**Please note that Cypress is an Infineon Technologies Company.**
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

**Continuity of document content**
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

**Continuity of ordering part numbers**
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

The following document contains information on Cypress products. The document has the series name, product name, and ordering part numbering with the prefix “MB”. However, Cypress will offer these products to new and existing customers with the series name, product name, and ordering part number with the prefix “CY”.

**How to Check the Ordering Part Number**

1. Go to www.cypress.com/pcn.
2. Enter the keyword (for example, ordering part number) in the **SEARCH PCNS** field and click **Apply**.
3. Click the corresponding title from the search results.
4. Download the Affected Parts List file, which has details of all changes

**For More Information**

Please contact your local sales office for additional information about Cypress products and solutions.

**About Cypress**

Cypress is the leader in advanced embedded system solutions for the world's most innovative automotive, industrial, smart home appliances, consumer electronics and medical products. Cypress' microcontrollers, analog ICs, wireless and USB-based connectivity solutions and reliable, high-performance memories help engineers design differentiated products and get them to market first. Cypress is committed to providing customers with the best support and development resources on the planet enabling them to disrupt markets by creating new product categories in record time. To learn more, go to www.cypress.com.

# MB91460X Series

# FR60 32-bit Microcontrollers

MB91460X series is a line of general-purpose 32-bit RISC microcontrollers designed for embedded control applications which require high-speed real-time processing, such as consumer devices and on-board vehicle systems. This series uses the FR60 CPU, which is compatible with the FR family of CPUs.

This series contains the LIN-USART, CAN and FlexRay controllers.

## Features

### FR60 CPU core

- 32-bit RISC, load/store architecture, five-stage pipeline
- 16-bit fixed-length instructions (basic instructions)
- Instruction execution speed: 1 instruction per cycle
- Instructions including memory-to-memory transfer, bit manipulation, and barrel shift instructions: Instructions suitable for embedded applications
- Function entry/exit instructions and register data multi-load store instructions: Instructions supporting C language
- Register interlock function: Facilitating assembly-language coding
- Built-in multiplier with instruction-level support
  - Signed 32-bit multiplication: 5 cycles
  - Signed 16-bit multiplication: 3 cycles
- Interrupts (save PC/PS): 6 cycles (16 priority levels)
- Harvard architecture enabling program access and data access to be performed simultaneously
- Instructions compatible with the FR family

### Internal peripheral resources

- General-purpose ports : Maximum 73 ports
- DMAC (DMA Controller)
  - Maximum of 5 channels able to operate simultaneously
  - 2 transfer sources (internal peripheral/software)
  - Activation source can be selected using software.
  - Addressing mode specifies full 32-bit addresses (increment/decrement/fixed)
  - Transfer mode (demand transfer/burst transfer/step transfer/block transfer)
  - Transfer data size selectable from 8/16/32-bit
  - Multi-byte transfer enabled (by software)
  - DMAC descriptor in I/O areas ($200_H$ to $240_H$, $1000_H$ to $1027_H$)
- Flexray controller: 2 channels
  - Conformance with FlexRay protocol specification v2.1
  - Maximum transfer speed: 10 Mbps
  - Up to 128 transmission/reception message buffers
- A/D converter (successive approximation type)
  - 10-bit resolution: 17 channels
  - Conversion time: minimum 1 µs
- External interrupt inputs : 11 channels
  - 3 channels shared with CAN RX or $I^2C$ pins
- Bit search module (for REALOS)
  - Function to search from the MSB (most significant bit) for the position of the first “0”, “1”, or changed bit in a word
- LIN-USART (full duplex double buffer): 3 channels
  - Clock synchronous/asynchronous selectable
  - Sync-break detection
  - Internal dedicated baud rate generator
- $I^2C$ bus interface (supports 400 kbps): 1 channel
  - Master/slave transmission and reception
  - Arbitration function, clock synchronization function
- CAN controller (C-CAN): 2 channels
  - Maximum transfer speed: 1 Mbps
  - 32 transmission/reception message buffers
- 16-bit PPG timer : 12 channels
- 16-bit reload timer: 8 channels
- 16-bit free-run timer: 8 channels (1 channel each for ICU and OCU)
- Input capture: 8 channels (operates in conjunction with the free-run timer)
- Output compare: 6 channels (operates in conjunction with the free-run timer)
- Watchdog timer
- Real-time clock
- Low-power consumption modes : Sleep/stop mode function
- Low voltage detection circuit
- Clock supervisor
  - Monitors the sub-clock (32 kHz) and the main clock (4 MHz) , and switches to a recovery clock (CR oscillator, etc.) when the oscillations stop.
- Clock modulator
- Clock monitor
- Sub-clock calibration
  - Corrects the real-time clock timer when operating with the 32 kHz or CR oscillator
- Main oscillator stabilization timer
  - Generates an interrupt in sub-clock mode after the stabilization wait time has elapsed on the 23-bit stabilization wait time counter

- Sub-oscillator stabilization timer
  - Generates an interrupt in main clock mode after the stabilization wait time has elapsed on the 15-bit stabilization wait time counter

### Package and technology

- Package: QFP-100
- CMOS 0.18 µm technology
- Power supply range 3 V to 5 V (1.8 V internal logic provided by a step-down voltage converter)
- Operating temperature range: between − 40°C and + 105°C

## Contents


**Product Lineup** ..... **4**
**Pin Assignment** ..... **6**
MB91F465XA ..... 6
**Pin Description** ..... **7**
MB91F465XA ..... 7
**I/O Circuit Types** ..... **11**
**Handling Devices** ..... **17**
Preventing Latch-up ..... 17
Handling of unused input pins ..... 17
Power supply pins ..... 17
Crystal oscillator circuit ..... 17
Notes on using external clock ..... 17
Mode pins (MD_x) ..... 18
Notes on operating in PLL clock mode ..... 18
Pull-up control ..... 18
Notes on PS register ..... 18
**Notes on Debugger** ..... **19**
Execution of the RETI Command ..... 19
Break function ..... 19
Operand break ..... 19
**Block Diagram** ..... **20**
MB91F465XA ..... 20
**CPU and Control Unit** ..... **21**
Features ..... 21
Internal architecture ..... 21
Programming model ..... 22
Registers ..... 23
**Embedded Program/Data Memory (Flash)** ..... **26**
Flash features ..... 26
Operation modes ..... 26
Flash access in CPU mode ..... 27
Parallel Flash programming mode ..... 30
Flash Security ..... 32
Notes About Flash Memory CRC Calculation ..... 34
**Memory Space** ..... **35**
**Memory Maps** ..... **36**
MB91F465XA ..... 36
**I/O Map** ..... **37**
MB91F465XA ..... 37
Flash memory and external bus area ..... 63
**Interrupt Vector Table** ..... **65**
**Recommended Settings** ..... **70**
PLL and Clockgear settings ..... 70
Clock Modulator settings ..... 71
FlexRay PLL, Clock and Port settings ..... 76
**Electrical Characteristics** ..... **78**
Absolute maximum ratings ..... 78
Recommended operating conditions ..... 80
DC characteristics ..... 81
A/D converter characteristics ..... 85
FLASH memory program/erase characteristics ..... 89
AC characteristics ..... 90
**E-Ray Overview** ..... **97**
Block Diagram ..... 98
**Generic Interface** ..... **101**
Generic CPU Interface ..... 102
Internal Signal and Flag Interface ..... 105
Physical Layer Interface ..... 107
Interface to embedded RAM Blocks ..... 108
**Programmer's Model** ..... **112**
Register Map ..... 112
Customer Registers ..... 116
Special Registers ..... 119
Interrupt Registers ..... 125
CC Control Registers ..... 139
CC Status Registers ..... 153
Message Buffer Control Registers ..... 163
Message Buffer Status Registers ..... 166
Identification Registers ..... 173
Input Buffer ..... 174
Output Buffer ..... 179
**Functional Description** ..... **186**
Communication Cycle ..... 186
Communication Modes ..... 188
Clock Synchronization ..... 188
Error Handling ..... 190
Communication Controller States ..... 191
Network Management ..... 204
Filtering and Masking ..... 205
Transmit Process ..... 207
Receive Process ..... 209
FIFO Function ..... 210
Message Handling ..... 211
Message RAM ..... 218
Module Interrupt ..... 224
**Appendix** ..... **226**
Register Bit Overview ..... 226
Assignment of FlexRay Configuration Parameters ..... 239
**Ordering Information** ..... **241**
**Package Dimension** ..... **242**
**Revision History** ..... **243**
**Main Changes** ..... **243**
**Document History** ..... **244**

