: The trap state is left (return to normal operation) immediately after TRPF has become0 again without any synchronization to T12 or T13.
TRPM1	1	rw	**Trap Mode Control Bit 1** Together with bit TRPM0, these two bits define the behavior of theselected outputs when leaving the trap state after the trap condition has become inactiveagain. A synchronization to the timer driving the PWM pattern avoids unintended pulses whenleaving the trap state. The behavior resulting from the combination [TRPM1, TRPM0]: 00: The trap state is left (return to normal operation) after TRPF has become 0 againwhen a zero-match of T12 (while counting up) is detected (synchronization to T12). 01: The trap state is left (return to normal operation) after TRPF has become 0 againwhen a zero-match of T13 is detected (synchronization to T13). 10: Reserved 11: The trap state is left (return to normal operation) immediately after TRPF has become0 again without any synchronization to T12 or T13.

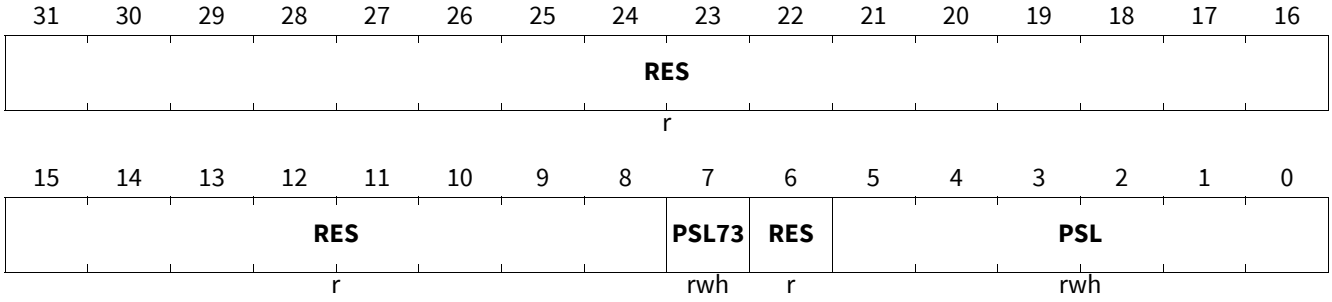
Field	Bits	Type	Description
TRPM2	2	rw	**Trap Mode Control Bit 2** This bit defines how the trap flag TRPF can be cleared after the trap input condition ($\overline{\text{CTRAP}}$ = 0 and TRPPEN = 1) is no longer valid (either by $\overline{\text{CTRAP}}$ = 1 or by TRPPEN = 0). 0_B Automatic Mode: Bit TRPF is cleared by HW if the trap input condition is no longer valid. 1_B Manual Mode: Bit TRPF stays 1 after the trap input condition is no longer valid. It has to be cleared by SW by writing ISR.RTRPF = 1.
RES	7:3, 31:16	r	**Reserved** Returns 0 if read.
TRPEN	13:8	rw	**Trap Enable Control** Setting these bits enables the trap functionality for the following corresponding output signals: Bit 0: trap functionality of CC70 Bit 1: trap functionality of COUT70 Bit 2: trap functionality of CC71 Bit 3: trap functionality of COUT71 Bit 4: trap functionality of CC72 Bit 5: trap functionality of COUT72 The enable feature of the trap functionality is defined as follows: 00_H **Disabled**, The trap functionality of the corresponding output signal is disabled. The output state is independent from bit TRPS. 01_H **Enabled**, The trap functionality of the corresponding output signal is enabled. The output is set to the passive state while TRPS = 1.
TRPEN13	14	rw	**Trap Enable Control for Timer T13** 0_B **Disabled**, The trap functionality for T13 is disabled. Timer T13 (if selected and enabled) provides PWM functionality even while TRPS = 1. 1_B **Enabled**, The trap functionality for T13 is enabled. The timer T13 PWM output signal is set to the passive state while TRPS = 1.
TRPPEN	15	rw	**Trap Pin Enable** This bit enables the input (pin) function for the trap generation. An interrupt can only be generated if a falling edge is detected at pin CTRAP while TRPPEN = 1. 0_B **Disable**, The CCU7 trapfunctionality based on the input $\overline{\text{CTRAP}}$ is disabled. A CCU7 trap can only be generated by SW by settingbit TRPF. 1_B **Enable**, The CCU7 trapfunctionality based on the input $\overline{\text{CTRAP}}$ is enabled. A CCU7 trap can be generated by SW by setting bitTRPF or by $\overline{\text{CTRAP}}$=0.

Passive State Level Register

Register PSLR defines the passive state level driven by the output pins of the module. The passive state level is the value that is driven by the port pin during the passive state of the output. During the active state, the corresponding output pin drives the active state level, which is the inverted passive state level. The passive state level permits the adaptation of the driven output levels to the driver polarity (inverted, not inverted) of the connected power stage. The bits in this register have shadow bit fields to permit a concurrent update of

all PWM-related parameters (bit field PSL is updated with T12_ST, whereas PSL73 is updated with T13_ST). The actually used values can be read (attribute rh), whereas the shadow bits can only be written (attribute w).

PSLR
Passive State Level Register **(00F4$_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**



Field	Bits	Type	Description
PSL	5:0	rwh	**Compare Outputs Passive State Level** The bits of this bit field define the passive level driven by the module outputs during the passive state. The bit positions are: Bit 0: passive level for output CC70 Bit 1: passive level for output COUT70 Bit 2: passive level for output CC71 Bit 3: passive level for output COUT71 Bit 4: passive level for output CC72 Bit 5: passive level for output COUT72 The value of each bit position is defined as: 00_H **Level0**, The passive level is 0. 01_H **Level1**, The passive level is 1.
RES	6, 15:8, 31:16	r	**Reserved** Returns 0 if read.
PSL73	7	rwh	**Passive State Level of Output COUT73** This bit field defines the passive level of the output pin COUT73. 0_B **Level0**, The passive level is 0. 1_B **Level1**, The passive level is 1.

Multi-Channel Mode Output Shadow Register

Register MCMOUTS contains bits used as pattern input for the multi-channel mode and the Hall mode. This register is a shadow register (that can be read and written) for register MCMOUT, which indicates the currently active signals.

MCMOUTS

Multi-Channel Mode Output Shadow Register (00F8$_H$) **RESET_TYPE_5 Value: 0000 0000$_H$**

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

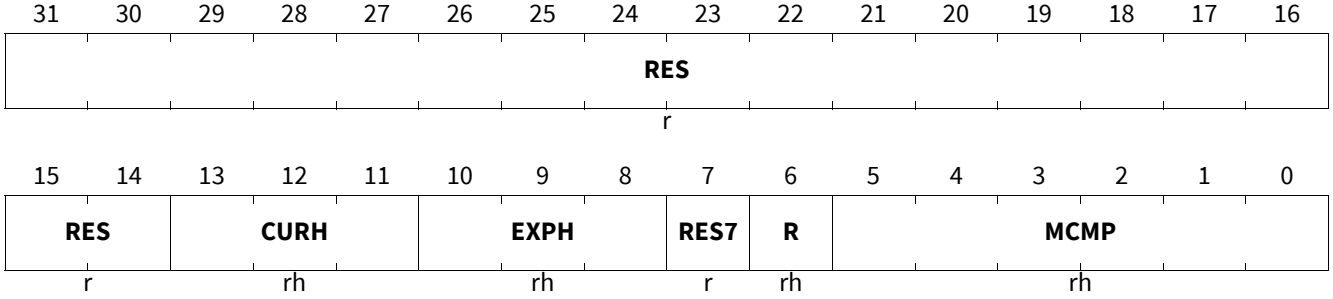
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STRHP	RES	CURHS			EXPHS			STRMCM	RES	MCMPS					
w	r	rw			rw			w	r	rw					

Field	Bits	Type	Description
MCMPS	5:0	rw	**Multi-Channel PWM Pattern Shadow** Bit field MCMPS is the shadow bit field for bitfield MCMP. The multi-channel shadow transfer is triggered according to the transfer conditions defined by register MCMCTR.
RES	6, 14, 31:16	r	**Reserved** Returns 0 if read.
STRMCM	7	w	**Shadow Transfer Request for MCMPS** Writing STRMCM = 1 leads to an immediate activation of MCM_ST to update bit field MCMP by the value of MCMPS. When read, this bit always delivers 0. 0_B **NoAction**, No action. 1_B **Update**, Bit field MCMP is updated.
EXPHS	10:8	rw	**Expected Hall Pattern Shadow** Bit field EXPHS is the shadow bit field for bit field EXPH. The shadow transfer takesplace when a correct Hall event is detected (CM_CHE).
CURHS	13:11	rw	**Current Hall Pattern Shadow** Bit field CURHS is the shadow bit field for bit field CURH. The shadow transfer takes place whena correct Hall event is detected (CM_CHE).
STRHP	15	w	**Shadow Transfer Request for the Hall Pattern** Writing STRHP = 1 leads to an immediate activation of HP_ST to update bit fields EXPH and CURH by EXPHS and CURHS. When read, this bit always delivers 0. 0_B **NoAction**, No action. 1_B **Update**, Bit fields EXPH and CURH are updated.

Multi-Channel Mode Output Register

Register MCMOUT shows the multi-channel control bits that are currently used. Register MCMOUT is defined as follows:

MCMOUT
Multi-Channel Mode Output Register **($00FC_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**



Field	Bits	Type	Description
MCMP	5:0	rh	**Multi-Channel PWM Pattern** Bit field MCMP is written by a shadow transfer from bit field MCMPS. It contains the output pattern for the multi-channel mode. If this mode is enabled by bit MCMEN in register MODCTR, the output state of the following output signal can be modified: Bit 0: multi-channel state for output CC70 Bit 1: multi-channel state for output COUT70 Bit 2: multi-channel state for output CC71 Bit 3: multi-channel state for output COUT71 Bit 4: multi-channel state for output CC72 Bit 5: multi-channel state for output COUT72 The multi-channel patterns can set the related output to the passive state. While IDLE = 1, bit field MCMP is cleared. 00_H **Passive**, The output is set to the passive state. The PWM generated by T12 or T13 is not taken into account. 01_H **PWM**, The output can deliver the PWM generated by T12 or T13 (according to register MODCTR).
R	6	rh	**Reminder Flag** This reminder flag indicates that the shadow transfer from bit field MCMPS to MCMP has been requested by the selected trigger source. This bit is cleared when the shadow transfer takes place or while MCMEN = 0. 0_B **Noshadowtransfer**, Currently, no shadow transfer from MCMPS to MCMP is requested. 1_B **Shadowtransfer**, A shadow transfer from MCMPS to MCMP has been requested by the selected trigger source, but it has not yet been executed, because the selected synchronization condition has not yet occurred.
RES7	7	r	**Reserved** Returns 0 if read.

Field	Bits	Type	Description
EXPH	10:8	rh	**Expected Hall Pattern** Bit field EXPH is updated by a shadow transfer HP_ST from bit field EXPHS. If HCRDY = 1, EXPH is compared to the sampled CCPOSx inputs in order to detect the occurrence of the next desired (=expected) hall pattern or a wrong pattern. If the sampled hall pattern at the hall input pins is equal to bit field EXPH, a correct Hall event has been detected (CM_CHE).
CURH	13:11	rh	**Current Hall Pattern** Bit field CURH is updated by a shadow transfer HP_ST from bit field CURHS. If HCRDY = 1, CURH is compared to the sampled CCPOSx inputs in order to detect a spike. If the sampled Hall pattern at the Hall input pins is equal to bit field CURH, no Hall event has been detected. If the sampled Hall input pattern is neither equal to CURH nor equal to EXPH, the Hall event was not the desired one and may be due to a fatal error (e.g. blocked rotor, etc.). In this case, a wrong Hall event has been detected (CM_WHE).
RES	15:14, 31:16	r	**Reserved** Returns 0 if read.

Multi-Channel Mode Control Register

Register MCMCTR contains control bits for the multi-channel functionality.

MCMCTR
Multi-Channel Mode Control Register **(0100_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES					STE13U	STE12D	STE12U	RES		SWSYN		RES	SWSEL		
r					rw	rw	rw	r		rw		r	rw		

Field	Bits	Type	Description
SWSEL	2:0	rw	**Switching Selection** Bit field SWSEL selects one of the following trigger request sources (next multi-channel event) for the shadow transfer from MCMPS to MCMP. The trigger request is stored in the reminder flag R until the shadow transfer is done and flag R is cleared automatically with the shadow transfer. The shadow transfer takes place synchronously with an event selected in bit field SWSYN. 000_B **Norequest**, no trigger request will be generated 001_B **Correctpattern**, correct hall pattern on CCPOSx detected 010_B **T13periodmatch**, T13 period-match detected (while counting up) 011_B **T12onematch**, T12 one-match (while counting down) 100_B **T12channel1comparematch**, T12 channel 1 compare-match detected (phase delay function) 101_B **T12periodmatch**, T12 period match detected (while counting up) else reserved, no trigger request will be generated
RES	3, 7:6, 15:11, 31:16	r	**Reserved**
SWSYN	5:4	rw	**Switching Synchronization** Bit field SWSYN triggers the shadow transfer between MCMPS and MCMP if it has been requested before (flag R set by an event selected by SWSEL). This feature permits the synchronization of the outputs to the PWM source, that is used for modulation (T12or T13). 00_B **Direct**, the trigger event directly causes the shadow transfer 01_B **T13zeromatch**, T13 zero-match triggers the shadow transfer 10_B **T12zeromatch**, a T12 zero-match (while counting up) triggers the shadow transfer 11_B **Reserved**, reserved; no action
STE12U	8	rw	**Shadow Transfer Enable for T12 Upcounting** This bit enables the shadow transfer T12_ST if flag MCMOUT.R is set or becomes set while a T12 period match is detected while counting up. 0_B **Noaction**, 1_B **Enabled**, The T12_ST shadow transfer mechanism is enabled if MCMEN = 1.
STE12D	9	rw	**Shadow Transfer Enable for T12 Downcounting** This bit enables the shadow transfer T12_ST if flag MCMOUT.R is set or becomes set while a T12 one match is detected while counting down. 0_B **Noaction**, 1_B **Enabled**, The T12_ST shadow transfer mechanism is enabled if MCMEN = 1.

Field	Bits	Type	Description
STE13U	10	rw	**Shadow Transfer Enable for T13 Upcounting** This bit enables the shadow transfer T13_ST if flag MCMOUT.R is set or becomes set while a T13 period match is detected. 0_B **Noaction**, 1_B **Enabled**, The T13_ST shadow transfer mechanism is enabled if MCMEN = 1.

Input Monitoring Register

The input monitoring register monitors the occurrence of a selected event for the input signals. If a hardware event triggers the setting of bit IMON.x and the same bit is written with 1 via software at the same time, then the corresponding bit is cleared (software overrules hardware). The lost bit event is indicated if an event is detected again at one or more input signals with its lost indicator enabled.

Note: The register is only applicable in capture modes if the edges areselected through T12MSEL.MSEL7x.

IMON
Input Monitoring Register **(0104_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						T13HRI	T12HRI	CTRAPI	CC72INI	CC71INI	CC70INI	CCPOS2I	CCPOS1I	CCPOS0I	LBE
r						rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
LBE	0	rwh	**Lost Bit Event** This bit determines if a lost bit event has occurred. A lost bit event occurs when a selected event occurs again with the previous event captured (IMON.x remains set) and its lost indicator is enabled, for at least one of the monitored input signals. The bit can be cleared by writing a 1 to the same bit position, while writing a 0 has no effect. 0_B The lost bit event has not occurred. 1_B The lost bit event has occurred.

Field	Bits	Type	Description
CCPOSxI (x=0-2)	x+1	rwh	**Event indication for input signal CCPOSx** The bit determines if the selected event has occurred via an edge detection. The bit can be cleared by writing a 1 to the same bit position, while writing a 0 has no effect. *Note: The dedicated edge is indicated for a selected event if Hysteretic-likeControl or Capture modes are initialized in T12MSEL.MSEL7x. If these modes are not selected,then all edges will be indicated as an event for the inputs.* 0_B A selected event has not occurred. 1_B Edge detection indicates a selected event has occurred.
CC7xINI (x=0-2)	x+4	rwh	**Event indication for input signal CC7xIN** The bit determines if the selected event has occurred via an edge detection. The bit can be cleared by writing a 1 to the same bit position, while writing a 0 has no effect. 0_B A selected event has not occurred. 1_B Edge detection indicates a selected event has occurred.
CTRAPI	7	rwh	**Event indication for input signal CTRAP** The bit determines if the selected event has occurred via an edge detection. The bit can be cleared by writing a 1 to the same bit position, while writing a 0 has no effect. 0_B An event has not occurred. 1_B Edge detection indicates an event has occurred.
T12HRI	8	rwh	**Event indication for input signal T12HR** The bit determines if the selected event has occurred via an edge detection. The bit can be cleared by writing a 1 to the same bit position, while writing a 0 has no effect. 0_B An event has not occurred. 1_B Edge detection indicates an event has occurred.
T13HRI	9	rwh	**Event indication for input signal T13HR** The bit determines if the selected event has occurred via an edge detection. The bit can be cleared by writing a 1 to the same bit position, while writing a 0 has no effect. 0_B An event has not occurred. 1_B Edge detection indicates an event has occurred.
RES	31:10	r	**Reserved; - 0** Returns 0 if read; should be written with 0.

Lost Indicator Register

The lost indicator register has the lost indicator enable bits for its detected event at the input signals. The lost bit event can then be enabled as an output signal through one of the service request lines.

LI

Lost Indicator Register **(0108_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INPLBE		LBEEN	RES			T13HREN	T12HREN	CTRAPEN	CC72INEN	CC71INEN	CC70INEN	CCPOS2EN	CCPOS1EN	CCPOS0EN	RES
rw		rw	r			rw	rw	rw	rw	rw	rw	rw	rw	rw	r

Field	Bits	Type	Description
RES	0, 12:10, 31:16	r	**Reserved; - 0** Returns 0 if read; should be written with 0.
CCPOSxEN (x=0-2)	x+1	rw	**Lost Indicator Enable for input signal CCPOSx** This bit determines if the monitored event at the input signal is enabled for the detection of a lost bit event. 0_B Input signal is disabled for a lost bit event detection. 1_B Input signal is enabled for a lost bit event detection.
CC7xINEN (x=0-2)	x+4	rw	**Lost Indicator Enable for input signal CC7xIN** This bit determines if the monitored event at the input signal is enabled for the detection of a lost bit event. 0_B Input signal is disabled for a lost bit event detection. 1_B Input signal is enabled for a lost bit event detection.
CTRAPEN	7	rw	**Lost Indicator Enable for input signal CTRAP** This bit determines if the monitored event at the input signal is enabled for the detection of a lost bit event. 0_B Input signal is disabled for a lost bit event detection. 1_B Input signal is enabled for a lost bit event detection.
T12HREN	8	rw	**Lost Indicator Enable for input signal T12HR** This bit determines if the monitored event at the input signal is enabled for the detection of a lost bit event. 0_B Input signal is disabled for a lost bit event detection. 1_B Input signal is enabled for a lost bit event detection.
T13HREN	9	rw	**Lost Indicator Enable for input signal T13HR** This bit determines if the monitored event at the input signal is enabled for the detection of a lost bit event. 0_B Input signal is disabled for a lost bit event detection. 1_B Input signal is enabled for a lost bit event detection.
LBEEN	13	rw	**Interrupt Enable for Lost Bit Event** This bit determines if a SRx line is activated if lost bit event is detected. 0_B Lost bit event is disabled for the activation of a SRx line. 1_B Lost bit event is enabled for the activation of a SRx line.

Field	Bits	Type	Description
INPLBE	15:14	rw	**Interrupt Node Pointer for lost bit event** This bit field defines which service request output line is selected to output an lost event alert for an enabled lost bit event. 00_B Service request output SR0 is selected. 01_B Service request output SR1 is selected. 10_B Service request output SR2 is selected. 11_B Service request output SR3 is selected.

Capture/Compare Interrupt Status Register 1

Register IS contains the individual interrupt request bits. This register can only be read; write actions have no impact on the contents of this register. The software can set or reset the bits individually by writing to the registers ISS (to set the bits) or to register ISR (to reset the bits).

The interrupt generation is independent from the value of the bits in register IS, e.g. the interrupt will be generated (if enabled) even if the corresponding bit is already set. The trigger for an interrupt generation is the detection of a set condition (by HW or SW) for the corresponding bit in register IS.

In compare mode (and hall mode), the timer-related interrupts are only generated while the timer is running (T1xR = 1). In capture mode, the capture interrupts are also generated while the timer T12 is stopped.

IS

Capture/Compare Interrupt Status Register 1 ($010C_H$) **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STR	IDLE	WHE	CHE	TRPS	TRPF	T13PM	T13CM	T12PM	T12OM	ICC72F	ICC72R	ICC71F	ICC71R	ICC70F	ICC70R
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ICC70R	0	rh	**Capture, Compare-Match Rising Edge Flag** In compare mode, a compare-match has been detected while T12 was counting up. In capture mode, a rising edge has been detected at the input CC70. 0_B **Notoccurred**, The event has not yet occurred since this bit has been reset for the last time. 1_B **Detected**, The event described above has been detected.
ICC70F	1	rh	**Capture, Compare-Match Falling Edge Flag** In compare mode, a compare-match has been detected while T12 was counting down. In capture mode, a falling edge has been detected at the input CC70. 0_B **Notoccurred**, The event has not yet occurred since this bit has been reset for the last time. 1_B **Detected**, The event described above has been detected.

Field	Bits	Type	Description
ICC71R	2	rh	**Capture, Compare-Match Rising Edge Flag** In compare mode, a compare-match has been detected while T12 was counting up. In capture mode, a rising edge has been detected at the input CC71. 0_B **Notoccurred**, The event has not yet occurred since this bit has been reset for the last time. 1_B **Detected**, The event described above has been detected.
ICC71F	3	rh	**Capture, Compare-Match Falling Edge Flag** In compare mode, a compare-match has been detected while T12 was counting down. In capture mode, a falling edge has been detected at the input CC71. 0_B **Notoccurred**, The event has not yet occurred since this bit has been reset for the last time. 1_B **Detected**, The event described above has been detected.
ICC72R	4	rh	**Capture, Compare-Match Rising Edge Flag** In compare mode, a compare-match has been detected while T12 was counting up. In capture mode, a rising edge has been detected at the input CC72. 0_B **Notoccurred**, The event has not yet occurred since this bit has been reset for the last time. 1_B **Detected**, The event described above has been detected.
ICC72F	5	rh	**Capture, Compare-Match Falling Edge Flag** In compare mode, a compare-match has been detected while T12 was counting down. In capture mode, a falling edge has been detected at the input CC72. 0_B **Notoccurred**, The event has not yet occurred since this bit has been reset for the last time. 1_B **Detected**, The event described above has been detected.
T12OM	6	rh	**Timer T12 One-Match Flag** 0_B **Notdetected**, A timer T12 one-match (while counting down) has not yet been detected since this bit has been reset for the last time. 1_B **Detected**, A timer T12 one-match (while counting down) has been detected.
T12PM	7	rh	**Timer T12 Period-Match Flag** 0_B **Notdetected**, A timer T12 period-match (while counting up) has not yet been detected since this bit has been reset for the last time. 1_B **Detected**, A timer T12 period-match (while counting up) has been detected.
T13CM	8	rh	**Timer T13 Compare-Match Flag** 0_B **Notdetected**, A timer T13 compare-match has not yet been detected since this bit has been reset for the last time. 1_B **Detected**, A timer T13 compare-match has been detected.
T13PM	9	rh	**Timer T13 Period-Match Flag** 0_B **Notdetected**, A timer T13 period-match has not yet been detected since this bit has been reset for the last time. 1_B **Detected**, A timer T13 period-match has been detected.

Field	Bits	Type	Description
TRPF	10	rh	**Trap Flag** The trap flag TRPF will be set by hardware if TRPPEN= 1 and $\overline{\text{CTRAP}}$ = 0 or by software. If TRPM2 = 0, bit TRPF is reset by hardware if the input $\overline{\text{CTRAP}}$ becomes inactive (TRPPEN= 1). If TRPM2 = 1, bit TRPF must be reset by software in order to leave the trap state. 0_B **Notdetected**, The trap condition has not been detected. 1_B **Detected**, The trap condition has been detected (input
TRPS	11	rh	**Trap State** During the trap state, the selected outputs a reset to the passive state. The logic level driven during the passive state is defined by the corresponding bit in register PSLR. Bit TRPS = 1 and TRPF = 0 can occur if the trap condition is no longer active but the selected synchronization has not yet taken place. 0_B **Notactive**, The trap state is not active. 1_B **Active**, The trap state is active. Bit TRPS is set while bit TRPF = 1. It is reset according to the mode selected in register TRPCTR.
CHE	12	rh	**Correct Hall Event** On every valid hall edge, the contents of EXPH are compared with the pattern on pin CCPOSx and if equal bit CHE is set. 0_B **Notdetected**, A transition to a correct (= expected) hall event has not yet been detected since this bit has been reset for the last time. 1_B **Detected**, A transition to a correct (= expected) hall event has been detected.
WHE	13	rh	**Wrong Hall Event** On every valid hall edge, the contents of EXPH are compared with the pattern on pin CCPOSx. If both comparisons(CURH and EXPH with CCPOSx) are not true, bit WHE (wrong hall event)is set. 0_B **Notdetected**, A transition to a wrong hall event (not the expected one) has not yet been detected since this bit has been reset for the last time. 1_B **Detected**, A transition to a wrong hall event (not the expected one) has been detected.
IDLE	14	rh	**IDLE State** This bit is set together with bit WHE (wrong hall event) and it must be reset by software. 0_B **Noaction**, 1_B **Idle**, Bit field MCMP is cleared and held to 0, the selected outputs are set to passive state.
STR	15	rh	**Multi-Channel Mode Shadow Transfer Request** This bit is set when a shadow transfer from MCMOUTS to MCMOUT takes places in multi-channel mode. 0_B **No**, The shadow transfer has not yet taken place. 1_B **Yes**, The shadow transfer has taken place.
RES	31:16	r	**Reserved** Returns 0 if read.

Capture/Compare Interrupt Status Set Register 1

Register ISS contains individual interrupt request set bits to generate a CCU7 interrupt request by software. Writing a 1 sets the bit(s) in register IS at the corresponding bit position(s) and can generate an interrupt event (if available and enabled). All bit positions read as 0.

ISS

Capture/Compare Interrupt Status Set Register 1(0110_H) **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSTR	SIDLE	SWHE	SCHE	SWHC	STRPF	ST13PM	ST13CM	ST12PM	ST12OM	SCC72F	SCC72R	SCC71F	SCC71R	SCC70F	SCC70R
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
SCC70R	0	w	**Set Capture, Compare-Match Rising Edge Flag** 0_B **Noaction**, 1_B **Set**, Bit CC70R in register IS will be set.
SCC70F	1	w	**Set Capture, Compare-Match Falling Edge Flag** 0_B **Noaction**, 1_B **Set**, Bit CC70F in register IS will be set.
SCC71R	2	w	**Set Capture, Compare-Match Rising Edge Flag** 0_B **Noaction**, 1_B **Set**, Bit CC71R in register IS will be set.
SCC71F	3	w	**Set Capture, Compare-Match Falling Edge Flag** 0_B **Noaction**, 1_B **Set**, Bit CC71F in register IS will be set.
SCC72R	4	w	**Set Capture, Compare-Match Rising Edge Flag** 0_B **Noaction**, 1_B **Set**, Bit CC72R in register IS will be set.
SCC72F	5	w	**Set Capture, Compare-Match Falling Edge Flag** 0_B **Noaction**, 1_B **Set**, Bit CC72F in register IS will be set.
ST12OM	6	w	**Set Timer T12 One-Match Flag** 0_B **Noaction**, 1_B **Set**, Bit T12OM in register IS will be set.
ST12PM	7	w	**Set Timer T12 Period-Match Flag** 0_B **Noaction**, 1_B **Set**, Bit T12PM in register IS will be set.
ST13CM	8	w	**Set Timer T13 Compare-Match Flag** 0_B **Noaction**, 1_B **Set**, Bit T13CM in register IS will be set.

Field	Bits	Type	Description
ST13PM	9	w	**Set Timer T13 Period-Match Flag** 0_B **Noaction**, 1_B **Set**, Bit T13PM in register IS will be set.
STRPF	10	w	**Set Trap Flag** 0_B **Noaction**, 1_B **Set**, Bits TRPF and TRPS in register IS will be set.
SWHC	11	w	**Software Hall Compare** 0_B **Noaction**, 1_B **Set**, The Hall compare action is triggered.
SCHE	12	w	**Set Correct Hall Event Flag** 0_B **Noaction**, 1_B **Set**, Bit CHE in register IS will be set.
SWHE	13	w	**Set Wrong Hall Event Flag** 0_B **Noaction**, 1_B **Set**, Bit WHE in register IS will be set.
SIDLE	14	w	**Set IDLE Flag** 0_B **Noaction**, 1_B **Set**, Bit IDLE in register IS will be set.
SSTR	15	w	**Set STR Flag** 0_B **Noaction**, 1_B **Set**, Bit STR in register IS will be set.
RES	31:16	r	**Reserved** Returns 0 if read.

Capture/Compare Interrupt Status Reset Register 1

Register ISR contains bits to individually clear the interrupt event flags by software. Writing a 1 clears the bit(s) in register IS at the corresponding bit position(s). All bit positions read as 0.

ISR

Capture/Compare Interrupt Status Reset Register 1(0114_H) **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSTR	**RIDLE**	**RWHE**	**RCHE**	**RES11**	**RTRPF**	**RT13PM**	**RT13CM**	**RT12PM**	**RT12OM**	**RCC72F**	**RCC72R**	**RCC71F**	**RCC71R**	**RCC70F**	**RCC70R**
w	w	w	w	r	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RCC70R	0	w	**Reset Capture, Compare-Match Rising Edge Flag** 0_B **Noaction**, 1_B **Reset**, Bit CC70R in register IS will be cleared.

Field	Bits	Type	Description
RCC70F	1	w	**Reset Capture, Compare-Match Falling Edge Flag** 0_B **Noaction**, 1_B **Reset**, Bit CC70F in register IS will be cleared.
RCC71R	2	w	**Reset Capture, Compare-Match Rising Edge Flag** 0_B **Noaction**, 1_B **Reset**, Bit CC71R in register IS will be cleared.
RCC71F	3	w	**Reset Capture, Compare-Match Falling Edge Flag** 0_B **Noaction**, 1_B **Reset**, Bit CC71F in register IS will be cleared.
RCC72R	4	w	**Reset Capture, Compare-Match Rising Edge Flag** 0_B **Noaction**, 1_B **Reset**, Bit CC72R in register IS will be cleared.
RCC72F	5	w	**Reset Capture, Compare-Match Falling Edge Flag** 0_B **Noaction**, 1_B **Reset**, Bit CC72F in register IS will be cleared.
RT12OM	6	w	**Reset Timer T12 One-Match Flag** 0_B **Noaction**, 1_B **Reset**, Bit T12OM in register IS will be cleared.
RT12PM	7	w	**Reset Timer T12 Period-Match Flag** 0_B **Noaction**, 1_B **Reset**, Bit T12PM in register IS will be cleared.
RT13CM	8	w	**Reset Timer T13 Compare-Match Flag** 0_B **Noaction**, 1_B **Reset**, Bit T13CM in register IS will be cleared.
RT13PM	9	w	**Reset Timer T13 Period-Match Flag** 0_B **Noaction**, 1_B **Reset**, Bit T13PM in register IS will be cleared.
RTRPF	10	w	**Reset Trap Flag** 0_B **Noaction**, 1_B **Reset**, Bit TRPF in register IS will be cleared (not taken into account while input
RES11	11	r	**Reserved** Returns 0 if read.
RCHE	12	w	**Reset Correct Hall Event Flag** 0_B **Noaction**, 1_B **Reset**, Bit CHE in register IS will be cleared.
RWHE	13	w	**Reset Wrong Hall Event Flag** 0_B **Noaction**, 1_B **Reset**, Bit WHE in register IS will be cleared.
RIDLE	14	w	**Reset IDLE Flag** 0_B **Noaction**, 1_B **Reset**, Bit IDLE in register IS will be cleared.
RSTR	15	w	**Reset STR Flag** 0_B **Noaction**, 1_B **Reset**, Bit STR in register IS will be cleared.

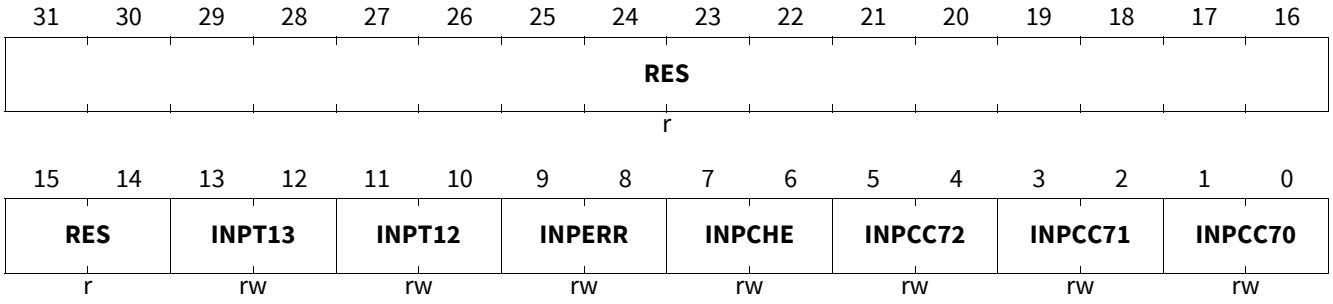
Field	Bits	Type	Description
RES	31:16	r	**Reserved** Returns 0 if read.

Capture/Compare Interrupt Node Pointer Register 1

Register INP contains the interrupt node pointers allowing a flexible interrupt handling. These bit fields define which service request output will be activated if the corresponding interrupt event occurs and the interrupt generation for this event is enabled.

INP

Capture/Compare Interrupt Node Pointer Register 1(0118_H) **RESET_TYPE_5 Value: $0000\ 3940_H$**



Field	Bits	Type	Description
INPCC70	1:0	rw	**Interrupt Node Pointer for Channel 0 Interrupts** This bit field defines the interrupt output line, which is activated due to a set condition for bit CC70R (if enabled by bit ENCC70R) or for bit CC70F (if enabled by bit ENCC70F). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.
INPCC71	3:2	rw	**Interrupt Node Pointer for Channel 1 Interrupts** This bit field defines the interrupt output line, which is activated due to a set condition for bit CC71R (if enabled by bit ENCC71R) or for bit CC71F (if enabled by bit ENCC71F). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.
INPCC72	5:4	rw	**Interrupt Node Pointer for Channel 2 Interrupts** This bit field defines the interrupt output line, which is activated due to a set condition for bit CC72R (if enabled by bit ENCC72R) or for bit CC72F (if enabled by bit ENCC72F). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.

Field	Bits	Type	Description
INPCHE	7:6	rw	**Interrupt Node Pointer for the CHE Interrupt** This bit field defines the interrupt output line, which is activated due to a set condition for bit CHE (if enabled by bit ENCHE) or for bit STR (if enabled by bit ENSTR). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.
INPERR	9:8	rw	**Interrupt Node Pointer for Error Interrupts** This bit field defines the interrupt output line,which is activated due to a set condition for bit TRPF (if enabledby bit ENTRPF) or for bit WHE (if enabled by bit ENWHE). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.
INPT12	11:10	rw	**Interrupt Node Pointer for Timer T12 Interrupts** This bit field defines the interrupt output line,which is activated due to a set condition for bit T12OM (if enabledby bit ENT12OM) or for bit T12PM (if enabled by bit ENT12PM). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.
INPT13	13:12	rw	**Interrupt Node Pointer for Timer T13 Interrupts** This bit field defines the interrupt output line,which is activated due to a set condition for bit T13CM (if enabledby bit ENT13CM) or for bit T13PM (if enabled by bit ENT13PM). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.
RES	15:14, 31:16	r	**Reserved** Returns 0 if read.

Capture/Compare Interrupt Enable Register 1

Register IEN contains the interrupt enable bits and a control bit to enable the automatic idle function in the case of a wrong hall pattern.

IEN

Capture/Compare Interrupt Enable Register 1 (011C$_H$) **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENSTR	ENIDLE	ENWHE	ENCHE	RES	ENTRPF	ENT13PM	ENT13CM	ENT12PM	ENT12OM	ENCC72F	ENCC72R	ENCC71F	ENCC71R	ENCC70F	ENCC70R
rw	rw	rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENCC70R	0	rw	**Capture, Compare-Match Rising Edge Interrupt Enable for Channel 0** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CC70R in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CC70R in register IS occurs. The interrupt line that will be activated is selected by bit field INPCC70.
ENCC70F	1	rw	**Capture, Compare-Match Falling Edge Interrupt Enable for Channel 0** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CC70F in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CC70F in register IS occurs. The interrupt line that will be activated is selected by bit field INPCC70.
ENCC71R	2	rw	**Capture, Compare-Match Rising Edge Interrupt Enable for Channel 1** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CC71R in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CC71R in register IS occurs. The interrupt line that will be activated is selected by bit field INPCC71.
ENCC71F	3	rw	**Capture, Compare-Match Falling Edge Interrupt Enable for Channel 1** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CC71F in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CC71F in register IS occurs. The interrupt line that will be activated is selected by bit field INPCC71.
ENCC72R	4	rw	**Capture, Compare-Match Rising Edge Interrupt Enable for Channel 2** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CC72R in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CC72R in register IS occurs. The interrupt line that will be activated is selected by bit field INPCC72.

Field	Bits	Type	Description
ENCC72F	5	rw	**Capture, Compare-Match Falling Edge Interrupt Enable for Channel 2** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CC72F in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CC72F in register IS occurs. The interrupt line that will be activated is selected by bit field INPCC72.
ENT12OM	6	rw	**Enable Interrupt for T12 One-Match** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit T12OM in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit T12OM in register IS occurs. The interrupt line that will be activated is selected by bit field INPT12.
ENT12PM	7	rw	**Enable Interrupt for T12 Period-Match** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit T12PM in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit T12PM in register IS occurs. The interrupt line that will be activated is selected by bit field INPT12.
ENT13CM	8	rw	**Enable Interrupt for T13 Compare-Match** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit T13CM in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit T13CM in register IS occurs. The interrupt line that will be activated is selected by bit field INPT13.
ENT13PM	9	rw	**Enable Interrupt for T13 Period-Match** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit T13PM in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit T13PM in register IS occurs. The interrupt line that will be activated is selected by bit field INPT13.
ENTRPF	10	rw	**Enable Interrupt for Trap Flag** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit TRPF in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit TRPF in register IS occurs. The interrupt line that will be activated is selected by bit field INPERR.
RES	11, 31:16	r	**Reserved** Returns 0 if read.
ENCHE	12	rw	**Enable Interrupt for Correct Hall Event** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CHE in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CHE in register IS occurs. The interrupt line that will be activated is selected by bit field INPCHE.

Field	Bits	Type	Description
ENWHE	13	rw	**Enable Interrupt for Wrong Hall Event** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit WHE in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit WHE in register IS occurs. The interrupt line that will be activated is selected by bit field INPERR.
ENIDLE	14	rw	**Enable Idle** This bit enables the automatic entering of the idle state (bit IDLE will be set) after a wrong hall event has been detected (bit WHE is set). During the idle state, the bit field MCMP is automatically cleared. 0_B **IDLEnotset**, The bit IDLE is not automatically set when a wrong hall event is detected. 1_B **IDLEset**, The bit IDLE is automatically set when a wrong hall event is detected.
ENSTR	15	rw	**Enable Multi-Channel Mode Shadow Transfer Interrupt** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit STR in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit STR in register IS occurs. The interrupt line that will be activated is selected by bit field INPCHE.

Capture/Compare Interrupt Status Register 2

IS_2
Capture/Compare Interrupt Status Register 2 (0120_H) **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		T16PM	T16CM	T15PM	T15CM	T14PM	T14CM	RES		ICC72BF	ICC72BR	ICC71BF	ICC71BR	ICC70BF	ICC70BR
r		rh	rh	rh	rh	rh	rh	r		rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ICC70BR	0	rh	**Capture, Compare-Match Rising Edge Flag** A compare-match has been detectedwhile T12 was counting up. 0_B **Notoccurred**, The event has not yet occurred since this bit has been reset for the last time. 1_B **Detected**, The event described above has been detected.
ICC70BF	1	rh	**Capture, Compare-Match Falling Edge Flag** A compare-match has been detectedwhile T12 was counting down. 0_B **Notoccurred**, The event has not yet occurred since this bit has been reset for the last time. 1_B **Detected**, The event described above has been detected.

Field	Bits	Type	Description
ICC71BR	2	rh	**Capture, Compare-Match Rising Edge Flag** A compare-match has been detectedwhile T12 was counting up. 0_B **Notoccurred**, The event has not yet occurred since this bit has been reset for the last time. 1_B **Detected**, The event described above has been detected.
ICC71BF	3	rh	**Capture, Compare-Match Falling Edge Flag** A compare-match has been detectedwhile T12 was counting down. 0_B **Notoccurred**, The event has not yet occurred since this bit has been reset for the last time. 1_B **Detected**, The event described above has been detected.
ICC72BR	4	rh	**Capture, Compare-Match Rising Edge Flag** A compare-match has been detectedwhile T12 was counting up. 0_B **Notoccurred**, The event has not yet occurred since this bit has been reset for the last time. 1_B **Detected**, The event described above has been detected.
ICC72BF	5	rh	**Capture, Compare-Match Falling Edge Flag** A compare-match has been detectedwhile T12 was counting down. 0_B **Notoccurred**, The event has not yet occurred since this bit has been reset for the last time. 1_B **Detected**, The event described above has been detected.
RES	7:6, 15:14, 31:16	r	**Reserved** Returns 0 if read.
T14CM	8	rh	**Timer T14 Compare-Match Flag** 0_B **Notdetected**, A timer T14 compare-match has not yet been detected since this bit has been reset for the last time. 1_B **Detected**, A timer T14 compare-match has been detected.
T14PM	9	rh	**Timer T14 Period-Match Flag** 0_B **Notdetected**, A timer T14 period-match has not yet been detected since this bit has been reset for the last time. 1_B **Detected**, A timer T14 period-match has been detected.
T15CM	10	rh	**Timer T15 Compare-Match Flag** 0_B **Notdetected**, A timer T15 compare-match has not yet been detected since this bit has been reset for the last time. 1_B **Detected**, A timer T15 compare-match has been detected.
T15PM	11	rh	**Timer T15 Period-Match Flag** 0_B **Notdetected**, A timer T15 period-match has not yet been detected since this bit has been reset for the last time. 1_B **Detected**, A timer T15 period-match has been detected.
T16CM	12	rh	**Timer T16 Compare-Match Flag** 0_B **Notdetected**, A timer T16 compare-match has not yet been detected since this bit has been reset for the last time. 1_B **Detected**, A timer T16 compare-match has been detected.