# 1. Product Lineup

| Feature | MB91V460A | MB91F465XA |
|---|---|---|
| Max. core frequency (CLKB) | 80MHz | 100MHz |
| Max. resource frequency (CLKP) | 40MHz | 50MHz |
| Max. external bus freq. (CLKT) | 40MHz | - |
| Max. CAN frequency (CLKCAN) | 20MHz | 50MHz |
| Max. FlexRay frequency (SCLK) | - | 80MHz |
| Technology | 0.35µm | 0.18µm |
| Watchdog | yes | yes |
| Watchdog (RC osc. based) | yes (disengageable) | yes |
| Bit Search | yes | yes |
| Reset input (INITX) | yes | yes |
| Hardware Standby input (HSTX) | yes | no |
| Clock Modulator | yes | yes |
| Clock Monitor | yes | yes |
| Low Power Mode | yes | yes |
| DMA | 5 ch | 5 ch |
| MAC (µDSP) | no | no |
| MMU/MPU | MPU (16 ch) [1)] | MPU (8 ch) [1)] |
| | | |
| FlexRay 2 channels (A/B) | no | yes |
| | | |
| Flash | Emulation SRAM 32bit read data | 544 KByte |
| Flash Protection | - | yes |
| | | |
| D-RAM | 64 KByte | 16 KByte |
| ID-RAM | 64 KByte | 16 KByte |
| Flash-Cache (Instruction cache) | 16 KByte | 8 KByte |
| Boot-ROM / BI-ROM | 4 KByte fixed | 4 KByte |
| | | |
| RTC | 1 ch | 1 ch |
| Free Running Timer | 8 ch | 8 ch |
| ICU | 8 ch | 8 ch |
| OCU | 8 ch | 6 ch |
| Reload Timer | 8 ch | 8 ch |
| PPG 16-bit | 16 ch | 12 ch |
| PFM 16-bit | 1 ch | - |
| Sound Generator | 1 ch | - |
| Up/Down Counter (8/16-bit) | 4 ch (8-bit) / 2 ch (16-bit) | - |

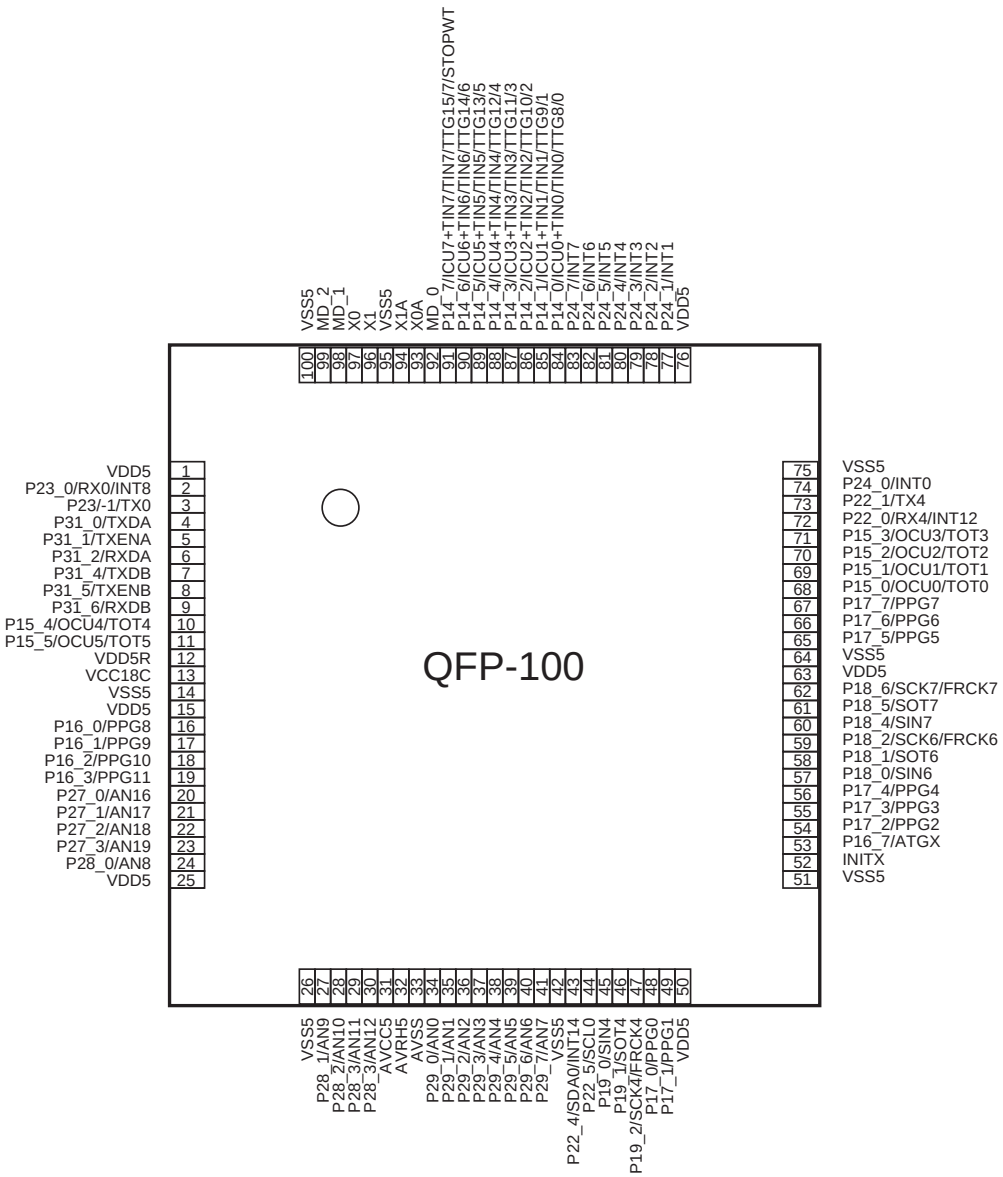
| Feature | MB91V460A | MB91F465XA |
|---|---|---|
| | | |
| C_CAN | 6 ch (128msg) | 2 ch (32msg) |
| LIN-USART | 4 ch + 4 ch FIFO + 8 ch | 3 ch FIFO |
| $I^2C$ (400k) | 4 ch | 1 ch |
| | | |
| FR external bus | yes (32bit addr, 32bit data) | - |
| | | |
| External Interrupts | 16 ch | 11 ch |
| NMI Interrupts | 1 ch | - |
| | | |
| SMC | 6 ch | - |
| LCD controller (40x4) | 1 ch | - |
| | | |
| ADC (10 bit) | 32 ch | 17 ch |
| Alarm Comparator | 2 ch | - |
| | | |
| Supply Supervisor | yes | yes |
| Clock Supervisor | yes | yes |
| | | |
| Main clock oscillator | 4MHz | 4MHz |
| Sub clock oscillator | 32kHz | 32kHz |
| RC Oscillator | 100kHz | 100kHz / 2MHz |
| PLL | x 20 | x 25 |
| | | |
| DSU4 | yes | - |
| EDSU | yes (32 BP) *1 | yes (16 BP) *1 |
| | | |
| Supply Voltage | 3V / 5V | 3V / 5V |
| Regulator | yes | yes |
| Power Consumption | n.a. | < 1 W |
| Temperature Range (Ta) | 0..70 C | -40..105 C |
| | | |
| Package | BGA660 | QFP100 |
| | | |
| Power on to PLL run | < 20 ms | < 20 ms |
| Flash Download Time | n.a. | < 5 sec. typical |
| | | |

*1: MPU channels use EDSU breakpoint registers (shared operation between MPU and EDSU).

# 2. Pin Assignment

## 2.1. MB91F465XA

(TOP VIEW)



# 3. Pin Description

## 3.1. MB91F465XA

| Pin no. | Pin name | I/O | I/O circuit type* | Function |
|---|---|---|---|---|
| 2 | P23_0 | I/O | A | General-purpose input/output port |
| | RX0 | | | RX input/output pin of CAN0 |
| | INT8 | | | External interrupt input pin |
| 3 | P23_1 | I/O | A | General-purpose input/output port |
| | TX0 | | | TX output pin of CAN0 |
| 4 | P31_0 | I/O | A | General-purpose input/output port |
| | TXDA | | | FlexRay transmit output pin |
| 5 | P31_1 | I/O | A | General-purpose input/output port |
| | TXENA | | | FlexRay transmit enable output pin |
| 6 | P31_2 | I/O | A | General-purpose input/output port |
| | RXDA | | | FlexRay receive input pin |
| 7 | P31_4 | I/O | A | General-purpose input/output port |
| | TXDB | | | FlexRay Transmit output pin |
| 8 | P31_5 | I/O | A | General-purpose input/output port |
| | TXENB | | | FlexRay transmit enable output pin |
| 9 | P31_6 | I/O | A | General-purpose input/output port |
| | RXDB | | | FlexRay receive input pin |
| 10, 11 | P15_4,P15_5 | I/O | A | General-purpose input/output ports |
| | OCU4, OCU5 | | | Output compare output pins |
| | TOT4, TOT5 | | | Reload timer output pins |
| 16 to 19 | P16_0 to P16_3 | I/O | A | General-purpose input/output ports |
| | PPG8 to PPG11 | | | Output pins of PPG timer |
| 20 to 23 | P27_0 to P27_3 | I/O | B | General-purpose input/output ports |
| | AN16 to AN19 | | | Analog input pins of A/D converter |
| 24 | P28_0 | I/O | B | General-purpose input/output port |
| | AN8 | | | Analog input pins of A/D converter |
| 27 to 30 | P28_1 to P28_4 | I/O | B | General-purpose input/output ports |
| | AN9 to AN12 | | | Analog input pins of A/D converter |
| 34 to 41 | P29_0 to P29_7 | I/O | B | General-purpose input/output ports |
| | AN0 to AN7 | | | Analog input pins of A/D converter |
| 43 | P22_4 | I/O | C | General-purpose input/output port |
| | SDA0 | | | $I^2C$ bus data input/output pin |
| | INT14 | | | External interrupt input pin |
| 44 | P22_5 | I/O | C | General-purpose input/output port |
| | SCL0 | | | I2C bus clock input/output pin |

| Pin no. | Pin name | I/O | I/O circuit type* | Function |
|---|---|---|---|---|
| 45 | P19_0 | I/O | A | General-purpose input/output port |
| | SIN4 | | | Data input pin of USART4 |
| 46 | P19_1 | I/O | A | General-purpose input/output port |
| | SOT4 | | | Data output pin of USART4 |
| 47 | P19_2 | I/O | A | General-purpose input/output port |
| | SCK4 | | | Clock input/output pin of USART4 |
| | FRCK4 | | | Free Run Timer external clock input pin |
| 48, 49 | P17_0, P17_1 | I/O | A | General-purpose input/output ports |
| | PPG0, PPG1 | | | Output pins of PPG timer |
| 52 | INITX | I | H | External reset input pin |
| 53 | P16_7 | I/O | A | General-purpose input/output port |
| | ATGX | | | A/D converter external trigger input pin |
| 54 to 56 | P17_2 to P17_4 | I/O | A | General-purpose input/output ports |
| | PPG2 to PPG4 | | | Output pins of PPG timer |
| 57 | P18_0 | I/O | A | General-purpose input/output port |
| | SIN6 | | | Data input pin of USART6 |
| 58 | P18_1 | I/O | A | General-purpose input/output port |
| | SOT6 | | | Data output pin of USART6 |
| 59 | P18_2 | I/O | A | General-purpose input/output port |
| | SCK6 | | | Clock input/output pin of USART6 |
| | FRCK6 | | | Free Run Timer external clock input pin |
| 60 | P18_4 | I/O | A | General-purpose input/output port |
| | SIN7 | | | Data input pin of USART7 |
| 61 | P18_5 | I/O | A | General-purpose input/output port |
| | SOT7 | | | Data output pin of USART7 |
| 62 | P18_6 | I/O | A | General-purpose input/output port |
| | SCK7 | | | Clock input/output pin of USART7 |
| | FRCK7 | | | Free Run Timer external clock input pin |
| 65 | P17_5 | I/O | A | General-purpose input/output port |
| | PPG5/<br>MONCLK | | | Output pin of PPG timer/<br>Clock monitor pin |
| 66, 67 | P17_6, P17_7 | I/O | A | General-purpose input/output ports |
| | PPG6, PPG7 | | | Output pins of PPG timer |
| 68 to 71 | P15_0 to P15_3 | I/O | A | General-purpose input/output ports |
| | OCU0 to OCU3 | | | Output compare output pins |
| | TOT0 to TOT3 | | | Reload timer output pins |
| 72 | P22_0 | I/O | A | General-purpose input/output port |
| | RX4 | | | RX input/output pin of CAN4 |
| | INT12 | | | External Interrupt input |

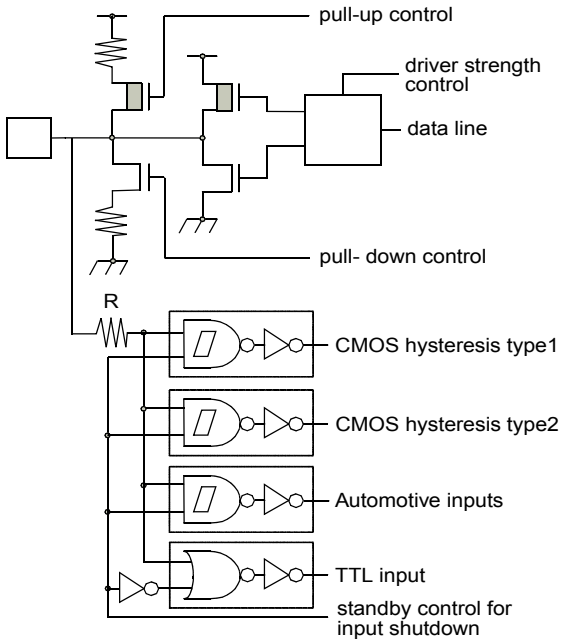
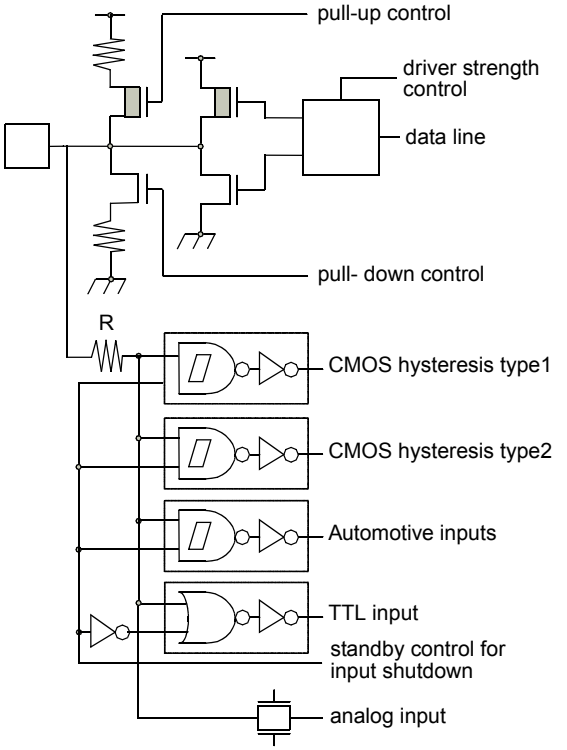
| Pin no. | Pin name | I/O | I/O circuit type* | Function |
|---|---|---|---|---|
| 73 | P22_1 | I/O | A | General-purpose input/output port |
| | TX4 | | | TX output pin of CAN4 |
| 74 to 83 | P24_0 to P24_7 | I/O | A | General-purpose input/output ports |
| | INT0 to INT7 | | | External interrupt input pins |
| 84 to 90 | P14_0 to P14_6 | I/O | A | General-purpose input/output ports |
| | ICU0 to ICU6 | | | Input capture input pins |
| | TIN0 to TIN6 | | | External trigger input pins of reload timer |
| | TTG8/0, TTG9/1 to TTG14/6 | | | External trigger input pins of PPG timer |
| 91 | P14_7 | I/O | A | General-purpose input/output port |
| | ICU7 | | | Input capture input pin |
| | TIN7 | | | External trigger input pins of reload timer |
| | TTG15/7 | | | External trigger input pins of PPG timer |
| | STOPWT | | | FlexRay Stop Watch input |
| 92 | MD_0 | I | G | Mode setting pin |
| 93 | X0A | — | J2 | Sub clock (oscillation) input |
| 94 | X1A | — | J2 | Sub clock (oscillation) output |
| 96 | X1 | — | J1 | Clock (oscillation) output |
| 97 | X0 | — | J1 | Clock (oscillation) input |
| 98 | MD_1 | I | G | Mode setting pin |
| 99 | MD_2 | I | G | Mode setting pin |