Field	Bits	Type	Description
T16PM	13	rh	**Timer T16 Period-Match Flag** 0_B **Notdetected**, A timer T16 period-match has not yet been detected since this bit has been reset for the last time. 1_B **Detected**, A timer T16 period-match has been detected.

Capture/Compare Interrupt Status Set Register 2

ISS_2

Capture/Compare Interrupt Status Set Register 2(0124_H) **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		ST16PM	ST16CM	ST15PM	ST15CM	ST14PM	ST14CM	RES		SCC72BF	SCC72BR	SCC71BF	SCC71BR	SCC70BF	SCC70BR
r		w	w	w	w	w	w	r		w	w	w	w	w	w

Field	Bits	Type	Description
SCC70BR	0	w	**Set Capture, Compare-Match Rising Edge Flag** 0_B **Noaction**, 1_B **Set**, Bit CC70BR in register IS_2 will be set.
SCC70BF	1	w	**Set Capture, Compare-Match Falling Edge Flag** 0_B **Noaction**, 1_B **Set**, Bit CC70BF in register IS_2 will be set.
SCC71BR	2	w	**Set Capture, Compare-Match Rising Edge Flag** 0_B **Noaction**, 1_B **Set**, Bit CC71BR in register IS_2 will be set.
SCC71BF	3	w	**Set Capture, Compare-Match Falling Edge Flag** 0_B **Noaction**, 1_B **Set**, Bit CC71BF in register IS_2 will be set.
SCC72BR	4	w	**Set Capture, Compare-Match Rising Edge Flag** 0_B **Noaction**, 1_B **Set**, Bit CC72BR in register IS_2 will be set.
SCC72BF	5	w	**Set Capture, Compare-Match Falling Edge Flag** 0_B **Noaction**, 1_B **Set**, Bit CC72BF in register IS_2 will be set.
RES	7:6, 15:14, 31:16	r	**Reserved** Returns 0 if read.
ST14CM	8	w	**Set Timer T14 Compare-Match Flag** 0_B **Noaction**, 1_B **Set**, Bit T14CM in register IS_2 will be set.

Field	Bits	Type	Description
ST14PM	9	w	**Set Timer T14 Period-Match Flag** 0_B **Noaction**, 1_B **Set**, Bit T14PM in register IS_2 will be set.
ST15CM	10	w	**Set Timer T15 Compare-Match Flag** 0_B **Noaction**, 1_B **Set**, Bit T15CM in register IS_2 will be set.
ST15PM	11	w	**Set Timer T15 Period-Match Flag** 0_B **Noaction**, 1_B **Set**, Bit T15PM in register IS_2 will be set.
ST16CM	12	w	**Set Timer T16 Compare-Match Flag** 0_B **Noaction**, 1_B **Set**, Bit T16CM in register IS_2 will be set.
ST16PM	13	w	**Set Timer T16 Period-Match Flag** 0_B **Noaction**, 1_B **Set**, Bit T16PM in register IS_2 will be set.

Capture/Compare Interrupt Status Reset Register 2

ISR_2

Capture/Compare Interrupt Status Reset Register 2(0128_H) **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		RT16PM	RT16CM	RT15PM	RT15CM	RT14PM	RT14CM	RES		RCC72BF	RCC72BR	RCC71BF	RCC71BR	RCC70BF	RCC70BR
r		w	w	w	w	w	w	r		w	w	w	w	w	w

Field	Bits	Type	Description
RCC70BR	0	w	**Reset Capture, Compare-Match Rising Edge Flag** 0_B **Noaction**, 1_B **Reset**, Bit CC70BR in register IS_2 will be cleared.
RCC70BF	1	w	**Reset Capture, Compare-Match Falling Edge Flag** 0_B **Noaction**, 1_B **Reset**, Bit CC70BF in register IS_2 will be cleared.
RCC71BR	2	w	**Reset Capture, Compare-Match Rising Edge Flag** 0_B **Noaction**, 1_B **Reset**, Bit CC71BR in register IS_2 will be cleared.
RCC71BF	3	w	**Reset Capture, Compare-Match Falling Edge Flag** 0_B **Noaction**, 1_B **Reset**, Bit CC71BF in register IS_2 will be cleared.

Field	Bits	Type	Description
RCC72BR	4	w	**Reset Capture, Compare-Match Rising Edge Flag** 0_B **Noaction**, 1_B **Reset**, Bit CC72BR in register IS_2 will be cleared.
RCC72BF	5	w	**Reset Capture, Compare-Match Falling Edge Flag** 0_B **Noaction**, 1_B **Reset**, Bit CC72BF in register IS_2 will be cleared.
RES	7:6, 15:14, 31:16	r	**Reserved** Returns 0 if read.
RT14CM	8	w	**Reset Timer T14 Compare-Match Flag** 0_B **Noaction**, 1_B **Reset**, Bit T14CM in register IS_2 will be cleared.
RT14PM	9	w	**Reset Timer T14 Period-Match Flag** 0_B **Noaction**, 1_B **Reset**, Bit T14PM in register IS_2 will be cleared.
RT15CM	10	w	**Reset Timer T15 Compare-Match Flag** 0_B **Noaction**, 1_B **Reset**, Bit T15CM in register IS_2 will be cleared.
RT15PM	11	w	**Reset Timer T15 Period-Match Flag** 0_B **Noaction**, 1_B **Reset**, Bit T15PM in register IS_2 will be cleared.
RT16CM	12	w	**Reset Timer T16 Compare-Match Flag** 0_B **Noaction**, 1_B **Reset**, Bit T16CM in register IS_2 will be cleared.
RT16PM	13	w	**Reset Timer T16 Period-Match Flag** 0_B **Noaction**, 1_B **Reset**, Bit T16PM in register IS_2 will be cleared.

Capture/Compare Interrupt Node Pointer Register 2

INP_2

Capture/Compare Interrupt Node Pointer Register 2($012C_H$) **RESET_TYPE_5 Value: 0000 3F00$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		INPT16		INPT15		INPT14		RES		INPCC72B		INPCC71B		INPCC70B	
r		rw		rw		rw		r		rw		rw		rw	

Field	Bits	Type	Description
INPCC70B	1:0	rw	**Interrupt Node Pointer for Channel 0 Interrupts** This bit field defines the interrupt output line,which is activated due to a set condition for bit CC70R (if enabledby bit ENCC70R) or for bit CC70F (if enabled by bit ENCC70F). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.
INPCC71B	3:2	rw	**Interrupt Node Pointer for Channel 1 Interrupts** This bit field defines the interrupt output line,which is activated due to a set condition for bit CC71R (if enabledby bit ENCC71R) or for bit CC71F (if enabled by bit ENCC71F). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.
INPCC72B	5:4	rw	**Interrupt Node Pointer for Channel 2 Interrupts** This bit field defines the interrupt output line,which is activated due to a set condition for bit CC72R (if enabledby bit ENCC72R) or for bit CC72F (if enabled by bit ENCC72F). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.
RES	7:6, 15:14, 31:16	r	**Reserved** Returns 0 if read.
INPT14	9:8	rw	**Interrupt Node Pointer for Timer T14 Interrupts** This bit field defines the interrupt output line,which is activated due to a set condition for bit T14CM (if enabledby bit ENT14CM) or for bit T14PM (if enabled by bit ENT14PM). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.
INPT15	11:10	rw	**Interrupt Node Pointer for Timer T15 Interrupts** This bit field defines the interrupt output line,which is activated due to a set condition for bit T15CM (if enabledby bit ENT15CM) or for bit T15PM (if enabled by bit ENT15PM). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.

Field	Bits	Type	Description
INPT16	13:12	rw	**Interrupt Node Pointer for Timer T16 Interrupts** This bit field defines the interrupt output line,which is activated due to a set condition for bit T16CM (if enabledby bit ENT16CM) or for bit T16PM (if enabled by bit ENT16PM). 00_B **SR0**, Interrupt output line SR0 is selected. 01_B **SR1**, Interrupt output line SR1 is selected. 10_B **SR2**, Interrupt output line SR2 is selected. 11_B **SR3**, Interrupt output line SR3 is selected.

Capture/Compare Interrupt Enable Register 2

IEN_2
Capture/Compare Interrupt Enable Register 2 (0130_H) **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		ENT16PM	ENT16CM	ENT15PM	ENT15CM	ENT14PM	ENT14CM	RES		ENCC72BF	ENCC72BR	ENCC71BF	ENCC71BR	ENCC70BF	ENCC70BR
r		rw	rw	rw	rw	rw	rw	r		rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENCC70BR	0	rw	**Capture, Compare-Match Rising Edge Interrupt Enable for Channel 0** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CC70R in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CC70R in register IS occurs. The interrupt line that will be activated is selected by bit field INPCC70.
ENCC70BF	1	rw	**Capture, Compare-Match Falling Edge Interrupt Enable for Channel 0** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CC70F in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CC70F in register IS occurs. The interrupt line that will be activated is selected by bit field INPCC70.
ENCC71BR	2	rw	**Capture, Compare-Match Rising Edge Interrupt Enable for Channel 1** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CC71R in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CC71R in register IS occurs. The interrupt line that will be activated is selected by bit field INPCC71.

Field	Bits	Type	Description
ENCC71BF	3	rw	**Capture, Compare-Match Falling Edge Interrupt Enable for Channel 1** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CC71F in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CC71F in register IS occurs. The interrupt line that will be activated is selected by bit field INPCC71.
ENCC72BR	4	rw	**Capture, Compare-Match Rising Edge Interrupt Enable for Channel 2** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CC72R in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CC72R in register IS occurs. The interrupt line that will be activated is selected by bit field INPCC72.
ENCC72BF	5	rw	**Capture, Compare-Match Falling Edge Interrupt Enable for Channel 2** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit CC72F in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit CC72F in register IS occurs. The interrupt line that will be activated is selected by bit field INPCC72.
RES	7:6, 15:14, 31:16	r	**Reserved** Returns 0 if read.
ENT14CM	8	rw	**Enable Interrupt for T14 Compare-Match** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit T14CM in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit T14CM in register IS occurs. The interrupt line that will be activated is selected by bit field INPT14.
ENT14PM	9	rw	**Enable Interrupt for T14 Period-Match** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit T14PM in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit T14PM in register IS occurs. The interrupt line that will be activated is selected by bit field INPT14.
ENT15CM	10	rw	**Enable Interrupt for T15 Compare-Match** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit T15CM in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit T15CM in register IS occurs. The interrupt line that will be activated is selected by bit field INPT15.
ENT15PM	11	rw	**Enable Interrupt for T15 Period-Match** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit T15PM in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit T15PM in register IS occurs. The interrupt line that will be activated is selected by bit field INPT15.

Field	Bits	Type	Description
ENT16CM	12	rw	**Enable Interrupt for T16 Compare-Match** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit T16CM in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit T16CM in register IS occurs. The interrupt line that will be activated is selected by bit field INPT16.
ENT16PM	13	rw	**Enable Interrupt for T16 Period-Match** 0_B **Nointerrupt**, No interrupt will be generated if the set condition for bit T16PM in register IS occurs. 1_B **Interrupt**, An interrupt will be generated if the set condition for bit T16PM in register IS occurs. The interrupt line that will be activated is selected by bit field INPT16.

OCDS Control and Status Register

The OCDS Control and Status register OCS controls the module's behavior in suspend mode (used for debugging). The OCS register is cleared by Debug Reset.

The OCS register can only be written when the OCDS is enabled. When OCDS is disabled the OCS suspend control is ineffective.

The default state of the OTGB bus outputs of a peripheral is disabled (all bits are 0). This allows to have an ORed combination of all peripheral OTGBs within the OTGM instead of a multiplexer.

OCS
OCDS Control and Status Register **(0134_H)** **RESET_TYPE_5 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		SUSSTA	SUS_P	SUS				RES							
r		rh	w	rw				r							

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

Field	Bits	Type	Description
RES	23:0, 31:30	r	**Reserved - 0** Read as 0; must be written with 0.
SUS	27:24	rw	**OCDS Suspend Control** Controls the sensitivity to the ocds_suspend input. 0_H Will not suspend 1_H Hard suspend. Clock is switched off immediately. 2_H Soft suspend, Stop Mode 0 3_H Soft suspend, Stop Mode 1 **others**, reserved
SUS_P	28	w	**SUS Write Protection** SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.

Field	Bits	Type	Description
SUSSTA	29	rh	**Suspend State** 0_B Module is not (yet) suspended 1_B Module is suspended

Fractional Dividerx

T_FDIVx (x=0-1)
Fractional Dividerx **(0138_H+x*4)** **RESET_TYPE_5 Value: 0000 0000_H**

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

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

Field	Bits	Type	Description
PQ	15:0	rw	**PQ** Dividend/Divisor settings for fractional Clock Divider Interpretation is defined by T_CLK_CTRL.FDIVx_SEL setting: FDIVx_SEL 0: Q = [7:0] ; P = [15:8] FDIVx_SEL 1: Q = [9:0]; P = [15:10]
RES	31:16	r	**Reserved** Returns 0 if read.

Timer Clock Selection

T_CLK_CTRL
Timer Clock Selection **(0140_H)** **RESET_TYPE_5 Value: 0000 0000_H**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		FDIV1_SEL	FDIV0_SEL	DT_CLK_SEL		T16_CLK_SEL		T15_CLK_SEL		T14_CLK_SEL		T13_CLK_SEL		T12_CLK_SEL	
r		rw	rw	rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
T12_CLK_SEL	1:0	rw	**Clock selector for Timer 12** 00_B **NODIV**, No Clock Division 01_B **FDIV0**, Clock from Clock Divider 0 10_B **FDIV1**, Clock from Clock Divider 1 11_B **NOTUSED**, No Clock Division
T13_CLK_SEL	3:2	rw	**Clock selector for Timer 13** 00_B **NODIV**, No Clock Division 01_B **FDIV0**, Clock from Clock Divider 0 10_B **FDIV1**, Clock from Clock Divider 1 11_B **NOTUSED**, No Clock Division
T14_CLK_SEL	5:4	rw	**Clock selector for Timer 14** 00_B **NODIV**, No Clock Division 01_B **FDIV0**, Clock from Clock Divider 0 10_B **FDIV1**, Clock from Clock Divider 1 11_B **NOTUSED**, No Clock Division
T15_CLK_SEL	7:6	rw	**Clock selector for Timer 15** 00_B **NODIV**, No Clock Division 01_B **FDIV0**, Clock from Clock Divider 0 10_B **FDIV1**, Clock from Clock Divider 1 11_B **NOTUSED**, No Clock Division
T16_CLK_SEL	9:8	rw	**Clock selector for Timer 16** 00_B **NODIV**, No Clock Division 01_B **FDIV0**, Clock from Clock Divider 0 10_B **FDIV1**, Clock from Clock Divider 1 11_B **NOTUSED**, No Clock Division
DT_CLK_SEL	11:10	rw	**Clock Selector for dead time control** 00_B **NODIV**, No Clock Division 01_B **FDIV0**, Clock from Clock Divider 0 10_B **FDIV1**, Clock from Clock Divider 1 11_B **NOTUSED**, No Clock Division
FDIV0_SEL	12	rw	**FDIV0 Mode Selection** 0_B **8by8**, T_FDIV0[7:0] = Q; TFDIV[15:8] = P 1_B **6by10**, T_FDIV0[9:0] = Q; TFDIV[15:10] = P
FDIV1_SEL	13	rw	**FDIV1 Mode Selection** 0_B **8by8**, T_FDIV0[7:0] = Q; TFDIV[15:8] = P 1_B **6by10**, T_FDIV0[9:0] = Q; TFDIV[15:10] = P
RES	15:14, 31:16	r	**Reserved** Returns 0 if read.

Timer Control Register 3

Register TCTR3 provides software-control for (set and clear) for the shadow transferfunctionality of all timers. (STE12-16)

TCTR3

Timer Control Register 3 **(0144_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES			T16STD	T15STD	T14STD	T13STD	T12STD	RES			T16STR	T15STR	T14STR	T13STR	T12STR
	r		w	w	w	w	w		r		w	w	w	w	w

Field	Bits	Type	Description
T12STR	0	w	**Timer T12 Shadow Transfer Request** 0_B **No_change**, no change to STE12 1_B **STE12_set**, STE12 is set, enabling the shadow transfer.
T13STR	1	w	**Timer T13 Shadow Transfer Request** 0_B **No_change**, No effect on STE13 1_B **STE13set**, STE13 is set, enabling the shadow transfer.
T14STR	2	w	**Timer T14 Shadow Transfer Request** 0_B **No_change**, no change to STE14 1_B **STE14_set**, STE14 is set, enabling the shadow transfer.
T15STR	3	w	**Timer T15 Shadow Transfer Request** 0_B **No_change**, no change to STE15 1_B **STE15_set**, STE15 is set, enabling the shadow transfer.
T16STR	4	w	**Timer T16 Shadow Transfer Request** 0_B **No_change**, no change to STE16 1_B **STE16_set**, STE16 is set, enabling the shadow transfer.
RES	7:5, 15:13, 31:16	r	**Reserved** Returns 0 if read.
T12STD	8	w	**Timer T12 Shadow Transfer Disable** 0_B **No_change**, no change to STE12 1_B **STE12reset**, STE12 is reset without triggering the shadow transfer.
T13STD	9	w	**Timer T13 Shadow Transfer Disable** 0_B **No_change**, No effect on STE13 1_B **STE13reset**, STE13 is reset without triggering the shadow transfer.
T14STD	10	w	**Timer T14 Shadow Transfer Disable** 0_B **No_change**, no change to STE14 1_B **STE14reset**, STE14 is reset without triggering the shadow transfer.
T15STD	11	w	**Timer T15 Shadow Transfer Disable** 0_B **No_change**, no change to STE15 1_B **STE15reset**, STE15 is reset without triggering the shadow transfer.

Field	Bits	Type	Description
T16STD	12	w	**Timer T16 Shadow Transfer Disable** 0_B **No_change**, no change to STE16 1_B **STE16reset**, STE16 is reset without triggering the shadow transfer.

Interupt gating Register

IGT

Interupt gating Register **(0148_H)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		**GT3**		**RES**		**GT2**		**RES**		**GT1**		**RES**		**GT0**	
r		rw		r		rw		r		rw		r		rw	

Field	Bits	Type	Description
GT0	1:0	rw	**Gating SR0** 00_B **NOGT**, No gating 01_B **T12CDIR**, Gated off with T12 CDIR = 0 10_B **T12CDIRNOT**, Gated off with T12 CDIR = 1 11_B **GTIN0**, Gated off with Input SR_GT0 = 0
RES	3:2, 7:6, 11:10, 31:14	r	**Reserved** Returns 0 if read.
GT1	5:4	rw	**Gating SR1** 00_B **NOGT**, No gating 01_B **T12CDIR**, Gated off with T12 CDIR = 0 10_B **T12CDIRNOT**, Gated off with T12 CDIR = 1 11_B **GTIN1**, Gated off with Input SR_GT1 = 0
GT2	9:8	rw	**Gating SR2** 00_B **NOGT**, No gating 01_B **T12CDIR**, Gated off with T12 CDIR = 0 10_B **T12CDIRNOT**, Gated off with T12 CDIR = 1 11_B **GTIN2**, Gated off with Input SR_GT2 = 0
GT3	13:12	rw	**Gating SR3** 00_B **NOGT**, No gating 01_B **T12CDIR**, Gated off with T12 CDIR = 0 10_B **T12CDIRNOT**, Gated off with T12 CDIR = 1 11_B **GTIN3**, Gated off with Input SR_GT3 = 0

26 Bridge Driver (BDRV)

26.1 Features overview

The BDRV module consists of 6 gate drivers to control external normal-level n-channel MOSFETs arranged in 3 half bridges for 3-phase motor control applications.

The BDRV provides the following features:

- Flexible control by SFRs of Bridge Driver module, PWM output signals of CCU7 module, or alternate functions of GPIOs (see **Chapter 26.6**)
- Current-driven output stages to control external n-channel MOSFET gates with flexibly programmable gate current profile (see **Chapter 26.7**)
- Adjustable cross-conduction protection (see **Chapter 26.8**)
- High-current discharge mode to reduce dead times and to keep external MOSFETs off during fast transients (see **Chapter 26.9**)
- Safe switch-off path to switch off the Bridge Driver in a defined way in the case of errors (see **Chapter 26.10**)
- Passive pull-down mode to keep external MOSFETs off if the Bridge Driver is disabled (see **Chapter 26.11**)
- Active brake mode with reduced current consumption to statically switch on external MOSFETs (see **Chapter 26.12**)
- Timing measurements of on/off delays and on/off slope durations (see **Chapter 26.13**)
- Adaptive control mode with automatic adjustment of gate current values (see **Chapter 26.14**)
- Integrated 2-stage charge pump for low-voltage operation and statical MOSFET gate control (see **Chapter 26.15**)
- Adjustable voltage monitoring of Bridge Driver supply voltage (VSD) and charge pump output voltage (VCP) (see **Chapter 26.16**)
- Adjustable short-circuit detection in on and off state (see **Chapter 26.17**)
- Open-load detection in off state (see **Chapter 26.18**)
- Overtemperature detection and shutdown (see **Chapter 26.19**)

26.2 Block diagram

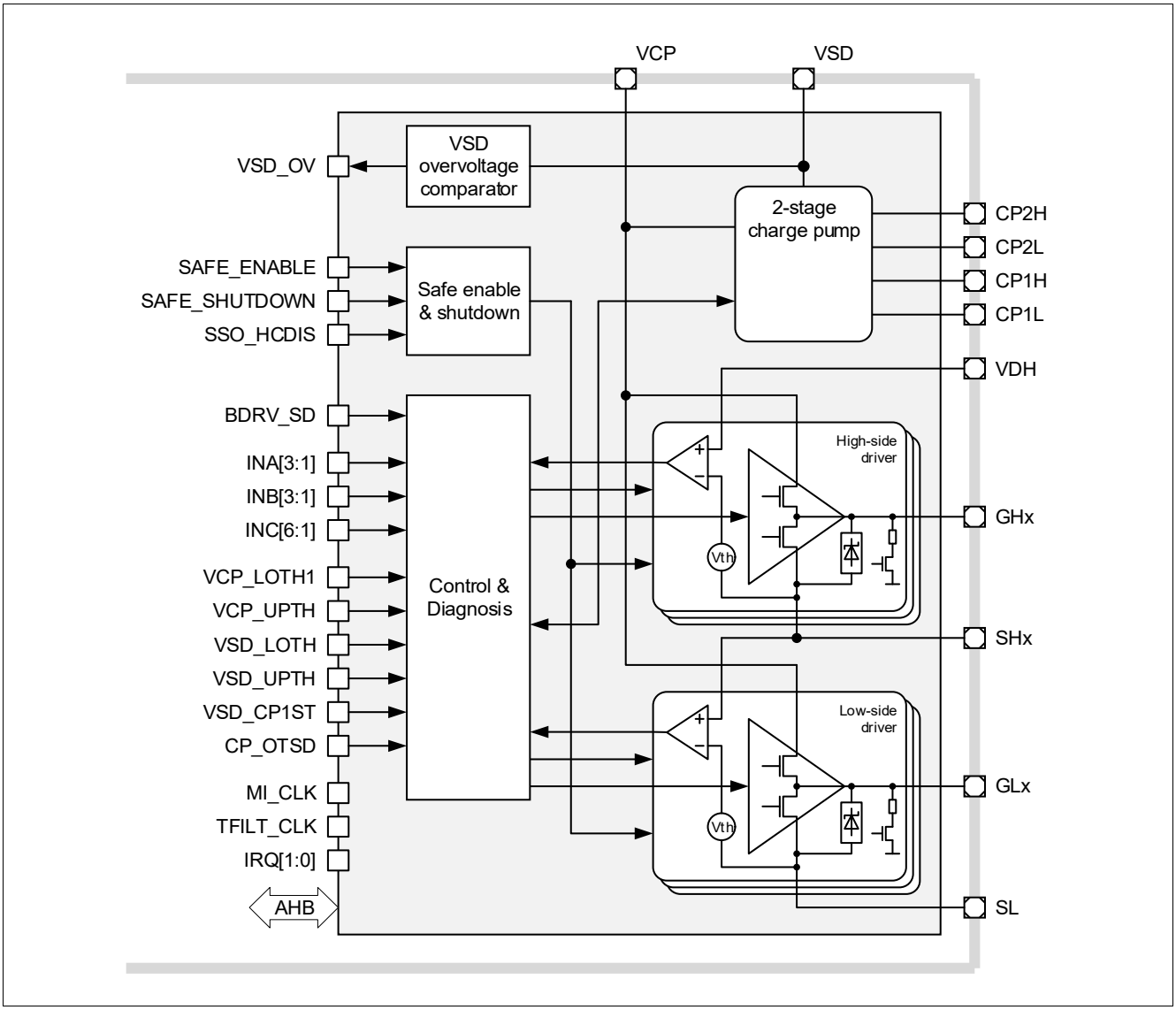


Figure 365 Block diagram BDRV

26.3 Toplevel signals

Table 315 BDRV clock domains

Signal	Direction	Description	From
MI_CLK	input	Peripheral clock	SCU
TFILT_CLK	input	Filter and blanking time clock	SCU

Table 316 BDRV control inputs

Signal	Direction	Description	From
INA[3:1]	input	PWM control signals for gate drivers	CCU7
INB[3:1]	input	PWM control signals for gate drivers	CCU7
INC[6:1]	input	PWM control signals for gate drivers	GPIOs
BDRV_SD	input	Shutdown request	SCU

Table 317 BDRV monitoring status inputs

Signal	Direction	Description	From
VCP_LOTH1, VCP_UPTH	input	VCP voltage monitoring status	ADC2
VSD_LOTH, VSD_UPTH, VSD_CP1ST	input	VSD voltage monitoring status	ADC2
CP_OTSD	input	Charge pump temperature sensor status	ADC2

Table 318 BDRV outputs

Signal	Direction	Description	To
VSD_OV	output	VSD overvoltage wake-up signal	PMU
IRQ[1:0]	output	Interrupt node output signals	SCU

Table 319 BDRV functional safety interface

Signal	Direction	Description	From
SAFE_ENABLE	input	Safe enable signal	PMU
SAFE_SHUTDOWN	input	Safe shutdown signal	PMU
SSO_HCDIS	input	Safe switch-off with high-current discharge mode	PMU

Table 320 BDRV pins

Signal	Direction	Description	From/to
VSD	input	Charge pump supply voltage	VSD pin
VCP	input/output	Charge pump output voltage/gate driver supply input voltage	VCP pin

Table 320 BDRV pins (cont'd)

Signal	Direction	Description	From/to
CP1L, CP1H, CP2L, CP2H	input/output	Charge pump external capacitors	CP1L, CP1H, CP2L, CP2H pins
VDH	input	High-side MOSFET drain voltage	VDH pin
GH1, GH2, GH3	output	High-side gate driver outputs	GH1, GH2, GH3 pins
SH1, SH2, SH3	input	High-side MOSFET source voltage	SH1, SH2, SH3 pins
GL1, GL2, GL3	output	Low-side gate driver outputs	GL1, GL2, GL3 pins
SL	input	Low-side MOSFET source voltage	SL pin

26.4 Interrupts

Events

The bridge driver provides the following events:

- Overcurrent event
- Drain-source monitoring event
- Sequencer issue event
- Cross-conduction event
- Active driver detection event
- Charge pump low detection event
- VSD overvoltage detection event during off mode (bridge driver disabled)

The events are propagated to the SCU for further processing and distribution.

Interrupt nodes

The bridge driver provides two interrupt nodes:

- IRQ0: interrupt requests from active driver detection
- IRQ1: interrupt requests from all failure monitoring features which can trigger an interrupt

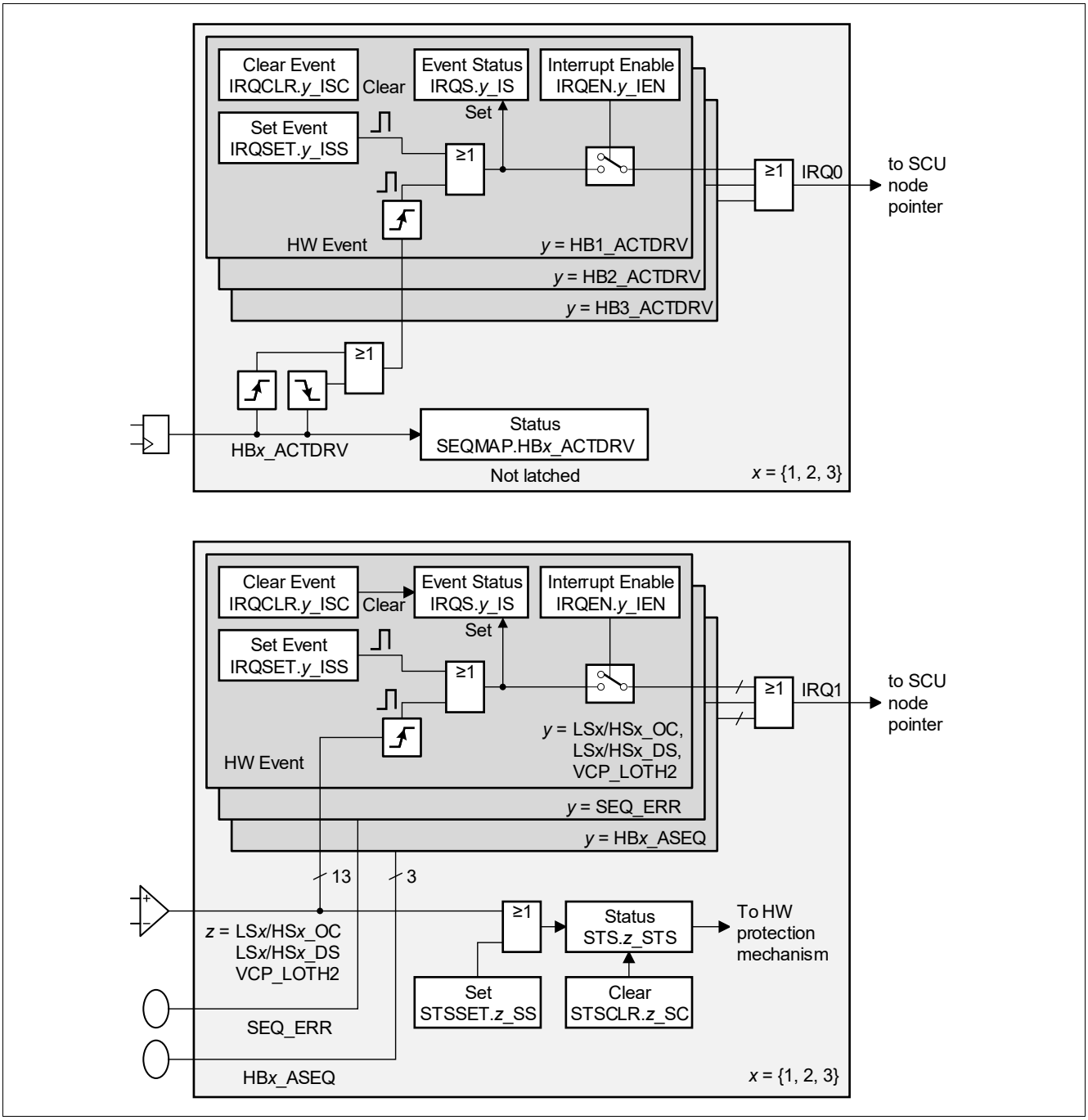


Figure 366 Interrupt handling

*Note: Details about the mentioned "HW protection mechanism" can be found in **Chapter 26.16.2** and **Chapter 26.17.1**.*

26.5 Operation mode behavior

Table 321 Operation mode behavior BDRV

Reset	• RESET_TYPE_5 resets all other bridge driver configuration settings
Power-up / power-down	• After power-up the bridge driver is disabled by default • At power-down the bridge driver is reset by RESET_TYPE_5
Active mode	• The bridge driver is off by default • The bridge driver needs to be enabled together with the CSA and the CSC by setting the respective registers
Stop mode	• The bridge driver is powered down automatically when stop mode is requested • The bridge driver configuration is lost in stop mode • The bridge driver enters safe state after wake-up from stop mode
Sleep / fail-sleep mode	• The bridge driver is powered down automatically when sleep mode is requested • The bridge driver configuration is lost in sleep mode • The bridge driver enters safe state after wake-up from sleep mode
Fail-safe state	• The bridge driver deactivates all current charge/discharge gate drivers • All gate drivers are passively pulled down to ground • The charge pump and half bridges are still enabled and register settings are kept

26.6 Flexible control

A gate driver can switch on and off an external MOSFET in two different ways, either statically by SFRs or PWM-controlled by the set of PWM input signals INAx, INBx, and INCx. In both cases all diagnostic and protection functions (short circuit, open-load, and overtemperature detection) are available.

The bridge driver enables the charge and discharge output currents of both the low-side and high-side gate driver of half bridge x if CTRL1.HBx_EN is set.

*Note: Before any half bridges can be enabled the BDRV module must be enabled by SAFE_ENABLE='1', i.e. the safe switch-off path must be released on system level (see **Chapter 26.20**).*

26.6.1 Static mode

The bridge driver switches on a gate driver LSx or HSx statically if the corresponding control bit LSx_ON or HSx_ON is set in the CTRL1 register.

26.6.2 PWM mode

The bridge driver controls a gate driver LSx or HSx with one of the PWM signals from the INAx, INBx, or INCx inputs if the corresponding control bit LSx_PWM or HSx_PWM is set in the CTRL1 register.

Notes

1. *A gate driver can only be set to PWM control if static control of both gate drivers of the corresponding half bridge x is disabled, i.e. CTRL1.LSx_ON=0 and CTRL1.HSx_ON=0.*
2. *It is recommended to configure the gate driver for PWM operation before enabling the PWM control signal generation (at INAx, INBx, and INCx) to avoid any unwanted glitches at the driver output.*

The bridge driver selects the PWM control signals for the gate drivers according to the bit fields LSx_SRC_SEL and HSx_SRC_SEL in the PWMSRCSEL register. The default (and recommended) selection is listed in the following table:

Table 322 Default gate driver mapping to the PWM input signals

Gate driver	PWMSRCSEL bit field	Reset value	Corresponding PWM input signal
HS1	HS1_SRC_SEL	000_B	INA1
HS2	HS2_SRC_SEL	001_B	INA2
HS3	HS3_SRC_SEL	010_B	INA3
LS1	LS1_SRC_SEL	100_B	INB1
LS2	LS2_SRC_SEL	101_B	INB2
LS3	LS3_SRC_SEL	110_B	INB3

Note: The bridge driver is only able to change the bit fields LSx_SRC_SEL and HSx_SRC_SEL of the PWMSRCSEL register if all LSx_PWM and HSx_PWM control bits are not set in the CTRL1 register.

26.7 Current-driven output stages

The Bridge Driver output stages generate source and sink currents to charge and discharge the gates of external n-channel MOSFETs. The gate current values are programmable to vary the slew rate at the output of the related half bridge. The maximum gate-to-source voltage which is applied to the external n-channel MOSFETs is limited internally by reducing the charge currents appropriately.

26.7.1 Overview

Figure 367 shows an overview of one switching cycle of an external MOSFET.

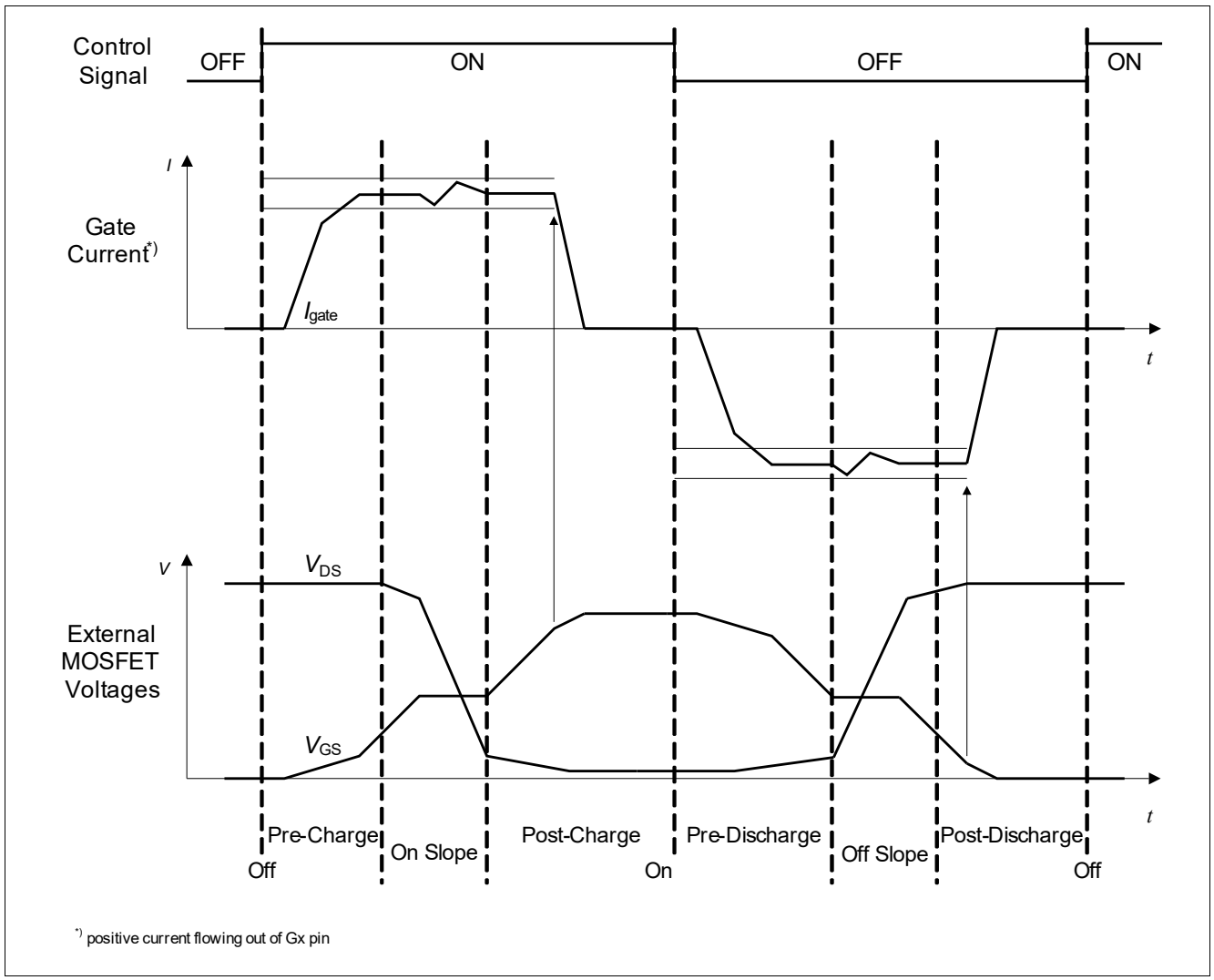


Figure 367 Switching cycle

The control input signal sets the gate driver either in charge or discharge mode, i.e. it generates a source current flowing out of the driver to charge the gate of an external MOSFET or a sink current flowing into the driver to discharge that gate.

Based on the changes on the drain-to-source voltage of the external MOSFET the charging and discharging phases can be divided into sub-phases:

- Switch-on pre-charge: the gate of the external MOSFET is pre-charged without change of the drain-to-source voltage; the external MOSFET is still off

- Switch-on slope: the gate of the external MOSFET is further charged while the external MOSFET turns on and generates the on slope at the drain-to-source voltage
- Switch-on post-charge: the gate of the external MOSFET is post-charged up to the high-level output gate-to-source voltage V_{Gxxy} of the gate driver; the $R_{DS(on)}$ of the external MOSFET decreases to its minimum value
- Switch-off pre-discharge: the gate of the external MOSFET is pre-discharged without significant change of the drain-to-source voltage; the external MOSFET is still on, but its $R_{DS(on)}$ increases
- Switch-off slope: the gate of the external MOSFET is further discharged while the external MOSFET turns off and generates the off slope at the drain-to-source voltage
- Switch-off post-discharge: the gate of the external MOSFET is post-discharged until the gate-to-source voltage is 0 V; the external MOSFET is already off

26.7.2 Control modes

There are two basic modes to program the gate current values of the output stages: constant mode and sequencer mode. After each mode has finished its respective gate current profile, both modes activate a clamping current that remains valid until the driver changes the state again. This clamping charge/discharge currents are defined per half bridge in the bit fields HBx_ICLMPON and HBx_ICLPMOFF of the HBxIGATECLMPC registers and are intended to statically keep on or off the external MOSFET, e.g. driving an external R_{GS} during on state or keeping the off state during fast voltage transients or EMI.

26.7.2.1 Constant mode

The constant mode is selected by default. One current value for the gate charging phase and one current value for the gate discharging phase is defined by SFRs, each with a duration value. The three switching sub-phases are covered with the same gate charge/discharge current value:

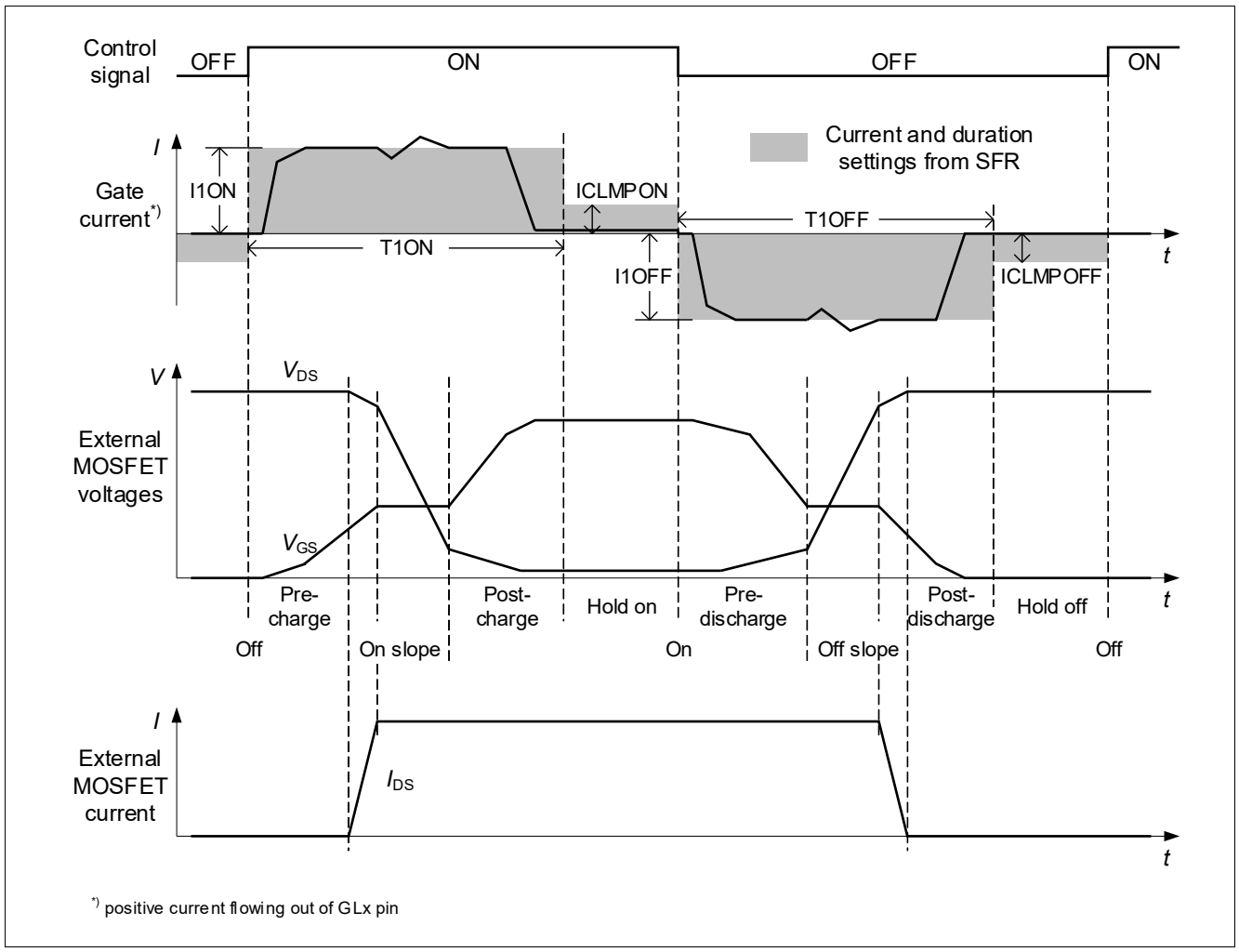


Figure 368 Gate current values in constant mode

The symbols I1ON, T1ON, ICLMPON, I1OFF, T1OFF, and ICLMPOFF shown in **Figure 368** expand to the following registers and bit fields:

- I1ON: defined per gate driver in LSxSEQONIC.LSx_I1ON and HSxSEQONIC.HSx_I1ON
- T1ON: defined per gate driver in LSxSEQONTC.LSx_T1ON and HSxSEQONTC.HSx_T1ON
- ICLMPON: defined per half bridge in HBxIGATECLMPC
- I1OFF: defined per gate driver in LSxSEQOFFIC.LSx_I1OFF and HSxSEQOFFIC.HSx_I1OFF
- T1OFF: defined per gate driver in LSxSEQOFFTC.LSx_T1OFF and HSxSEQOFFTC.HSx_T1OFF
- ICLMPOFF: defined per half bridge in HBxIGATECLMPC

26.7.2.2 Sequencer mode

The sequencer mode can be activated for each half bridge x and for switching on and switching off events separately by the respective bits HBxONSEQCNF or HBxOFFSEQCNF in the CTRL2 register. The gate charging phase and the gate discharging phase are split into consecutive sub-phases. Each of the three switching sub-phases have individual SFR settings for current values and durations. Since the sequencer mode is designed for use with inductive loads in PWM operation, the slope sub-phase has two separate charge/discharge current value and duration time settings for the current slope and the voltage slope:

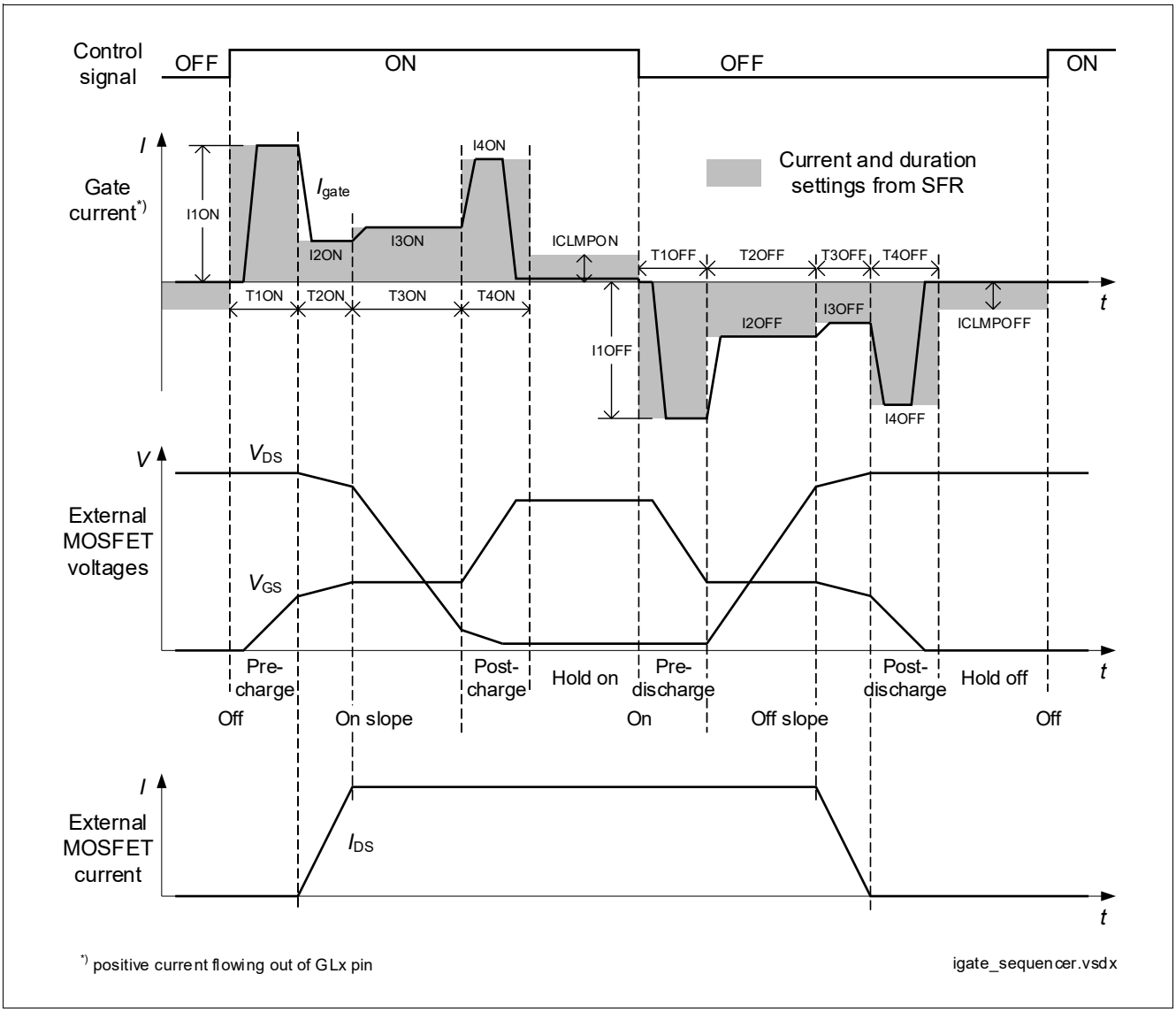


Figure 369 Gate current values in sequencer mode

The symbols IyON, TyON, ICLMPON, IyOFF, TyOFF, and ICLMPOFF shown in **Figure 369** expand to following registers and bit fields:

- IyON (y = 1...4): defined per gate driver in LSxSEQONIC.LSx_IyON and HSxSEQONIC.HSx_IyON
- TyON (y = 1...4): defined per gate driver in LSxSEQONTC.LSx_TyON and HSxSEQONTC.HSx_TyON
- ICLMPON: defined per half bridge in HBxIGATECLMPC
- IyOFF (y = 1...3): defined per gate driver in LSxSEQOFFIC.LSx_IyOFF and HSxSEQOFFIC.HSx_IyOFF
- I4OFF: defined once for all gate drivers in SEQOFFT4I4.I4OFF
- TyOFF (y = 1...3): defined per gate driver in LSxSEQOFFTC.LSx_TyOFF and HSxSEQOFFTC.HSx_TyOFF
- T4OFF: defined once for all gate drivers in SEQOFFT4I4.T4OFF
- ICLMPOFF: defined per half bridge in HBxIGATECLMPC

26.7.3 Gate current values

The nominal gate currents (I_{gate}) for the respective set point values (x) can be estimated by:

(26.1)

$$I_{gate}(x) = 5\ mA + 345\ mA * (\frac{x}{63})^{1.49}$$

and are shown in **Table 323**.

Table 323 Nominal gate currents

x	I_{gate}	x	I_{gate}	x	I_{gate}	x	I_{gate}
0	5 mA	16	50 mA	32	131 mA	48	235 mA
1	6 mA	17	54 mA	33	137 mA	49	242 mA
2	7 mA	18	58 mA	34	143 mA	50	249 mA
3	9 mA	19	63 mA	35	149 mA	51	257 mA
4	11 mA	20	67 mA	36	155 mA	52	264 mA
5	13 mA	21	72 mA	37	161 mA	53	272 mA
6	15 mA	22	77 mA	38	167 mA	54	279 mA
7	18 mA	23	82 mA	39	174 mA	55	287 mA
8	21 mA	24	87 mA	40	180 mA	56	294 mA
9	24 mA	25	92 mA	41	187 mA	57	302 mA
10	27 mA	26	97 mA	42	194 mA	58	310 mA
11	31 mA	27	103 mA	43	200 mA	59	318 mA
12	34 mA	28	108 mA	44	207 mA	60	326 mA
13	38 mA	29	114 mA	45	214 mA	61	334 mA
14	42 mA	30	119 mA	46	221 mA	62	342 mA
15	46 mA	31	125 mA	47	228 mA	63	350 mA

26.7.4 Active driver

The “active driver” is the gate driver in a half bridge which drives the “active MOSFET”. It generates the current and voltage slopes during PWM operation of an inductive load. Depending on the direction of the load current this is either the low-side or the high-side gate driver. Therefore, the sequencer mode which is capable to fine-tune the switching behavior of an active MOSFET can be mapped per half bridge to either the low-side or the high-side gate driver by SEQMAP.HBx_SEQMAP. The other gate driver of the half bridge which drives the “free-wheeling” MOSFET is then controlled by the constant mode, but taking the respective gate current and duration settings from the registers LSx_AFIC, LSx_AFTC, HSx_AFIC, and HSx_AFTC.

Figure 370 shows typical waveforms during PWM actuation for both mappings and it shows also that the active driver can be automatically determined by observing the output signals of the high-speed comparators (see **Chapter 26.13**):

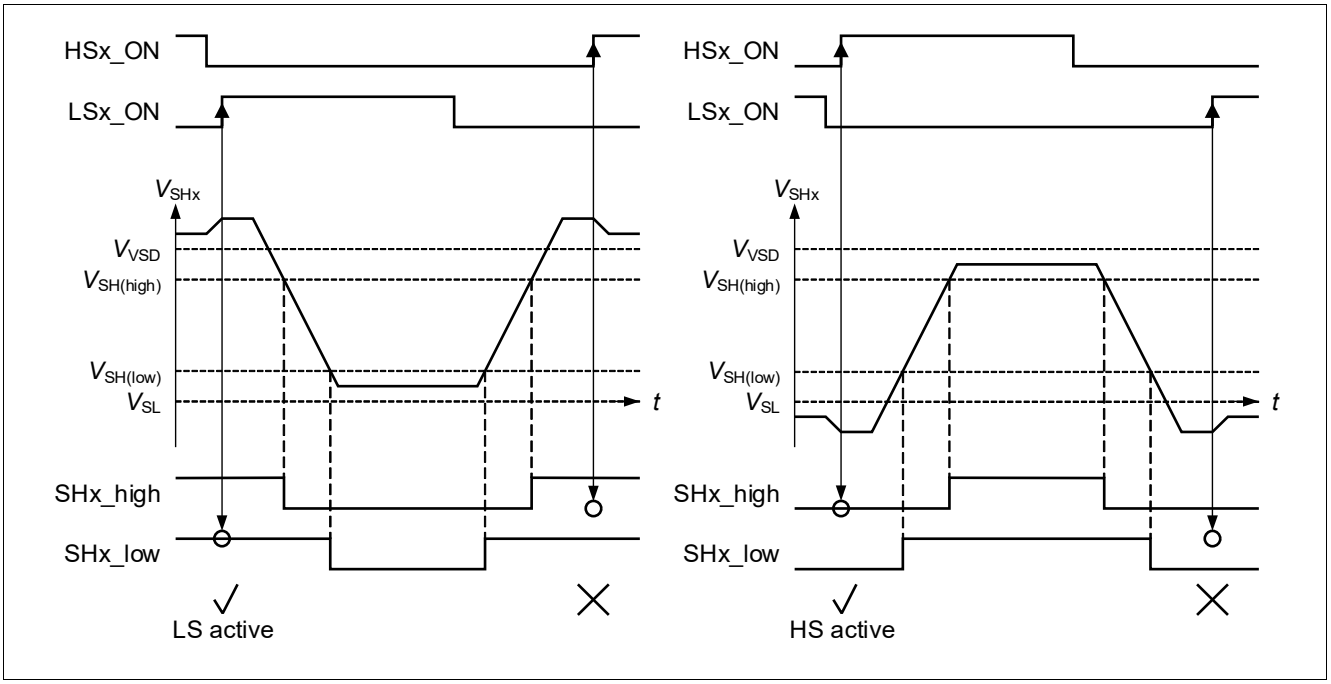


Figure 370 Automatic detection of active driver

If automatic detection of the active driver is enabled by CTRL2.ACTDRV_DET_EN='1', each time a control signal of a low-side or high-side gate driver (LSx_ON or HSx_ON) changes from '0' to '1' the corresponding low-side or high-side high-speed comparator output (SHx_low or SHx_high) is checked if the half bridge voltage has not yet reached the corresponding supply rail (i.e. $V_{SHx} > V_{SH(low)}$ or $V_{SHx} < V_{SH(high)}$) and thus the corresponding driver is automatically chosen by hardware to be the active driver for the next switching cycle.

That is in short: At the next switching cycle the hardware automatically

- Turns on the low-side gate driver as active driver if SHx_low='1'
- Turns on the high-side gate driver as active driver if SHx_high='0'

Each time the hardware changes the role of a gate driver of one half bridge to "active driver" the corresponding status flag SEQMAP.HBx_ACTDRV is updated and the corresponding interrupt status flag IRQS.HBx_ACTDRV_IS is set (which triggers an interrupt in the case of IRQEN.HBx_ACTDRV_IEN='1').

26.8 Adjustable cross-conduction protection

The Bridge Driver protects the external MOSFETs of each half bridge against cross-conduction. After switching off one of the two gate drivers the other gate driver does not switch on for a programmable protection time t_{CCP} to prevent both MOSFETs to be conducting at the same time. After t_{CCP} has expired, the other gate driver is switched on:

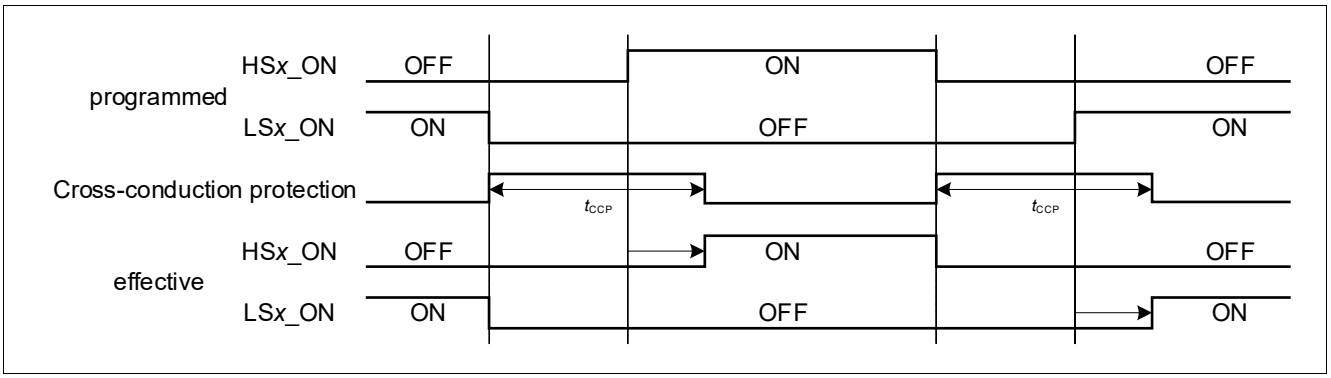


Figure 371 Cross-conduction protection

The configuration of the cross-conduction protection is done by the following bit fields of the CTRL2 register:

- DRV_CCP_DIS can be set to disable the cross-condition protection
- DRV_CCP_TIMSEL configures the value of t_{CCP}

Notes

1. *The cross-conduction protection feature of the Bridge Driver is only able to increase the effective dead time between the complementary control signals of one half bridge (LSx_ON, HSx_ON) to the selected cross-conduction protection time. If the dead time of the programmed control signals at INAx and INBx is already longer than the selected cross-conduction protection time this feature has no effect.*
2. *The cross-conduction protection feature of the Bridge Driver switches off both gate drivers of one half bridge if both control signals, LSx_ON and HSx_ON, are '1' at the same time. In this case a "driver sequence error" event is generated and the interrupt status flag IRQS.SEQ_ERR_IS is set.*

26.9 High-current discharge mode

The high-current discharge mode provides a low-ohmic path between the related Gx and Sx pins to do a fast discharge of the external MOSFET gate and to provide a strong pull-down mechanism in case the MOSFET is already off. It is realized by the active discharge path of each gate driver, and the current value that is used for the high-current discharge mode can be adjusted by the bitfield IHCDIS of the register HCDIS.

The high-current discharge mode is activated in the following situations:

- **Shut down:** the gate of the external MOSFET is discharged with the current setting HCDIS.IHCDIS
  - In the case of an emergency shutdown after detection of an error condition (see **Chapter 26.16**, **Chapter 26.17**, and **Chapter 26.19**)
  - In the case of system-level events like power-mode changes

  Note: The high-current discharge mode overrides only the discharge current settings until the clamping current gets active and keeps the switch-off timing settings as described in **Chapter 26.7.2**.
  The high-current discharge mode can be disabled per gate driver by setting the corresponding bits LSxDRV_HCDISCH_DIS or HSxDRV_HCDISCH_DIS in the PROT_CTRL register.

- **Keep off:** the gate of the external MOSFET is kept discharged with the current setting HCDIS.IHCDIS while the complementary external MOSFET is switching on

  Note: The high-current discharge mode is active only during the switch-on phase of the complementary gate driver. Afterwards, both gate drivers activate their respective clamping currents.
  The high-current discharge mode can be disabled per gate driver by setting the corresponding bits LSxDRV_HCDISCH_DIS or HSxDRV_HCDISCH_DIS in the PROT_CTRL register.

- **Support safe switch-off:** the gate of the external MOSFET is discharged with the current setting HCDIS.IHCDIS to support the safe switch-off path if SAFE_ENABLE=’1’ and SSO_HCDIS=’1’

  Note: The high-current discharge mode can be disabled by HCDIS.HCDIS_SSO=’0’ during safe switch-off events.

26.10 Safe switch-off path

In case of an error detected by a safety mechanism, the bridge driver switches off all external MOSFETs in a defined and reliable way. The safe switch-off path is controlled by the fail-safe supervision functional block of the PMU. In a first step, if enabled by HCDIS.HCDIS_SSO=’1’, the high-current discharge mode is activated by SSO_HCDIS=’1’ and discharges the gates of the external MOSFETs with the gate current setting HCDIS.IHCDIS. Then, after a delay, the PMU sets SAFE_ENABLE=’0’ and SAFE_SHUTDOWN=’1’ which disables all active gate driver currents (including the possibly before activated high-current discharge mode) and enables the passive pull-down mode for all half bridges.

Note: After power-up, the safe switch-off path is by default active and must be released explicitly (see PMU).

26.11 Passive pull-down mode

If a half bridge is disabled (i.e. CTRL1.HBx_EN='0') or the safe switch-off path is activated (i.e. SAFE_ENABLE='0' and SAFE_SHUTDOWN='1') both gate drivers passively discharge the gates of the external MOSFETs by activating:

- Resistors from Gxy to ground (VS supply needed) and
- Resistors between Gxy and Sxy (no supply needed)

During active operation of the gate drivers these pull-down resistors are switched off. The passive pull-down mode is available in all system modes, in particular in the sleep and stop modes.

26.12 Active brake mode

In active brake mode either all external high-side MOSFETs or all external low-side MOSFETs are statically switched on to short-circuit the motor coils which brakes the motor or keeps it actively blocked during standstill. Since in active brake mode no PWM capability is needed the charge pump can be set into single-stage mode (high-side MOSFETs active) or can even be switched off (low-side MOSFETs active) to reduce the overall current consumption I_{VSD_ab} from the VSD pin.

Active brake mode with high-side drivers is activated by switching on all external high-side MOSFETs (CTRL1.HSx_ON = '1') and putting the charge pump into single stage mode:

- CP_CTRL.CP_EN = '1'
- CP_CTRL.CP_1STAGE = '1'

Active brake mode with low-side drivers is activated by switching on all external low-side MOSFETs (CTRL1.LSx_ON = '1') and turning off the charge pump:

- CP_CTRL.CP_EN = '1'
- CP_CTRL.CP_RDY_EN = '0'
- CP_CLK_CTRL.CPCLK_EN = '0'
- PROT_CTRL.DRVx_VCPLO_SDEN = '0'

Note: Both active brake mode configurations reduce the performance of the charge pump in order to reduce the current consumption from the VSD pin. This means on the other hand that the voltage at the VSD pin must be high enough to get proper gate voltages at the external MOSFETs.

26.13 Timing measurements

Each half bridge provides two high-speed voltage comparators at the SHx pin to measure the beginning and the end of the SHx voltage slope during switch-on or switch-off of an external MOSFET. The measured values are stored in SFRs for further evaluation by software or by the adaptive control mode (see **Chapter 26.14**).

Figure 372 shows the thresholds $V_{SH(high)}$ and $V_{SH(low)}$ of the high-speed voltage comparators as well as the measured slope timing values TONDLY, TONDUR, TOFFDLY, and TOFFDUR during PWM operation of an external low-side MOSFET.

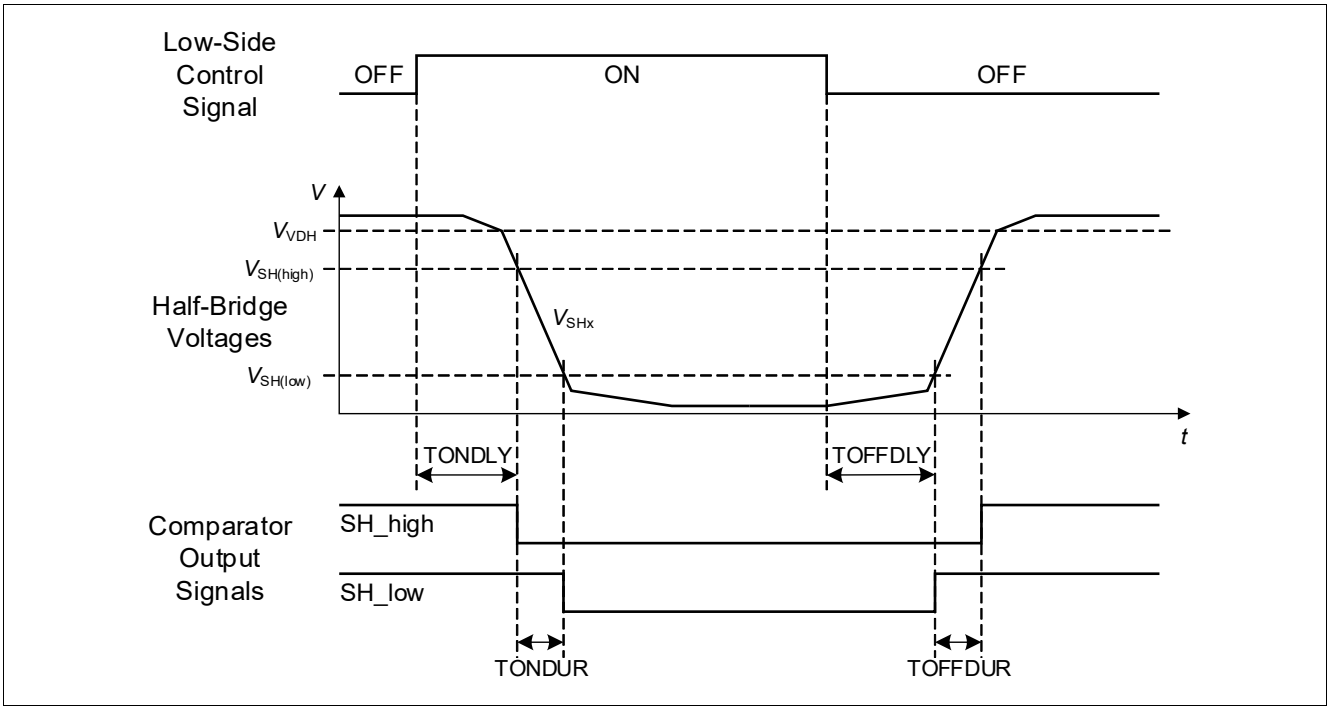


Figure 372 Comparator thresholds and timing parameters during low-side PWM

Figure 373 shows the thresholds $V_{SH(high)}$ and $V_{SH(low)}$ of the high-speed voltage comparators as well as the measured slope timing values TONDLY, TONDUR, TOFFDLY, and TOFFDUR during PWM operation of an external high-side MOSFET.

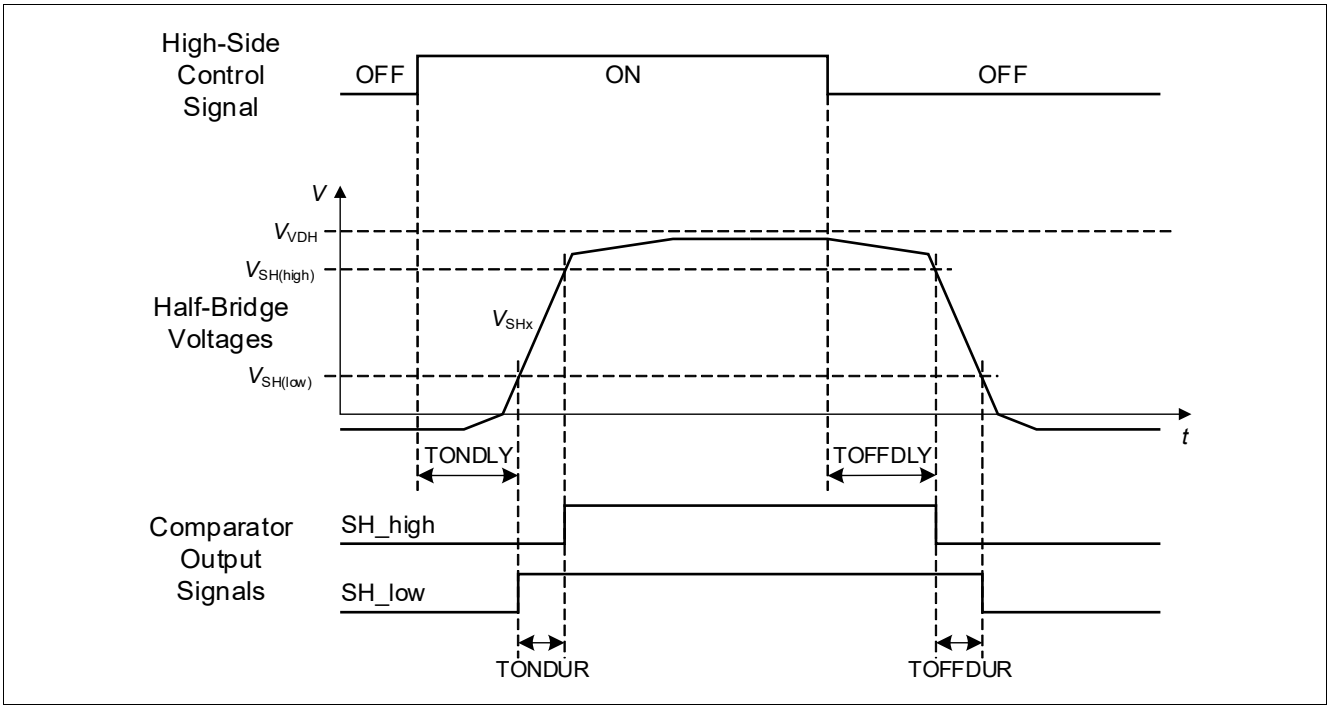


Figure 373 Comparator thresholds and timing parameters during high-side PWM

The symbols TONDLY, TONDUR, TOFFDLY, and TOFFDUR shown in **Figure 372** and **Figure 373** expand to following registers and bit fields for each half bridge x:

- TONDLY: HBxONVAL.HBx_TONDLY
- TONDUR: HBxONVAL.HBx_TONDUR
- TOFFDLY: HBxOFFVAL.HBx_TOFFDLY
- TOFFDUR: HBxOFFVAL.HBx_TOFFDUR

The timing measurements are performed at each transition of the corresponding PWM control signal and the contents of the HBxONVAL/HBxOFFVAL registers are updated if the HBx_ONVALVF/HBx_OFFVALVF flags are cleared. This means that the software can control when to get an update by writing to the HBx_ONVALVF_CLR/HBx_OFFVALVF_CLR clear bits.

For TONDLY/TOFFDLY measurements, if successful, the results are stored in the HBx_TONDLY/HBx_TOFFDLY bit fields and the HBx_ONVALVF/HBx_OFFVALVF flags are set. If not successful, i.e. the corresponding high-speed comparator signal did not change during the on/off period of the gate driver control signal, the results are ignored. The HBx_ONVALVF/HBx_OFFVALVF flags stay cleared until a successful measurement could be achieved in the following PWM cycles, indicated by the HBx_ONVALVF/HBx_OFFVALVF flags being set.

For TONDUR/TOFFDUR measurements, if successful, the results are stored in the HBx_TONDUR/HBx_TOFFDUR bit fields. If not successful, i.e. the second high-speed comparator signal did not change after the signal change of the first high-speed comparator and before the next change of the gate driver control signal, the HBx_TONDURMERR/HBx_TOFFDURMERR flags are set and the HBx_TONDUR/HBx_TOFFDUR bit fields are set to their maximum value.

The HBxONVAL/HBxOFFVAL registers additionally hold the information which gate driver (low-side or high-side) has been the “active driver” during the measurement in the HBx_ACTDRV_ON/HBx_ACTDRV_OFF bit.

The current output state of each high-speed comparator can be read by the corresponding SHx_LOW_STS or SHx_HIGH_STS flags in the STS register.

For plausibility checks of the high-speed comparators and the assigned timing measurement counters there is an additional counter using the drain-to-source voltage monitoring comparator outputs for alternative switch-on and switch-off timing measurements which can be put on each half bridge:

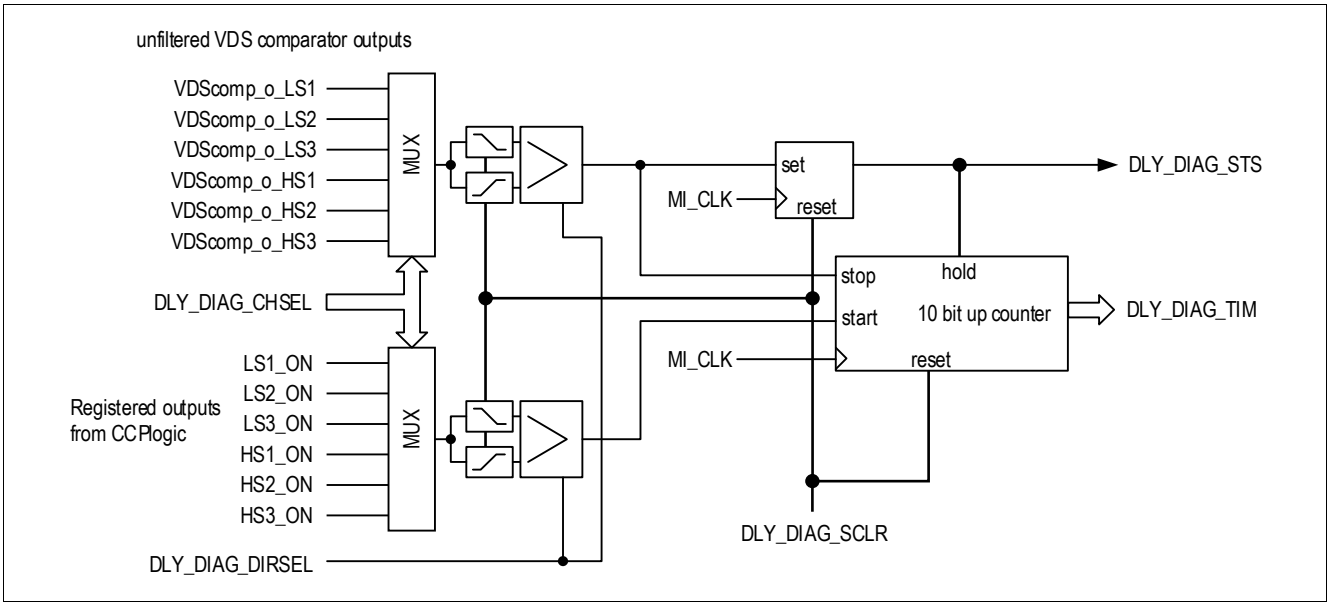


Figure 374 Principle of channel turn on/off timing measurement with VDS comparators

The bit fields shown in **Figure 374** can be found in the register DLY_DIAG. The timer exists once and can be assigned to each gate driver by DLY_DIAG_CHSEL and to its switch-on or switch-off event by DLY_DIAG_DIRSEL. It measures the delay from the selected change in the corresponding PWM input signal until the selected drain-to-source comparator output signal reacts. After completion of the measurement the result is stored in DLY_DIAG_TIM and the status flag DLY_DIAG_STS is set. A new measurement can be triggered by clearing the status flag DLY_DIAG_STS by DLY_DIAG_SCLR.

*Note: This timer uses the drain-to-source comparators described in **Chapter 26.17** with their threshold settings defined by CTRL2.DSMONVTH. Since the unfiltered output signals of the comparators are used in this function the settings for blanking time and filter time have no effect on this function.*

26.14 Adaptive control mode

The adaptive control mode is able to compensate MOSFET and gate driver parameter variations in the running system by providing a closed-loop regulation of the delay from changing PWM control signals until the resulting voltage slopes at the SHx pins. The adaptive control mode uses the information from the timing measurement comparators (TONDLY/TOFFDLY, see **Chapter 26.13**) to adjust the settings for the pre-charge (I1ON) and pre-discharge (I1OFF) currents of the sequencer mode (see **Chapter 26.7.2.2**):

- If the measured time is longer than a target time defined by SFRs then the respective gate current value is increased by one digit for the next switching cycle
- If the measured time is shorter than a target time defined by SFRs then the respective gate current value is decreased by one digit for the next switching cycle

26.14.1 Target settings

The target times for the adaptive control mode are defined as follows (see also **Figure 375** and **Figure 376**):

- On delay target: T1ON + T2ON
- Off delay target: T1OFF + T1OFFADDDLY

The parameters T1ON, T2ON, and T1OFF are directly taken from the sequencer timing settings (see **Chapter 26.7.2.2**). The additional delay T1OFFADDDLY which is needed to let the gate current settle to its I2OFF value is defined for each gate driver in the register ASEQOFFADDDLY and the full bit field name expands to LSxT1OFFADDDLY for low-side gate driver x and HSxT1OFFADDDLY for high-side gate driver x.

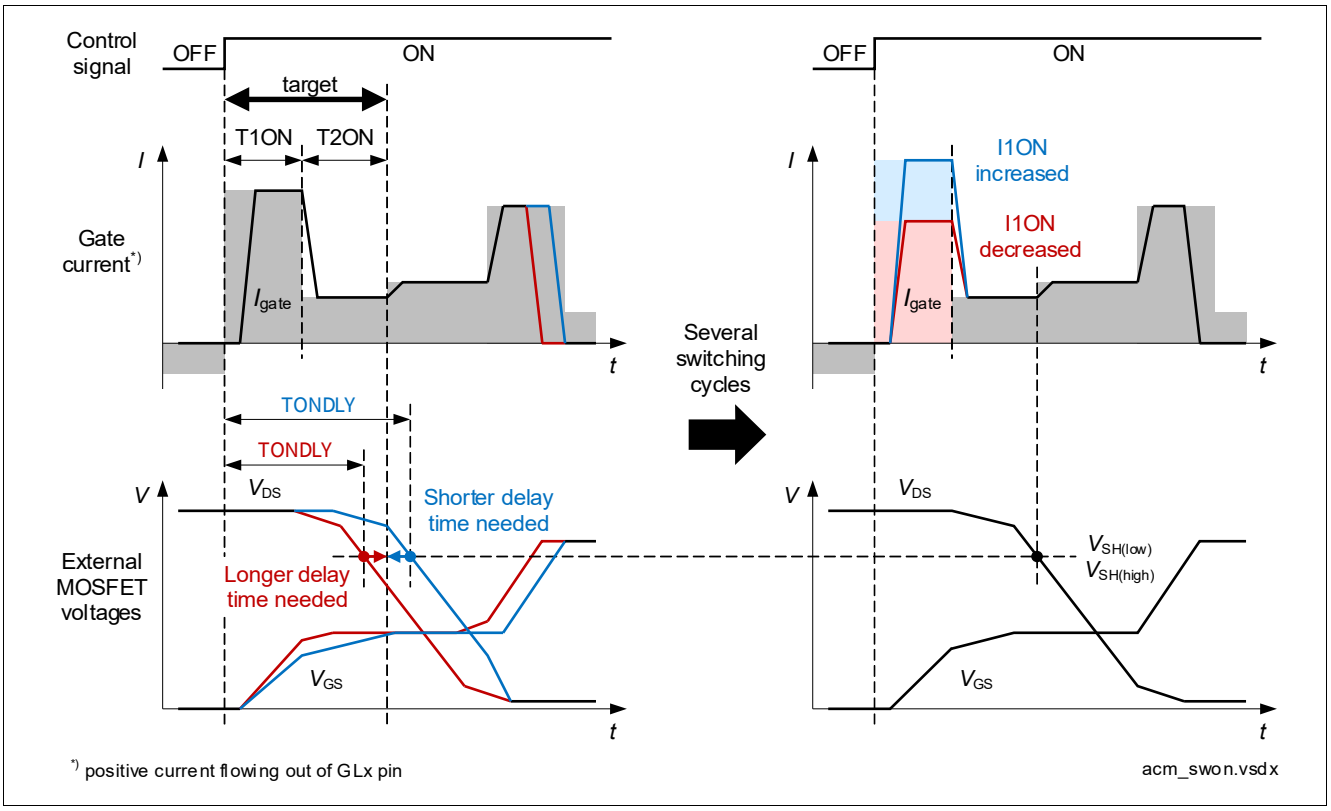


Figure 375 Adaptive control of the switch-on pre-charge current setting of a gate driver

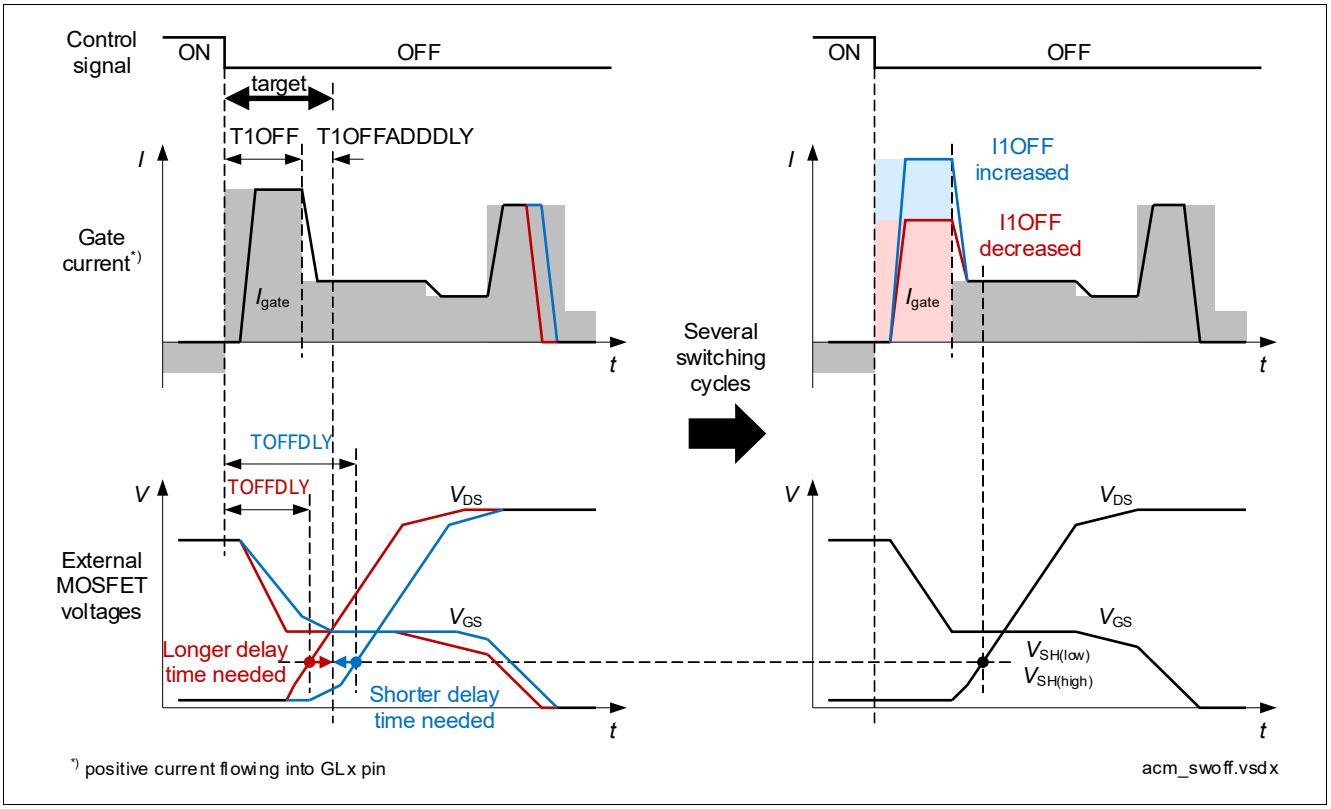


Figure 376 Adaptive control of the first switch-off sub-phase current setting of a gate driver

26.14.2 Activation and operation

The adaptive control mode is available for each “active driver”, i.e. the gate driver to which the sequencer is mapped to (see **Chapter 26.7.4**). **Table 324** summarizes the mapping of the enable bits and the initial gate current values to the corresponding register bit fields.