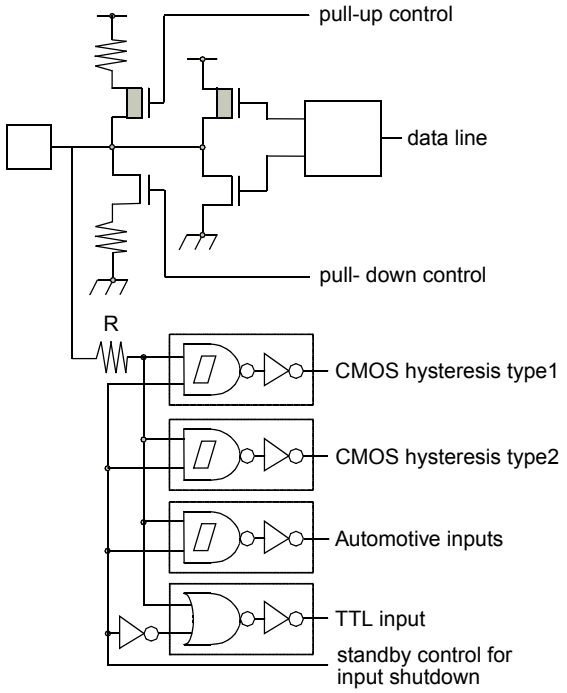
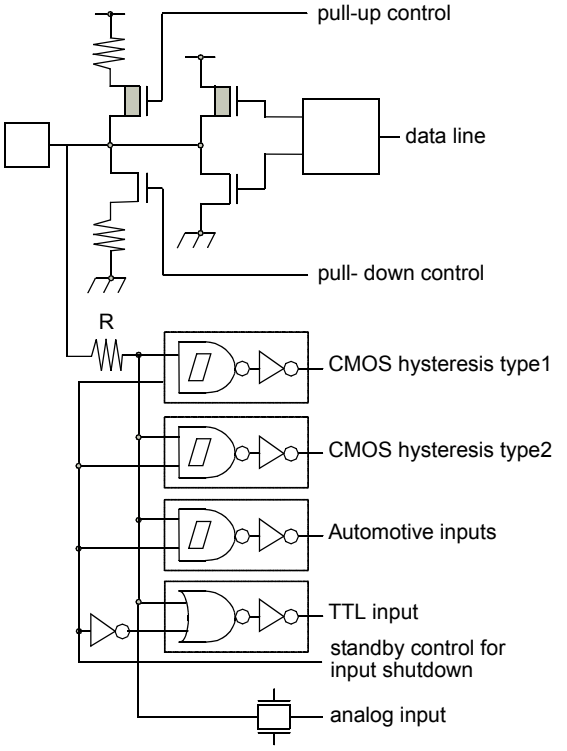
* : For information about the I/O circuit type, refer to “9. I/O Circuit Types”.

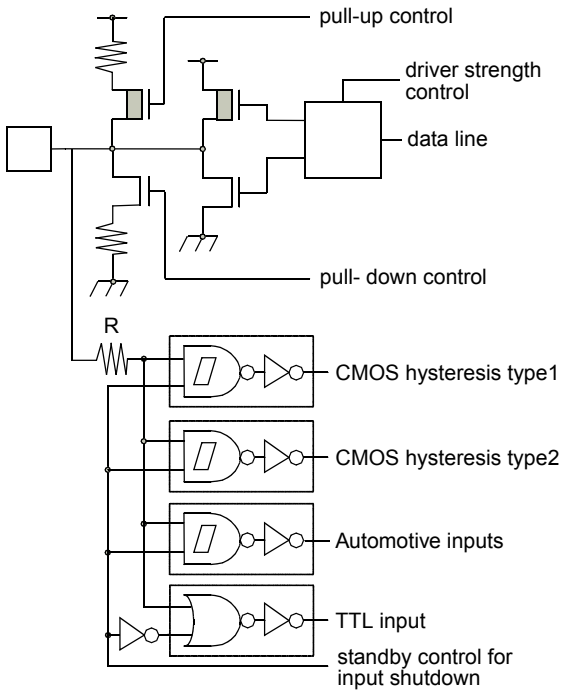
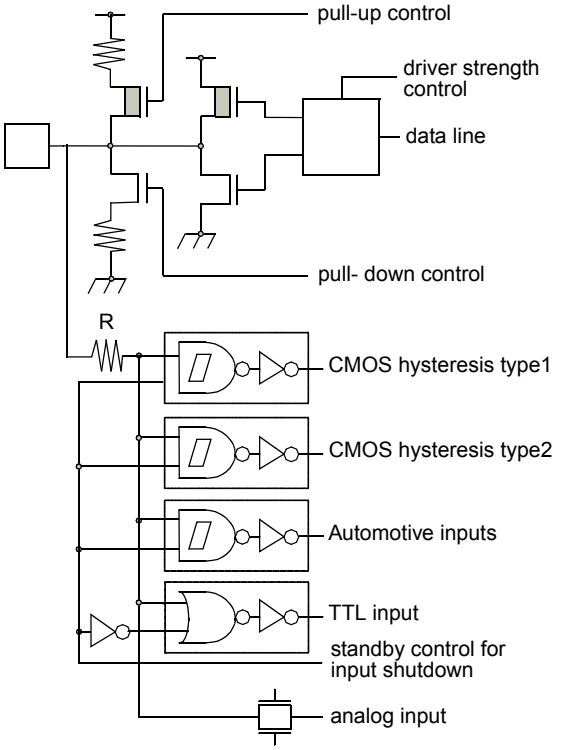
**[Power supply/Ground pins]**

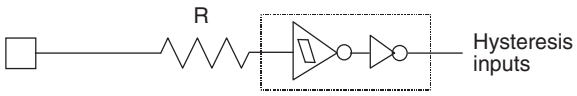
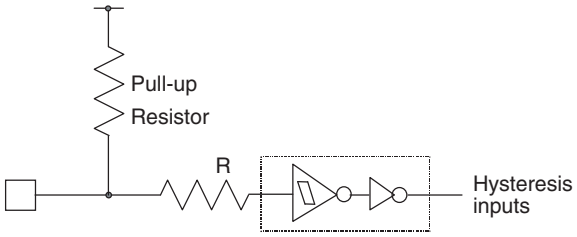
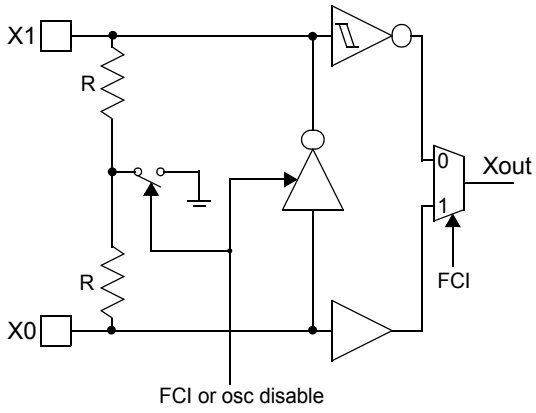
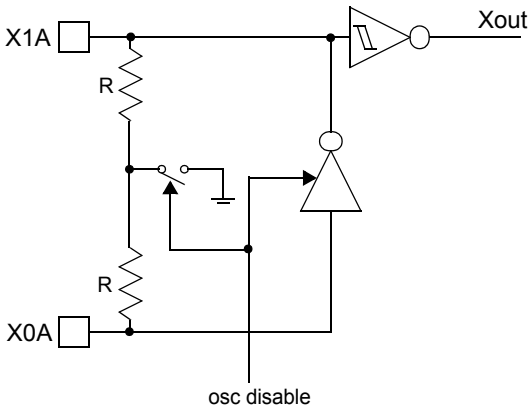
| Pin no. | Pin name | I/O | Function |
|---|---|---|---|
| 14, 26, 42, 51, 64, 75, 95, 100 | VSS5 | Supply | Ground pins |
| 1, 15, 25, 50, 63, 76 | VDD5 | | Power supply pins |
| 12 | VDD5R | | Power supply pins for internal regulator |
| 33 | AVSS5 | | Analog ground pin for A/D converter |
| 31 | AVCC5 | | Power supply pin for A/D converter |
| 32 | AVRH5 | | Reference power supply pin for A/D converter |
| 13 | VCC18C | | Capacitor connection pin for internal regulator |