Table 324 Register bit field mapping

	HBx switch-on	HBx switch-off
Enable adaptive control mode	ASEQC.HBxASMONEN	ASEQC.HBxASMOFFEN
Enable hysteresis[1)]	ASEQC.HBxONHYSTEN	ASEQC.HBxOFFHYSTEN
Initial values for LS as active driver (i.e. SEQMAP.HBx_SEQMAP=0)	LSxSEQONIC.LSx_I1ON	LSxSEQOFFIC.LSx_I1OFF
Initial values for HS as active driver (i.e. SEQMAP.HBx_SEQMAP=1)	HSxSEQONIC.HSx_I1ON	HSxSEQOFFIC.HSx_I1OFF
Resulting gate current value[2)]	HBxONVAL.HBx_I1ONVAL	HBxOFFVAL.HBx_I1OFFVAL

1) If the hysteresis is enabled the adapted current is kept for two switching cycles before changing the direction (increase/decrease) of the adaption.

2) As for the timing measurement results (see **Chapter 26.13**) the validity of this value can be checked by the HBxONVAL.HBx_ONVALVF flag and clearing this flag by HBxONVAL.HBx_ONVALVF_CLR triggers its update. This mechanism does not interfere with the timing of the adaptive control mode.

Note: Enabling the adaptive control mode for a gate driver will shadow its configuration, i.e. a change of the registers afterwards will have no effect until the adaptive control mode will be disabled and enabled again the next time.

26.14.3 Monitoring

The adaptive control mode is monitored against the limits defined by the register bit fields shown in **Table 325** and corresponding errors are reported by the status flags as shown in **Table 326**:

Table 325 Limit definitions

Limit	Switch-on	Switch-off
Min. gate current[1)]	ASEQONIMIN.I1ONMIN	ASEQOFFIMIN.I1OFFMIN
Max. gate current[1)]	ASEQONIMAX.I1ONMAX	ASEQOFFIMAX.I1OFFMAX
Min. delay	ASEQONTMIN.T12ONMIN	ASEQOFFTMIN.T1OFFMIN
Max. delay	ASEQONTMAX.T12ONMAX	ASEQOFFTMAX.T1OFFMAX

1) The adaptive control mode does not change the gate currents to values beyond these limits.

Table 326 Status reporting

Limit	Switch-on: ASEQONSTS	Switch-off: ASEQOFFSTS
Min. gate current	HBxI1ONMIN	HBxI1OFFMIN
Max. gate current	HBxI1ONMAX	HBxI1OFFMAX
Min. delay[1)]	HBxT12ONMIN	HBxT1OFFMIN
Max. delay[1)]	HBxT12ONMAX	HBxT1OFFMAX

1) The timing status flags are set if the timing measurement results are beyond the defined min./max. delay limits for a number of consecutive times defined in ASEQERRCNT.HBxT12ONERRCNT or ASEQERRCNT.HBxT1OFFERRNCT.

Additionally, the plausibility of the timing measurement comparator events is monitored and if the events occur in the wrong order or events are missing an error is reported in the corresponding status flag ASEQONSTS.HBxONMF or ASEQOFFSTS.HBxOFFMF after a number of consecutive times defined in ASEQERRCNT.HBxMFERRCNT. The adaptive control mode does not change gate current values if the underlying delay measurement is invalid, i.e. the corresponding comparator did not switch within the last on or off cycle.

Notes

1. *If one of status flags in ASEQONSTS or ASEQOFFSTS is set also the corresponding IRQS.HBx_ASEQ_IS is set and the affected driver (low-side or high-side) is stored in the corresponding ASEQONSTS.HBxONFAILDRV or ASEQOFFSTS.HBxOFFFAILDRV bit.*
2. *The status flags in ASEQONSTS and ASEQOFFSTS are cleared when read by software.*

26.15 Integrated 2-stage charge pump

The Bridge Driver is supplied by an integrated 2-stage charge pump which provides a stable voltage V_{CP} above the battery voltage. This allows the Bridge Driver to operate down to low battery voltage values and to statically switch on the external MOSFETs.

26.15.1 Operating modes

The charge pump is supplied by the VSD pin and is enabled by CP_CTRL.CP_EN. Based on the settings of CP_CTRL.CP_1STAGE and CP_CTRL.CP_STG_AUTO it can be operated in three modes:

Table 327 Charge pump operating modes

CP_1STAGE	CP_STG_AUTO	Charge pump operating mode
0	0	2-stage mode
0	1	automatic switch between 2-stage mode at low VSD voltages and single-stage mode at high VSD voltages[1)]
1	x	single-stage mode

1) The decision is done based on the digital comparator of the ADC2 VSD voltage measurement (CMPLO0) and the status is shown in STS.VSD_CP1ST_STS.

26.15.2 Clock configuration

The charge pump clock configuration is done by the register CP_CLK_CTRL. By default the clock is enabled (CP_CLK_CTRL.CPCLK_EN=1). If CP_CLK_CTRL.CPCLK_EN is cleared the charge pump clock signal state is defined by CP_CLK_CTRL.CPCLKDIS_SET.

The frequency of the charge pump is prepared for dithering to reduce the emission spectrum on the VSD supply. A frequency divider uses a 7-bit divider ratio that is built from two registers, a 2-bit MSB and a 5-bit LSB. The MSB has a fixed value defined by CP_CLK_CTRL.F_CP while the LSB counts up and down in a triangle frequency modulation. The register bit field CP_CLK_CTRL.DITH_UPPER represents the upper frequency and thus stores the lower counter value limit of the up-down counter. Accordingly, the register bit field CP_CLK_CTRL.DITH_LOWER stores the upper counter value limit.

Table 328 shows the default settings as an example.

Table 328 Default charge pump clock settings

CP_CLK_CTRL registers	Reset settings (binary)	Counter value	Resulting typical dithering border frequency
F_CP & DITH_UPPER	10 & 01010	74	20 MHz[1)] / 74 = 270 kHz
F_CP & DITH_LOWER	10 & 10110	86	20 MHz[1)] / 86 = 233 kHz

1) Nominal value of MI_CLK.

26.16 Adjustable voltage monitoring

The supply voltages of the Bridge Driver, VSD and VCP, are monitored by the ADC2 digital comparators and by two dedicated analog comparators. The Bridge Driver can be optionally disabled at undervoltage or overvoltage conditions of the monitored signals. In addition to the details described in the subsequent sections there is one combined status flag showing a general “supply error”: CTRL1.SUPERR_STS.

26.16.1 ADC2 digital comparators

The absolute voltages of the VSD and VCP pins are measured by the ADC2 and evaluated by its digital comparators. The comparators are configured to generate undervoltage and overvoltage events which by default switch all gate drivers to “high-current discharge” mode for the maximum programmable time. Afterwards, the gate drivers switch to the clamping current. **Table 329** shows the mapping to the ADC2 digital comparators and to the BDRV status and control bits.

Table 329 Status and control bits

Event	ADC2 comparator	Status flag	Disable gate driver discharge[1)]
VCP undervoltage	CMPLO2	STS.VCP_LOTH1_STS	PROT_CTRL.DRVx_VCPLO_DIS
VCP overvoltage	CMPUP2	STS.VCP_UPTH_STS	PROT_CTRL.DRVx_VCPUP_DIS
VSD undervoltage	CMPLO1	STS.VSD_LOTH_STS	PROT_CTRL.DRVx_VSDLO_DIS
VSD overvoltage	CMPUP0	STS.VSD_UPTH_STS	PROT_CTRL.DRVx_VSDUP_DIS

1) If set to ‘1’ the corresponding event does not trigger a gate driver discharge.

The VCP undervoltage event can be selected by CP_CTRL.VCP_LOWSRC_SEL=’0’ to disable all gate drivers if CP_CTRL.CP_RDY_EN and PROT_CTRL.DRVx_VCPLO_SDEN both are set.

26.16.2 VCP vs. VSD comparator

The voltage difference between VCP and VSD pins is monitored by an analog comparator and reported by the STS.VCP_LOTH2_STS status flag.

The comparator can be configured by the following bits:

- CP_CTRL.CP_LOWTH2: selects between two thresholds
- CP_CTRL.CPLOW_TFILT_SEL: selects one of four filter times

The comparator can be selected by CP_CTRL.VCP_LOWSRC_SEL=’1’ to disable all gate drivers if CP_CTRL.CP_RDY_EN and PROT_CTRL.DRVx_VCPLO_SDEN both are set.

26.16.3 VSD overvoltage comparator

The VSD overvoltage comparator is active in the power-down modes to detect reverse supply conditions due to a motor in generator mode and can trigger a wake-up of the system. The comparator status is shown in STS.VSD_OV_STS. The software can then decide to switch on the low-side gate drivers to brake the motor.

Note: The charge pump must not be switched on if the absolute maximum ratings of the VCP pin cannot be fulfilled.

26.17 Adjustable short circuit detection

For short circuit detection the drain-to-source comparators of the Bridge Driver are used to compare the voltage drops across the external MOSFETs to the threshold voltage defined by CTRL2.DSMONVTH. During transitions from off to on and vice versa the comparator output signals are ignored for the blanking time defined by CTRL2.LS_HS_BT_TFILT_SEL.

26.17.1 On state

In on state, if a stable short-circuit condition is detected for a time longer than the filter time defined by CTRL2.DRV_DS_TFILT_SEL, the corresponding status flags are set:

- STS.LSx_OC_STS or STS.HSx_OC_STS
- IRQS.LSx_OC_IS or IRQS.HSx_OC_IS (interrupts are generated according to the settings in IRQEN)

By default, all external MOSFETs are switched off automatically using the high-current discharge mode in the case of a short-circuit detected at one MOSFET. Possible configurations for each gate driver are:

- Disable automatic switch-off by setting the corresponding LSxDRV_OCSDN_DIS or HSxDRV_OCSDN_DIS bit in PROT_CTRL
- Reduce the automatic switch-off behavior to the one MOSFET where the short-circuit was detected by setting the corresponding LSx_OC_SEL or HSx_OC_SEL bit in CTRL1. In this configuration, the respective MOSFET is switched off using the currently programmed gate current control mode instead of the high-current discharge mode

Note: After being switched off by an overcurrent event, a gate driver can be switched on again only after clearing the corresponding bit in the STS register.

26.17.2 Off state

In off state the motor phases are terminated by internal bias currents. By default, the motor phases are pulled up and can be pulled down at each half bridge separately by activating a diagnostic current source by setting the corresponding CTRL1.HSx_DCS_EN while setting the off clamping current of the same half bridge to HBxIGATECLMPC.HBx_ICLMPOFF=0.

In the case of a short circuit it is not possible for the motor phase either to follow the pull-up or the pull-down current source and the corresponding drain-to-source comparator stays below its threshold which is reported in the corresponding status flags:

- STS.LSx_DS_STS or STS.HSx_DS_STS
- IRQS.LSx_DS_IS or IRQS.HSx_DS_IS (interrupts are generated according to the settings in IRQEN)

Note: The timing behavior of the short-circuit detection in off state is strongly application-dependent. After changing the configuration of the diagnostic current sources a “long-enough” settling time has to be considered. It is recommended to clear the status bits after this settling time and check them again after the filter time defined by CTRL2.DRV_DS_TFILT_SEL.

26.18 Open-load detection

In off state the motor phases are terminated by internal bias currents. By default, the motor phases are pulled up and can be pulled down at each half bridge separately by activating a diagnostic current source by setting the corresponding CTRL1.HSx_DCS_EN while setting the off clamping current of the same half bridge to HBxIGATECLMPC.HBx_ICLMPOFF=0.

For open-load detection in off state all diagnostic current sources have to be activated except the one of the half bridge/motor phase to be tested. The activated diagnostic current sources are able to override the pull-up bias current of the half bridge to be tested and therefore - if the motor is connected - pull down all motor phases. If the connection to the tested motor phase is missing this phase will be pulled up and the corresponding drain-to-source comparator will set its status flags:

- STS.LSx_DS_STS or STS.HSx_DS_STS
- IRQS.LSx_DS_IS or IRQS.HSx_DS_IS (interrupts are generated according to the settings in IRQEN)

Note: The timing behavior of the open-load detection in off state is strongly application-dependent. After changing the configuration of the diagnostic current sources a “long-enough” settling time has to be considered. It is recommended to clear the status bits after this settling time and check them again after the filter time defined by CTRL2.DRV_DS_TFILT_SEL.

26.19 Overtemperature

The temperature of the Bridge Driver Charge Pump is monitored by a dedicated temperature sensor and evaluated by the ADC2. In the case the ADC2 digital comparator CMPUP4 detects an overtemperature event the Bridge Driver switches off all gate drivers with the “high-current discharge” mode for the maximum programmable time and sets the status flags STS.CP_OTSD_STS and CTRL1.SUPERR_STS. Afterwards, the gate drivers are kept off with the corresponding off clamping current as long as the overtemperature condition is present.

Note: Interrupts on the overtemperature event can be triggered by the ADC2 digital comparators.

26.20 Programmer’s guide

26.20.1 Recommended switch-on sequence

The Bridge Driver requires some other modules to be enabled and running until it can be enabled and used itself. Therefore a proper switch-on sequence has to be fulfilled:

Step 1	Power-up the device or wake-up from Sleep mode or Stop mode
Step 2	Enable charge pump
Step 3	Enable CSA and CSC including built-in self-test
Step 4	Leave safe state
Step 5	Check VCP measurement of ADC2 for sufficient charge pump voltage
Step 6	Clear ADC2 compare unit status bits
Step 7	Clear bridge driver status bit STS.VCP_LOTH2_STS
Step 8	Enable bridge driver

26.21 Register description BDRV

26.21.1 BDRV Address Maps

Table 330 Register Address Space - BDRV

Module	Base Address	End Address	Note
BDRV	$4000C000_H$	$4000FFFF_H$	

Table 331 Register Overview - BDRV (ascending Offset Address)

Short Name	Long Name	Offset Address	Page Number
CTRL1	Bridge driver control 1	0000_H	1137
CTRL2	Bridge driver control 2	0004_H	1139
PWMSRCSEL	PWM source selection	0008_H	1141
SEQMAP	Sequencer mapping	$000C_H$	1143
DLY_DIAG	Delay diagnosis timer	0010_H	1144
CP_CTRL	Charge pump control	0014_H	1145
CP_CLK_CTRL	Charge pump clock control	0018_H	1146
PROT_CTRL	Protection function control	$001C_H$	1147
STS	Bridge driver status	0020_H	1149
STSCLR	Bridge driver status clear	0024_H	1152
STSSET	Bridge driver status set	0028_H	1153
IRQS	Bridge driver interrupt status	$002C_H$	1155
IRQCLR	Bridge driver interrupt status clear	0030_H	1157
IRQSET	Bridge driver interrupt status set	0034_H	1159
IRQEN	Bridge driver interrupt enable	0038_H	1161
HB1IGATECLMPC	Half bridge 1 gate current clamping value	$003C_H$	1163
HB2IGATECLMPC	Half bridge 2 gate current clamping value	0040_H	1164
HB3IGATECLMPC	Half bridge 3 gate current clamping value	0044_H	1164
LS1AFTC	Low-side driver 1 active free-wheeling time control	0048_H	1165
LS1AFIC	Low-side driver 1 active free-wheeling current control	$004C_H$	1165
HS1AFTC	High-side driver 1 active free-wheeling time control	0050_H	1166
HS1AFIC	High-side driver 1 active free-wheeling current control	0054_H	1166
LS1SEQOFFTC	Low-side driver 1 switch-off time control	0058_H	1167
LS1SEQOFFIC	Low-side driver 1 switch-off current control	$005C_H$	1167
LS1SEQONTC	Low-side driver 1 switch-on time control	0060_H	1168
LS1SEQONIC	Low-side driver 1 switch-on current control	0064_H	1169
HS1SEQOFFTC	High-side driver 1 switch-off time control	0068_H	1169

Table 331 Register Overview - BDRV (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Page Number
HS1SEQOFFIC	High-side driver 1 switch-off current control	$006C_H$	1170
HS1SEQONTC	High-side driver 1 switch-on time control	0070_H	1171
HS1SEQONIC	High-side driver 1 switch-on current control	0074_H	1171
LS2AFTC	Low-side driver 2 active free-wheeling time control	0078_H	1172
LS2AFIC	Low-side driver 2 active free-wheeling current control	$007C_H$	1173
HS2AFTC	High-side driver 2 active free-wheeling time control	0080_H	1173
HS2AFIC	High-side driver 2 active free-wheeling current control	0084_H	1174
LS2SEQOFFTC	Low-side driver 2 switch-off time control	0088_H	1174
LS2SEQOFFIC	Low-side driver 2 switch-off current control	$008C_H$	1175
LS2SEQONTC	Low-side driver 2 switch-on time control	0090_H	1175
LS2SEQONIC	Low-side driver 2 switch-on current control	0094_H	1176
HS2SEQOFFTC	High-side driver 2 switch-off time control	0098_H	1177
HS2SEQOFFIC	High-side driver 2 switch-off current control	$009C_H$	1177
HS2SEQONTC	High-side driver 2 switch-on time control	$00A0_H$	1178
HS2SEQONIC	High-side driver 2 switch-on current control	$00A4_H$	1179
LS3AFTC	Low-side driver 3 active free-wheeling time control	$00A8_H$	1179
LS3AFIC	Low-side driver 3 active free-wheeling current control	$00AC_H$	1180
HS3AFTC	High-side driver 3 active free-wheeling time control	$00B0_H$	1180
HS3AFIC	High-side driver 3 active free-wheeling current control	$00B4_H$	1181
LS3SEQOFFTC	Low-side driver 3 switch-off time control	$00B8_H$	1181
LS3SEQOFFIC	Low-side driver 3 switch-off current control	$00BC_H$	1182
LS3SEQONTC	Low-side driver 3 switch-on time control	$00C0_H$	1182
LS3SEQONIC	Low-side driver 3 switch-on current control	$00C4_H$	1183
HS3SEQOFFTC	High-side driver 3 switch-off time control	$00C8_H$	1184
HS3SEQOFFIC	High-side driver 3 switch-off current control	$00CC_H$	1184
HS3SEQONTC	High-side driver 3 switch-on time control	$00D0_H$	1185
HS3SEQONIC	High-side driver 3 switch-on current control	$00D4_H$	1186
SEQOFFT4I4	Sequencer switch-off phase 4 time and current control	$00D8_H$	1186
HCDIS	High-current discharge mode control	$00DC_H$	1187
HB1ONVAL	Half bridge 1 switch-on measurement values	$00E0_H$	1187
HB1OFFVAL	Half bridge 1 switch-off measurement values	$00E4_H$	1188
HB2ONVAL	Half bridge 2 switch-on measurement values	$00E8_H$	1189
HB2OFFVAL	Half bridge 2 switch-off measurement values	$00EC_H$	1190
HB3ONVAL	Half bridge 3 switch-on measurement values	$00F0_H$	1191
HB3OFFVAL	Half bridge 3 switch-off measurement values	$00F4_H$	1192
ASEQC	Adaptive sequencer control	$00F8_H$	1194
ASEQONSTS	Adaptive sequencer switch-on status	$00FC_H$	1195

Table 331 Register Overview - BDRV (ascending Offset Address) (cont'd)

Short Name	Long Name	Offset Address	Page Number
ASEQOFFSTS	Adaptive sequencer switch-off status	0100_H	1197
ASEQERRCNT	Adaptive sequencer error counter control	0104_H	1199
ASEQONTMIN	Adaptive sequencer minimum switch-on time setting	0108_H	1200
ASEQOFFTMIN	Adaptive sequencer minimum switch-off time setting	$010C_H$	1201
ASEQONIMIN	Adaptive sequencer minimum switch-on current setting	0110_H	1201
ASEQOFFIMIN	Adaptive sequencer minimum switch-off current setting	0114_H	1202
ASEQONTMAX	Adaptive sequencer maximum switch-on time setting	0118_H	1202
ASEQOFFTMAX	Adaptive sequencer maximum switch-off time setting	$011C_H$	1203
ASEQONIMAX	Adaptive sequencer maximum switch-on current setting	0120_H	1203
ASEQOFFIMAX	Adaptive sequencer maximum switch-off current setting	0124_H	1204
ASEQOFFADDDLY	Adaptive sequencer additional switch-off delay setting	0128_H	1204
BEMFC_CTRL	BEMF comparator control and status	$012C_H$	1206
BEMFC_IRQS	BEMF comparator interrupt status	0130_H	1208
BEMFC_IRQCLR	BEMF comparator interrupt status clear	0134_H	1209
BEMFC_IRQSET	BEMF comparator interrupt status set	0138_H	1210
BEMFC_IRQEN	BEMF comparator interrupt enable	$013C_H$	1211

26.21.2 BDRV Registers

Bridge driver control 1

CTRL1
Bridge driver control 1 **(0000_H)** **RESET_TYPE_5 Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES			SUPERR_STS	RES	HS3_DCS_EN	HS2_DCS_EN	HS1_DCS_EN	HS3_OC_SEL	HS3_ON	HS3_PWM	RES	LS3_OC_SEL	LS3_ON	LS3_PWM	HB3_EN
	r		r	r	rw	rw	rw	rw	rw	rwpht	r	rw	rw	rwpht	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HS2_OC_SEL	HS2_ON	HS2_PWM	RES	LS2_OC_SEL	LS2_ON	LS2_PWM	HB2_EN	HS1_OC_SEL	HS1_ON	HS1_PWM	RES	LS1_OC_SEL	LS1_ON	LS1_PWM	HB1_EN
rw	rw	rwpht	r	rw	rw	rwpht	rw	rw	rw	rwpht	r	rw	rw	rwpht	rw

Field	Bits	Type	Description
HB1_EN	0	rw	**Half bridge 1 enable** 0_B **DISABLE**, Driver circuit power off 1_B **ENABLE**, Driver circuit power on
LS1_PWM	1	rwpht	**Low-side driver 1 PWM enable** *Note: This bit can only be set if HS1_ON and LS1_ON are 0.* 0_B **DISABLE**, Control by PWM input disabled 1_B **ENABLE**, Control by PWM input enabled
LS1_ON	2	rw	**Low-side driver 1 on** 0_B **OFF**, Driver off 1_B **ON**, Driver on
LS1_OC_SEL	3	rw	**Low-side driver 1 overcurrent shutdown select** 0_B **GLOBALSD**, All drivers will be shut down in case of overcurrent 1_B **LOCALSD**, Only local driver will be shut down in case of overcurrent
RES	4, 12, 20, 27, 31:29	r	**Reserved** Always read as 0
HS1_PWM	5	rwpht	**High-side driver 1 PWM enable** *Note: This bit can only be set if HS1_ON and LS1_ON are 0.* 0_B **DISABLE**, Control by PWM input disabled 1_B **ENABLE**, Control by PWM input enabled
HS1_ON	6	rw	**High-side driver 1 on** 0_B **OFF**, Driver off 1_B **ON**, Driver on

Field	Bits	Type	Description
HS1_OC_SEL	7	rw	**High-side driver 1 overcurrent shutdown select** 0_B **GLOBALSD**, All drivers will be shut down in case of overcurrent 1_B **LOCALSD**, Only local driver will be shut down in case of overcurrent
HB2_EN	8	rw	**Half bridge 2 enable** 0_B **DISABLE**, Driver circuit power off 1_B **ENABLE**, Driver circuit power on
LS2_PWM	9	rwpht	**Low-side driver 2 PWM enable** *Note: This bit can only be set if HS2_ON and LS2_ON are 0.* 0_B **DISABLE**, Control by PWM input disabled 1_B **ENABLE**, Control by PWM input enabled
LS2_ON	10	rw	**Low-side driver 2 on** 0_B **OFF**, Driver off 1_B **ON**, Driver on
LS2_OC_SEL	11	rw	**Low-side driver 2 overcurrent shutdown select** 0_B **GLOBALSD**, All drivers will be shut down in case of overcurrent 1_B **LOCALSD**, Only local driver will be shut down in case of overcurrent
HS2_PWM	13	rwpht	**High-side driver 2 PWM enable** *Note: This bit can only be set if HS2_ON and LS2_ON are 0.* 0_B **DISABLE**, Control by PWM input disabled 1_B **ENABLE**, Control by PWM input enabled
HS2_ON	14	rw	**High-side driver 2 on** 0_B **OFF**, Driver off 1_B **ON**, Driver on
HS2_OC_SEL	15	rw	**High-side driver 2 overcurrent shutdown select** 0_B **GLOBALSD**, All drivers will be shut down in case of overcurrent 1_B **LOCALSD**, Only local driver will be shut down in case of overcurrent
HB3_EN	16	rw	**Half bridge 3 enable** 0_B **DISABLE**, Driver circuit power off 1_B **ENABLE**, Driver circuit power on
LS3_PWM	17	rwpht	**Low-side driver 3 PWM enable** *Note: This bit can only be set if HS3_ON and LS3_ON are 0.* 0_B **DISABLE**, Control by PWM input disabled 1_B **ENABLE**, Control by PWM input enabled
LS3_ON	18	rw	**Low-side driver 3 on** 0_B **OFF**, Driver off 1_B **ON**, Driver on
LS3_OC_SEL	19	rw	**Low-side driver 3 overcurrent shutdown select** 0_B **GLOBALSD**, All drivers will be shut down in case of overcurrent 1_B **LOCALSD**, Only local driver will be shut down in case of overcurrent
HS3_PWM	21	rwpht	**High-side driver 3 PWM enable** *Note: This bit can only be set if HS3_ON and LS3_ON are 0.* 0_B **DISABLE**, Control by PWM input disabled 1_B **ENABLE**, Control by PWM input enabled

Field	Bits	Type	Description
HS3_ON	22	rw	**High-side driver 3 on** 0_B **OFF**, Driver off 1_B **ON**, Driver on
HS3_OC_SEL	23	rw	**High-side driver 3 overcurrent shutdown select** 0_B **GLOBALSD**, All drivers will be shut down in case of overcurrent 1_B **LOCALSD**, Only local driver will be shut down in case of overcurrent
HS1_DCS_EN	24	rw	**High-side driver 1 diagnosis current source enable** *Note: HB1IGATECLMPC.HB1_ICLMPOFF has to be programmed to 0h.* 0_B **DISABLE**, 1_B **ENABLE**, Short diagnosis can be performed by evaluating the LS1/HS1_DS_STS flag
HS2_DCS_EN	25	rw	**High-side driver 2 diagnosis current source enable** *Note: HB2IGATECLMPC.HB2_ICLMPOFF has to be programmed to 0h.* 0_B **DISABLE**, 1_B **ENABLE**, Short diagnosis can be performed by evaluating the LS2/HS2_DS_STS flag
HS3_DCS_EN	26	rw	**High-side driver 3 diagnosis current source enable** *Note: HB3IGATECLMPC.HB3_ICLMPOFF has to be programmed to 0h.* 0_B **DISABLE**, 1_B **ENABLE**, Short diagnosis can be performed by evaluating the LS3/HS3_DS_STS flag
SUPERR_STS	28	r	**Supply error status** 0_B **NORMAL**, Supply is in required range 1_B **SUPERR**, Supply error detected; this flag is an OR of the VSD_x, VCP_x and CP_OTSD status flags from the ADC2 and IRQS.VCP_LOTH2_STS (if selected)

Bridge driver control 2

CTRL2
Bridge driver control 2 **(0004_H)** **RESET_TYPE_5 Value: $0001\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DRV_CCP_DIS	**DRV_CCP_TIMSEL**			**RES**		**LS_HS_BT_TFILT_SEL**		**HSDRV_DS_TFILT_SEL**		**LSDRV_DS_TFILT_SEL**		**RES**	**DSMONVTH**		
rw	rw			r		rw		rw		rw		r	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES			**ACTDRV_DET_EN**	**RES**		**HB3OFFSEQCNF**	**HB3ONSEQCNF**	**RES**		**HB2OFFSEQCNF**	**HB2ONSEQCNF**	**RES**		**HB1OFFSEQCNF**	**HB1ONSEQCNF**
r			rw	r		rw	rw	r		rw	rw	r		rw	rw

Field	Bits	Type	Description
HB1ONSEQCNF	0	rw	**Half bridge 1 sequencer switch-on configuration** 0_B **CONST**, Half bridge drivers operate with constant current 1_B **SEQ**, Sequencer is enabled
HB1OFFSEQCNF	1	rw	**Half bridge 1 sequencer switch-off configuration** 0_B **CONST**, Half bridge drivers operate with constant current 1_B **SEQ**, Sequencer is enabled
RES	3:2, 7:6, 11:10, 15:13, 19, 27:26	r	**Reserved** Always read as 0
HB2ONSEQCNF	4	rw	**Half bridge 2 sequencer switch-on configuration** 0_B **CONST**, Half bridge drivers operate with constant current 1_B **SEQ**, Sequencer is enabled
HB2OFFSEQCNF	5	rw	**Half bridge 2 sequencer switch-off configuration** 0_B **CONST**, Half bridge drivers operate with constant current 1_B **SEQ**, Sequencer is enabled
HB3ONSEQCNF	8	rw	**Half bridge 3 sequencer switch-on configuration** 0_B **CONST**, Half bridge drivers operate with constant current 1_B **SEQ**, Sequencer is enabled
HB3OFFSEQCNF	9	rw	**Half bridge 3 sequencer switch-off configuration** 0_B **CONST**, Half bridge drivers operate with constant current 1_B **SEQ**, Sequencer is enabled
ACTDRV_DET_EN	12	rw	**Detection of active / free-wheeling MOSFET** 0_B **DISABLE**, 1_B **ENABLE**,
DSMONVTH	18:16	rw	**Voltage threshold for drain-source monitoring of external MOSFETs** 000_B **0V125**, Threshold 0 for VDS at 125 mV 001_B **0V25**, Threshold 1 for VDS at 250 mV 010_B **0V5**, Threshold 2 for VDS at 500 mV 011_B **0V75**, Threshold 3 for VDS at 750 mV 100_B **1V0**, Threshold 4 for VDS at 1.00 V 101_B **1V25**, Threshold 5 for VDS at 1.25 V 110_B **1V5**, Threshold 6 for VDS at 1.50 V 111_B **1V75**, Threshold 7 for VDS at 1.75 V

Register description BDRV

Field	Bits	Type	Description
LSDRV_DS_TFILT_SEL	21:20	rw	**Filter time for drain-source monitoring of low-side drivers** 00_B **1u**, 1 µs filter time 01_B **2u**, 2 µs filter time 10_B **4u**, 4 µs filter time 11_B **8u**, 8 µs filter time
HSDRV_DS_TFILT_SEL	23:22	rw	**Filter time for drain-source monitoring of high-side drivers** 00_B **1u**, 1 µs filter time 01_B **2u**, 2 µs filter time 10_B **4u**, 4 µs filter time 11_B **8u**, 8 µs filter time
LS_HS_BT_TFILT_SEL	25:24	rw	**Blanking time for drain-source monitoring of low-side/high-side drivers** 00_B **1u**, 1 µs filter time 01_B **2u**, 2 µs filter time 10_B **4u**, 4 µs filter time 11_B **8u**, 8 µs filter time
DRV_CCP_TIMSEL	30:28	rw	**Minimum cross-conduction protection time setting** 000_B **0u2**, 200 ns cross-conduction protection time 001_B **0u4**, 400 ns cross-conduction protection time 010_B **0u8**, 800 ns cross-conduction protection time 011_B **1u6**, 1.6 µs cross-conduction protection time 100_B **3u2**, 3.2 µs cross-conduction protection time 101_B **6u4**, 6.4 µs cross-conduction protection time 110_B **12u8**, 12.8 µs cross-conduction protection time 111_B **NU**, Not used, 12.8 µs cross-conduction protection time
DRV_CCP_DIS	31	rw	**Dynamic cross-conduction protection disable** 0_B **ENABLE**, Dynamic CCP is active 1_B **DISABLE**, Dynamic CCP is disabled

PWM source selection

PWMSRCSEL
PWM source selection **(0008_H)** **RESET_TYPE_5 Value: $0026\ 1504_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES									HS3_SRC_SEL			RES	LS3_SRC_SEL		
r									rwpht			r	rwpht		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	HS2_SRC_SEL			RES	LS2_SRC_SEL			RES	HS1_SRC_SEL			RES	LS1_SRC_SEL		
r	rwpht			r	rwpht			r	rwpht			r	rwpht		

Field	Bits	Type	Description
LS1_SRC_SEL	2:0	rwpht	**Low-side driver 1 PWM source selection** *Note: Can only be written while all CTRL1.*_PWM are 0.* 000_B **INA1**, PWM control input 001_B **INA2**, PWM control input 010_B **INA3**, PWM control input 011_B **INC1N**, Inverted PWM control input 100_B **INB1**, PWM control input 101_B **INB2**, PWM control input 110_B **INB3**, PWM control input 111_B **INC4**, PWM control input
RES	3, 7, 11, 15, 19, 31:23	r	**Reserved** Always read as 0
HS1_SRC_SEL	6:4	rwpht	**High-side driver 1 PWM source selection** *Note: Can only be written while all CTRL1.*_PWM are 0.* 000_B **INA1**, PWM control input 001_B **INA2**, PWM control input 010_B **INA3**, PWM control input 011_B **INC1**, PWM control input 100_B **INB1**, PWM control input 101_B **INB2**, PWM control input 110_B **INB3**, PWM control input 111_B **INC4**, PWM control input
LS2_SRC_SEL	10:8	rwpht	**Low-side driver 2 PWM source selection** *Note: Can only be written while all CTRL1.*_PWM are 0.* 000_B **INA1**, PWM control input 001_B **INA2**, PWM control input 010_B **INA3**, PWM control input 011_B **INC2N**, Inverted PWM control input 100_B **INB1**, PWM control input 101_B **INB2**, PWM control input 110_B **INB3**, PWM control input 111_B **INC5**, PWM control input
HS2_SRC_SEL	14:12	rwpht	**High-side driver 2 PWM source selection** *Note: Can only be written while all CTRL1.*_PWM are 0.* 000_B **INA1**, PWM control input 001_B **INA2**, PWM control input 010_B **INA3**, PWM control input 011_B **INC2**, PWM control input 100_B **INB1**, PWM control input 101_B **INB2**, PWM control input 110_B **INB3**, PWM control input 111_B **INC5**, PWM control input

Register description BDRV

Field	Bits	Type	Description
LS3_SRC_SEL	18:16	rwpht	**Low-side driver 3 PWM source selection** *Note: Can only be written while all CTRL1.*_PWM are 0.* 000_B **INA1**, PWM control input 001_B **INA2**, PWM control input 010_B **INA3**, PWM control input 011_B **INC3N**, Inverted PWM control input 100_B **INB1**, PWM control input 101_B **INB2**, PWM control input 110_B **INB3**, PWM control input 111_B **INC6**, PWM control input
HS3_SRC_SEL	22:20	rwpht	**High-side driver 3 PWM source selection** *Note: Can only be written while all CTRL1.*_PWM are 0.* 000_B **INA1**, PWM control input 001_B **INA2**, PWM control input 010_B **INA3**, PWM control input 011_B **INC3**, PWM control input 100_B **INB1**, PWM control input 101_B **INB2**, PWM control input 110_B **INB3**, PWM control input 111_B **INC6**, PWM control input

Sequencer mapping

SEQMAP
Sequencer mapping **($000C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES														HB3_ACTDRV	HB3_SEQMAP
r														r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						HB2_ACTDRV	HB2_SEQMAP	RES						HB1_ACTDRV	HB1_SEQMAP
r						r	rw	r						r	rw

Field	Bits	Type	Description
HB1_SEQMAP	0	rw	**Half bridge 1 sequencer mapping** 0_B **LS**, Sequencer is mapped to low-side driver 1_B **HS**, Sequencer is mapped to high-side driver
HB1_ACTDRV	1	r	**Half bridge 1 active driver detected at switch-on** 0_B **LS**, Low-side driver detected as active 1_B **HS**, High-side driver detected as active

Register description BDRV

Field	Bits	Type	Description
RES	7:2, 15:10, 31:18	r	**Reserved** Always read as 0
HB2_SEQMAP	8	rw	**Half bridge 2 sequencer mapping** 0_B **LS**, Sequencer is mapped to low-side driver 1_B **HS**, Sequencer is mapped to high-side driver
HB2_ACTDRV	9	r	**Half bridge 2 active driver detected at switch-on** 0_B **LS**, Low-side driver detected as active 1_B **HS**, High-side driver detected as active
HB3_SEQMAP	16	rw	**Half bridge 3 sequencer mapping** 0_B **LS**, Sequencer is mapped to low-side driver 1_B **HS**, Sequencer is mapped to high-side driver
HB3_ACTDRV	17	r	**Half bridge 3 active driver detected at switch-on** 0_B **LS**, Low-side driver detected as active 1_B **HS**, High-side driver detected as active

Delay diagnosis timer

DLY_DIAG
Delay diagnosis timer **(0010_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES					**DLY_DIAG_SSET**	**DLY_DIAG_SCLR**	**DLY_DIAG_STS**	**RES**				**DLY_DIAG_DIRSEL**	**DLY_DIAG_CHSEL**		
r					w	w	rhxr	r				rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						**DLY_DIAG_TIM**									
r						r									

Field	Bits	Type	Description
DLY_DIAG_TIM	9:0	r	**External power MOSFET switch-on/-off timer result register** Nominal t_{delay} = 50 ns * (DLY_DIAG_TIM + 1), max. 51.2 µs
RES	15:10, 23:20, 31:27	r	**Reserved** Always read as 0

Field	Bits	Type	Description
DLY_DIAG_CHSEL	18:16	rw	**External power MOSFET switch-on/-off timer channel select** 000_B **DISABLE**, Switch-on/-off time measurement deactivated 001_B **LS1**, Measure low-side driver 1 switch-on/-off delay time 010_B **LS2**, Measure low-side driver 2 switch-on/-off delay time 011_B **LS3**, Measure low-side driver 3 switch-on/-off delay time 100_B **NU**, Not used, switch-on/-off time measurement deactivated 101_B **HS1**, Measure high-side driver 1 switch-on/-off delay time 110_B **HS2**, Measure high-side driver 2 switch-on/-off delay time 111_B **HS3**, Measure high-side driver 3 switch-on/-off delay time
DLY_DIAG_DIRSEL	19	rw	**External power MOSFET timer on/off select** 0_B **TURNOFF**, Measure switch-off time 1_B **TURNON**, Measure switch-on time
DLY_DIAG_STS	24	rhxr	**External power MOSFET switch-on/-off timer valid flag** *Note: Clear flag to initiate a measurement.* 0_B **NOTVALID**, Timer measurement ongoing 1_B **VALID**, Timer measurement finished
DLY_DIAG_SCLR	25	w	**External power MOSFET switch-on/-off timer valid flag clear** 0_B **KEEP**, Keep timer status 1_B **CLEAR**, Clear timer status
DLY_DIAG_SSET	26	w	**External power MOSFET switch-on/-off timer valid flag set** 0_B **KEEP**, Keep timer status 1_B **SET**, Set timer status

Charge pump control

CP_CTRL
Charge pump control **(0014_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES														CP_STG_AUTO	CP_1STAGE
r														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				CPLOW_TFILT_SEL		VCP_LOWSRC_SEL	VCP_LOWTH2	RES					CP_RDY_EN	RES	CP_EN
r				rw		rw	rw	r					rw	r	rw

Field	Bits	Type	Description
CP_EN	0	rw	**Charge pump enable** 0_B **DISABLE**, Charge pump power off 1_B **ENABLE**, Charge pump power on

Field	Bits	Type	Description
RES	1, 7:3, 15:12, 31:18	r	**Reserved** Always read as 0
CP_RDY_EN	2	rw	**Bridge driver on charge pump ready enable** 0_B **OFF**, Bridge driver can be enabled without charge pump voltage 1_B **ON**, Bridge driver can only be enabled when charge pump is ready; bridge driver is disabled when charge pump voltage drops below UV threshold
VCP_LOWTH2	8	rw	**Charge pump output voltage comparator falling threshold** 0_B **7V5**, Threshold 0 for VCP above VSD at 7.5 V 1_B **9V5**, Threshold 1 for VCP above VSD at 9.5 V
VCP_LOWSRC_SEL	9	rw	**Charge pump low voltage detection source select** 0_B **ADC2**, Charge pump voltage measurement in ADC2 is selected as UV detection source 1_B **COMP**, Charge pump voltage comparator is selected as UV detection source
CPLOW_TFILT_SEL	11:10	rw	**Filter time for charge pump voltage low diagnosis** 00_B **4u**, 4 µs filter time 01_B **8u**, 8 µs filter time 10_B **16u**, 16 µs filter time 11_B **32u**, 32 µs filter time
CP_1STAGE	16	rw	**Charge pump 1-stage mode select** 0_B **2STAGE**, Double stage mode enabled 1_B **1STAGE**, Single stage mode enabled
CP_STG_AUTO	17	rw	**Automatic switch to 1-stage mode and back to 2-stage mode depending on VSD** 0_B **FIXED**, Fixed stage mode as set in CP_1STAGE 1_B **AUTO**, Automatic stage selection mode enabled

Charge pump clock control

CP_CLK_CTRL
Charge pump clock control **(0018_H)** **RESET_TYPE_5 Value: 0001 4A16$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES														CPCLK_DIS_SET	CPCLK_EN
r														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	F_CP		DITH_UPPER					RES			DITH_LOWER				
r	rw		rw					r			rw				