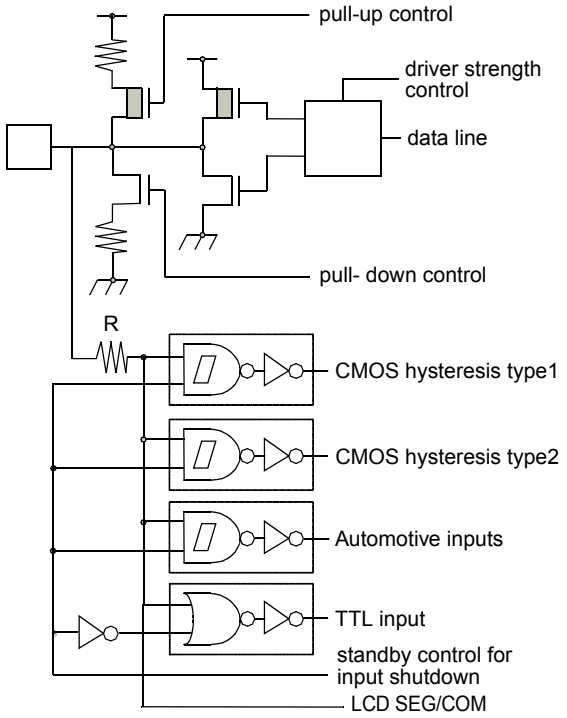
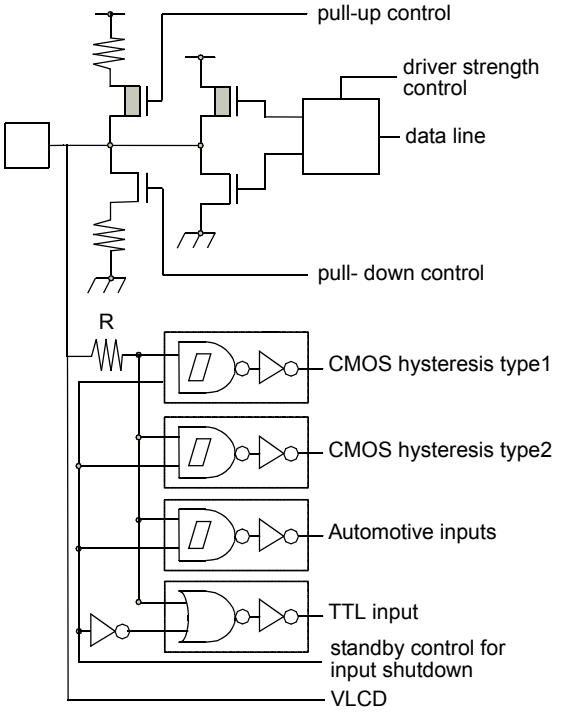
## 4. I/O Circuit Types

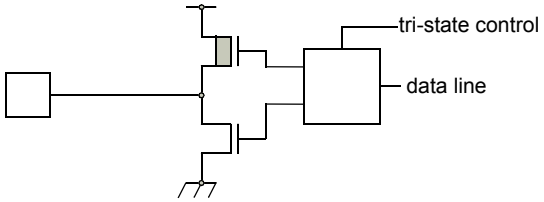
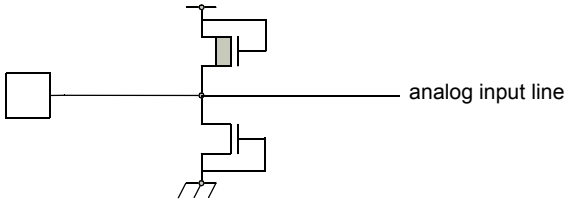
| Type | Circuit | Remarks |
|---|---|---|
| A |  | CMOS level output<br>(programmable $I_{OL}$ = 5mA, $I_{OH}$ = -5mA<br>and $I_{OL}$ = 2mA, $I_{OH}$ = -2mA)<br>2 different CMOS hysteresis inputs with input shutdown function<br>Automotive input with input shutdown function<br>TTL input with input shutdown function<br>Programmable pull-up resistor: 50kΩ approx. |
| B |  | CMOS level output<br>(programmable $I_{OL}$ = 5mA, $I_{OH}$ = -5mA<br>and $I_{OL}$ = 2mA, $I_{OH}$ = -2mA)<br>2 different CMOS hysteresis inputs with input shutdown function<br>Automotive input with input shutdown function)<br>TTL input with input shutdown function<br>Programmable pull-up resistor: 50kΩ approx.<br>Analog input |

| Type | Circuit | Remarks |
|---|---|---|
| C |  | CMOS level output ($I_{OL}$ = 3mA, $I_{OH}$ = -3mA)<br>2 different CMOS hysteresis inputs with input shutdown function<br>Automotive input with input shutdown function<br>TTL input with input shutdown function<br>Programmable pull-up resistor: 50kΩ approx. |
| D |  | CMOS level output ($I_{OL}$ = 3mA, $I_{OH}$ = -3mA)<br>2 different CMOS hysteresis inputs with input shutdown function<br>Automotive input with input shutdown function<br>TTL input with input shutdown function<br>Programmable pull-up resistor: 50kΩ approx.<br>Analog input |

| Type | Circuit | Remarks |
|---|---|---|
| E |  | CMOS level output<br>(programmable $I_{OL}$ = 5mA, $I_{OH}$ = -5mA<br>and $I_{OL}$ = 2mA, $I_{OH}$ = -2mA,<br>and $I_{OL}$ = 30mA, $I_{OH}$ = -30mA)<br>2 different CMOS hysteresis inputs with input shutdown function<br>Automotive input with input shutdown function<br>TTL input with input shutdown function<br>Programmable pull-up resistor: 50kΩ approx. |
| F |  | CMOS level output<br>(programmable $I_{OL}$ = 5mA, $I_{OH}$ = -5mA<br>and $I_{OL}$ = 2mA, $I_{OH}$ = -2mA,<br>and $I_{OL}$ = 30mA, $I_{OH}$ = -30mA)<br>2 different CMOS hysteresis inputs with input shutdown function<br>Automotive input with input shutdown function<br>TTL input with input shutdown function<br>Programmable pull-up resistor: 50kΩ approx.<br>Analog input |

| Type | Circuit | Remarks |
|---|---|---|
| G |  | Mask ROM and EVA device:<br>CMOS Hysteresis input pin<br>Flash device:<br>CMOS input pin<br>12 V withstand (for MD [2:0]) |
| H |  | CMOS Hysteresis input pin<br>Pull-up resistor value: 50 kΩ approx. |
| J1 |  | High-speed oscillation circuit:<br>• Programmable between oscillation mode (external crystal or resonator connected to X0/X1 pins) and Fast external Clock Input (FCI) mode (external clock connected to X0 pin)<br>• Feedback resistor = approx. 2 * 0.5 MΩ. Feedback resistor is grounded in the center when the oscillator is disabled or in FCI mode. |
| J2 |  | Low-speed oscillation circuit:<br>• Feedback resistor = approx. 2 * 5 MΩ. Feedback resistor is grounded in the center when the oscillator is disabled. |

| Type | Circuit | Remarks |
|---|---|---|
| K |  | CMOS level output<br>(programmable $I_{OL}$ = 5mA, $I_{OH}$ = -5mA<br>and $I_{OL}$ = 2mA, $I_{OH}$ = -2mA)<br>2 different CMOS hysteresis inputs with input shutdown function<br>Automotive input with input shutdown function<br>TTL input with input shutdown function<br>Programmable pull-up resistor: 50kΩ approx.<br>LCD SEG/COM output |
| L |  | CMOS level output<br>(programmable $I_{OL}$ = 5mA, $I_{OH}$ = -5mA<br>and $I_{OL}$ = 2mA, $I_{OH}$ = -2mA)<br>2 different CMOS hysteresis inputs with input shutdown function<br>Automotive input with input shutdown function)<br>TTL input with input shutdown function<br>Programmable pull-up resistor: 50kΩ approx.<br>Analog input<br>LCD Voltage input |

| Type | Circuit | Remarks |
|---|---|---|
| M |  | CMOS level tri-state output<br>($I_{OL}$ = 5mA, $I_{OH}$ = -5mA) |
| N |  | Analog input pin with protection |

# 5. Handling Devices

## 5.1. Preventing Latch-up

Latch-up may occur in a CMOS IC if a voltage higher than ($V_{DD}5$) or less than ($V_{SS}5$) is applied to an input or output pin or if a voltage exceeding the rating is applied between the power supply pins and ground pins. If latch-up occurs, the power supply current increases rapidly, sometimes resulting in thermal breakdown of the device. Therefore, be very careful not to apply voltages in excess of the absolute maximum ratings.

## 5.2. Handling of unused input pins

If unused input pins are left open, abnormal operation may result. Any unused input pins should be connected to pull-up or pull-down resistor (2KΩ to 10KΩ) or enable internal pullup or pulldown resisters (PPER/PPCR) before the input enable (PORTEN) is activated by software. The mode pins MD_x can be connected to $V_{SS}5$ or $V_{DD}5$ directly. Unused ALARM input pins can be connected to $AV_{SS}5$ directly.

## 5.3. Power supply pins

In MB91460X series, devices including multiple power supply pins and ground pins are designed as follows; pins necessary to be at the same potential are interconnected internally to prevent malfunctions such as latch-up. All of the power supply pins and ground pins must be externally connected to the power supply and ground respectively in order to reduce unnecessary radiation, to prevent strobe signal malfunctions due to the ground level rising and to follow the total output current ratings. Furthermore, the power supply pins and ground pins of the MB91460X series must be connected to the current supply source via a low impedance.
It is also recommended to connect a ceramic capacitor of approximately 0.1 µF as a bypass capacitor between power supply pin and ground pin near this device.
This series has a built-in step-down regulator. Connect a bypass capacitor of 4.7 µF (use a X7R ceramic capacitor) to VCC18C pin for the regulator.

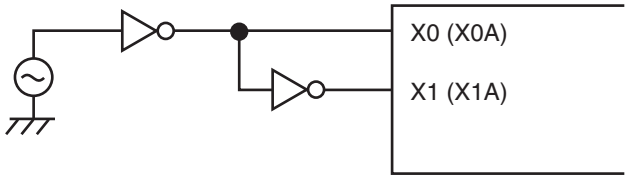
## 5.4. Crystal oscillator circuit

Noise in proximity to the X0 (X0A) and X1 (X1A) pins can cause the device to operate abnormally. Printed circuit boards should be designed so that the X0 (X0A) and X1 (X1A) pins, and crystal oscillator, as well as bypass capacitors connected to ground, are located near the device and ground.
It is recommended that the printed circuit board layout be designed such that the X0 and X1 pins or X0A and X1A pins are surrounded by ground plane for the stable operation.
Please request the oscillator manufacturer to evaluate the oscillational characteristics of the crystal and this device.

## 5.5. Notes on using external clock

When using the external clock, it is necessary to simultaneously supply the X0 (X0A) and the X1 (X1A) pins. In the described combination, X1 (X1A) should be supplied with a clock signal which has the opposite phase to the X0 (X0A) pins. At X0 and X1, a frequency up to 16 MHz is possible.

### Example of using opposite phase supply




*(Continued)*

*(Continued)*

### 5.6. Mode pins (MD_x)

These pins should be connected directly to the power supply or ground pins. To prevent the device from entering test mode accidentally due to noise, minimize the lengths of the patterns between each mode pin and power supply pin or ground pin on the printed circuit board as possible and connect them with low impedance.

### 5.7. Notes on operating in PLL clock mode

If the oscillator is disconnected or the clock input stops when the PLL clock is selected, the microcontroller may continue to operate at the free-running frequency of the self-oscillating circuit of the PLL. However, this self-running operation cannot be guaranteed.

### 5.8. Pull-up control

The AC standard is not guaranteed in case a pull-up resistor is connected to the pin serving as an external bus pin.

### 5.9. Notes on PS register

As the PS register is processed in advance by some instructions, when the debugger is being used, the exception handling may result in execution breaking in an interrupt handling routine or the displayed values of the flags in the PS register being updated.
As the microcontroller is designed to carry out reprocessing correctly upon returning from such an EIT event,
the operation before and after the EIT always proceeds according to specification.

**The following behavior may occur if any of the following occurs in the instruction immediately after a DIV0U/DIV0S instruction:**

a. a user interrupt or NMI is accepted;
b. single-step execution is performed;
c. execution breaks due to a data event or from the emulator menu.
    1. D0 and D1 flags are updated in advance.
    2. An EIT handling routine (user interrupt/NMI or emulator) is executed.
    3. Upon returning from the EIT, the DIV0U/DIV0S instruction is executed
       and the D0 and D1 flags are updated to the same values as those in 1.

**The following behavior occurs when an ORCCR, STILM, MOV Ri,PS instruction is executed to enable a user interrupt or NMI source while that interrupt is in the active state.**

1. The PS register is updated in advance.
2. An EIT handling routine (user interrupt/NMI or emulator) is executed.
3. Upon returning from the EIT, the above instructions are executed and the PS register is updated to the same value as in 1.

# 6. Notes on Debugger

## 6.1. Execution of the RETI Command

If single-step execution is used in an environment where an interrupt occurs frequently, the corresponding interrupt handling routine will be executed repeatedly to the exclusion of other processing. This will prevent the main routine and the handlers for low priority level interrupts from being executed (For example, if the time-base timer interrupt is enabled, stepping over the RETI instruction will always break on the first line of the time-base timer interrupt handler).

Disable the corresponding interrupts when the corresponding interrupt handling routine no longer needs debugging.

## 6.2. Break function

If the range of addresses that cause a hardware break (including event breaks) is set to the address of the current system stack pointer or to an area that contains the stack pointer, execution will break after each instruction regardless of whether the user program actually contains data access instructions.

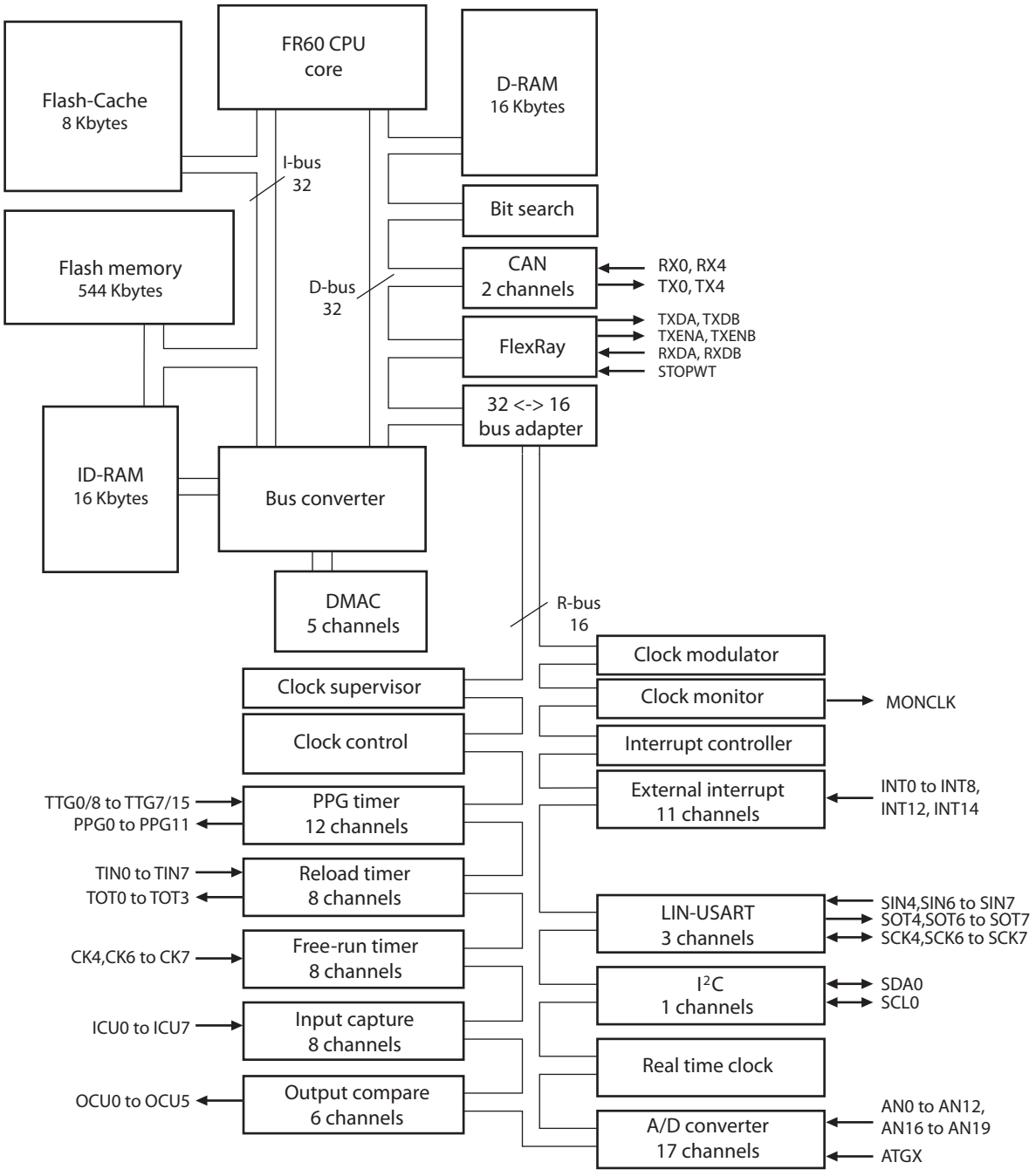
To prevent this, do not set (word) access to the area containing the address of the system stack pointer as the target of the hardware break (including an event breaks).