Field	Bits	Type	Description
DITH_LOWER	4:0	rw	**CP clock divider boundary for lower frequency during dithering** Must be equal or greater than DITH_UPPER, see definition of F_CP
RES	7:5, 15, 31:18	r	**Reserved** Always read as 0
DITH_UPPER	12:8	rw	**CP clock divider boundary for upper frequency during dithering** Must be equal or less than DITH_LOWER, see definition of F_CP
F_CP	14:13	rw	**MSB of CP clock divider** The CP_CLK frequency divider (input frequency 20 MHz) is defined during dithering by the concatenation of F_CP and DITH_UPPER for the upper frequency boundary and DITH_LOWER for the lower frequency boundary 00_B **NU**, Don't use this setting 01_B **420k**, Frequency range 318 kHz ... 625 kHz 10_B **250k**, Frequency range 211 kHz ... 312 kHz 11_B **180k**, Frequency range 158 kHz ... 208 kHz
CPCLK_EN	16	rw	**Charge pump clock enable** 0_B **DISABLE**, Charge pump clock is switched off and has value according to CPCLKDIS_SET 1_B **ENABLE**, Charge pump clock is running
CPCLKDIS_SET	17	rw	**Charge pump clock value if disabled** 0_B **LOW**, Charge pump clock is 0 if disabled 1_B **HIGH**, Charge pump clock is 1 if disabled

Protection function control

PROT_CTRL
Protection function control **($001C_H$)** **RESET_TYPE_5 Value: $0100\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES			DRVx_VSDUP_DIS	DRVx_VSDLO_DIS	DRVx_VCPUP_DIS	DRVx_VCPLO_DIS	DRVx_VCPLO_SDEN	RES		HS3DRV_OCSDN_DIS	HS3DRV_HCDISCHG_DI	RES		LS3DRV_OCSDN_DIS	LS3DRV_HCDISCHG_DIS
r			rw	rw	rw	rw	rw	r		rw	rw	r		rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS2DRV_OCSDN_DIS	HS2DRV_HCDISCHG_DI	RES		LS2DRV_OCSDN_DIS	LS2DRV_HCDISCHG_DIS	RES		HS1DRV_OCSDN_DIS	HS1DRV_HCDISCHG_DI	RES		LS1DRV_OCSDN_DIS	LS1DRV_HCDISCHG_DIS
r		rw	rw	r		rw	rw	r		rw	rw	r		rw	rw

Field	Bits	Type	Description
LS1DRV_HCDISCHG_DIS	0	rw	**Low-side driver 1 high-current discharge disable** 0_B **ENABLE**, 1_B **DISABLE**,
LS1DRV_OCSDN_DIS	1	rw	**Low-side driver 1 overcurrent shutdown disable** 0_B **ENABLE**, 1_B **DISABLE**,
RES	3:2, 7:6, 11:10, 15:14, 19:18, 23:22, 31:29	r	**Reserved** Always read as 0
HS1DRV_HCDISCHG_DIS	4	rw	**High-side driver 1 high-current discharge disable** 0_B **ENABLE**, 1_B **DISABLE**,
HS1DRV_OCSDN_DIS	5	rw	**High-side driver 1 overcurrent shutdown disable** 0_B **ENABLE**, 1_B **DISABLE**,
LS2DRV_HCDISCHG_DIS	8	rw	**Low-side driver 2 high-current discharge disable** 0_B **ENABLE**, 1_B **DISABLE**,
LS2DRV_OCSDN_DIS	9	rw	**Low-side driver 2 overcurrent shutdown disable** 0_B **ENABLE**, 1_B **DISABLE**,
HS2DRV_HCDISCHG_DIS	12	rw	**High-side driver 2 high-current discharge disable** 0_B **ENABLE**, 1_B **DISABLE**,
HS2DRV_OCSDN_DIS	13	rw	**High-side driver 2 overcurrent shutdown disable** 0_B **ENABLE**, 1_B **DISABLE**,
LS3DRV_HCDISCHG_DIS	16	rw	**Low-side driver 3 high-current discharge disable** 0_B **ENABLE**, 1_B **DISABLE**,
LS3DRV_OCSDN_DIS	17	rw	**Low-side driver 3 overcurrent shutdown disable** 0_B **ENABLE**, 1_B **DISABLE**,
HS3DRV_HCDISCHG_DIS	20	rw	**High-side driver 3 high-current discharge disable** 0_B **ENABLE**, 1_B **DISABLE**,
HS3DRV_OCSDN_DIS	21	rw	**High-side driver 3 overcurrent shutdown disable** 0_B **ENABLE**, 1_B **DISABLE**,

Register description BDRV

Field	Bits	Type	Description
DRVx_VCPLO_SDEN	24	rw	**Driver shutdown on charge pump low voltage** 0_B **SDDIS**, Bridge driver not disabled on charge pump undervoltage 1_B **SDEN**, Bridge driver disabled on charge pump undervoltage
DRVx_VCPLO_DIS	25	rw	**Disable driver discharge on charge pump low voltage** 0_B **DRVOFF**, DRVx discharged on charge pump undervoltage 1_B **DRVNOTOFF**, DRVx keep state on charge pump undervoltage
DRVx_VCPUP_DIS	26	rw	**Disable driver discharge on charge pump high voltage** 0_B **DRVOFF**, DRVx discharged on charge pump overvoltage 1_B **DRVNOTOFF**, DRVx keep state on charge pump overvoltage
DRVx_VSDLO_DIS	27	rw	**Disable driver discharge on VSD low voltage** 0_B **DRVOFF**, DRVx discharged on VSD undervoltage 1_B **DRVNOTOFF**, DRVx keep state on VSD undervoltage
DRVx_VSDUP_DIS	28	rw	**Disable driver discharge on VSD high voltage** 0_B **DRVOFF**, DRVx discharged on VSD overvoltage 1_B **DRVNOTOFF**, DRVx keep state on VSD overvoltage

Bridge driver status

STS
Bridge driver status **(0020_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VCP_LOTH2_STS	**VSD_OV_STS**	**VSD_CP1ST_STS**	**VSD_UPTH_STS**	**VSD_LOTH_STS**	**VCP_UPTH_STS**	**VCP_LOTH1_STS**	**CP_OTSD_STS**	**SH3_HIGH_STS**	**SH3_LOW_STS**	**HS3_DS_STS**	**HS3_OC_STS**	**RES**		**LS3_DS_STS**	**LS3_OC_STS**
rhxr	r	r	r	r	r	r	r	r	r	rhxr	rhxr	r		rhxr	rhxr

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SH2_HIGH_STS	**SH2_LOW_STS**	**HS2_DS_STS**	**HS2_OC_STS**	**RES**		**LS2_DS_STS**	**LS2_OC_STS**	**SH1_HIGH_STS**	**SH1_LOW_STS**	**HS1_DS_STS**	**HS1_OC_STS**	**RES**		**LS1_DS_STS**	**LS1_OC_STS**
r	r	rhxr	rhxr	r		rhxr	rhxr	r	r	rhxr	rhxr	r		rhxr	rhxr

Field	Bits	Type	Description
LS1_OC_STS	0	rhxr	**External low-side 1 MOSFET overcurrent status** 0_B **VDSLOW**, No overcurrent condition detected 1_B **VDSHIGH**, Overcurrent detected; switch is turned off
LS1_DS_STS	1	rhxr	**Low-side driver 1 off-state drain source monitoring status** 0_B **VDSHIGH**, No failure on external MOSFET detected 1_B **VDSLOW**, Failure on external MOSFET detected

Field	Bits	Type	Description
RES	3:2, 11:10, 19:18	r	**Reserved** Always read as 0
HS1_OC_STS	4	rhxr	**External high-side 1 MOSFET overcurrent status** 0_B **VDSLOW**, No overcurrent condition detected 1_B **VDSHIGH**, Overcurrent detected; switch is turned off
HS1_DS_STS	5	rhxr	**High-side driver 1 off-state drain source monitoring status** 0_B **VDSHIGH**, No failure on external MOSFET detected 1_B **VDSLOW**, Failure on external MOSFET detected
SH1_LOW_STS	6	r	**SH1 voltage low comparator status** 0_B **VSHLOW**, SH voltage below low comparator threshold 1_B **VSHHIGH**, SH voltage above low comparator threshold
SH1_HIGH_STS	7	r	**SH1 voltage high comparator status** 0_B **VSHLOW**, SH voltage below high comparator threshold 1_B **VSHHIGH**, SH voltage above high comparator threshold
LS2_OC_STS	8	rhxr	**External low-side 2 MOSFET overcurrent status** 0_B **VDSLOW**, No overcurrent condition detected 1_B **VDSHIGH**, Overcurrent detected; switch is turned off
LS2_DS_STS	9	rhxr	**Low-side driver 2 off-state drain source monitoring status** 0_B **VDSHIGH**, No failure on external MOSFET detected 1_B **VDSLOW**, Failure on external MOSFET detected
HS2_OC_STS	12	rhxr	**External high-side 2 MOSFET overcurrent status** 0_B **VDSLOW**, No overcurrent condition detected 1_B **VDSHIGH**, Overcurrent detected; switch is turned off
HS2_DS_STS	13	rhxr	**High-side driver 2 off-state drain source monitoring status** 0_B **VDSHIGH**, No failure on external MOSFET detected 1_B **VDSLOW**, Failure on external MOSFET detected
SH2_LOW_STS	14	r	**SH2 voltage low comparator status** 0_B **VSHLOW**, SH voltage below low comparator threshold 1_B **VSHHIGH**, SH voltage above low comparator threshold
SH2_HIGH_STS	15	r	**SH2 voltage high comparator status** 0_B **VSHLOW**, SH voltage below high comparator threshold 1_B **VSHHIGH**, SH voltage above high comparator threshold
LS3_OC_STS	16	rhxr	**External low-side 3 MOSFET overcurrent status** 0_B **VDSLOW**, No overcurrent condition detected 1_B **VDSHIGH**, Overcurrent detected; switch is turned off
LS3_DS_STS	17	rhxr	**Low-side driver 3 off-state drain source monitoring status** 0_B **VDSHIGH**, No failure on external MOSFET detected 1_B **VDSLOW**, Failure on external MOSFET detected
HS3_OC_STS	20	rhxr	**External high-side 3 MOSFET overcurrent status** 0_B **VDSLOW**, No overcurrent condition detected 1_B **VDSHIGH**, Overcurrent detected; switch is turned off
HS3_DS_STS	21	rhxr	**High-side driver 3 off-state drain source monitoring status** 0_B **VDSHIGH**, No failure on external MOSFET detected 1_B **VDSLOW**, Failure on external MOSFET detected

Field	Bits	Type	Description
SH3_LOW_STS	22	r	**SH3 voltage low comparator status** 0_B **VSHLOW**, SH voltage below low comparator threshold 1_B **VSHHIGH**, SH voltage above low comparator threshold
SH3_HIGH_STS	23	r	**SH3 voltage high comparator status** 0_B **VSHLOW**, SH voltage below high comparator threshold 1_B **VSHHIGH**, SH voltage above high comparator threshold
CP_OTSD_STS	24	r	**Charge pump temperature measurement status** This bit only replicates the ADC2 status signal CP_OTSD, which is a latched status and not a real-time value. 0_B **NO_OT**, No overtemperature detected 1_B **OT**, Overtemperature in charge pump detected
VCP_LOTH1_STS	25	r	**VCP measurement low voltage status** This bit only replicates the ADC2 status signal VCP_LOTH, which is a latched status and not a real-time value. 0_B **VCP_OK**, No undervoltage detected 1_B **VCP_LOW**, Undervoltage on charge pump output detected
VCP_UPTH_STS	26	r	**VCP measurement high voltage status** This bit only replicates the ADC2 status signal VCP_UPTH, which is a latched status and not a real-time value. 0_B **VCP_OK**, No overvoltage detected 1_B **VCP_HIGH**, Overvoltage on charge pump output detected
VSD_LOTH_STS	27	r	**VSD measurement low voltage status** This bit only replicates the ADC2 status signal VSD_LOTH, which is a latched status and not a real-time value. 0_B **VSD_OK**, No undervoltage detected 1_B **VSD_LOW**, Undervoltage on VSD supply detected
VSD_UPTH_STS	28	r	**VSD measurement high voltage status** This bit only replicates the ADC2 status signal VSD_UPTH, which is a latched status and not a real-time value. 0_B **VSD_OK**, No overvoltage detected 1_B **VSD_HIGH**, Overvoltage on VSD supply detected
VSD_CP1ST_STS	29	r	**VSD measurement status for automatic CP stage selection** This bit only replicates the ADC2 status signal VSD_CP1ST, which is a latched status and not a real-time value. 0_B **2STAGE**, VSD range for CP two-stage operation 1_B **1STAGE**, VSD range for CP one-stage operation
VSD_OV_STS	30	r	**VSD overvoltage comparator status** 0_B **VSD_OK**, No overvoltage condition detected 1_B **VSD_HIGH**, Overvoltage on VSD supply detected in power-down mode
VCP_LOTH2_STS	31	rhxr	**Charge pump analog comparator low voltage status** 0_B **VCP_OK**, No undervoltage detected 1_B **VCP_LOW**, Undervoltage on charge pump output detected

Register description BDRV

Bridge driver status clear

STSCLR
Bridge driver status clear **(0024$_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VCP_LOTH2_SC				RES						HS3_DS_SC	HS3_OC_SC	RES		LS3_DS_SC	LS3_OC_SC
w				r						w	w	r		w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS2_DS_SC	HS2_OC_SC	RES		LS2_DS_SC	LS2_OC_SC	RES		HS1_DS_SC	HS1_OC_SC	RES		LS1_DS_SC	LS1_OC_SC
r		w	w	r		w	w	r		w	w	r		w	w

Field	Bits	Type	Description
LS1_OC_SC	0	w	**External low-side 1 MOSFET overcurrent status clear** 0_B **KEEP**, 1_B **CLEAR**,
LS1_DS_SC	1	w	**Low-side driver 1 off-state drain source monitoring status clear** 0_B **KEEP**, 1_B **CLEAR**,
RES	3:2, 7:6, 11:10, 15:14, 19:18, 30:22	r	**Reserved** Always read as 0
HS1_OC_SC	4	w	**External high-side 1 MOSFET overcurrent status clear** 0_B **KEEP**, 1_B **CLEAR**,
HS1_DS_SC	5	w	**High-side driver 1 off-state drain source monitoring status clear** 0_B **KEEP**, 1_B **CLEAR**,
LS2_OC_SC	8	w	**External low-side 2 MOSFET overcurrent status clear** 0_B **KEEP**, 1_B **CLEAR**,
LS2_DS_SC	9	w	**Low-side driver 2 off-state drain source monitoring status clear** 0_B **KEEP**, 1_B **CLEAR**,
HS2_OC_SC	12	w	**External high-side 2 MOSFET overcurrent status clear** 0_B **KEEP**, 1_B **CLEAR**,

Register description BDRV

Field	Bits	Type	Description
HS2_DS_SC	13	w	**High-side driver 2 off-state drain source monitoring status clear** 0_B **KEEP**, 1_B **CLEAR**,
LS3_OC_SC	16	w	**External low-side 3 MOSFET overcurrent status clear** 0_B **KEEP**, 1_B **CLEAR**,
LS3_DS_SC	17	w	**Low-side driver 3 off-state drain source monitoring status clear** 0_B **KEEP**, 1_B **CLEAR**,
HS3_OC_SC	20	w	**External high-side 3 MOSFET overcurrent status clear** 0_B **KEEP**, 1_B **CLEAR**,
HS3_DS_SC	21	w	**High-side driver 3 off-state drain source monitoring status clear** 0_B **KEEP**, 1_B **CLEAR**,
VCP_LOTH2_SC	31	w	**Charge pump comparator low voltage status clear** 0_B **KEEP**, 1_B **CLEAR**,

Bridge driver status set

STSSET
Bridge driver status set **(0028_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VCP_LOTH2_SS				**RES**						**HS3_DS_SS**	**HS3_OC_SS**	**RES**		**LS3_DS_SS**	**LS3_OC_SS**
w				r						w	w	r		w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		**HS2_DS_SS**	**HS2_OC_SS**	**RES**		**LS2_DS_SS**	**LS2_OC_SS**	**RES**		**HS1_DS_SS**	**HS1_OC_SS**	**RES**		**LS1_DS_SS**	**LS1_OC_SS**
r		w	w	r		w	w	r		w	w	r		w	w

Field	Bits	Type	Description
LS1_OC_SS	0	w	**External low-side 1 MOSFET overcurrent status set** 0_B **KEEP**, 1_B **SET**,
LS1_DS_SS	1	w	**Low-side driver 1 off-state drain source monitoring status set** 0_B **KEEP**, 1_B **SET**,

Field	Bits	Type	Description
RES	3:2, 7:6, 11:10, 15:14, 19:18, 30:22	r	**Reserved** Always read as 0
HS1_OC_SS	4	w	**External high-side 1 MOSFET overcurrent status set** 0_B **KEEP**, 1_B **SET**,
HS1_DS_SS	5	w	**High-side driver 1 off-state drain source monitoring status set** 0_B **KEEP**, 1_B **SET**,
LS2_OC_SS	8	w	**External low-side 2 MOSFET overcurrent status set** 0_B **KEEP**, 1_B **SET**,
LS2_DS_SS	9	w	**Low-side driver 2 off-state drain source monitoring status set** 0_B **KEEP**, 1_B **SET**,
HS2_OC_SS	12	w	**External high-side 2 MOSFET overcurrent status set** 0_B **KEEP**, 1_B **SET**,
HS2_DS_SS	13	w	**High-side driver 2 off-state drain source monitoring status set** 0_B **KEEP**, 1_B **SET**,
LS3_OC_SS	16	w	**External low-side 3 MOSFET overcurrent status set** 0_B **KEEP**, 1_B **SET**,
LS3_DS_SS	17	w	**Low-side driver 3 off-state drain source monitoring status set** 0_B **KEEP**, 1_B **SET**,
HS3_OC_SS	20	w	**External high-side 3 MOSFET overcurrent status set** 0_B **KEEP**, 1_B **SET**,
HS3_DS_SS	21	w	**High-side driver 3 off-state drain source monitoring status set** 0_B **KEEP**, 1_B **SET**,
VCP_LOTH2_SS	31	w	**Charge pump comparator low voltage status set** 0_B **KEEP**, 1_B **SET**,

Register description BDRV

Bridge driver interrupt status

IRQS

Bridge driver interrupt status **(002C$_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VCP_LOTH2_IS	HB3_ACTDRV_IS	HB2_ACTDRV_IS	HB1_ACTDRV_IS	SEQ_ERR_IS	HB3_ASEQ_IS	HB2_ASEQ_IS	HB1_ASEQ_IS	RES		HS3_DS_IS	HS3_OC_IS	RES		LS3_DS_IS	LS3_OC_IS
rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	rhxre	r		rhxre	rhxre	r		rhxre	rhxre

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS2_DS_IS	HS2_OC_IS	RES		LS2_DS_IS	LS2_OC_IS	RES		HS1_DS_IS	HS1_OC_IS	RES		LS1_DS_IS	LS1_OC_IS
r		rhxre	rhxre	r		rhxre	rhxre	r		rhxre	rhxre	r		rhxre	rhxre

Field	Bits	Type	Description
LS1_OC_IS	0	rhxre	**External low-side 1 MOSFET overcurrent interrupt status** 0_B **VDSLOW**, No overcurrent condition detected 1_B **VDSHIGH**, Overcurrent detected; switch is automatically shut down
LS1_DS_IS	1	rhxre	**Low-side driver 1 off-state drain source monitoring interrupt status** 0_B **VDSHIGH**, No failure on external MOSFET detected 1_B **VDSLOW**, Failure on external MOSFET detected
RES	3:2, 7:6, 11:10, 15:14, 19:18, 23:22	r	**Reserved** Always read as 0
HS1_OC_IS	4	rhxre	**External high-side 1 MOSFET overcurrent interrupt status** 0_B **VDSLOW**, No overcurrent condition detected 1_B **VDSHIGH**, Overcurrent detected; switch is automatically shut down
HS1_DS_IS	5	rhxre	**High-side driver 1 off-state drain source monitoring interrupt status** 0_B **VDSHIGH**, No failure on external MOSFET detected 1_B **VDSLOW**, Failure on external MOSFET detected
LS2_OC_IS	8	rhxre	**External low-side 2 MOSFET overcurrent interrupt status** 0_B **VDSLOW**, No overcurrent condition detected 1_B **VDSHIGH**, Overcurrent detected; switch is automatically shut down
LS2_DS_IS	9	rhxre	**Low-side driver 2 off-state drain source monitoring interrupt status** 0_B **VDSHIGH**, No failure on external MOSFET detected 1_B **VDSLOW**, Failure on external MOSFET detected

Field	Bits	Type	Description
HS2_OC_IS	12	rhxre	**External high-side 2 MOSFET overcurrent interrupt status** 0_B **VDSLOW**, No overcurrent condition detected 1_B **VDSHIGH**, Overcurrent detected; switch is automatically shut down
HS2_DS_IS	13	rhxre	**High-side driver 2 off-state drain source monitoring interrupt status** 0_B **VDSHIGH**, No failure on external MOSFET detected 1_B **VDSLOW**, Failure on external MOSFET detected
LS3_OC_IS	16	rhxre	**External low-side 3 MOSFET overcurrent interrupt status** 0_B **VDSLOW**, No overcurrent condition detected 1_B **VDSHIGH**, Overcurrent detected; switch is automatically shut down
LS3_DS_IS	17	rhxre	**Low-side driver 3 off-state drain source monitoring interrupt status** 0_B **VDSHIGH**, No failure on external MOSFET detected 1_B **VDSLOW**, Failure on external MOSFET detected
HS3_OC_IS	20	rhxre	**External high-side 3 MOSFET overcurrent interrupt status** 0_B **VDSLOW**, No overcurrent condition detected 1_B **VDSHIGH**, Overcurrent detected; switch is automatically shut down
HS3_DS_IS	21	rhxre	**High-side driver 3 off-state drain source monitoring interrupt status** 0_B **VDSHIGH**, No failure on external MOSFET detected 1_B **VDSLOW**, Failure on external MOSFET detected
HB1_ASEQ_IS	24	rhxre	**Half bridge 1 adaptive sequencer interrupt status** *Note: Interrupt is set on any half bridge 1 error reported in ASEQONSTS or ASEQOFFSTS.* 0_B **NOASEQISS**, No adaptive sequencer issue detected 1_B **ASEQISS**, Adaptive sequencer issue detected
HB2_ASEQ_IS	25	rhxre	**Half bridge 2 adaptive sequencer interrupt status** *Note: Interrupt is set on any half bridge 2 error reported in ASEQONSTS or ASEQOFFSTS.* 0_B **NOASEQISS**, No adaptive sequencer issue detected 1_B **ASEQISS**, Adaptive sequencer issue detected
HB3_ASEQ_IS	26	rhxre	**Half bridge 3 adaptive sequencer interrupt status** *Note: Interrupt is set on any half bridge 3 error reported in ASEQONSTS or ASEQOFFSTS.* 0_B **NOASEQISS**, No adaptive sequencer issue detected 1_B **ASEQISS**, Adaptive sequencer issue detected
SEQ_ERR_IS	27	rhxre	**Driver sequence error interrupt status** 0_B **SEQOK**, No cross current 1_B **SEQFAIL**, High-side and low-side drivers of same half bridge concurrently activated, output protection activated
HB1_ACTDRV_IS	28	rhxre	**Half bridge 1 active driver detection interrupt status** 0_B **STAYED**, Active driver did not change 1_B **CHANGED**, Active driver change detected

Register description BDRV

Field	Bits	Type	Description
HB2_ACTDRV_IS	29	rhxre	**Half bridge 2 active driver detection interrupt status** 0_B **STAYED**, Active driver did not change 1_B **CHANGED**, Active driver change detected
HB3_ACTDRV_IS	30	rhxre	**Half bridge 3 active driver detection interrupt status** 0_B **STAYED**, Active driver did not change 1_B **CHANGED**, Active driver change detected
VCP_LOTH2_IS	31	rhxre	**Charge pump comparator low voltage interrupt status** 0_B **VCP_OK**, No undervoltage detected 1_B **VCP_LOW**, Undervoltage on charge pump output detected

Bridge driver interrupt status clear

IRQCLR
Bridge driver interrupt status clear **(0030_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VCP_LOTH2_ISC	HB3_ACTDRV_ISC	HB2_ACTDRV_ISC	HB1_ACTDRV_ISC	SEQ_ERR_ISC	HB3_ASEQ_ISC	HB2_ASEQ_ISC	HB1_ASEQ_ISC	RES		HS3_DS_ISC	HS3_OC_ISC	RES		LS3_DS_ISC	LS3_OC_ISC
w	w	w	w	w	w	w	w	r		w	w	r		w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS2_DS_ISC	HS2_OC_ISC	RES		LS2_DS_ISC	LS2_OC_ISC	RES		HS1_DS_ISC	HS1_OC_ISC	RES		LS1_DS_ISC	LS1_OC_ISC
r		w	w	r		w	w	r		w	w	r		w	w

Field	Bits	Type	Description
LS1_OC_ISC	0	w	**External low-side 1 MOSFET overcurrent interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
LS1_DS_ISC	1	w	**Low-side driver 1 off-state drain source monitoring interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
RES	3:2, 7:6, 11:10, 15:14, 19:18, 23:22	r	**Reserved** Always read as 0
HS1_OC_ISC	4	w	**External high-side 1 MOSFET overcurrent interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,

Field	Bits	Type	Description
HS1_DS_ISC	5	w	**High-side driver 1 off-state drain source monitoring interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
LS2_OC_ISC	8	w	**External low-side 2 MOSFET overcurrent interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
LS2_DS_ISC	9	w	**Low-side driver 2 off-state drain source monitoring interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
HS2_OC_ISC	12	w	**External high-side 2 MOSFET overcurrent interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
HS2_DS_ISC	13	w	**High-side driver 2 off-state drain source monitoring interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
LS3_OC_ISC	16	w	**External low-side 3 MOSFET overcurrent interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
LS3_DS_ISC	17	w	**Low-side driver 3 off-state drain source monitoring interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
HS3_OC_ISC	20	w	**External high-side 3 MOSFET overcurrent interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
HS3_DS_ISC	21	w	**High-side driver 3 off-state drain source monitoring interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
HB1_ASEQ_ISC	24	w	**Half bridge 1 adaptive sequencer interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
HB2_ASEQ_ISC	25	w	**Half bridge 2 adaptive sequencer interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
HB3_ASEQ_ISC	26	w	**Half bridge 3 adaptive sequencer interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
SEQ_ERR_ISC	27	w	**Driver sequence error interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,

Register description BDRV

Field	Bits	Type	Description
HB1_ACTDRV_ISC	28	w	**Half bridge 1 active driver detection interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
HB2_ACTDRV_ISC	29	w	**Half bridge 2 active driver detection interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
HB3_ACTDRV_ISC	30	w	**Half bridge 3 active driver detection interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
VCP_LOTH2_ISC	31	w	**Charge pump comparator low voltage interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,

Bridge driver interrupt status set

IRQSET
Bridge driver interrupt status set **(0034_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VCP_LOTH2_ISS	HB3_ACTDRV_ISS	HB2_ACTDRV_ISS	HB1_ACTDRV_ISS	SEQ_ERR_ISS	HB3_ASEQ_ISS	HB2_ASEQ_ISS	HB1_ASEQ_ISS	RES		HS3_DS_ISS	HS3_OC_ISS	RES		LS3_DS_ISS	LS3_OC_ISS
w	w	w	w	w	w	w	w	r		w	w	r		w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS2_DS_ISS	HS2_OC_ISS	RES		LS2_DS_ISS	LS2_OC_ISS	RES		HS1_DS_ISS	HS1_OC_ISS	RES		LS1_DS_ISS	LS1_OC_ISS
r		w	w	r		w	w	r		w	w	r		w	w

Field	Bits	Type	Description
LS1_OC_ISS	0	w	**External low-side 1 MOSFET overcurrent interrupt status set** 0_B **KEEP**, 1_B **SET**,
LS1_DS_ISS	1	w	**Low-side driver 1 off-state drain source monitoring interrupt status set** 0_B **KEEP**, 1_B **SET**,
RES	3:2, 7:6, 11:10, 15:14, 19:18, 23:22	r	**Reserved** Always read as 0

Field	Bits	Type	Description
HS1_OC_ISS	4	w	**External high-side 1 MOSFET overcurrent interrupt status set** 0_B **KEEP**, 1_B **SET**,
HS1_DS_ISS	5	w	**High-side driver 1 off-state drain source monitoring interrupt status set** 0_B **KEEP**, 1_B **SET**,
LS2_OC_ISS	8	w	**External low-side 2 MOSFET overcurrent interrupt status set** 0_B **KEEP**, 1_B **SET**,
LS2_DS_ISS	9	w	**Low-side driver 2 off-state drain source monitoring interrupt status set** 0_B **KEEP**, 1_B **SET**,
HS2_OC_ISS	12	w	**External high-side 2 MOSFET overcurrent interrupt status set** 0_B **KEEP**, 1_B **SET**,
HS2_DS_ISS	13	w	**High-side driver 2 off-state drain source monitoring interrupt status set** 0_B **KEEP**, 1_B **SET**,
LS3_OC_ISS	16	w	**External low-side 3 MOSFET overcurrent interrupt status set** 0_B **KEEP**, 1_B **SET**,
LS3_DS_ISS	17	w	**Low-side driver 3 off-state drain source monitoring interrupt status set** 0_B **KEEP**, 1_B **SET**,
HS3_OC_ISS	20	w	**External high-side 3 MOSFET overcurrent interrupt status set** 0_B **KEEP**, 1_B **SET**,
HS3_DS_ISS	21	w	**High-side driver 3 off-state drain source monitoring interrupt status set** 0_B **KEEP**, 1_B **SET**,
HB1_ASEQ_ISS	24	w	**Half bridge 1 adaptive sequencer interrupt status set** 0_B **KEEP**, 1_B **SET**,
HB2_ASEQ_ISS	25	w	**Half bridge 2 adaptive sequencer interrupt status set** 0_B **KEEP**, 1_B **SET**,
HB3_ASEQ_ISS	26	w	**Half bridge 3 adaptive sequencer interrupt status set** 0_B **KEEP**, 1_B **SET**,

Register description BDRV

Field	Bits	Type	Description
SEQ_ERR_ISS	27	w	**Driver sequence error interrupt status set** 0_B **KEEP**, 1_B **SET**,
HB1_ACTDRV_ISS	28	w	**Half bridge 1 active driver detection interrupt status set** 0_B **KEEP**, 1_B **SET**,
HB2_ACTDRV_ISS	29	w	**Half bridge 2 active driver detection interrupt status set** 0_B **KEEP**, 1_B **SET**,
HB3_ACTDRV_ISS	30	w	**Half bridge 3 active driver detection interrupt status set** 0_B **KEEP**, 1_B **SET**,
VCP_LOTH2_ISS	31	w	**Charge pump comparator low voltage interrupt status set** 0_B **KEEP**, 1_B **SET**,

Bridge driver interrupt enable

IRQEN
Bridge driver interrupt enable **(0038_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VCP_LOTH2_IEN	HB3_ACTDRV_IEN	HB2_ACTDRV_IEN	HB1_ACTDRV_IEN	SEQ_ERR_IEN	HB3_ASEQ_IEN	HB2_ASEQ_IEN	HB1_ASEQ_IEN	RES		HS3_DS_IEN	HS3_OC_IEN	RES		LS3_DS_IEN	LS3_OC_IEN
rw	rw	rw	rw	rw	rw	rw	rw	r		rw	rw	r		rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS2_DS_IEN	HS2_OC_IEN	RES		LS2_DS_IEN	LS2_OC_IEN	RES		HS1_DS_IEN	HS1_OC_IEN	RES		LS1_DS_IEN	LS1_OC_IEN
r		rw	rw	r		rw	rw	r		rw	rw	r		rw	rw

Field	Bits	Type	Description
LS1_OC_IEN	0	rw	**External low-side 1 MOSFET overcurrent interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
LS1_DS_IEN	1	rw	**Low-side driver 1 off-state drain source monitoring interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,

Field	Bits	Type	Description
RES	3:2, 7:6, 11:10, 15:14, 19:18, 23:22	r	**Reserved** Always read as 0
HS1_OC_IEN	4	rw	**External high-side 1 MOSFET overcurrent interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
HS1_DS_IEN	5	rw	**High-side driver 1 off-state drain source monitoring interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
LS2_OC_IEN	8	rw	**External low-side 2 MOSFET overcurrent interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
LS2_DS_IEN	9	rw	**Low-side driver 2 off-state drain source monitoring interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
HS2_OC_IEN	12	rw	**External high-side 2 MOSFET overcurrent interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
HS2_DS_IEN	13	rw	**High-side driver 2 off-state drain source monitoring interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
LS3_OC_IEN	16	rw	**External low-side 3 MOSFET overcurrent interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
LS3_DS_IEN	17	rw	**Low-side driver 3 off-state drain source monitoring interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
HS3_OC_IEN	20	rw	**External high-side 3 MOSFET overcurrent interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
HS3_DS_IEN	21	rw	**High-side driver 3 off-state drain source monitoring interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
HB1_ASEQ_IEN	24	rw	**Half bridge 1 adaptive sequencer interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,

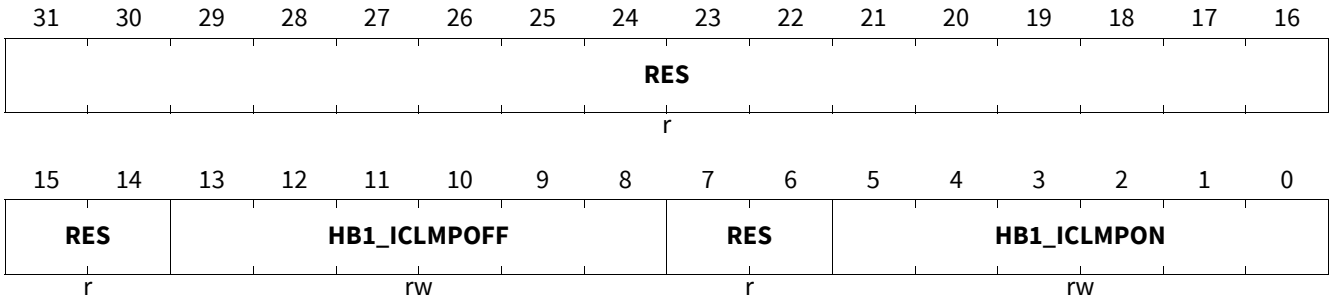
Register description BDRV

Field	Bits	Type	Description
HB2_ASEQ_IEN	25	rw	**Half bridge 2 adaptive sequencer interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
HB3_ASEQ_IEN	26	rw	**Half bridge 3 adaptive sequencer interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
SEQ_ERR_IEN	27	rw	**Driver sequence error interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
HB1_ACTDRV_IEN	28	rw	**Half bridge 1 active driver detection interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
HB2_ACTDRV_IEN	29	rw	**Half bridge 2 active driver detection interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
HB3_ACTDRV_IEN	30	rw	**Half bridge 3 active driver detection interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
VCP_LOTH2_IEN	31	rw	**Charge pump comparator low voltage interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,

Half bridge 1 gate current clamping value

HB1IGATECLMPC

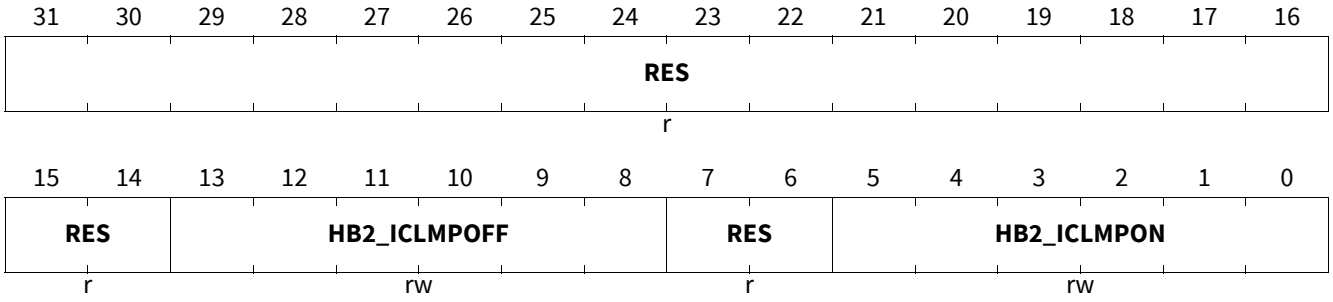
Half bridge 1 gate current clamping value **($003C_H$)** **RESET_TYPE_5 Value: 0000 0404$_H$**



Field	Bits	Type	Description
HB1_ICLMPON	5:0	rw	**Half bridge 1 current clamping value for on state** Nominal $I_{clmp(on)}$ = 5 mA ... 350 mA in 64 steps
RES	7:6, 31:14	r	**Reserved** Always read as 0
HB1_ICLMPOFF	13:8	rw	**Half bridge 1 current clamping value for off state** Nominal $I_{clmp(off)}$ = 5 mA ... 350 mA in 64 steps

Half bridge 2 gate current clamping value

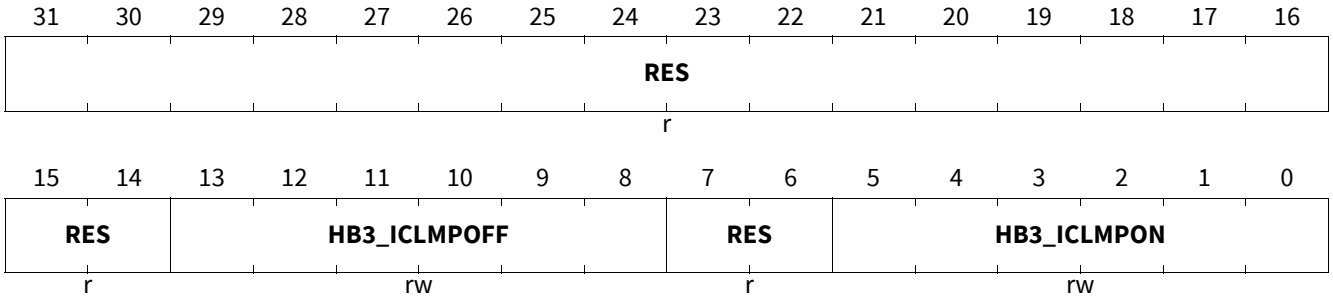
HB2IGATECLMPC
Half bridge 2 gate current clamping value (0040$_H$) RESET_TYPE_5 Value: 0000 0404$_H$



Field	Bits	Type	Description
HB2_ICLMPON	5:0	rw	**Half bridge 2 current clamping value for on state** Nominal $I_{clmp(on)}$ = 5 mA ... 350 mA in 64 steps
RES	7:6, 31:14	r	**Reserved** Always read as 0
HB2_ICLMPOFF	13:8	rw	**Half bridge 2 current clamping value for off state** Nominal $I_{clmp(off)}$ = 5 mA ... 350 mA in 64 steps

Half bridge 3 gate current clamping value

HB3IGATECLMPC
Half bridge 3 gate current clamping value (0044$_H$) RESET_TYPE_5 Value: 0000 0404$_H$



Field	Bits	Type	Description
HB3_ICLMPON	5:0	rw	**Half bridge 3 current clamping value for on state** Nominal $I_{clmp(on)}$ = 5 mA ... 350 mA in 64 steps
RES	7:6, 31:14	r	**Reserved** Always read as 0
HB3_ICLMPOFF	13:8	rw	**Half bridge 3 current clamping value for off state** Nominal $I_{clmp(off)}$ = 5 mA ... 350 mA in 64 steps

Low-side driver 1 active free-wheeling time control

LS1AFTC

Low-side driver 1 active free-wheeling time control(0048_H) **RESET_TYPE_5 Value: $0000\ 1414_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LS1_TAFON								LS1_TAFOFF							
rw								rw							

Field	Bits	Type	Description
LS1_TAFOFF	7:0	rw	**Low-side driver 1 active free-wheeling switch-off time setting** Nominal $t_{af(off)}$ = 50 ns * (LS1_TAFOFF + 1), max. 12.8 µs
LS1_TAFON	15:8	rw	**Low-side driver 1 active free-wheeling switch-on time setting** Nominal $t_{af(on)}$ = 50 ns * (LS1_TAFON + 1), max. 12.8 µs
RES	31:16	r	**Reserved** Always read as 0

Low-side driver 1 active free-wheeling current control

LS1AFIC

Low-side driver 1 active free-wheeling current control($004C_H$) **RESET_TYPE_5 Value: $0000\ 1616_H$**

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

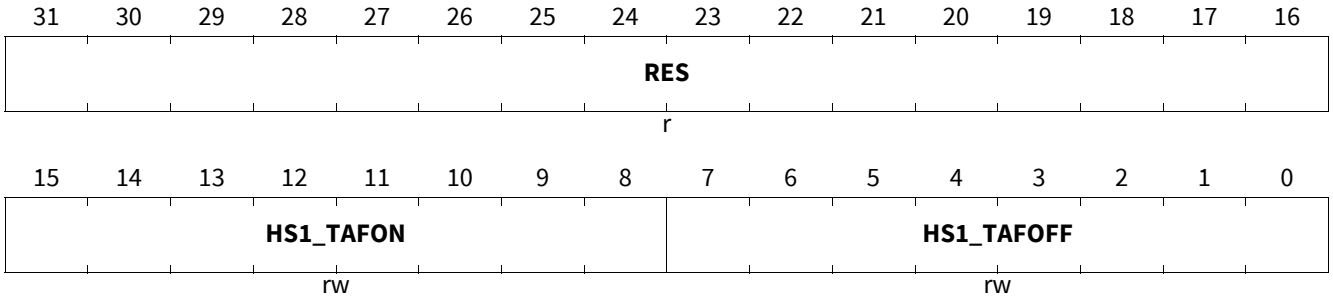
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		LS1_IAFON						RES		LS1_IAFOFF					
r		rw						r		rw					

Field	Bits	Type	Description
LS1_IAFOFF	5:0	rw	**Low-side driver 1 active free-wheeling switch-off current setting** Nominal $I_{af(off)}$ = 5 mA ... 350 mA in 64 steps
RES	7:6, 31:14	r	**Reserved** Always read as 0
LS1_IAFON	13:8	rw	**Low-side driver 1 active free-wheeling switch-on current setting** Nominal $I_{af(on)}$ = 5 mA ... 350 mA in 64 steps