## 6.3. Operand break

It may cause malfunctions if a stack pointer exists in the area which is set as the DSU operand break. Do not set the access to the areas containing the address of system stack pointer as a target of data event break.

# 7. Block Diagram

## 7.1. MB91F465XA



# 8. CPU and Control Unit

The FR family CPU is a high performance core that is designed based on the RISC architecture with advanced instructions for embedded applications.

## 8.1. Features

- Adoption of RISC architecture
  Basic instruction: 1 instruction per cycle
- General-purpose registers: 32-bit × 16 registers
- 4 Gbytes linear memory space
- Multiplier installed
  32-bit × 32-bit multiplication: 5 cycles
  16-bit × 16-bit multiplication: 3 cycles
- Enhanced interrupt processing function
  Quick response speed (6 cycles)
  Multiple-interrupt support
  Level mask function (16 levels)
- Enhanced instructions for I/O operation
  Memory-to-memory transfer instruction
  Bit processing instruction
  Basic instruction word length: 16 bits
- Low-power consumption
  Sleep mode/stop mode

## 8.2. Internal architecture

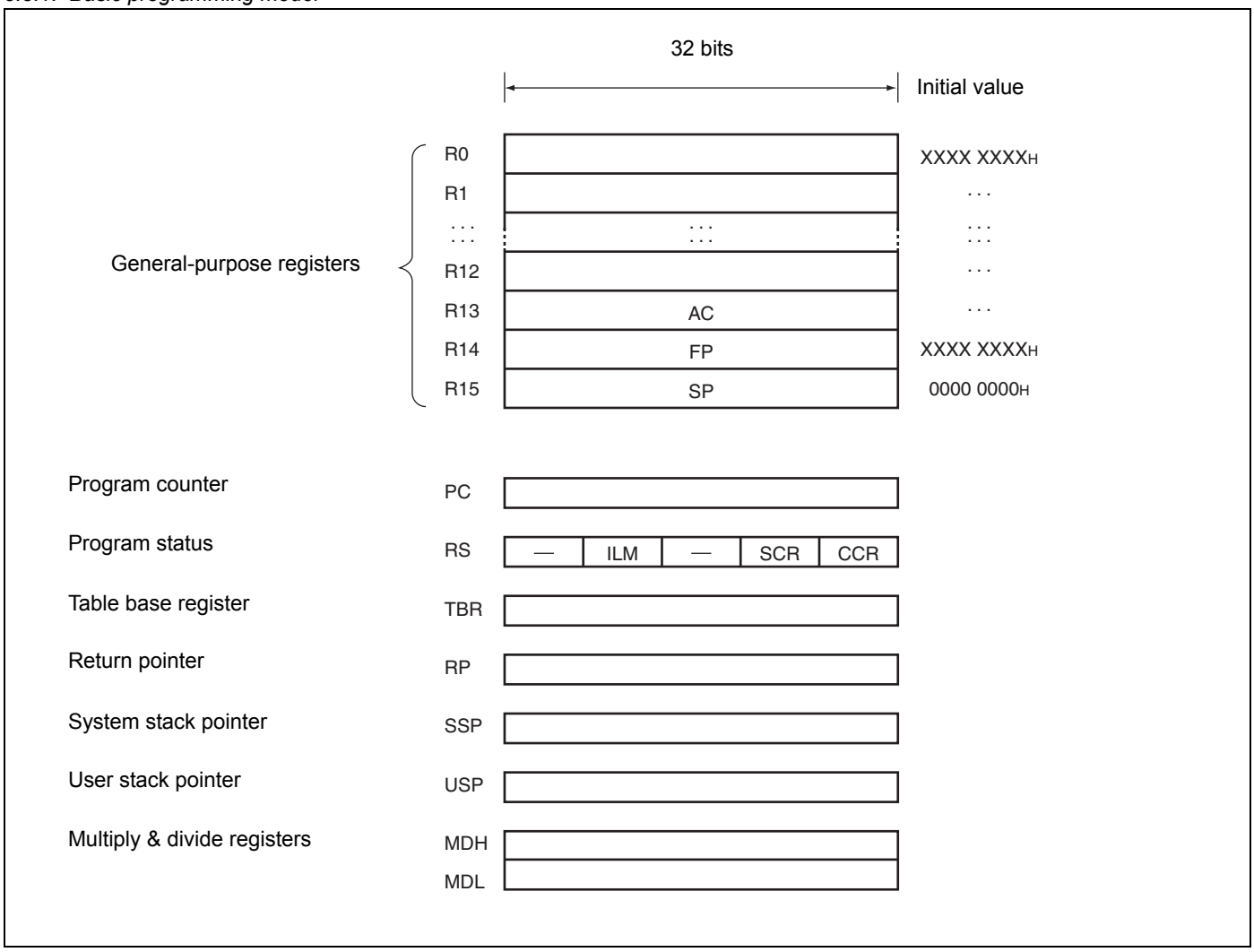
■ The FR family CPU uses the Harvard architecture in which the instruction bus and data bus are independent of each other.

■ A 32-bit ↔ 16-bit buffer is connected to the 32-bit bus (D-bus) to provide an interface between the CPU and peripheral resources.

■ A Harvard ↔ Princeton bus converter is connected to both the I-bus and D-bus to provide an interface between the CPU and the bus controller.