High-side driver 1 active free-wheeling time control

HS1AFTC

High-side driver 1 active free-wheeling time control(0050_H) **RESET_TYPE_5 Value: $0000\ 1414_H$**

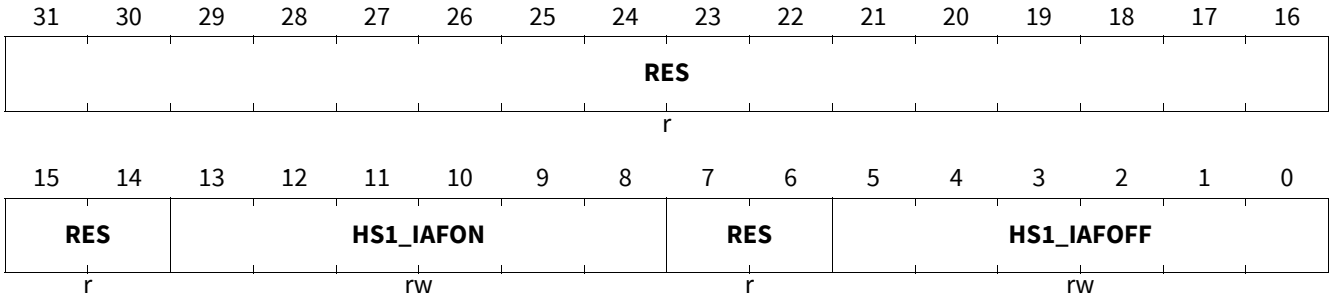


Field	Bits	Type	Description
HS1_TAFOFF	7:0	rw	**High-side driver 1 active free-wheeling switch-off time setting** Nominal $t_{af(off)}$ = 50 ns * (HS1_TAFOFF + 1), max. 12.8 µs
HS1_TAFON	15:8	rw	**High-side driver 1 active free-wheeling switch-on time setting** Nominal $t_{af(on)}$ = 50 ns * (HS1_TAFON + 1), max. 12.8 µs
RES	31:16	r	**Reserved** Always read as 0

High-side driver 1 active free-wheeling current control

HS1AFIC

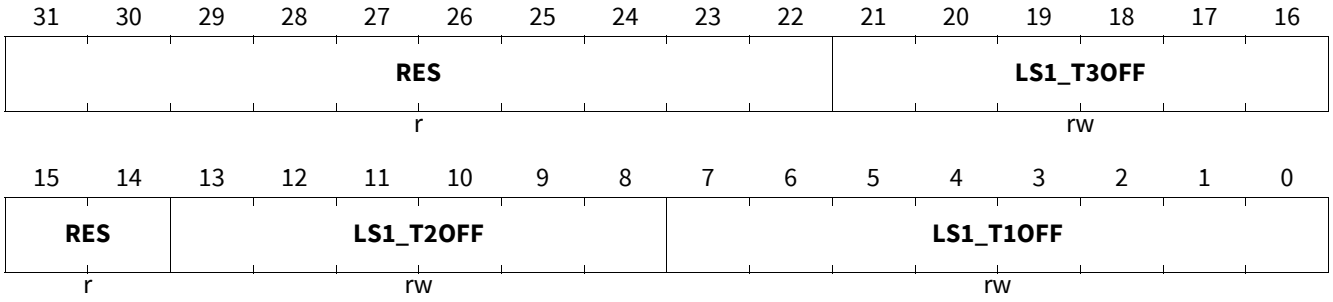
High-side driver 1 active free-wheeling current control(0054_H) **RESET_TYPE_5 Value: $0000\ 1616_H$**



Field	Bits	Type	Description
HS1_IAFOFF	5:0	rw	**High-side driver 1 active free-wheeling switch-off current setting** Nominal $I_{af(off)}$ = 5 mA ... 350 mA in 64 steps
RES	7:6, 31:14	r	**Reserved** Always read as 0
HS1_IAFON	13:8	rw	**High-side driver 1 active free-wheeling switch-on current setting** Nominal $I_{af(on)}$ = 5 mA ... 350 mA in 64 steps

Low-side driver 1 switch-off time control

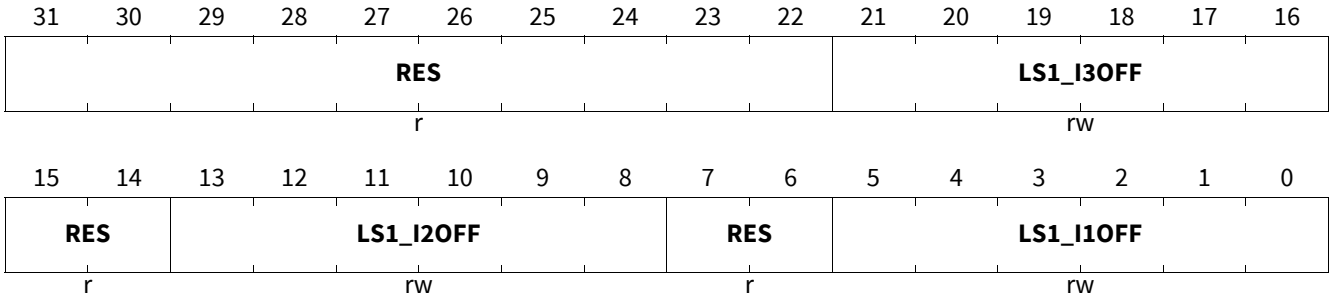
LS1SEQOFFTC
Low-side driver 1 switch-off time control (0058$_H$) RESET_TYPE_5 Value: 0005 0506$_H$



Field	Bits	Type	Description
LS1_T1OFF	7:0	rw	**Low-side driver 1 sequencer switch-off phase 1 and constant switch-off time setting** Nominal $t_{1(off)}$ = 50 ns * (LS1_T1OFF + 1), max. 12.8 µs *Note: This setting is used as an optimization target in adaptive mode.*
LS1_T2OFF	13:8	rw	**Low-side driver 1 sequencer switch-off phase 2 time setting** Nominal $t_{2(off)}$ = 50 ns * (LS1_T2OFF + 1), max. 3.2 µs
RES	15:14, 31:22	r	**Reserved** Always read as 0
LS1_T3OFF	21:16	rw	**Low-side driver 1 sequencer switch-off phase 3 time setting** Nominal $t_{3(off)}$ = 50 ns * (LS1_T3OFF + 1), max. 3.2 µs

Low-side driver 1 switch-off current control

LS1SEQOFFIC
Low-side driver 1 switch-off current control (005C$_H$) RESET_TYPE_5 Value: 0007 0720$_H$



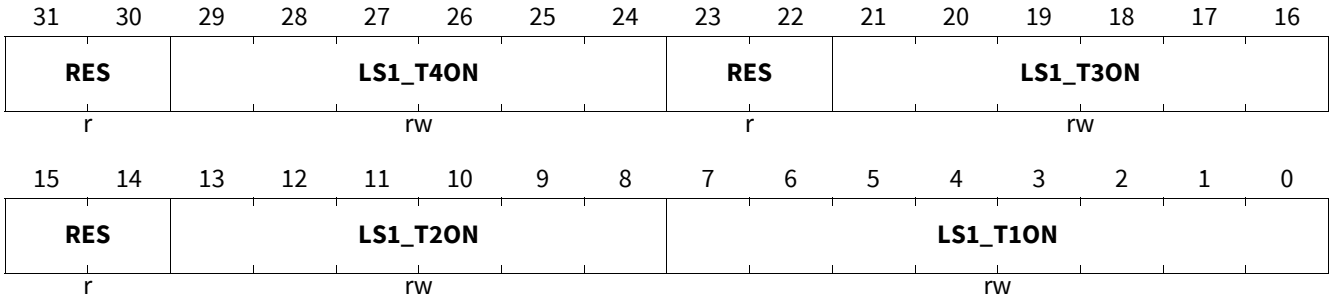
Field	Bits	Type	Description
LS1_I1OFF	5:0	rw	**Low-side driver 1 sequencer switch-off phase 1 and constant switch-off current setting** Nominal $I_{1(off)}$ = 5 mA ... 350 mA in 64 steps *Note: This setting is used as the initial value for optimization in adaptive mode.*

Field	Bits	Type	Description
RES	7:6, 15:14, 31:22	r	**Reserved** Always read as 0
LS1_I2OFF	13:8	rw	**Low-side driver 1 sequencer switch-off phase 2 current setting** Nominal $I_{2(off)}$ = 5 mA ... 350 mA in 64 steps
LS1_I3OFF	21:16	rw	**Low-side driver 1 sequencer switch-off phase 3 current setting** Nominal $I_{3(off)}$ = 5 mA ... 350 mA in 64 steps

Low-side driver 1 switch-on time control

LS1SEQONTC

Low-side driver 1 switch-on time control **(0060_H)** **RESET_TYPE_5 Value: $0405\ 0506_H$**



Field	Bits	Type	Description
LS1_T1ON	7:0	rw	**Low-side driver 1 sequencer switch-on phase 1 and constant switch-on time setting** Nominal $t_{1(on)}$ = 50 ns * (LS1_T1ON + 1), max. 12.8 µs *Note: This setting (together with LS1_T2ON) is used as an optimization target in adaptive mode.*
LS1_T2ON	13:8	rw	**Low-side driver 1 sequencer switch-on phase 2 time setting** Nominal $t_{2(on)}$ = 50 ns * (LS1_T2ON + 1), max. 3.2 µs *Note: This setting (together with LS1_T1ON) is used as an optimization target in adaptive mode.*
RES	15:14, 23:22, 31:30	r	**Reserved** Always read as 0
LS1_T3ON	21:16	rw	**Low-side driver 1 sequencer switch-on phase 3 time setting** Nominal $t_{3(on)}$ = 50 ns * (LS1_T3ON + 1), max. 3.2 µs
LS1_T4ON	29:24	rw	**Low-side driver 1 sequencer switch-on phase 4 time setting** Nominal $t_{4(on)}$ = 50 ns * (LS1_T4ON + 1), max. 3.2 µs

Low-side driver 1 switch-on current control

LS1SEQONIC
Low-side driver 1 switch-on current control (0064$_H$) RESET_TYPE_5 Value: 2007 0720$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		LS1_I4ON						RES		LS1_I3ON					
r		rw						r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		LS1_I2ON						RES		LS1_I1ON					
r		rw						r		rw					

Field	Bits	Type	Description
LS1_I1ON	5:0	rw	**Low-side driver 1 sequencer switch-on phase 1 and constant switch-on current setting** Nominal $I_{1(on)}$ = 5 mA ... 350 mA in 64 steps *Note: This setting is used as the initial value for optimization in adaptive mode.*
RES	7:6, 15:14, 23:22, 31:30	r	**Reserved** Always read as 0
LS1_I2ON	13:8	rw	**Low-side driver 1 sequencer switch-on phase 2 current setting** Nominal $I_{2(on)}$ = 5 mA ... 350 mA in 64 steps
LS1_I3ON	21:16	rw	**Low-side driver 1 sequencer switch-on phase 3 current setting** Nominal $I_{3(on)}$ = 5 mA ... 350 mA in 64 steps
LS1_I4ON	29:24	rw	**Low-side driver 1 sequencer switch-on phase 4 current setting** Nominal $I_{4(on)}$ = 5 mA ... 350 mA in 64 steps

High-side driver 1 switch-off time control

HS1SEQOFFTC
High-side driver 1 switch-off time control (0068$_H$) RESET_TYPE_5 Value: 0005 0506$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										HS1_T3OFF					
r										rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS1_T2OFF						HS1_T1OFF							
r		rw						rw							

Field	Bits	Type	Description
HS1_T1OFF	7:0	rw	**High-side driver 1 sequencer switch-off phase 1 and constant switch-off time setting** Nominal $t_{1(off)}$ = 50 ns * (HS1_T1OFF + 1), max. 12.8 µs *Note: This setting is used as an optimization target in adaptive mode.*
HS1_T2OFF	13:8	rw	**High-side driver 1 sequencer switch-off phase 2 time setting** Nominal $t_{2(off)}$ = 50 ns * (HS1_T2OFF + 1), max. 3.2 µs
RES	15:14, 31:22	r	**Reserved** Always read as 0
HS1_T3OFF	21:16	rw	**High-side driver 1 sequencer switch-off phase 3 time setting** Nominal $t_{3(off)}$ = 50 ns * (HS1_T3OFF + 1), max. 3.2 µs

High-side driver 1 switch-off current control

HS1SEQOFFIC

High-side driver 1 switch-off current control (006C$_H$) RESET_TYPE_5 Value: 0007 0720$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										HS1_I3OFF					
r										rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS1_I2OFF						RES		HS1_I1OFF					
r		rw						r		rw					

Field	Bits	Type	Description
HS1_I1OFF	5:0	rw	**High-side driver 1 sequencer switch-off phase 1 and constant switch-off current setting** Nominal $I_{1(off)}$ = 5 mA ... 350 mA in 64 steps *Note: This setting is used as the initial value for optimization in adaptive mode.*
RES	7:6, 15:14, 31:22	r	**Reserved** Always read as 0
HS1_I2OFF	13:8	rw	**High-side driver 1 sequencer switch-off phase 2 current setting** Nominal $I_{2(off)}$ = 5 mA ... 350 mA in 64 steps
HS1_I3OFF	21:16	rw	**High-side driver 1 sequencer switch-off phase 3 current setting** Nominal $I_{3(off)}$ = 5 mA ... 350 mA in 64 steps

High-side driver 1 switch-on time control

HS1SEQONTC

High-side driver 1 switch-on time control (0070$_H$) RESET_TYPE_5 Value: 0405 0506$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		HS1_T4ON						RES		HS1_T3ON					
r		rw						r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS1_T2ON						HS1_T1ON							
r		rw						rw							

Field	Bits	Type	Description
HS1_T1ON	7:0	rw	**High-side driver 1 sequencer switch-on phase 1 and constant switch-on time setting** Nominal $t_{1(on)}$ = 50 ns * (HS1_T1ON + 1), max. 12.8 µs *Note: This setting (together with HS1_T2ON) is used as an optimization target in adaptive mode.*
HS1_T2ON	13:8	rw	**High-side driver 1 sequencer switch-on phase 2 time setting** Nominal $t_{2(on)}$ = 50 ns * (HS1_T2ON + 1), max. 3.2 µs *Note: This setting (together with HS1_T1ON) is used as an optimization target in adaptive mode.*
RES	15:14, 23:22, 31:30	r	**Reserved** Always read as 0
HS1_T3ON	21:16	rw	**High-side driver 1 sequencer switch-on phase 3 time setting** Nominal $t_{3(on)}$ = 50 ns * (HS1_T3ON + 1), max. 3.2 µs
HS1_T4ON	29:24	rw	**High-side driver 1 sequencer switch-on phase 4 time setting** Nominal $t_{4(on)}$ = 50 ns * (HS1_T4ON + 1), max. 3.2 µs

High-side driver 1 switch-on current control

HS1SEQONIC

High-side driver 1 switch-on current control (0074$_H$) RESET_TYPE_5 Value: 2007 0720$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		HS1_I4ON						RES		HS1_I3ON					
r		rw						r		rw					

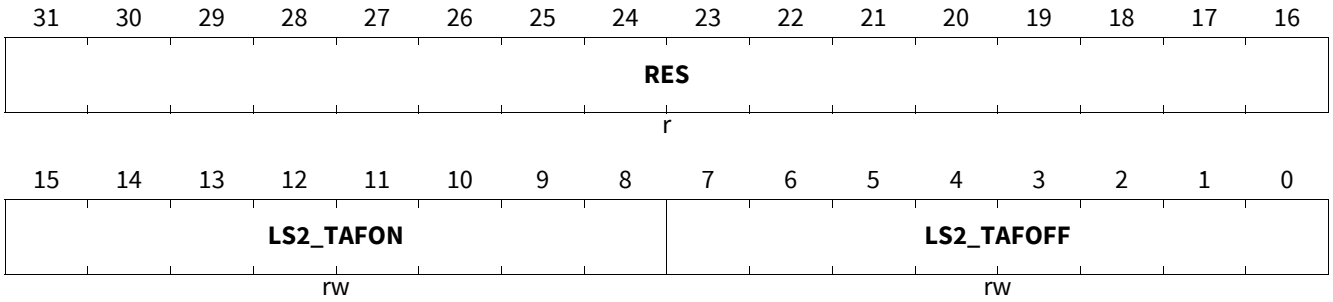
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS1_I2ON						RES		HS1_I1ON					
r		rw						r		rw					

Field	Bits	Type	Description
HS1_I1ON	5:0	rw	**High-side driver 1 sequencer switch-on phase 1 and constant switch-on current setting** Nominal $I_{1(on)}$ = 5 mA ... 350 mA in 64 steps *Note: This setting is used as the initial value for optimization in adaptive mode.*
RES	7:6, 15:14, 23:22, 31:30	r	**Reserved** Always read as 0
HS1_I2ON	13:8	rw	**High-side driver 1 sequencer switch-on phase 2 current setting** Nominal $I_{2(on)}$ = 5 mA ... 350 mA in 64 steps
HS1_I3ON	21:16	rw	**High-side driver 1 sequencer switch-on phase 3 current setting** Nominal $I_{3(on)}$ = 5 mA ... 350 mA in 64 steps
HS1_I4ON	29:24	rw	**High-side driver 1 sequencer switch-on phase 4 current setting** Nominal $I_{4(on)}$ = 5 mA ... 350 mA in 64 steps

Low-side driver 2 active free-wheeling time control

LS2AFTC

Low-side driver 2 active free-wheeling time control(0078_H) **RESET_TYPE_5 Value: $0000\ 1414_H$**



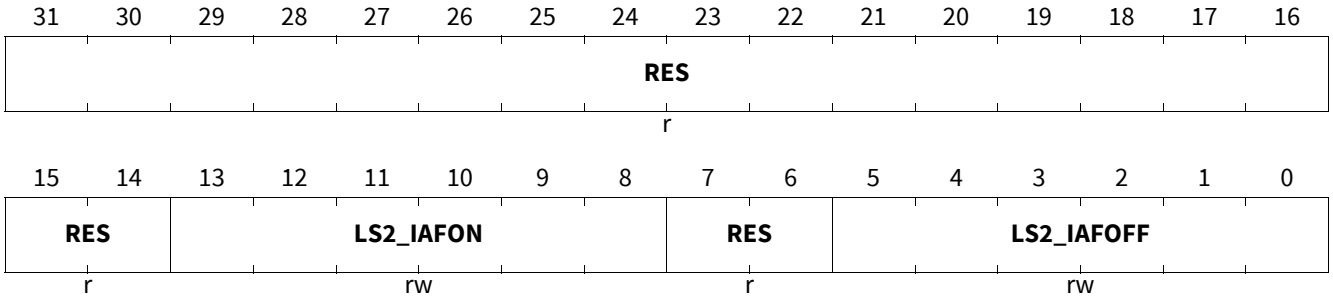
Field	Bits	Type	Description
LS2_TAFOFF	7:0	rw	**Low-side driver 2 active free-wheeling switch-off time setting** Nominal $t_{af(off)}$ = 50 ns * (LS2_TAFOFF + 1), max. 12.8 µs
LS2_TAFON	15:8	rw	**Low-side driver 2 active free-wheeling switch-on time setting** Nominal $t_{af(on)}$ = 50 ns * (LS2_TAFON + 1), max. 12.8 µs
RES	31:16	r	**Reserved** Always read as 0

Register description BDRV

Low-side driver 2 active free-wheeling current control

LS2AFIC

Low-side driver 2 active free-wheeling current control($007C_H$) **RESET_TYPE_5 Value: $0000\ 1616_H$**

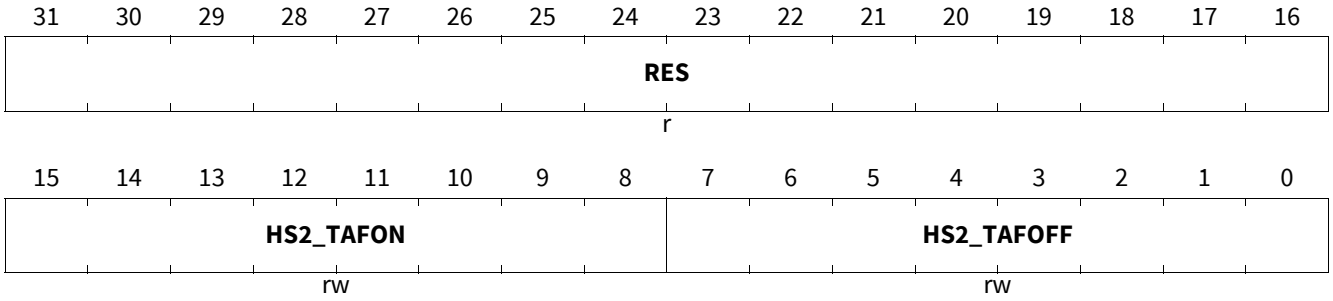


Field	Bits	Type	Description
LS2_IAFOFF	5:0	rw	**Low-side driver 2 active free-wheeling switch-off current setting** Nominal $I_{af(off)}$ = 5 mA ... 350 mA in 64 steps
RES	7:6, 31:14	r	**Reserved** Always read as 0
LS2_IAFON	13:8	rw	**Low-side driver 2 active free-wheeling switch-on current setting** Nominal $I_{af(on)}$ = 5 mA ... 350 mA in 64 steps

High-side driver 2 active free-wheeling time control

HS2AFTC

High-side driver 2 active free-wheeling time control(0080_H) **RESET_TYPE_5 Value: $0000\ 1414_H$**



Field	Bits	Type	Description
HS2_TAFOFF	7:0	rw	**High-side driver 2 active free-wheeling switch-off time setting** Nominal $t_{af(off)}$ = 50 ns * (HS2_TAFOFF + 1), max. 12.8 µs
HS2_TAFON	15:8	rw	**High-side driver 2 active free-wheeling switch-on time setting** Nominal $t_{af(on)}$ = 50 ns * (HS2_TAFON + 1), max. 12.8 µs
RES	31:16	r	**Reserved** Always read as 0

Register description BDRV

High-side driver 2 active free-wheeling current control

HS2AFIC

High-side driver 2 active free-wheeling current control(0084_H) **RESET_TYPE_5 Value: $0000\ 1616_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS2_IAFON						RES		HS2_IAFOFF					
r		rw						r		rw					

Field	Bits	Type	Description
HS2_IAFOFF	5:0	rw	**High-side driver 2 active free-wheeling switch-off current setting** Nominal $I_{af(off)}$ = 5 mA ... 350 mA in 64 steps
RES	7:6, 31:14	r	**Reserved** Always read as 0
HS2_IAFON	13:8	rw	**High-side driver 2 active free-wheeling switch-on current setting** Nominal $I_{af(on)}$ = 5 mA ... 350 mA in 64 steps

Low-side driver 2 switch-off time control

LS2SEQOFFTC

Low-side driver 2 switch-off time control (0088_H) **RESET_TYPE_5 Value: $0005\ 0506_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										LS2_T3OFF					
r										rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		LS2_T2OFF						LS2_T1OFF							
r		rw						rw							

Field	Bits	Type	Description
LS2_T1OFF	7:0	rw	**Low-side driver 2 sequencer switch-off phase 1 and constant switch-off time setting** Nominal $t_{1(off)}$ = 50 ns * (LS2_T1OFF + 1), max. 12.8 µs *Note: This setting is used as an optimization target in adaptive mode.*
LS2_T2OFF	13:8	rw	**Low-side driver 2 sequencer switch-off phase 2 time setting** Nominal $t_{2(off)}$ = 50 ns * (LS2_T2OFF + 1), max. 3.2 µs
RES	15:14, 31:22	r	**Reserved** Always read as 0
LS2_T3OFF	21:16	rw	**Low-side driver 2 sequencer switch-off phase 3 time setting** Nominal $t_{3(off)}$ = 50 ns * (LS2_T3OFF + 1), max. 3.2 µs

Low-side driver 2 switch-off current control

LS2SEQOFFIC
Low-side driver 2 switch-off current control (008C$_H$) RESET_TYPE_5 Value: 0007 0720$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										LS2_I3OFF					
r										rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		LS2_I2OFF						RES		LS2_I1OFF					
r		rw						r		rw					

Field	Bits	Type	Description
LS2_I1OFF	5:0	rw	**Low-side driver 2 sequencer switch-off phase 1 and constant switch-off current setting** Nominal $I_{1(off)}$ = 5 mA ... 350 mA in 64 steps *Note: This setting is used as the initial value for optimization in adaptive mode.*
RES	7:6, 15:14, 31:22	r	**Reserved** Always read as 0
LS2_I2OFF	13:8	rw	**Low-side driver 2 sequencer switch-off phase 2 current setting** Nominal $I_{2(off)}$ = 5 mA ... 350 mA in 64 steps
LS2_I3OFF	21:16	rw	**Low-side driver 2 sequencer switch-off phase 3 current setting** Nominal $I_{3(off)}$ = 5 mA ... 350 mA in 64 steps

Low-side driver 2 switch-on time control

LS2SEQONTC
Low-side driver 2 switch-on time control (0090$_H$) RESET_TYPE_5 Value: 0405 0506$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		LS2_T4ON						RES		LS2_T3ON					
r		rw						r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		LS2_T2ON						LS2_T1ON							
r		rw						rw							

Field	Bits	Type	Description
LS2_T1ON	7:0	rw	**Low-side driver 2 sequencer switch-on phase 1 and constant switch-on time setting** Nominal $t_{1(on)}$ = 50 ns * (LS2_T1ON + 1), max. 12.8 µs *Note: This setting (together with LS2_T2ON) is used as an optimization target in adaptive mode.*

Field	Bits	Type	Description
LS2_T2ON	13:8	rw	**Low-side driver 2 sequencer switch-on phase 2 time setting** Nominal $t_{2(on)}$ = 50 ns * (LS2_T2ON + 1), max. 3.2 µs *Note: This setting (together with LS2_T1ON) is used as an optimization target in adaptive mode.*
RES	15:14, 23:22, 31:30	r	**Reserved** Always read as 0
LS2_T3ON	21:16	rw	**Low-side driver 2 sequencer switch-on phase 3 time setting** Nominal $t_{3(on)}$ = 50 ns * (LS2_T3ON + 1), max. 3.2 µs
LS2_T4ON	29:24	rw	**Low-side driver 2 sequencer switch-on phase 4 time setting** Nominal $t_{4(on)}$ = 50 ns * (LS2_T4ON + 1), max. 3.2 µs

Low-side driver 2 switch-on current control

LS2SEQONIC
Low-side driver 2 switch-on current control (0094$_H$) RESET_TYPE_5 Value: 2007 0720$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		LS2_I4ON						RES		LS2_I3ON					
r		rw						r		rw					

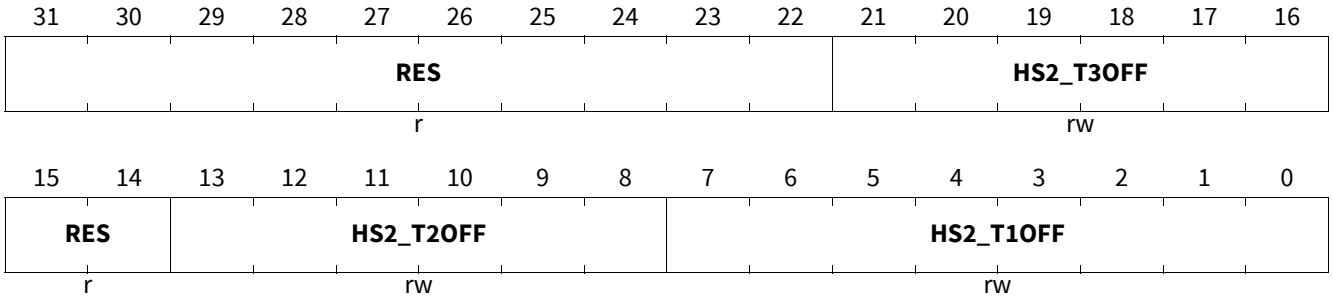
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		LS2_I2ON						RES		LS2_I1ON					
r		rw						r		rw					

Field	Bits	Type	Description
LS2_I1ON	5:0	rw	**Low-side driver 2 sequencer switch-on phase 1 and constant switch-on current setting** Nominal $I_{1(on)}$ = 5 mA ... 350 mA in 64 steps *Note: This setting is used as the initial value for optimization in adaptive mode.*
RES	7:6, 15:14, 23:22, 31:30	r	**Reserved** Always read as 0
LS2_I2ON	13:8	rw	**Low-side driver 2 sequencer switch-on phase 2 current setting** Nominal $I_{2(on)}$ = 5 mA ... 350 mA in 64 steps
LS2_I3ON	21:16	rw	**Low-side driver 2 sequencer switch-on phase 3 current setting** Nominal $I_{3(on)}$ = 5 mA ... 350 mA in 64 steps
LS2_I4ON	29:24	rw	**Low-side driver 2 sequencer switch-on phase 4 current setting** Nominal $I_{4(on)}$ = 5 mA ... 350 mA in 64 steps

High-side driver 2 switch-off time control

HS2SEQOFFTC

High-side driver 2 switch-off time control (0098$_H$) RESET_TYPE_5 Value: 0005 0506$_H$

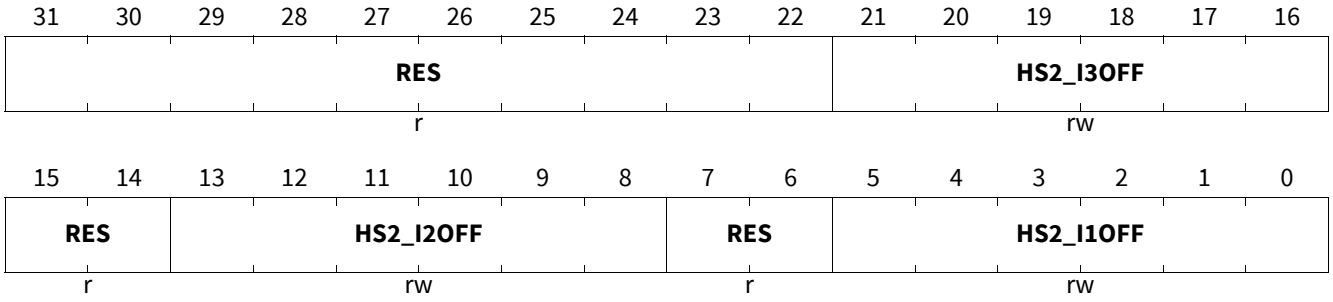


Field	Bits	Type	Description
HS2_T1OFF	7:0	rw	**High-side driver 2 sequencer switch-off phase 1 and constant switch-off time setting** Nominal $t_{1(off)}$ = 50 ns * (HS2_T1OFF + 1), max. 12.8 µs *Note: This setting is used as an optimization target in adaptive mode.*
HS2_T2OFF	13:8	rw	**High-side driver 2 sequencer switch-off phase 2 time setting** Nominal $t_{2(off)}$ = 50 ns * (HS2_T2OFF + 1), max. 3.2 µs
RES	15:14, 31:22	r	**Reserved** Always read as 0
HS2_T3OFF	21:16	rw	**High-side driver 2 sequencer switch-off phase 3 time setting** Nominal $t_{3(off)}$ = 50 ns * (HS2_T3OFF + 1), max. 3.2 µs

High-side driver 2 switch-off current control

HS2SEQOFFIC

High-side driver 2 switch-off current control (009C$_H$) RESET_TYPE_5 Value: 0007 0720$_H$



Field	Bits	Type	Description
HS2_I1OFF	5:0	rw	**High-side driver 2 sequencer switch-off phase 1 and constant switch-off current setting** Nominal $I_{1(off)}$ = 5 mA ... 350 mA in 64 steps *Note: This setting is used as the initial value for optimization in adaptive mode.*

Field	Bits	Type	Description
RES	7:6, 15:14, 31:22	r	**Reserved** Always read as 0
HS2_I2OFF	13:8	rw	**High-side driver 2 sequencer switch-off phase 2 current setting** Nominal $I_{2(off)}$ = 5 mA ... 350 mA in 64 steps
HS2_I3OFF	21:16	rw	**High-side driver 2 sequencer switch-off phase 3 current setting** Nominal $I_{3(off)}$ = 5 mA ... 350 mA in 64 steps

High-side driver 2 switch-on time control

HS2SEQONTC

High-side driver 2 switch-on time control **($00A0_H$)** **RESET_TYPE_5 Value: $0405\ 0506_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		**HS2_T4ON**						**RES**		**HS2_T3ON**					
r		rw						r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		**HS2_T2ON**						**HS2_T1ON**							
r		rw						rw							

Field	Bits	Type	Description
HS2_T1ON	7:0	rw	**High-side driver 2 sequencer switch-on phase 1 and constant switch-on time setting** Nominal $t_{1(on)}$ = 50 ns * (HS2_T1ON + 1), max. 12.8 µs *Note: This setting (together with HS2_T2ON) is used as an optimization target in adaptive mode.*
HS2_T2ON	13:8	rw	**High-side driver 2 sequencer switch-on phase 2 time setting** Nominal $t_{2(on)}$ = 50 ns * (HS2_T2ON + 1), max. 3.2 µs *Note: This setting (together with HS2_T1ON) is used as an optimization target in adaptive mode.*
RES	15:14, 23:22, 31:30	r	**Reserved** Always read as 0
HS2_T3ON	21:16	rw	**High-side driver 2 sequencer switch-on phase 3 time setting** Nominal $t_{3(on)}$ = 50 ns * (HS2_T3ON + 1), max. 3.2 µs
HS2_T4ON	29:24	rw	**High-side driver 2 sequencer switch-on phase 4 time setting** Nominal $t_{4(on)}$ = 50 ns * (HS2_T4ON + 1), max. 3.2 µs

Register description BDRV

High-side driver 2 switch-on current control

HS2SEQONIC
High-side driver 2 switch-on current control (00A4$_H$) RESET_TYPE_5 Value: 2007 0720$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		HS2_I4ON						RES		HS2_I3ON					
r		rw						r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS2_I2ON						RES		HS2_I1ON					
r		rw						r		rw					

Field	Bits	Type	Description
HS2_I1ON	5:0	rw	**High-side driver 2 sequencer switch-on phase 1 and constant switch-on current setting** Nominal $I_{1(on)}$ = 5 mA ... 350 mA in 64 steps *Note: This setting is used as the initial value for optimization in adaptive mode.*
RES	7:6, 15:14, 23:22, 31:30	r	**Reserved** Always read as 0
HS2_I2ON	13:8	rw	**High-side driver 2 sequencer switch-on phase 2 current setting** Nominal $I_{2(on)}$ = 5 mA ... 350 mA in 64 steps
HS2_I3ON	21:16	rw	**High-side driver 2 sequencer switch-on phase 3 current setting** Nominal $I_{3(on)}$ = 5 mA ... 350 mA in 64 steps
HS2_I4ON	29:24	rw	**High-side driver 2 sequencer switch-on phase 4 current setting** Nominal $I_{4(on)}$ = 5 mA ... 350 mA in 64 steps

Low-side driver 3 active free-wheeling time control

LS3AFTC
Low-side driver 3 active free-wheeling time control(00A8$_H$) RESET_TYPE_5 Value: 0000 1414$_H$

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LS3_TAFON								LS3_TAFOFF							
rw								rw							

Field	Bits	Type	Description
LS3_TAFOFF	7:0	rw	**Low-side driver 3 active free-wheeling switch-off time setting** Nominal $t_{af(off)}$ = 50 ns * (LS3_TAFOFF + 1), max. 12.8 µs

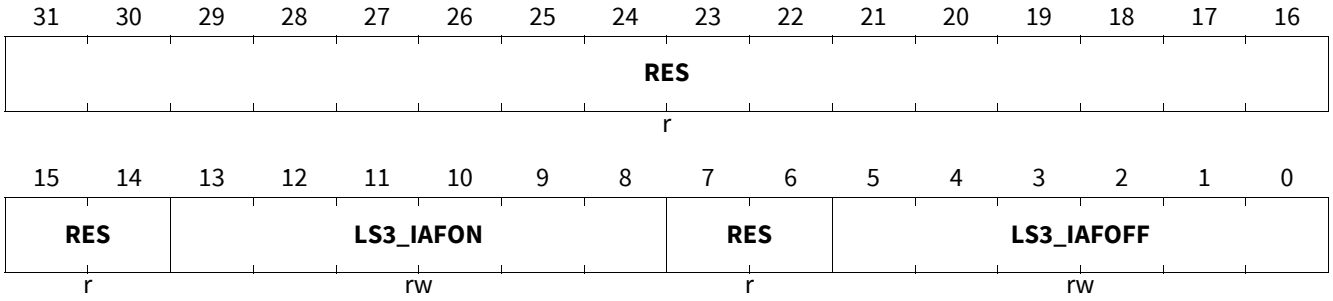
Register description BDRV

Field	Bits	Type	Description
LS3_TAFON	15:8	rw	**Low-side driver 3 active free-wheeling switch-on time setting** Nominal $t_{af(on)}$ = 50 ns * (LS3_TAFON + 1), max. 12.8 µs
RES	31:16	r	**Reserved** Always read as 0

Low-side driver 3 active free-wheeling current control

LS3AFIC

Low-side driver 3 active free-wheeling current control(00AC$_H$) **RESET_TYPE_5 Value: 0000 1616$_H$**

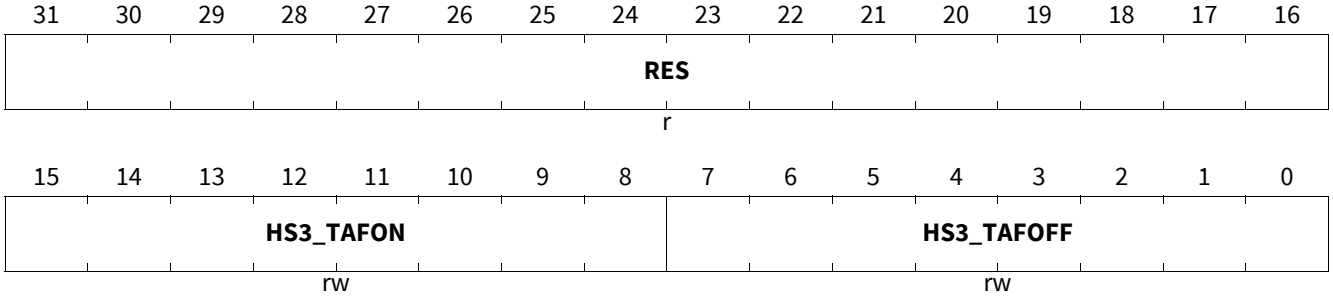


Field	Bits	Type	Description
LS3_IAFOFF	5:0	rw	**Low-side driver 3 active free-wheeling switch-off current setting** Nominal $I_{af(off)}$ = 5 mA ... 350 mA in 64 steps
RES	7:6, 31:14	r	**Reserved** Always read as 0
LS3_IAFON	13:8	rw	**Low-side driver 3 active free-wheeling switch-on current setting** Nominal $I_{af(on)}$ = 5 mA ... 350 mA in 64 steps

High-side driver 3 active free-wheeling time control

HS3AFTC

High-side driver 3 active free-wheeling time control(00B0$_H$) **RESET_TYPE_5 Value: 0000 1414$_H$**



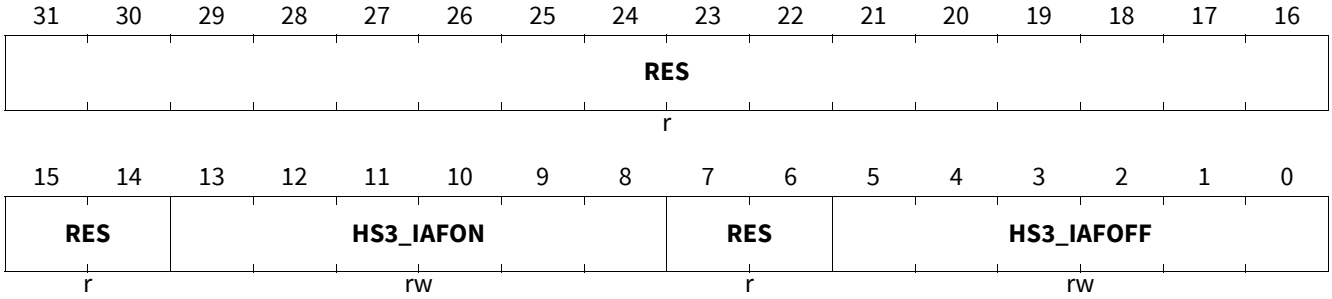
Field	Bits	Type	Description
HS3_TAFOFF	7:0	rw	**High-side driver 3 active free-wheeling switch-off time setting** Nominal $t_{af(off)}$ = 50 ns * (HS3_TAFOFF + 1), max. 12.8 µs
HS3_TAFON	15:8	rw	**High-side driver 3 active free-wheeling switch-on time setting** Nominal $t_{af(on)}$ = 50 ns * (HS3_TAFON + 1), max. 12.8 µs

Field	Bits	Type	Description
RES	31:16	r	**Reserved** Always read as 0

High-side driver 3 active free-wheeling current control

HS3AFIC

High-side driver 3 active free-wheeling current control($00B4_H$) **RESET_TYPE_5 Value: $0000\ 1616_H$**

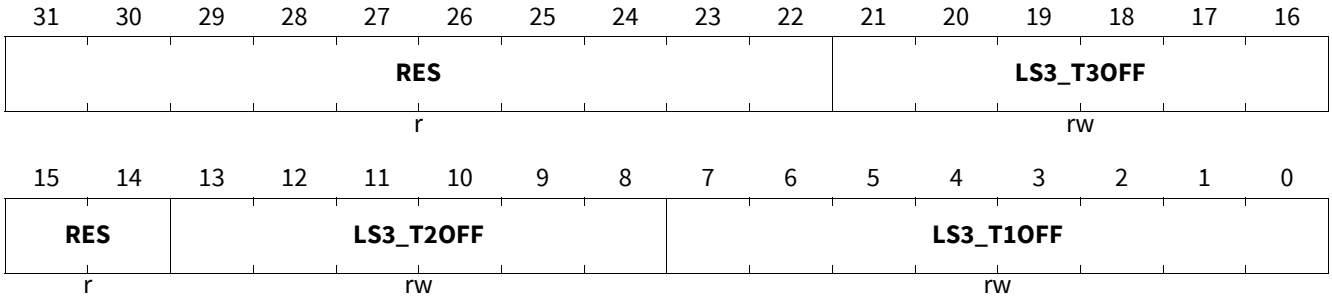


Field	Bits	Type	Description
HS3_IAFOFF	5:0	rw	**High-side driver 3 active free-wheeling switch-off current setting** Nominal $I_{af(off)}$ = 5 mA ... 350 mA in 64 steps
RES	7:6, 31:14	r	**Reserved** Always read as 0
HS3_IAFON	13:8	rw	**High-side driver 3 active free-wheeling switch-on current setting** Nominal $I_{af(on)}$ = 5 mA ... 350 mA in 64 steps

Low-side driver 3 switch-off time control

LS3SEQOFFTC

Low-side driver 3 switch-off time control ($00B8_H$) **RESET_TYPE_5 Value: $0005\ 0506_H$**

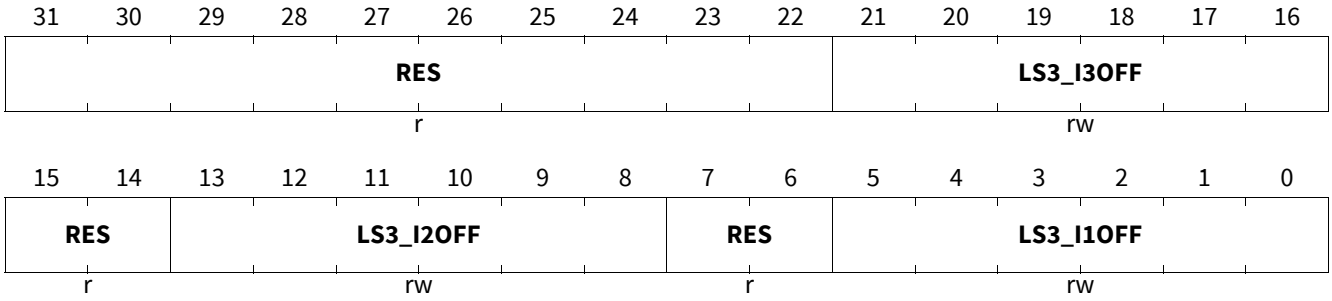


Field	Bits	Type	Description
LS3_T1OFF	7:0	rw	**Low-side driver 3 sequencer switch-off phase 1 and constant switch-off time setting** Nominal $t_{1(off)}$ = 50 ns * (LS3_T1OFF + 1), max. 12.8 µs *Note: This setting is used as an optimization target in adaptive mode.*
LS3_T2OFF	13:8	rw	**Low-side driver 3 sequencer switch-off phase 2 time setting** Nominal $t_{2(off)}$ = 50 ns * (LS3_T2OFF + 1), max. 3.2 µs

Field	Bits	Type	Description
RES	15:14, 31:22	r	**Reserved** Always read as 0
LS3_T3OFF	21:16	rw	**Low-side driver 3 sequencer switch-off phase 3 time setting** Nominal $t_{3(off)}$ = 50 ns * (LS3_T3OFF + 1), max. 3.2 µs

Low-side driver 3 switch-off current control

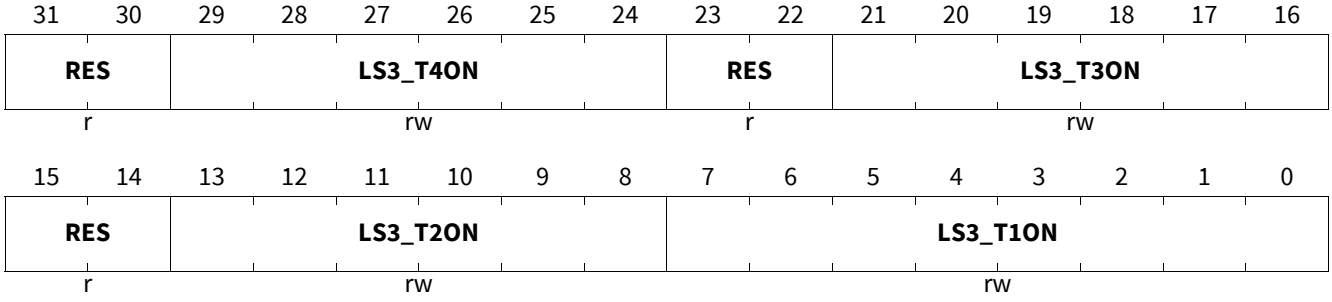
LS3SEQOFFIC
Low-side driver 3 switch-off current control (00BC$_H$) RESET_TYPE_5 Value: 0007 0720$_H$



Field	Bits	Type	Description
LS3_I1OFF	5:0	rw	**Low-side driver 3 sequencer switch-off phase 1 and constant switch-off current setting** Nominal $I_{1(off)}$ = 5 mA ... 350 mA in 64 steps *Note: This setting is used as the initial value for optimization in adaptive mode.*
RES	7:6, 15:14, 31:22	r	**Reserved** Always read as 0
LS3_I2OFF	13:8	rw	**Low-side driver 3 sequencer switch-off phase 2 current setting** Nominal $I_{2(off)}$ = 5 mA ... 350 mA in 64 steps
LS3_I3OFF	21:16	rw	**Low-side driver 3 sequencer switch-off phase 3 current setting** Nominal $I_{3(off)}$ = 5 mA ... 350 mA in 64 steps

Low-side driver 3 switch-on time control

LS3SEQONTC
Low-side driver 3 switch-on time control (00C0$_H$) RESET_TYPE_5 Value: 0405 0506$_H$



Field	Bits	Type	Description
LS3_T1ON	7:0	rw	**Low-side driver 3 sequencer switch-on phase 1 and constant switch-on time setting** Nominal $t_{1(on)}$ = 50 ns * (LS3_T1ON + 1), max. 12.8 µs *Note: This setting (together with LS3_T2ON) is used as an optimization target in adaptive mode.*
LS3_T2ON	13:8	rw	**Low-side driver 3 sequencer switch-on phase 2 time setting** Nominal $t_{2(on)}$ = 50 ns * (LS3_T2ON + 1), max. 3.2 µs *Note: This setting (together with LS3_T1ON) is used as an optimization target in adaptive mode.*
RES	15:14, 23:22, 31:30	r	**Reserved** Always read as 0
LS3_T3ON	21:16	rw	**Low-side driver 3 sequencer switch-on phase 3 time setting** Nominal $t_{3(on)}$ = 50 ns * (LS3_T3ON + 1), max. 3.2 µs
LS3_T4ON	29:24	rw	**Low-side driver 3 sequencer switch-on phase 4 time setting** Nominal $t_{4(on)}$ = 50 ns * (LS3_T4ON + 1), max. 3.2 µs

Low-side driver 3 switch-on current control

LS3SEQONIC
Low-side driver 3 switch-on current control (00C4$_H$) RESET_TYPE_5 Value: 2007 0720$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		LS3_I4ON						RES		LS3_I3ON					
r		rw						r		rw					

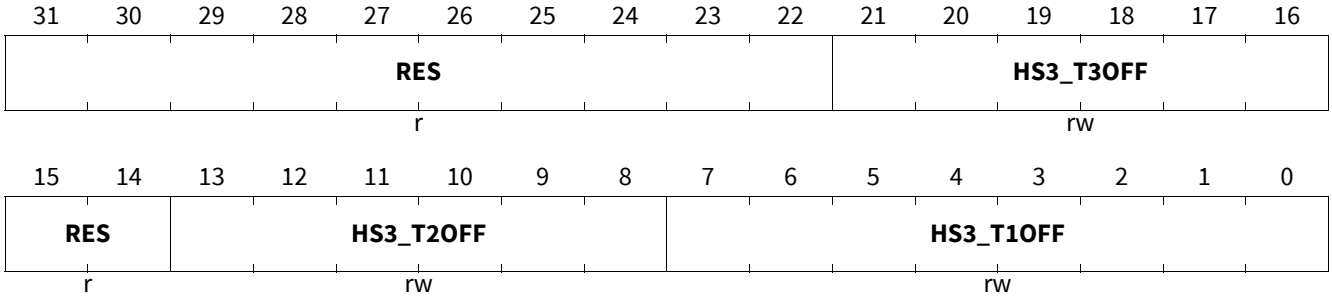
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		LS3_I2ON						RES		LS3_I1ON					
r		rw						r		rw					

Field	Bits	Type	Description
LS3_I1ON	5:0	rw	**Low-side driver 3 sequencer switch-on phase 1 and constant switch-on current setting** Nominal $I_{1(on)}$ = 5 mA ... 350 mA in 64 steps *Note: This setting is used as the initial value for optimization in adaptive mode.*
RES	7:6, 15:14, 23:22, 31:30	r	**Reserved** Always read as 0
LS3_I2ON	13:8	rw	**Low-side driver 3 sequencer switch-on phase 2 current setting** Nominal $I_{2(on)}$ = 5 mA ... 350 mA in 64 steps
LS3_I3ON	21:16	rw	**Low-side driver 3 sequencer switch-on phase 3 current setting** Nominal $I_{3(on)}$ = 5 mA ... 350 mA in 64 steps

Field	Bits	Type	Description
LS3_I4ON	29:24	rw	**Low-side driver 3 sequencer switch-on phase 4 current setting** Nominal $I_{4(on)}$ = 5 mA ... 350 mA in 64 steps

High-side driver 3 switch-off time control

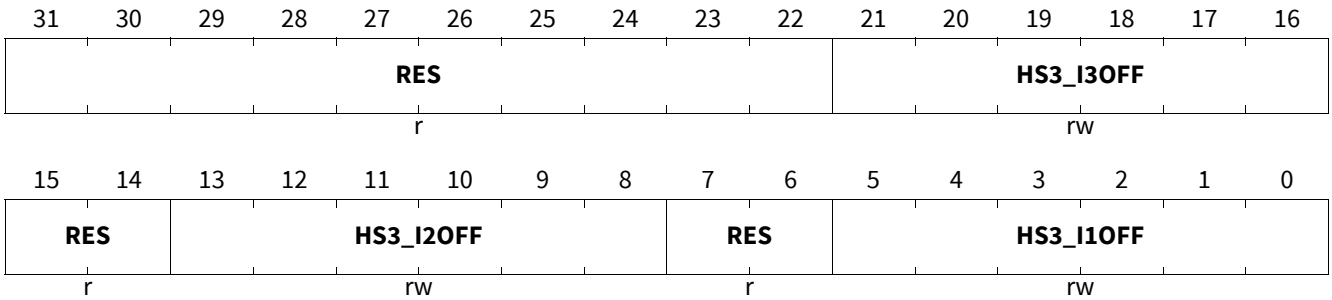
HS3SEQOFFTC
High-side driver 3 switch-off time control (00C8$_H$) RESET_TYPE_5 Value: 0005 0506$_H$



Field	Bits	Type	Description
HS3_T1OFF	7:0	rw	**High-side driver 3 sequencer switch-off phase 1 and constant switch-off time setting** Nominal $t_{1(off)}$ = 50 ns * (HS3_T1OFF + 1), max. 12.8 µs *Note: This setting is used as an optimization target in adaptive mode.*
HS3_T2OFF	13:8	rw	**High-side driver 3 sequencer switch-off phase 2 time setting** Nominal $t_{2(off)}$ = 50 ns * (HS3_T2OFF + 1), max. 3.2 µs
RES	15:14, 31:22	r	**Reserved** Always read as 0
HS3_T3OFF	21:16	rw	**High-side driver 3 sequencer switch-off phase 3 time setting** Nominal $t_{3(off)}$ = 50 ns * (HS3_T3OFF + 1), max. 3.2 µs

High-side driver 3 switch-off current control

HS3SEQOFFIC
High-side driver 3 switch-off current control (00CC$_H$) RESET_TYPE_5 Value: 0007 0720$_H$



Field	Bits	Type	Description
HS3_I1OFF	5:0	rw	**High-side driver 3 sequencer switch-off phase 1 and constant switch-off current setting** Nominal $I_{1(off)}$ = 5 mA ... 350 mA in 64 steps *Note: This setting is used as the initial value for optimization in adaptive mode.*
RES	7:6, 15:14, 31:22	r	**Reserved** Always read as 0
HS3_I2OFF	13:8	rw	**High-side driver 3 sequencer switch-off phase 2 current setting** Nominal $I_{2(off)}$ = 5 mA ... 350 mA in 64 steps
HS3_I3OFF	21:16	rw	**High-side driver 3 sequencer switch-off phase 3 current setting** Nominal $I_{3(off)}$ = 5 mA ... 350 mA in 64 steps

High-side driver 3 switch-on time control

HS3SEQONTC

High-side driver 3 switch-on time control (00D0$_H$) RESET_TYPE_5 Value: 0405 0506$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		**HS3_T4ON**						**RES**		**HS3_T3ON**					
r		rw						r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		**HS3_T2ON**						**HS3_T1ON**							
r		rw						rw							

Field	Bits	Type	Description
HS3_T1ON	7:0	rw	**High-side driver 3 sequencer switch-on phase 1 and constant switch-on time setting** Nominal $t_{1(on)}$ = 50 ns * (HS3_T1ON + 1), max. 12.8 µs *Note: This setting (together with HS3_T2ON) is used as an optimization target in adaptive mode.*
HS3_T2ON	13:8	rw	**High-side driver 3 sequencer switch-on phase 2 time setting** Nominal $t_{2(on)}$ = 50 ns * (HS3_T2ON + 1), max. 3.2 µs *Note: This setting (together with HS3_T1ON) is used as an optimization target in adaptive mode.*
RES	15:14, 23:22, 31:30	r	**Reserved** Always read as 0
HS3_T3ON	21:16	rw	**High-side driver 3 sequencer switch-on phase 3 time setting** Nominal $t_{3(on)}$ = 50 ns * (HS3_T3ON + 1), max. 3.2 µs
HS3_T4ON	29:24	rw	**High-side driver 3 sequencer switch-on phase 4 time setting** Nominal $t_{4(on)}$ = 50 ns * (HS3_T4ON + 1), max. 3.2 µs

Register description BDRV

High-side driver 3 switch-on current control

HS3SEQONIC
High-side driver 3 switch-on current control (00D4$_H$) RESET_TYPE_5 Value: 2007 0720$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		HS3_I4ON						RES		HS3_I3ON					
r		rw						r		rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HS3_I2ON						RES		HS3_I1ON					
r		rw						r		rw					

Field	Bits	Type	Description
HS3_I1ON	5:0	rw	**High-side driver 3 sequencer switch-on phase 1 and constant switch-on current setting** Nominal $I_{1(on)}$ = 5 mA ... 350 mA in 64 steps *Note: This setting is used as the initial value for optimization in adaptive mode.*
RES	7:6, 15:14, 23:22, 31:30	r	**Reserved** Always read as 0
HS3_I2ON	13:8	rw	**High-side driver 3 sequencer switch-on phase 2 current setting** Nominal $I_{2(on)}$ = 5 mA ... 350 mA in 64 steps
HS3_I3ON	21:16	rw	**High-side driver 3 sequencer switch-on phase 3 current setting** Nominal $I_{3(on)}$ = 5 mA ... 350 mA in 64 steps
HS3_I4ON	29:24	rw	**High-side driver 3 sequencer switch-on phase 4 current setting** Nominal $I_{4(on)}$ = 5 mA ... 350 mA in 64 steps

Sequencer switch-off phase 4 time and current control

SEQOFFT4I4
Sequencer switch-off phase 4 time and current control(00D8$_H$) RESET_TYPE_5 Value: 0020 0005$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										I4OFF					
r										rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										T4OFF					
r										rw					

Field	Bits	Type	Description
T4OFF	5:0	rw	**Sequencer switch-off phase 4 time setting for all drivers** Nominal $t_{4(off)}$ = 50 ns * (T4OFF + 1), max. 3.2 µs

Register description BDRV

Field	Bits	Type	Description
RES	15:6, 31:22	r	**Reserved** Always read as 0
I4OFF	21:16	rw	**Sequencer switch-off phase 4 current setting for all drivers** Nominal $I_{4(off)}$ = 5 mA ... 350 mA in 64 steps

High-current discharge mode control

HCDIS

High-current discharge mode control (00DC$_H$) RESET_TYPE_5 Value: 0001 003F$_H$

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

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

Field	Bits	Type	Description
IHCDIS	5:0	rw	**High-current discharge mode current setting for all drivers** Nominal I_{hcdis} = 5 mA ... 350 mA in 64 steps
RES	15:6, 31:17	r	**Reserved** Always read as 0
HCDIS_SSO	16	rw	**High-current discharge mode with safe switch-off** 0_B **DISABLE**, No high-current discharge mode in case of safe switch-off 1_B **ENABLE**, In case of safe switch-off all gates are discharged with high-current discharge mode

Half bridge 1 switch-on measurement values

HB1ONVAL

Half bridge 1 switch-on measurement values (00E0$_H$) RESET_TYPE_5 Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HB1_ONVALVF_SET	HB1_ONVALVF_CLR	HB1_ONVALVF	HB1_ACTDRV_ON	RES					HB1_TONDURMERR	HB1_TONDUR					
w	w	rhxr	r	r					r	r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HB1_I1ONVAL						HB1_TONDLY							
r		r						r							

Field	Bits	Type	Description
HB1_TONDLY	7:0	r	**Half bridge 1 switch-on delay time value measured until VSH voltage starts to change** Nominal $t_{sdly(on)}$ = 50 ns * (HB1_TONDLY + 1), max. 12.8 µs
HB1_I1ONVAL	13:8	r	**Half bridge 1 switch-on phase 1 current setting from adaptive sequencer** This is the adapted value of I1ON after having applied the T12ON optimization, nominal $I_{1(on)}$ = 5 mA ... 350 mA in 64 steps.
RES	15:14, 27:23	r	**Reserved** Always read as 0
HB1_TONDUR	21:16	r	**Half bridge 1 switch-on voltage slope duration time value** Nominal $t_{sdur(on)}$ = 50 ns * (HB1_TONDUR + 1), max. 3.2 µs
HB1_TONDURMERR	22	r	**Half bridge 1 switch-on voltage slope time measurement error** 0_B **VALID**, Slope duration measurement valid 1_B **ERROR**, Slope duration measurement error
HB1_ACTDRV_ON	28	r	**Half bridge 1 switch-on active driver** 0_B **LS**, Measurement values from low-side driver 1_B **HS**, Measurement values from high-side driver
HB1_ONVALVF	29	rhxr	**Half bridge 1 switch-on measurement values valid flag** *Note: Clear flag to update measurement values.* 0_B **NOTVALID**, Measurement values not valid 1_B **VALID**, Registers contain valid data
HB1_ONVALVF_CLR	30	w	**Half bridge 1 switch-on measurement values valid flag clear** 0_B **KEEP**, Keep measurement values 1_B **CLEAR**, Query new measurement values
HB1_ONVALVF_SET	31	w	**Half bridge 1 switch-on measurement values valid flag set** 0_B **KEEP**, Keep measurement status 1_B **SET**, Set new measurement status

Half bridge 1 switch-off measurement values

HB1OFFVAL
Half bridge 1 switch-off measurement values (00E4$_H$) **RESET_TYPE_5 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HB1_OFFVALVF_SET	HB1_OFFVALVF_CLR	HB1_OFFVALVF	HB1_ACTDRV_OFF	RES					HB1_TOFFDURMERR	HB1_TOFFDUR					
w	w	rhxr	r	r					r	r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HB1_I1OFFVAL						HB1_TOFFDLY							
r		r						r							

Field	Bits	Type	Description
HB1_TOFFDLY	7:0	r	**Half bridge 1 switch-off delay time value measured until VSH voltage starts to change** Nominal $t_{sdly(off)}$ = 50 ns * (HB1_TOFFDLY + 1), max. 12.8 µs
HB1_I1OFFVAL	13:8	r	**Half bridge 1 switch-off phase 1 current setting from adaptive sequencer** This is the adapted value of I1OFF after having applied the T1OFF optimization, nominal $I_{1(off)}$ = 5 mA ... 350 mA in 64 steps.
RES	15:14, 27:23	r	**Reserved** Always read as 0
HB1_TOFFDUR	21:16	r	**Half bridge 1 switch-off voltage slope duration time value** Nominal $t_{sdur(off)}$ = 50 ns * (HB1_TOFFDUR + 1), max. 3.2 µs
HB1_TOFFDURMERR	22	r	**Half bridge 1 switch-off voltage slope time measurement error** 0_B **VALID**, Slope duration measurement valid 1_B **ERROR**, Slope duration measurement error
HB1_ACTDRV_OFF	28	r	**Half bridge 1 switch-off active driver** 0_B **LS**, Measurement values from low-side driver 1_B **HS**, Measurement values from high-side driver
HB1_OFFVALVF	29	rhxr	**Half bridge 1 switch-off measurement values valid flag** *Note: Clear flag to update measurement values.* 0_B **NOTVALID**, Measurement values not valid 1_B **VALID**, Registers contain valid data
HB1_OFFVALVF_CLR	30	w	**Half bridge 1 switch-off measurement values valid flag clear** 0_B **KEEP**, Keep measurement values 1_B **CLEAR**, Query new measurement values
HB1_OFFVALVF_SET	31	w	**Half bridge 1 switch-off measurement values valid flag set** 0_B **KEEP**, Keep measurement status 1_B **SET**, Set new measurement status

Half bridge 2 switch-on measurement values

HB2ONVAL
Half bridge 2 switch-on measurement values (00E8$_H$) **RESET_TYPE_5 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HB2_ONVALVF_SET	HB2_ONVALVF_CLR	HB2_ONVALVF	HB2_ACTDRV_ON	RES					HB2_TONDURMERR	HB2_TONDUR					
w	w	rhxr	r	r					r	r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HB2_I1ONVAL						HB2_TONDLY							
r		r						r							

Field	Bits	Type	Description
HB2_TONDLY	7:0	r	**Half bridge 2 switch-on delay time value measured until VSH voltage starts to change** Nominal $t_{sdly(on)}$ = 50 ns * (HB2_TONDLY + 1), max. 12.8 µs
HB2_I1ONVAL	13:8	r	**Half bridge 2 switch-on phase 1 current setting from adaptive sequencer** This is the adapted value of I1ON after having applied the T12ON optimization, nominal $I_{1(on)}$ = 5 mA ... 350 mA in 64 steps.
RES	15:14, 27:23	r	**Reserved** Always read as 0
HB2_TONDUR	21:16	r	**Half bridge 2 switch-on voltage slope duration time value** Nominal $t_{sdur(on)}$ = 50 ns * (HB2_TONDUR + 1), max. 3.2 µs
HB2_TONDURMERR	22	r	**Half bridge 2 switch-on voltage slope time measurement error** 0_B **VALID**, Slope duration measurement valid 1_B **ERROR**, Slope duration measurement error
HB2_ACTDRV_ON	28	r	**Half bridge 2 switch-on active driver** 0_B **LS**, Measurement values from low-side driver 1_B **HS**, Measurement values from high-side driver
HB2_ONVALVF	29	rhxr	**Half bridge 2 switch-on measurement values valid flag** *Note: Clear flag to update measurement values.* 0_B **NOTVALID**, Measurement values not valid 1_B **VALID**, Registers contain valid data
HB2_ONVALVF_CLR	30	w	**Half bridge 2 switch-on measurement values valid flag clear** 0_B **KEEP**, Keep measurement values 1_B **CLEAR**, Query new measurement values
HB2_ONVALVF_SET	31	w	**Half bridge 2 switch-on measurement values valid flag set** 0_B **KEEP**, Keep measurement status 1_B **SET**, Set new measurement status

Half bridge 2 switch-off measurement values

HB2OFFVAL
Half bridge 2 switch-off measurement values (00EC$_H$) **RESET_TYPE_5 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HB2_OFFVALVF_SET	HB2_OFFVALVF_CLR	HB2_OFFVALVF	HB2_ACTDRV_OFF	RES					HB2_TOFFDURMERR	HB2_TOFFDUR					
w	w	rhxr	r	r					r	r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HB2_I1OFFVAL						HB2_TOFFDLY							
r		r						r							

Field	Bits	Type	Description
HB2_TOFFDLY	7:0	r	**Half bridge 2 switch-off delay time value measured until VSH voltage starts to change** Nominal $t_{sdly(off)}$ = 50 ns * (HB2_TOFFDLY + 1), max. 12.8 µs
HB2_I1OFFVAL	13:8	r	**Half bridge 2 switch-off phase 1 current setting from adaptive sequencer** This is the adapted value of I1OFF after having applied the T1OFF optimization, nominal $I_{1(off)}$ = 5 mA ... 350 mA in 64 steps.
RES	15:14, 27:23	r	**Reserved** Always read as 0
HB2_TOFFDUR	21:16	r	**Half bridge 2 switch-off voltage slope duration time value** Nominal $t_{sdur(off)}$ = 50 ns * (HB2_TOFFDUR + 1), max. 3.2 µs
HB2_TOFFDURMERR	22	r	**Half bridge 2 switch-off voltage slope time measurement error** 0_B **VALID**, Slope duration measurement valid 1_B **ERROR**, Slope duration measurement error
HB2_ACTDRV_OFF	28	r	**Half bridge 2 switch-off active driver** 0_B **LS**, Measurement values from low-side driver 1_B **HS**, Measurement values from high-side driver
HB2_OFFVALVF	29	rhxr	**Half bridge 2 switch-off measurement values valid flag** *Note: Clear flag to update measurement values.* 0_B **NOTVALID**, Measurement values not valid 1_B **VALID**, Registers contain valid data
HB2_OFFVALVF_CLR	30	w	**Half bridge 2 switch-off measurement values valid flag clear** 0_B **KEEP**, Keep measurement values 1_B **CLEAR**, Query new measurement values
HB2_OFFVALVF_SET	31	w	**Half bridge 2 switch-off measurement values valid flag set** 0_B **KEEP**, Keep measurement status 1_B **SET**, Set new measurement status

Half bridge 3 switch-on measurement values

HB3ONVAL
Half bridge 3 switch-on measurement values (00F0$_H$) RESET_TYPE_5 Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HB3_ONVALVF_SET	HB3_ONVALVF_CLR	HB3_ONVALVF	HB3_ACTDRV_ON	RES					HB3_TONDURMERR	HB3_TONDUR					
w	w	rhxr	r	r					r	r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HB3_I1ONVAL						HB3_TONDLY							
r		r						r							

Field	Bits	Type	Description
HB3_TONDLY	7:0	r	**Half bridge 3 switch-on delay time value measured until VSH voltage starts to change** Nominal $t_{sdly(on)}$ = 50 ns * (HB3_TONDLY + 1), max. 12.8 µs
HB3_I1ONVAL	13:8	r	**Half bridge 3 switch-on phase 1 current setting from adaptive sequencer** This is the adapted value of I1ON after having applied the T12ON optimization, nominal $I_{1(on)}$ = 5 mA ... 350 mA in 64 steps.
RES	15:14, 27:23	r	**Reserved** Always read as 0
HB3_TONDUR	21:16	r	**Half bridge 3 switch-on voltage slope duration time value** Nominal $t_{sdur(on)}$ = 50 ns * (HB3_TONDUR + 1), max. 3.2 µs
HB3_TONDURMERR	22	r	**Half bridge 3 switch-on voltage slope time measurement error** 0_B **VALID**, Slope duration measurement valid 1_B **ERROR**, Slope duration measurement error
HB3_ACTDRV_ON	28	r	**Half bridge 3 switch-on active driver** 0_B **LS**, Measurement values from low-side driver 1_B **HS**, Measurement values from high-side driver
HB3_ONVALVF	29	rhxr	**Half bridge 3 switch-on measurement values valid flag** *Note: Clear flag to update measurement values.* 0_B **NOTVALID**, Measurement values not valid 1_B **VALID**, Registers contain valid data
HB3_ONVALVF_CLR	30	w	**Half bridge 3 switch-on measurement values valid flag clear** 0_B **KEEP**, Keep measurement values 1_B **CLEAR**, Query new measurement values
HB3_ONVALVF_SET	31	w	**Half bridge 3 switch-on measurement values valid flag set** 0_B **KEEP**, Keep measurement status 1_B **SET**, Set new measurement status

Half bridge 3 switch-off measurement values

HB3OFFVAL
Half bridge 3 switch-off measurement values (00F4$_H$) **RESET_TYPE_5 Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HB3_OFFVALVF_SET	**HB3_OFFVALVF_CLR**	**HB3_OFFVALVF**	**HB3_ACTDRV_OFF**	**RES**					**HB3_TOFFDURMERR**	**HB3_TOFFDUR**					
w	w	rhxr	r	r					r	r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		**HB3_I1OFFVAL**						**HB3_TOFFDLY**							
r		r						r							

Field	Bits	Type	Description
HB3_TOFFDLY	7:0	r	**Half bridge 3 switch-off delay time value measured until VSH voltage starts to change** Nominal $t_{sdly(off)}$ = 50 ns * (HB3_TOFFDLY + 1), max. 12.8 µs
HB3_I1OFFVAL	13:8	r	**Half bridge 3 switch-off phase 1 current setting from adaptive sequencer** This is the adapted value of I1OFF after having applied the T1OFF optimization, nominal $I_{1(off)}$ = 5 mA ... 350 mA in 64 steps.
RES	15:14, 27:23	r	**Reserved** Always read as 0
HB3_TOFFDUR	21:16	r	**Half bridge 3 switch-off voltage slope duration time value** Nominal $t_{sdur(off)}$ = 50 ns * (HB3_TOFFDUR + 1), max. 3.2 µs
HB3_TOFFDURMERR	22	r	**Half bridge 3 switch-off voltage slope time measurement error** 0_B **VALID**, Slope duration measurement valid 1_B **ERROR**, Slope duration measurement error
HB3_ACTDRV_OFF	28	r	**Half bridge 3 switch-off active driver** 0_B **LS**, Measurement values from low-side driver 1_B **HS**, Measurement values from high-side driver
HB3_OFFVALVF	29	rhxr	**Half bridge 3 switch-off measurement values valid flag** *Note: Clear flag to update measurement values.* 0_B **NOTVALID**, Measurement values not valid 1_B **VALID**, Registers contain valid data
HB3_OFFVALVF_CLR	30	w	**Half bridge 3 switch-off measurement values valid flag clear** 0_B **KEEP**, Keep measurement values 1_B **CLEAR**, Query new measurement values
HB3_OFFVALVF_SET	31	w	**Half bridge 3 switch-off measurement values valid flag set** 0_B **KEEP**, Keep measurement status 1_B **SET**, Set new measurement status

Register description BDRV

Adaptive sequencer control

ASEQC
Adaptive sequencer control **($00F8_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES								HB3OFFHYSTEN	HB3ONHYSTEN	RES				HB3ASMOFFEN	HB3ASMONEN
r								rw	rw	r				rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HB2OFFHYSTEN	HB2ONHYSTEN	RES				HB2ASMOFFEN	HB2ASMONEN	HB1OFFHYSTEN	HB1ONHYSTEN	RES				HB1ASMOFFEN	HB1ASMONEN
rw	rw	r				rw	rw	rw	rw	r				rw	rw

Field	Bits	Type	Description
HB1ASMONEN	0	rw	**Half bridge 1 adaptive sequencer for switch-on** 0_B **DISABLE**, 1_B **ENABLE**,
HB1ASMOFFEN	1	rw	**Half bridge 1 adaptive sequencer for switch-off** 0_B **DISABLE**, 1_B **ENABLE**,
RES	5:2, 13:10, 21:18, 31:24	r	**Reserved** Always read as 0
HB1ONHYSTEN	6	rw	**Half bridge 1 optimizer hysteresis for switch-on** 0_B **DISABLE**, 1_B **ENABLE**,
HB1OFFHYSTEN	7	rw	**Half bridge 1 optimizer hysteresis for switch-off** 0_B **DISABLE**, 1_B **ENABLE**,
HB2ASMONEN	8	rw	**Half bridge 2 adaptive sequencer for switch-on** 0_B **DISABLE**, 1_B **ENABLE**,
HB2ASMOFFEN	9	rw	**Half bridge 2 adaptive sequencer for switch-off** 0_B **DISABLE**, 1_B **ENABLE**,
HB2ONHYSTEN	14	rw	**Half bridge 2 optimizer hysteresis for switch-on** 0_B **DISABLE**, 1_B **ENABLE**,
HB2OFFHYSTEN	15	rw	**Half bridge 2 optimizer hysteresis for switch-off** 0_B **DISABLE**, 1_B **ENABLE**,

Register description BDRV

Field	Bits	Type	Description
HB3ASMONEN	16	rw	**Half bridge 3 adaptive sequencer for switch-on** 0_B **DISABLE**, 1_B **ENABLE**,
HB3ASMOFFEN	17	rw	**Half bridge 3 adaptive sequencer for switch-off** 0_B **DISABLE**, 1_B **ENABLE**,
HB3ONHYSTEN	22	rw	**Half bridge 3 optimizer hysteresis for switch-on** 0_B **DISABLE**, 1_B **ENABLE**,
HB3OFFHYSTEN	23	rw	**Half bridge 3 optimizer hysteresis for switch-off** 0_B **DISABLE**, 1_B **ENABLE**,

Adaptive sequencer switch-on status

ASEQONSTS
Adaptive sequencer switch-on status **($00FC_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES								HB3ONFAILDRV	RES		HB3ONMF	HB3I1ONMIN	HB3T12ONMIN	HB3I1ONMAX	HB3T12ONMAX
r								rc	r		rc	rc	rc	rc	rc

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HB2ONFAILDRV	RES		HB2ONMF	HB2I1ONMIN	HB2T12ONMIN	HB2I1ONMAX	HB2T12ONMAX	HB1ONFAILDRV	RES		HB1ONMF	HB1I1ONMIN	HB1T12ONMIN	HB1I1ONMAX	HB1T12ONMAX
rc	r		rc	rc	rc	rc	rc	rc	r		rc	rc	rc	rc	rc

Field	Bits	Type	Description
HB1T12ONMAX	0	rc	**Half bridge 1 max T12ON value reached** 0_B **NOERROR**, Max value not reached 1_B **ERROR**, Max value reached
HB1I1ONMAX	1	rc	**Half bridge 1 max I1ON value reached** 0_B **NOERROR**, Max value not reached 1_B **ERROR**, Max value reached
HB1T12ONMIN	2	rc	**Half bridge 1 min T12ON value reached** 0_B **NOERROR**, Min value not reached 1_B **ERROR**, Min value reached
HB1I1ONMIN	3	rc	**Half bridge 1 min I1ON value reached** 0_B **NOERROR**, Min value not reached 1_B **ERROR**, Min value reached

Field	Bits	Type	Description
HB1ONMF	4	rc	**Half bridge 1 adaptive sequencer switch-on measurement failure** 0_B **NOERROR**, No measurement failure 1_B **ERROR**, Measurement failure
RES	6:5, 14:13, 22:21, 31:24	r	**Reserved** Always read as 0
HB1ONFAILDRV	7	rc	**Half bridge 1 switch-on failed gate driver** 0_B **LS**, Failure occurred with low-side driver active 1_B **HS**, Failure occurred with high-side driver active
HB2T12ONMAX	8	rc	**Half bridge 2 max T12ON value reached** 0_B **NOERROR**, Max value not reached 1_B **ERROR**, Max value reached
HB2I1ONMAX	9	rc	**Half bridge 2 max I1ON value reached** 0_B **NOERROR**, Max value not reached 1_B **ERROR**, Max value reached
HB2T12ONMIN	10	rc	**Half bridge 2 min T12ON value reached** 0_B **NOERROR**, Min value not reached 1_B **ERROR**, Min value reached
HB2I1ONMIN	11	rc	**Half bridge 2 min I1ON value reached** 0_B **NOERROR**, Min value not reached 1_B **ERROR**, Min value reached
HB2ONMF	12	rc	**Half bridge 2 adaptive sequencer switch-on measurement failure** 0_B **NOERROR**, No measurement failure 1_B **ERROR**, Measurement failure
HB2ONFAILDRV	15	rc	**Half bridge 2 switch-on failed gate driver** 0_B **LS**, Failure occurred with low-side driver active 1_B **HS**, Failure occurred with high-side driver active
HB3T12ONMAX	16	rc	**Half bridge 3 max T12ON value reached** 0_B **NOERROR**, Max value not reached 1_B **ERROR**, Max value reached
HB3I1ONMAX	17	rc	**Half bridge 3 max I1ON value reached** 0_B **NOERROR**, Max value not reached 1_B **ERROR**, Max value reached
HB3T12ONMIN	18	rc	**Half bridge 3 min T12ON value reached** 0_B **NOERROR**, Min value not reached 1_B **ERROR**, Min value reached
HB3I1ONMIN	19	rc	**Half bridge 3 min I1ON value reached** 0_B **NOERROR**, Min value not reached 1_B **ERROR**, Min value reached

Register description BDRV

Field	Bits	Type	Description
HB3ONMF	20	rc	**Half bridge 3 adaptive sequencer switch-on measurement failure** 0_B **NOERROR**, No measurement failure 1_B **ERROR**, Measurement failure
HB3ONFAILDRV	23	rc	**Half bridge 3 switch-on failed gate driver** 0_B **LS**, Failure occurred with low-side driver active 1_B **HS**, Failure occurred with high-side driver active

Adaptive sequencer switch-off status

ASEQOFFSTS
Adaptive sequencer switch-off status **(0100_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES								HB3OFFFAILDRV	RES		HB3OFFMF	HB3I1OFFMIN	HB3T1OFFMIN	HB3I1OFFMAX	HB3T1OFFMAX
r								rc	r		rc	rc	rc	rc	rc

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HB2OFFFAILDRV	RES		HB2OFFMF	HB2I1OFFMIN	HB2T1OFFMIN	HB2I1OFFMAX	HB2T1OFFMAX	HB1OFFFAILDRV	RES		HB1OFFMF	HB1I1OFFMIN	HB1T1OFFMIN	HB1I1OFFMAX	HB1T1OFFMAX
rc	r		rc	rc	rc	rc	rc	rc	r		rc	rc	rc	rc	rc

Field	Bits	Type	Description
HB1T1OFFMAX	0	rc	**Half bridge 1 max T1OFF value reached** 0_B **NOERROR**, Max value not reached 1_B **ERROR**, Max value reached
HB1I1OFFMAX	1	rc	**Half bridge 1 max I1OFF value reached** 0_B **NOERROR**, Max value not reached 1_B **ERROR**, Max value reached
HB1T1OFFMIN	2	rc	**Half bridge 1 min T1OFF value reached** 0_B **NOERROR**, Min value not reached 1_B **ERROR**, Min value reached
HB1I1OFFMIN	3	rc	**Half bridge 1 min I1OFF value reached** 0_B **NOERROR**, Min value not reached 1_B **ERROR**, Min value reached
HB1OFFMF	4	rc	**Half bridge 1 adaptive sequencer switch-off measurement failure** 0_B **NOERROR**, No measurement failure 1_B **ERROR**, Measurement failure
RES	6:5, 14:13, 22:21, 31:24	r	**Reserved** Always read as 0

Field	Bits	Type	Description
HB1OFFFAILDRV	7	rc	**Half bridge 1 switch-off failed gate driver** 0_B **LS**, Failure occurred with low-side driver active 1_B **HS**, Failure occurred with high-side driver active
HB2T1OFFMAX	8	rc	**Half bridge 2 max T1OFF value reached** 0_B **NOERROR**, Max value not reached 1_B **ERROR**, Max value reached
HB2I1OFFMAX	9	rc	**Half bridge 2 max I1OFF value reached** 0_B **NOERROR**, Max value not reached 1_B **ERROR**, Max value reached
HB2T1OFFMIN	10	rc	**Half bridge 2 min T1OFF value reached** 0_B **NOERROR**, Min value not reached 1_B **ERROR**, Min value reached
HB2I1OFFMIN	11	rc	**Half bridge 2 min I1OFF value reached** 0_B **NOERROR**, Min value not reached 1_B **ERROR**, Min value reached
HB2OFFMF	12	rc	**Half bridge 2 adaptive sequencer switch-off measurement failure** 0_B **NOERROR**, No measurement failure 1_B **ERROR**, Measurement failure
HB2OFFFAILDRV	15	rc	**Half bridge 2 switch-off failed gate driver** 0_B **LS**, Failure occurred with low-side driver active 1_B **HS**, Failure occurred with high-side driver active
HB3T1OFFMAX	16	rc	**Half bridge 3 max T1OFF value reached** 0_B **NOERROR**, Max value not reached 1_B **ERROR**, Max value reached
HB3I1OFFMAX	17	rc	**Half bridge 3 max I1OFF value reached** 0_B **NOERROR**, Max value not reached 1_B **ERROR**, Max value reached
HB3T1OFFMIN	18	rc	**Half bridge 3 min T1OFF value reached** 0_B **NOERROR**, Min value not reached 1_B **ERROR**, Min value reached
HB3I1OFFMIN	19	rc	**Half bridge 3 min I1OFF value reached** 0_B **NOERROR**, Min value not reached 1_B **ERROR**, Min value reached
HB3OFFMF	20	rc	**Half bridge 3 adaptive sequencer switch-off measurement failure** 0_B **NOERROR**, No measurement failure 1_B **ERROR**, Measurement failure
HB3OFFFAILDRV	23	rc	**Half bridge 3 switch-off failed gate driver** 0_B **LS**, Failure occurred with low-side driver active 1_B **HS**, Failure occurred with high-side driver active

Adaptive sequencer error counter control

ASEQERRCNT
Adaptive sequencer error counter control **(0104_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										HB3MFERRCNT		HB3T12ONERRCNT		HB3T1OFFERRCNT	
r										rw		rw		rw	

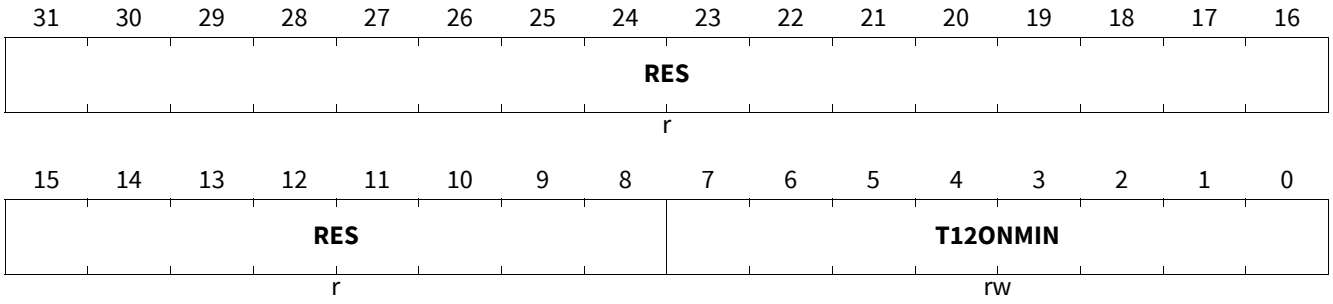
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		HB2MFERRCNT		HB2T12ONERRCNT		HB2T1OFFERRCNT		RES		HB1MFERRCNT		HB1T12ONERRCNT		HB1T1OFFERRCNT	
r		rw		rw		rw		r		rw		rw		rw	

Field	Bits	Type	Description
HB1T1OFFERRCNT	1:0	rw	**Half bridge 1 T1OFF error counter setting** 00_B **2_EVENTS**, Error flag is set after 2 consecutive events 01_B **4_EVENTS**, Error flag is set after 4 consecutive events 10_B **8_EVENTS**, Error flag is set after 8 consecutive events 11_B **15_EVENTS**, Error flag is set after 15 consecutive events
HB1T12ONERRCNT	3:2	rw	**Half bridge 1 T12ON error counter setting** 00_B **2_EVENTS**, Error flag is set after 2 consecutive events 01_B **4_EVENTS**, Error flag is set after 4 consecutive events 10_B **8_EVENTS**, Error flag is set after 8 consecutive events 11_B **15_EVENTS**, Error flag is set after 15 consecutive events
HB1MFERRCNT	5:4	rw	**Half bridge 1 measurement failure error counter setting** 00_B **2_EVENTS**, Error flag is set after 2 consecutive events 01_B **4_EVENTS**, Error flag is set after 4 consecutive events 10_B **8_EVENTS**, Error flag is set after 8 consecutive events 11_B **15_EVENTS**, Error flag is set after 15 consecutive events
RES	7:6, 15:14, 31:22	r	**Reserved** Always read as 0
HB2T1OFFERRCNT	9:8	rw	**Half bridge 2 T1OFF error counter setting** 00_B **2_EVENTS**, Error flag is set after 2 consecutive events 01_B **4_EVENTS**, Error flag is set after 4 consecutive events 10_B **8_EVENTS**, Error flag is set after 8 consecutive events 11_B **15_EVENTS**, Error flag is set after 15 consecutive events
HB2T12ONERRCNT	11:10	rw	**Half bridge 2 T12ON error counter setting** 00_B **2_EVENTS**, Error flag is set after 2 consecutive events 01_B **4_EVENTS**, Error flag is set after 4 consecutive events 10_B **8_EVENTS**, Error flag is set after 8 consecutive events 11_B **15_EVENTS**, Error flag is set after 15 consecutive events

Field	Bits	Type	Description
HB2MFERRCNT	13:12	rw	**Half bridge 2 measurement failure error counter setting** 00_B **2_EVENTS**, Error flag is set after 2 consecutive events 01_B **4_EVENTS**, Error flag is set after 4 consecutive events 10_B **8_EVENTS**, Error flag is set after 8 consecutive events 11_B **15_EVENTS**, Error flag is set after 15 consecutive events
HB3T1OFFERRCNT	17:16	rw	**Half bridge 3 T1OFF error counter setting** 00_B **2_EVENTS**, Error flag is set after 2 consecutive events 01_B **4_EVENTS**, Error flag is set after 4 consecutive events 10_B **8_EVENTS**, Error flag is set after 8 consecutive events 11_B **15_EVENTS**, Error flag is set after 15 consecutive events
HB3T12ONERRCNT	19:18	rw	**Half bridge 3 T12ON error counter setting** 00_B **2_EVENTS**, Error flag is set after 2 consecutive events 01_B **4_EVENTS**, Error flag is set after 4 consecutive events 10_B **8_EVENTS**, Error flag is set after 8 consecutive events 11_B **15_EVENTS**, Error flag is set after 15 consecutive events
HB3MFERRCNT	21:20	rw	**Half bridge 3 measurement failure error counter setting** 00_B **2_EVENTS**, Error flag is set after 2 consecutive events 01_B **4_EVENTS**, Error flag is set after 4 consecutive events 10_B **8_EVENTS**, Error flag is set after 8 consecutive events 11_B **15_EVENTS**, Error flag is set after 15 consecutive events

Adaptive sequencer minimum switch-on time setting

ASEQONTMIN
Adaptive sequencer minimum switch-on time setting(0108_H) **RESET_TYPE_5 Value: $0000\ 0001_H$**

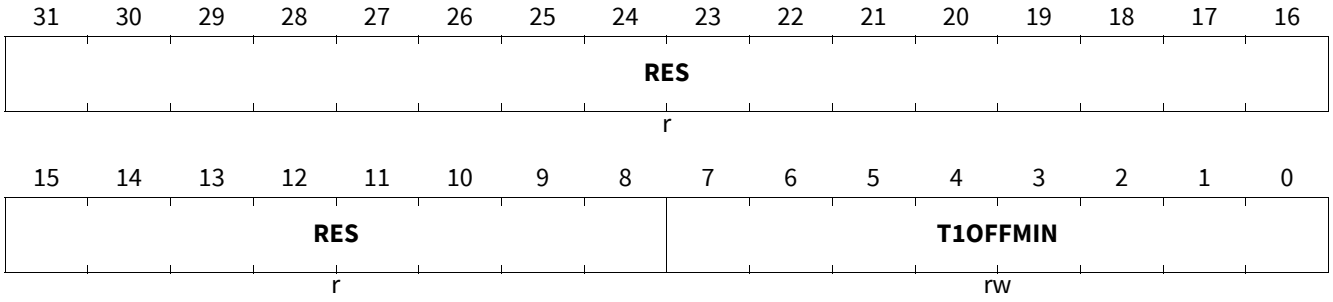


Field	Bits	Type	Description
T12ONMIN	7:0	rw	**Switch-on phases 1 and 2 minimum time setting** Nominal $t_{12on,min}$ = 50 ns * (T12ONMIN + 1), max. 12.8 µs
RES	31:8	r	**Reserved** Always read as 0

Register description BDRV

Adaptive sequencer minimum switch-off time setting

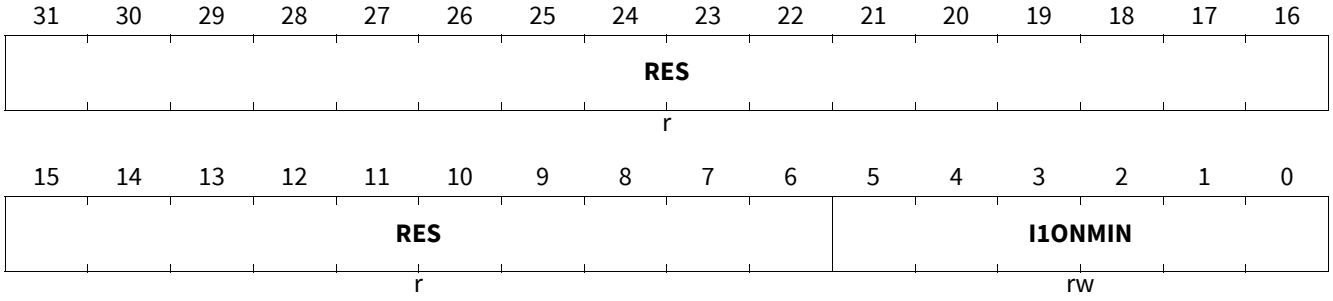
ASEQOFFTMIN
Adaptive sequencer minimum switch-off time setting($010C_H$) **RESET_TYPE_5 Value: $0000\ 0001_H$**



Field	Bits	Type	Description
T1OFFMIN	7:0	rw	**Switch-off phase 1 minimum time setting** Nominal $t_{1off,min}$ = 50 ns * (T1OFFMIN + 1), max. 12.8 µs
RES	31:8	r	**Reserved** Always read as 0

Adaptive sequencer minimum switch-on current setting

ASEQONIMIN
Adaptive sequencer minimum switch-on current setting(0110_H) **RESET_TYPE_5 Value: $0000\ 0006_H$**

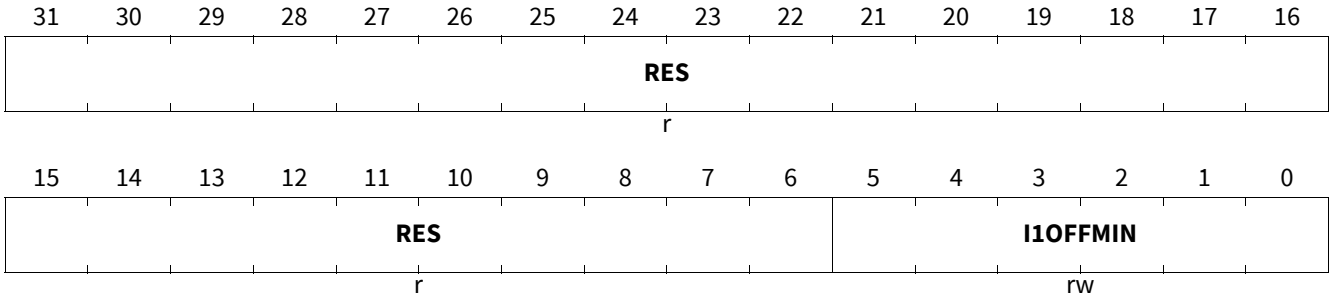


Field	Bits	Type	Description
I1ONMIN	5:0	rw	**Switch-on phase 1 minimum current setting** Nominal $I_{1(on),min}$ = 5 mA ... 350 mA in 64 steps
RES	31:6	r	**Reserved** Always read as 0

Register description BDRV

Adaptive sequencer minimum switch-off current setting

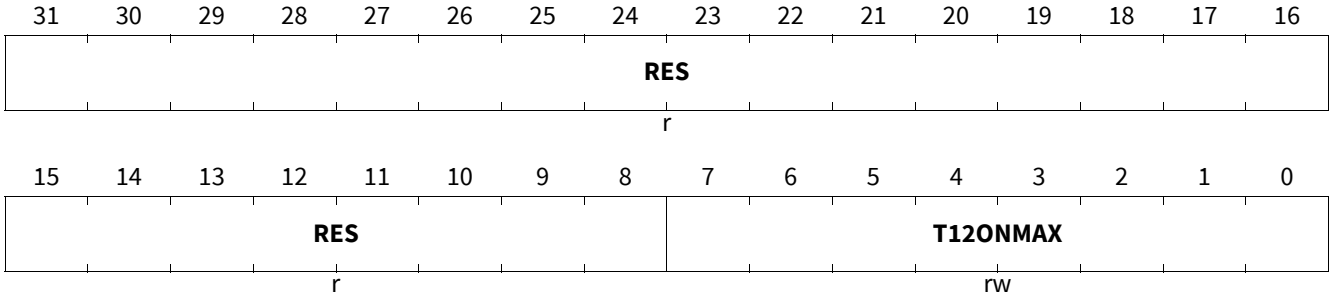
ASEQOFFIMIN
Adaptive sequencer minimum switch-off current setting(0114_H) **RESET_TYPE_5 Value: 0000 0006$_H$**



Field	Bits	Type	Description
I1OFFMIN	5:0	rw	**Switch-off phase 1 minimum current setting** Nominal $I_{1(off),min}$ = 5 mA ... 350 mA in 64 steps
RES	31:6	r	**Reserved** Always read as 0

Adaptive sequencer maximum switch-on time setting

ASEQONTMAX
Adaptive sequencer maximum switch-on time setting(0118_H) **RESET_TYPE_5 Value: 0000 0028$_H$**

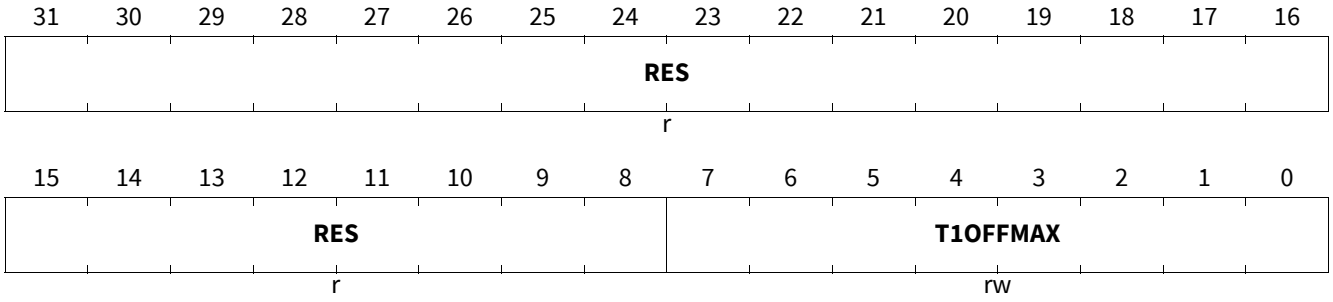


Field	Bits	Type	Description
T12ONMAX	7:0	rw	**Switch-on phases 1 and 2 maximum time setting** Nominal $t_{12on,max}$ = 50 ns * (T12ONMAX + 1), max. 12.8 µs
RES	31:8	r	**Reserved** Always read as 0

Register description BDRV

Adaptive sequencer maximum switch-off time setting

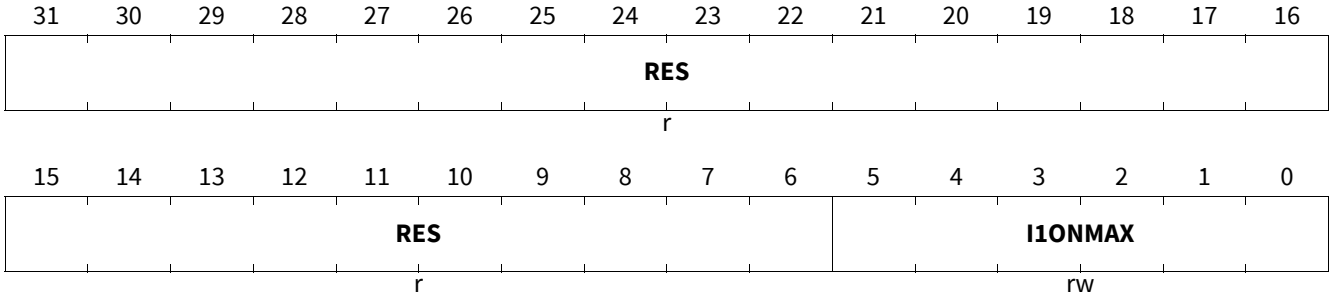
ASEQOFFTMAX
Adaptive sequencer maximum switch-off time setting($011C_H$) **RESET_TYPE_5 Value: $0000\ 0028_H$**



Field	Bits	Type	Description
T1OFFMAX	7:0	rw	**Switch-off phase 1 maximum time setting** Nominal $t_{1off,max}$ = 50 ns * (T1OFFMAX + 1), max. 12.8 µs
RES	31:8	r	**Reserved** Always read as 0

Adaptive sequencer maximum switch-on current setting

ASEQONIMAX
Adaptive sequencer maximum switch-on current setting(0120_H) **RESET_TYPE_5 Value: $0000\ 0037_H$**

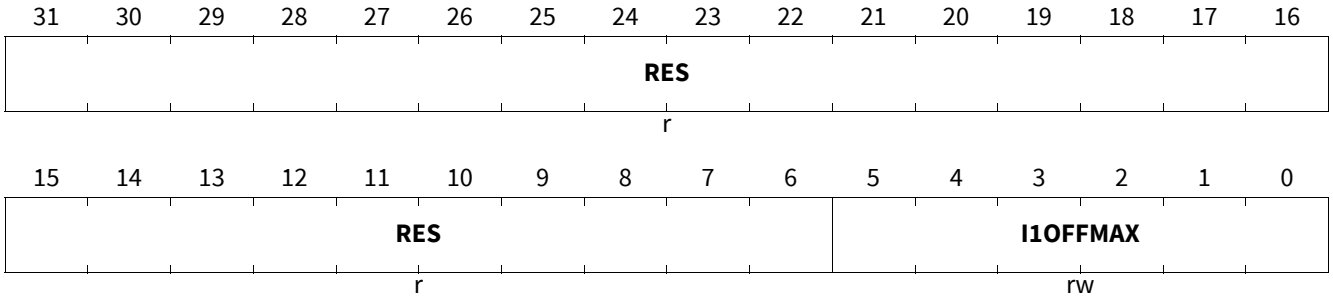


Field	Bits	Type	Description
I1ONMAX	5:0	rw	**Switch-on phase 1 maximum current setting** Nominal $I_{1(on),max}$ = 5 mA ... 350 mA in 64 steps
RES	31:6	r	**Reserved** Always read as 0

Register description BDRV

Adaptive sequencer maximum switch-off current setting

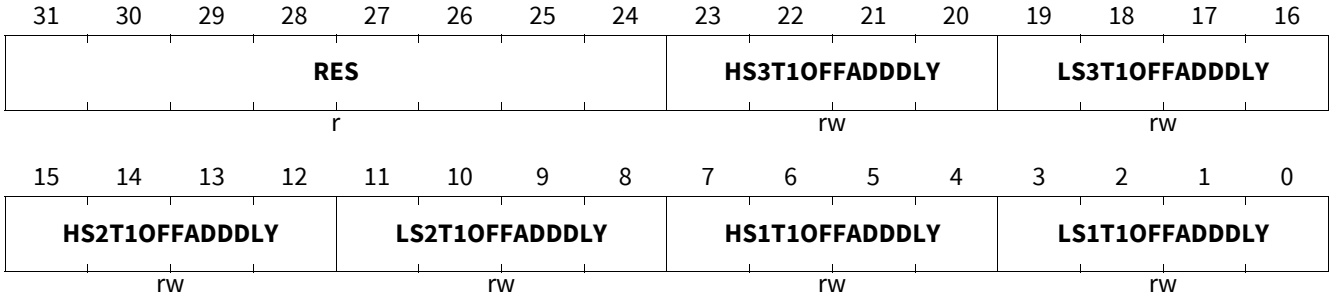
ASEQOFFIMAX
Adaptive sequencer maximum switch-off current setting(0124_H) **RESET_TYPE_5 Value: 0000 0037$_H$**



Field	Bits	Type	Description
I1OFFMAX	5:0	rw	**Switch-off phase 1 maximum current setting** Nominal $I_{1(off),max}$ = 5 mA ... 350 mA in 64 steps
RES	31:6	r	**Reserved** Always read as 0

Adaptive sequencer additional switch-off delay setting

ASEQOFFADDDLY
Adaptive sequencer additional switch-off delay setting(0128_H) **RESET_TYPE_5 Value: 0000 0000$_H$**



Field	Bits	Type	Description
LS1T1OFFADDDLY	3:0	rw	**Low-side driver 1 adaptive sequencer T1OFF additional delay setting** Nominal $t_{1off,add}$ = 50 ns * LS1T1OFFADDDLY 0_H **0n**, 0 ns added to LS1T1OFF for delay time adaption 1_H **DEL1**, Additional delay according to formula … E_H **DEL14**, Additional delay according to formula F_H **750n**, 750 ns added to LS1T1OFF for delay time adaption

Field	Bits	Type	Description
HS1T1OFFADDDLY	7:4	rw	**High-side driver 1 adaptive sequencer T1OFF additional delay setting** Nominal $t_{1off,add}$ = 50 ns * HS1T1OFFADDDLY 0_H **0n**, 0 ns added to HS1T1OFF for delay time adaption 1_H **DEL1**, Additional delay according to formula … E_H **DEL14**, Additional delay according to formula F_H **750n**, 750 ns added to HS1T1OFF for delay time adaption
LS2T1OFFADDDLY	11:8	rw	**Low-side driver 2 adaptive sequencer T1OFF additional delay setting** Nominal $t_{1off,add}$ = 50 ns * LS2T1OFFADDDLY 0_H **0n**, 0 ns added to LS2T1OFF for delay time adaption 1_H **DEL1**, Additional delay according to formula … E_H **DEL14**, Additional delay according to formula F_H **750n**, 750 ns added to LS2T1OFF for delay time adaption
HS2T1OFFADDDLY	15:12	rw	**High-side driver 2 adaptive sequencer T1OFF additional delay setting** Nominal $t_{1off,add}$ = 50 ns * HS2T1OFFADDDLY 0_H **0n**, 0 ns added to HS2T1OFF for delay time adaption 1_H **DEL1**, Additional delay according to formula … E_H **DEL14**, Additional delay according to formula F_H **750n**, 750 ns added to HS2T1OFF for delay time adaption
LS3T1OFFADDDLY	19:16	rw	**Low-side driver 3 adaptive sequencer T1OFF additional delay setting** Nominal $t_{1off,add}$ = 50 ns * LS3T1OFFADDDLY 0_H **0n**, 0 ns added to LS3T1OFF for delay time adaption 1_H **DEL1**, Additional delay according to formula … E_H **DEL14**, Additional delay according to formula F_H **750n**, 750 ns added to LS3T1OFF for delay time adaption
HS3T1OFFADDDLY	23:20	rw	**High-side driver 3 adaptive sequencer T1OFF additional delay setting** Nominal $t_{1off,add}$ = 50 ns * HS3T1OFFADDDLY 0_H **0n**, 0 ns added to HS3T1OFF for delay time adaption 1_H **DEL1**, Additional delay according to formula … E_H **DEL14**, Additional delay according to formula F_H **750n**, 750 ns added to HS3T1OFF for delay time adaption
RES	31:24	r	**Reserved** Always read as 0

Register description BDRV

BEMF comparator control and status

BEMFC_CTRL
BEMF comparator control and status **($012C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES				PH3_ZC_STS	PH2_ZC_STS	PH1_ZC_STS	SW_TRIG	RES			TRIGB_SEL	TRIGA_SEL	IN_SEL	TRIG_SEL	
r				r	r	r	rw	r			rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	BLNK_FILT_BYP	DEMAG_FILT_BYP	TBLNK_SEL			CMP_TFILT_SEL		RES	PH3_COMP_DIS_SET	PH2_COMP_DIS_SET	PH1_COMP_DIS_SET	RES	PH3_COMP_EN	PH2_COMP_EN	PH1_COMP_EN
r	rw	rw	rw			rw		r	rw	rw	rw	r	rw	rw	rw

Field	Bits	Type	Description
PH1_COMP_EN	0	rw	**Phase 1 comparator enable** *Note: For a proper operation of BEMF functionality all three comparators need to be enabled.* 0_B **DISABLE**, 1_B **ENABLE**,
PH2_COMP_EN	1	rw	**Phase 2 comparator enable** *Note: For a proper operation of BEMF functionality all three comparators need to be enabled.* 0_B **DISABLE**, 1_B **ENABLE**,
PH3_COMP_EN	2	rw	**Phase 3 comparator enable** *Note: For a proper operation of BEMF functionality all three comparators need to be enabled.* 0_B **DISABLE**, 1_B **ENABLE**,
RES	3, 7, 15, 23:21, 31:28	r	**Reserved** Always read as 0
PH1_COMP_DIS_SET	4	rw	**Phase 1 comparator output status value if disabled** 0_B **LOW**, PH1_ZC_STS is '0' if disabled 1_B **HIGH**, PH1_ZC_STS is '1' if disabled
PH2_COMP_DIS_SET	5	rw	**Phase 2 comparator output status value if disabled** 0_B **LOW**, PH2_ZC_STS is '0' if disabled 1_B **HIGH**, PH2_ZC_STS is '1' if disabled
PH3_COMP_DIS_SET	6	rw	**Phase 3 comparator output status value if disabled** 0_B **LOW**, PH3_ZC_STS is '0' if disabled 1_B **HIGH**, PH3_ZC_STS is '1' if disabled

Field	Bits	Type	Description
CMP_TFILT_SEL	9:8	rw	**Symmetrical spike filter time for BEMF comparators** 00_B **1u**, 1 µs filter time 01_B **2u**, 2 µs filter time 10_B **4u**, 4 µs filter time 11_B **8u**, 8 µs filter time
TBLNK_SEL	12:10	rw	**Blanking time for BEMF comparators** 000_B **6u**, 6 µs blanking time 001_B **8u**, 8 µs blanking time 010_B **12u**, 12 µs blanking time 011_B **16u**, 16 µs blanking time 100_B **3u**, 3 µs blanking time 101_B **NU**, Not used (3 µs blanking time selected) 110_B **NU**, Not used (3 µs blanking time selected) 111_B **NU**, Not used (3 µs blanking time selected)
DEMAG_FILT_BYP	13	rw	**Demagnetisation filter bypass** 0_B **ENABLE**, Filter enabled 1_B **BYPASS**, Filter bypassed
BLNK_FILT_BYP	14	rw	**Blanking time and demagnetisation filter bypass** 0_B **ENABLE**, Filter enabled 1_B **BYPASS**, Filter bypassed, direct signal output
TRIG_SEL	17:16	rw	**Trigger output selector** 00_B **SWTRIG**, Trigger from BEMFC_CTRL.SW_TRIG 01_B **PH1TRIG**, Trigger from BEMFC_CTRL.PH1_ZC_STS 10_B **PH2TRIG**, Trigger from BEMFC_CTRL.PH2_ZC_STS 11_B **PH3TRIG**, Trigger from BEMFC_CTRL.PH3_ZC_STS
IN_SEL	18	rw	**Deactivate INA/B/C as sample pulse sources** 0_B **ACT**, INA/B/C sample inputs are used 1_B **NOTACT**, INA/B/C are not used
TRIGA_SEL	19	rw	**Deactivate TRIGA as sample pulse source** 0_B **ACT**, TRIGA sample input is used 1_B **NOTACT**, TRIGA is not used
TRIGB_SEL	20	rw	**Deactivate TRIGB as sample pulse source** 0_B **ACT**, TRIGB sample input is used 1_B **NOTACT**, TRIGB is not used
SW_TRIG	24	rw	**Software trigger for output to timer** TRIG_SEL needs to be set to '00' for this bit to become relevant. 0_B **VALUE_0**, Software trigger reset 1_B **VALUE_1**, Software trigger set
PH1_ZC_STS	25	r	**Phase 1 zero crossing comparator status** 0_B **NEGATIVE**, Phase voltage is below the average of the other phases 1_B **POSITIVE**, Phase voltage is above the average of the other phases

Field	Bits	Type	Description
PH2_ZC_STS	26	r	**Phase 2 zero crossing comparator status** 0_B **NEGATIVE**, Phase voltage is below the average of the other phases 1_B **POSITIVE**, Phase voltage is above the average of the other phases
PH3_ZC_STS	27	r	**Phase 3 zero crossing comparator status** 0_B **NEGATIVE**, Phase voltage is below the average of the other phases 1_B **POSITIVE**, Phase voltage is above the average of the other phases

BEMF comparator interrupt status

BEMFC_IRQS
BEMF comparator interrupt status **(0130_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										PH3_ZCRISE_IS	PH3_ZCFALL_IS	PH2_ZCRISE_IS	PH2_ZCFALL_IS	PH1_ZCRISE_IS	PH1_ZCFALL_IS
r										rhxre	rhxre	rhxre	rhxre	rhxre	rhxre

Field	Bits	Type	Description
PH1_ZCFALL_IS	0	rhxre	**Phase 1 zero crossing falling interrupt status** 0_B **NOZCFALL**, No falling zero crossing detected 1_B **ZCFALL**, Falling zero crossing detected
PH1_ZCRISE_IS	1	rhxre	**Phase 1 zero crossing rising interrupt status** 0_B **NOZCRISE**, No rising zero crossing detected 1_B **ZCRISE**, Rising zero crossing detected
PH2_ZCFALL_IS	2	rhxre	**Phase 2 zero crossing falling interrupt status** 0_B **NOZCFALL**, No falling zero crossing detected 1_B **ZCFALL**, Falling zero crossing detected
PH2_ZCRISE_IS	3	rhxre	**Phase 2 zero crossing rising interrupt status** 0_B **NOZCRISE**, No rising zero crossing detected 1_B **ZCRISE**, Rising zero crossing detected
PH3_ZCFALL_IS	4	rhxre	**Phase 3 zero crossing falling interrupt status** 0_B **NOZCFALL**, No falling zero crossing detected 1_B **ZCFALL**, Falling zero crossing detected
PH3_ZCRISE_IS	5	rhxre	**Phase 3 zero crossing rising interrupt status** 0_B **NOZCRISE**, No rising zero crossing detected 1_B **ZCRISE**, Rising zero crossing detected

Register description BDRV

Field	Bits	Type	Description
RES	31:6	r	**Reserved** Always read as 0

BEMF comparator interrupt status clear

BEMFC_IRQCLR

BEMF comparator interrupt status clear **(0134$_H$)** **RESET_TYPE_5 Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										PH3_ZCRISE_ISC	PH3_ZCFALL_ISC	PH2_ZCRISE_ISC	PH2_ZCFALL_ISC	PH1_ZCRISE_ISC	PH1_ZCFALL_ISC
r										w	w	w	w	w	w

Field	Bits	Type	Description
PH1_ZCFALL_ISC	0	w	**Phase 1 zero crossing falling interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
PH1_ZCRISE_ISC	1	w	**Phase 1 zero crossing rising interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
PH2_ZCFALL_ISC	2	w	**Phase 2 zero crossing falling interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
PH2_ZCRISE_ISC	3	w	**Phase 2 zero crossing rising interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
PH3_ZCFALL_ISC	4	w	**Phase 3 zero crossing falling interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
PH3_ZCRISE_ISC	5	w	**Phase 3 zero crossing rising interrupt status clear** 0_B **KEEP**, 1_B **CLEAR**,
RES	31:6	r	**Reserved** Always read as 0

Register description BDRV

BEMF comparator interrupt status set

BEMFC_IRQSET
BEMF comparator interrupt status set **(0138_H)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										PH3_ZCRISE_ISS	PH3_ZCFALL_ISS	PH2_ZCRISE_ISS	PH2_ZCFALL_ISS	PH1_ZCRISE_ISS	PH1_ZCFALL_ISS
r										w	w	w	w	w	w

Field	Bits	Type	Description
PH1_ZCFALL_ISS	0	w	**Phase 1 zero crossing falling interrupt status set** 0_B **KEEP**, 1_B **SET**,
PH1_ZCRISE_ISS	1	w	**Phase 1 zero crossing rising interrupt status set** 0_B **KEEP**, 1_B **SET**,
PH2_ZCFALL_ISS	2	w	**Phase 2 zero crossing falling interrupt status set** 0_B **KEEP**, 1_B **SET**,
PH2_ZCRISE_ISS	3	w	**Phase 2 zero crossing rising interrupt status set** 0_B **KEEP**, 1_B **SET**,
PH3_ZCFALL_ISS	4	w	**Phase 3 zero crossing falling interrupt status set** 0_B **KEEP**, 1_B **SET**,
PH3_ZCRISE_ISS	5	w	**Phase 3 zero crossing rising interrupt status set** 0_B **KEEP**, 1_B **SET**,
RES	31:6	r	**Reserved** Always read as 0

Register description BDRV

BEMF comparator interrupt enable

BEMFC_IRQEN
BEMF comparator interrupt enable **($013C_H$)** **RESET_TYPE_5 Value: $0000\ 0000_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES										PH3_ZCRISE_IEN	PH3_ZCFALL_IEN	PH2_ZCRISE_IEN	PH2_ZCFALL_IEN	PH1_ZCRISE_IEN	PH1_ZCFALL_IEN
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PH1_ZCFALL_IEN	0	rw	**Phase 1 zero crossing falling interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
PH1_ZCRISE_IEN	1	rw	**Phase 1 zero crossing rising interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
PH2_ZCFALL_IEN	2	rw	**Phase 2 zero crossing falling interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
PH2_ZCRISE_IEN	3	rw	**Phase 2 zero crossing rising interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
PH3_ZCFALL_IEN	4	rw	**Phase 3 zero crossing falling interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
PH3_ZCRISE_IEN	5	rw	**Phase 3 zero crossing rising interrupt enable** 0_B **DISABLE**, 1_B **ENABLE**,
RES	31:6	r	**Reserved** Always read as 0

27 Package information

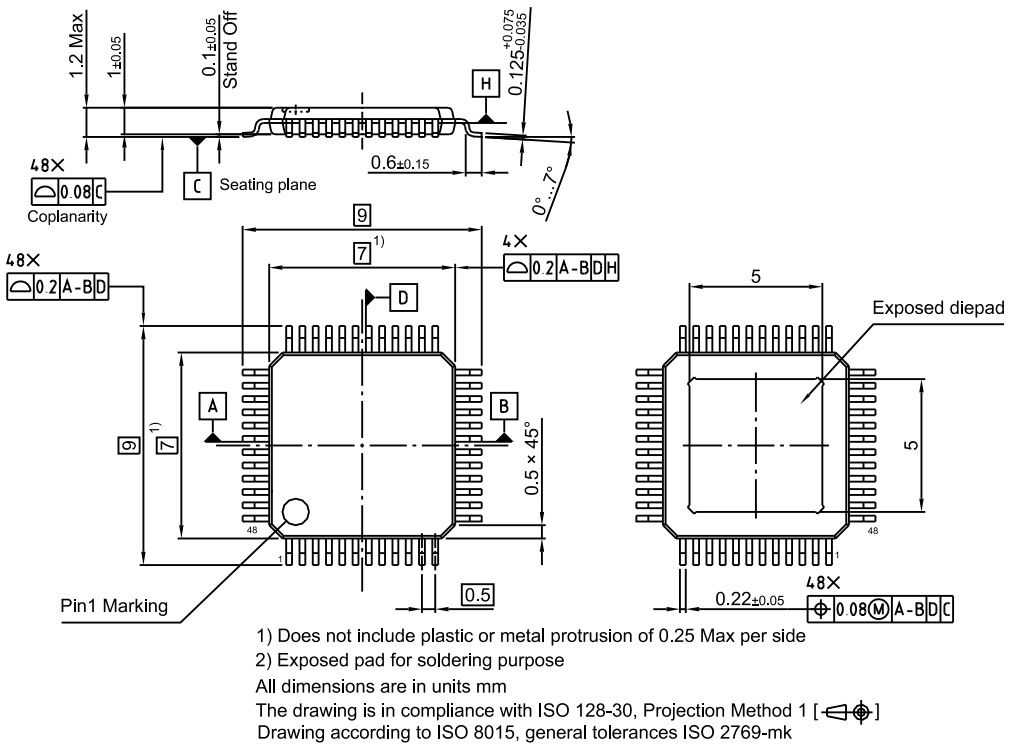


Figure 377 PG-TQFP-48-10

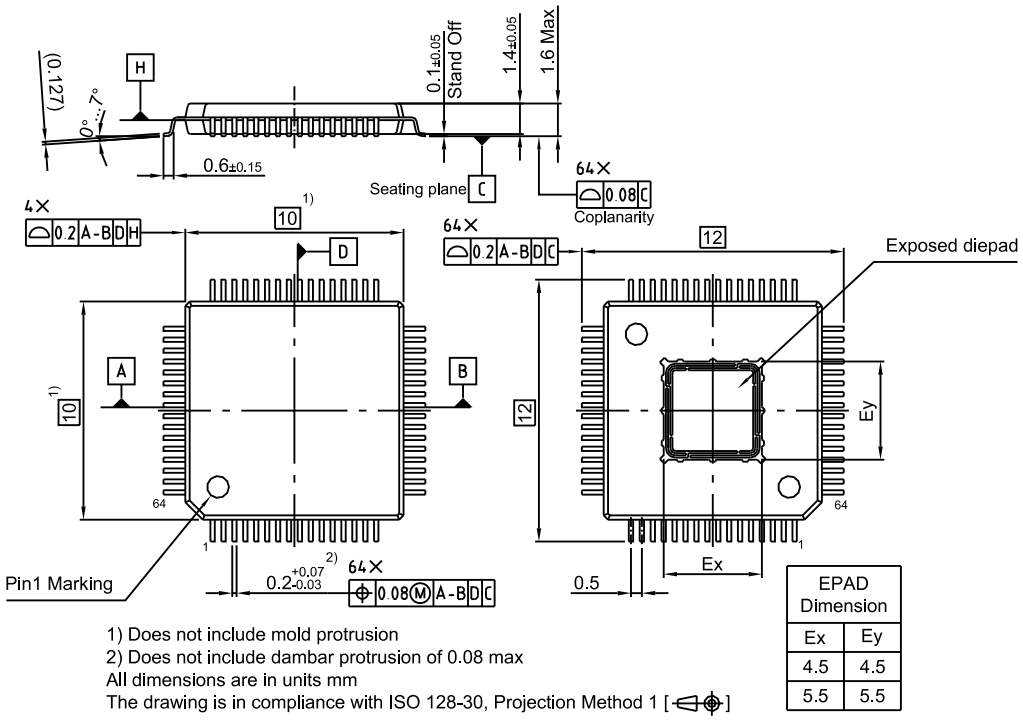


Figure 378 PG-LQFP-64-28

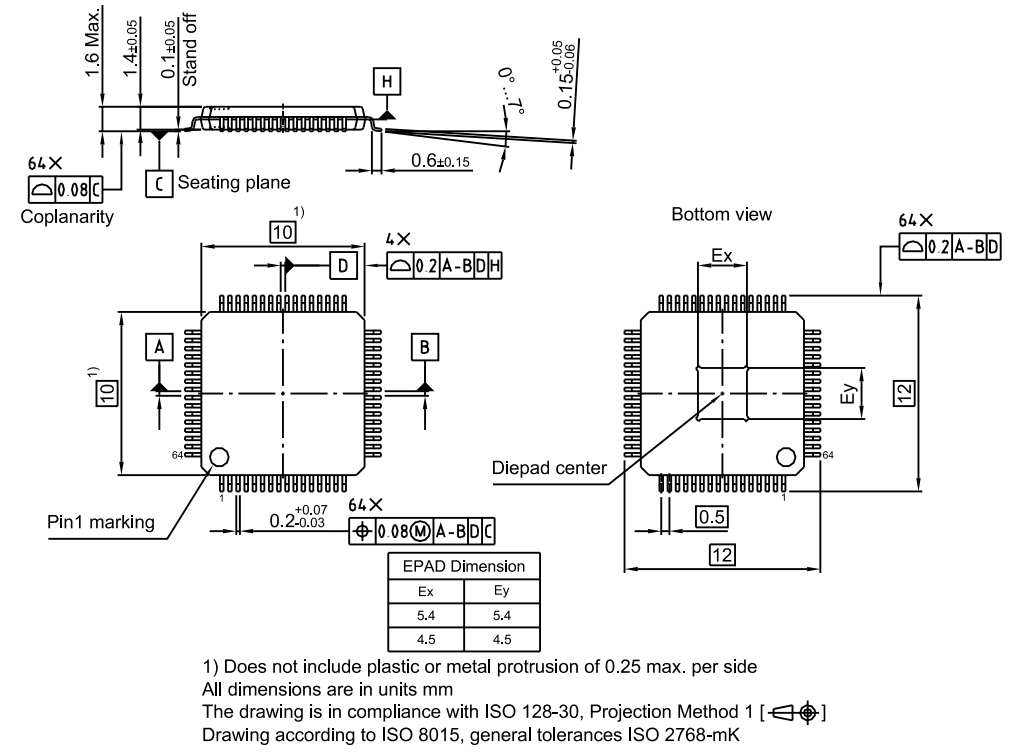


Figure 379 PG-LQFP-64-31

Package information

Green Product (RoHS compliant)

To meet the world-wide customer requirements for environmentally friendly products and to be compliant with government regulations the device is available as a green product. Green products are RoHS-Compliant (i.e Pb-free finish on leads and suitable for Pb-free soldering according to IPC/JEDEC J-STD-020).

Further information on packages

https://www.infineon.com/packages

28 Abbreviations

The following acronyms and terms are used within this document. List see in **Table 332**.

Table 332 Acronyms

Acronyms	Name
100TP	100 Time Programmable
AHB	Advanced High-performance Bus
APB	Advanced Peripheral Bus
ASIL	Automotive Safety Integrity Level
BEMF	Back Electro Magnetic Force
CAN	Controller Area Network
CP	Charge Pump for MOSFET driver
DMA	Direct Memory Access
ECC	Error Correction Code
EEPROM	Electrically Erasable Programmable Read Only Memory
FS	Functional Safety
FSM	Finite State Machine
GPIO	General Purpose Input Output
HiZ	High impedance
IEN	Interrupt Enable
LDO	Low DropOut voltage regulator
LOW	Long Open Window (for WDT)
LSB	Least Significant Bit
LQFP	Low profile Quad Flat Package
MCTRL	Motor control
MCU	Micro Controller Unit
MPU	Memory Protection Unit
MRST	Master Receive Slave Transmit
MSB	Most Significant Bit
MTSR	Master Transmit Slave Receive
N-FET	N-channel Field Effect Transistor
NMI	Non-Maskable Interrupt
NVIC	Nested Vector Interrupt Controller
NVM	Non-Volatile Memory
OSC	Oscillator
OT	Overtemperature
OTP	One Time Programmable
PBA	Peripheral Bridge
PC	Program Counter

Table 332 Acronyms (cont'd)

Acronyms	Name
PD	Pull Down
PLL	Phase Locked Loop
PMU	Power Management Unit
PPB	Private Peripheral Bus
PSRAM	Program Static Random Access Memory
PSW	Program Status Word
PU	Pull Up
PWM	Pulse Width Modulation
RAM	Random Access Memory
ROM	Read Only Memory
SCB	Short Circuit to Battery
SCG	Short Circuit to Ground
SECDED	Single Error Correction Double Error Detection
SFR	Special Function Register
SoC	System on Chip
SOW	Short Open Window (for WDT)
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
SSO	Safe Switch Off (path)
SWD	Arm® Serial Wire Debug
TAP	Test Access Port (for test and debug)
TCCR	Temperature Compensation Control Register
TMS	Test Mode Select
TSD	Thermal Shut Down
TQFP	Thin Quad Flat Package
UART	Universal Asynchronous Receiver Transmitter
UV	Undervoltage
VBG	Voltage reference Band Gap
VCO	Voltage Controlled Oscillator
WDT	Watchdog timer in SCU-DM

29 Revision history

Revision	Date	Changes
Rev. 1.0	2023-05-16	Initial version

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2023-05-16
Published by
Infineon Technologies AG
81726 Munich, Germany

Do you have a question about any aspect of this document?
Email: erratum@infineon.com

Document reference
Z8F69685615

IMPORTANT NOTICE

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenheitsgarantie").

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on technology, delivery terms and conditions and prices, please contact the nearest Infineon Technologies Office (**www.infineon.com**).

WARNINGS

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.