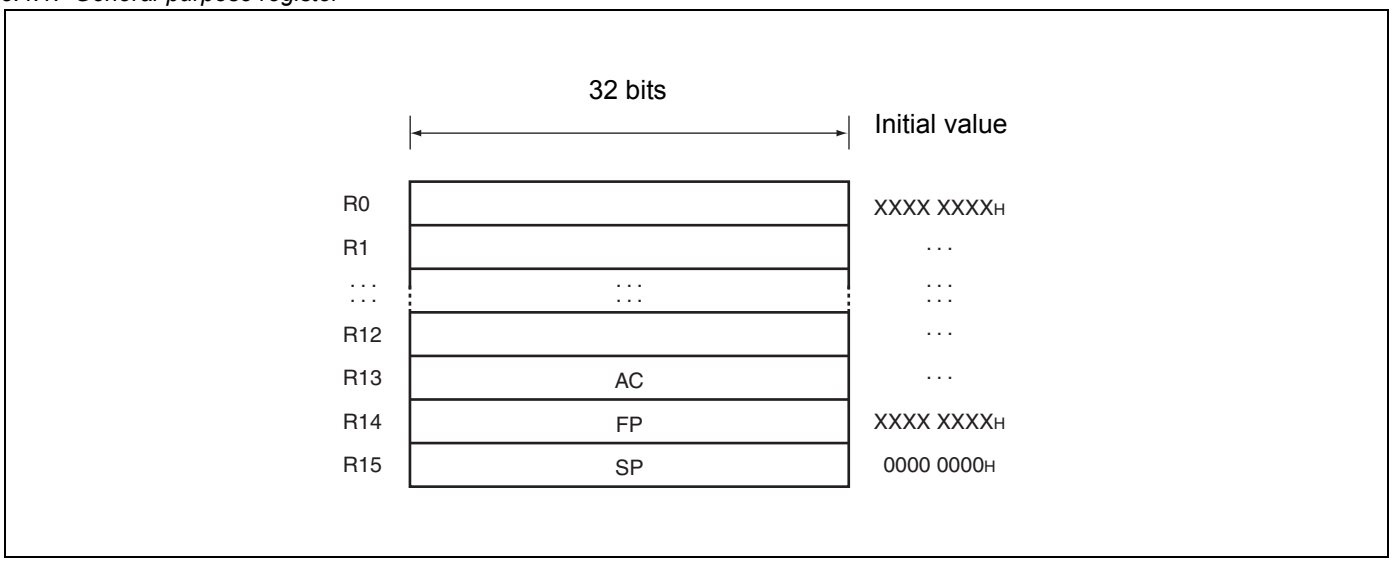
## 8.3. Programming model

### *8.3.1. Basic programming model*



## 8.4. Registers

### *8.4.1. General-purpose register*




Registers R0 to R15 are general-purpose registers. These registers can be used as accumulators for computation operations and as pointers for memory access.

Of the 16 registers, enhanced commands are provided for the following registers to enable their use for particular applications.

R13 : Virtual accumulator

R14 : Frame pointer

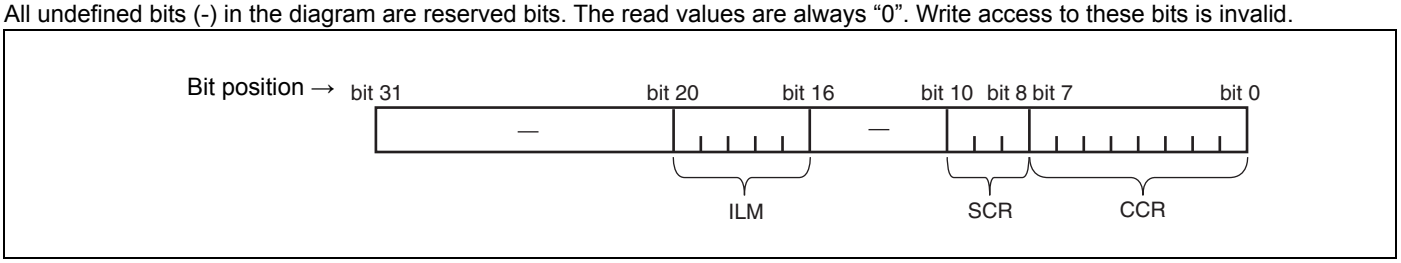
R15 : Stack pointer

Initial values at reset are undefined for R0 to R14. The value for R15 is $00000000_H$ (SSP value).

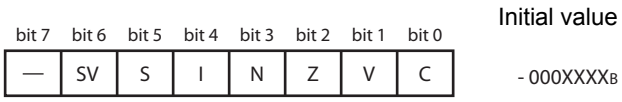
### *8.4.2. PS (Program Status)*

This register holds the program status, and is divided into three parts, ILM, SCR, and CCR.

All undefined bits (-) in the diagram are reserved bits. The read values are always “0”. Write access to these bits is invalid.

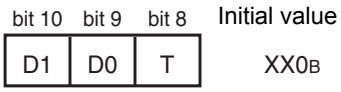


### 8.4.3. CCR (Condition Code Register)




SV: Supervisor flag
S : Stack flag
I : Interrupt enable flag
N : Negative enable flag
Z : Zero flag
V : Overflow flag
C : Carry flag

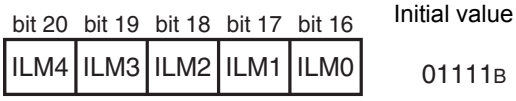
### 8.4.4. SCR (System Condition Register)




Flag for step division (D1, D0)
This flag stores interim data during execution of step division.

Step trace trap flag (T)
This flag indicates whether the step trace trap is enabled or disabled.
The step trace trap function is used by emulators. When an emulator is in use, it cannot be used in execution of user programs.

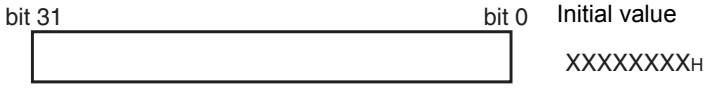
### 8.4.5. ILM (Interrupt Level Mask register)




This register stores interrupt level mask values, and the values stored in ILM4 to ILM0 are used for level masking.

The register is initialized to value "$01111_B$" at reset.

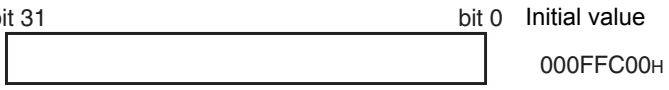
### 8.4.6. PC (Program Counter)




The program counter indicates the address of the instruction that is being executed.

The initial value at reset is undefined.

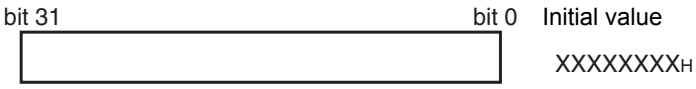
### 8.4.7. TBR (Table Base Register)




The table base register stores the starting address of the vector table used in EIT processing.

The initial value at reset is $000FFC00_H$.

### 8.4.8. RP (Return Pointer)



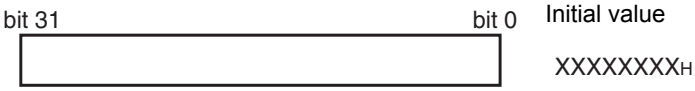

The return pointer stores the address for return from subroutines.

During execution of a CALL instruction, the PC value is transferred to this RP register.

During execution of a RET instruction, the contents of the RP register are transferred to PC.

The initial value at reset is undefined.

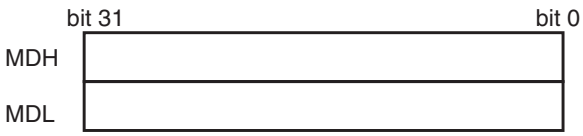
### 8.4.9. USP (User Stack Pointer)




The user stack pointer, when the S flag is “1”, this register functions as the R15 register.

- The USP register can also be explicitly specified.
  The initial value at reset is undefined.
- This register cannot be used with RETI instructions.

### 8.4.10. Multiply & divide registers




These registers are for multiplication and division, and are each 32 bits in length.

The initial value at reset is undefined.

# 9. Embedded Program/Data Memory (Flash)

## 9.1. Flash features

- 544 Kbytes (8 × 64 Kbytes + 4 × 8 Kbytes = 4.25 Mbits)
- Programmable wait state for read/write access
- Flash and Boot security with security vector at 0x0014:8000 - 0x0014:800F
- Boot security
- Basic specification: Same as MBM29LV400TC (except size and part of sector configuration)

## 9.2. Operation modes

1. 32-bit CPU mode :
   - CPU reads and executes programs in word (32-bit) length units.
   - Flash writing is not possible
   - Actual Flash Memory access is performed in word (32-bit) length units.
2. 16-bit CPU mode :
   - CPU reads and writes in half-word (16-bit) length units.
   - Program execution from the Flash is not possible.
   - Actual Flash Memory access is performed in half-word (16-bit) length units.
3. Flash memory mode (external access to Flash memory enabled)

Note: The operation mode of the flash memory can be selected using a Boot-ROM function. The function start address is 0xBF60. The parameter description is given in the Hardware Manual in chapter 54.6 “Flash Access Mode Switching”.

## 9.3. Flash access in CPU mode

### *9.3.1. Flash configuration*

*Flash memory map MB91F465XA*

| Addr | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0014:FFFFh<br>0014:C000h | SA6 (8KB) | | | | SA7 (8KB) | | | | ROMS7 |
| 0014:BFFFh<br>0014:8000h | SA4 (8KB) | | | | SA5 (8KB) | | | | |
| 0014:7FFFh<br>0014:4000h | SA2 (8KB) | | | | SA3 (8KB) | | | | |
| 0014:3FFFh<br>0014:0000h | SA0 (8KB) | | | | SA1 (8KB) | | | | |
| 0013:FFFFh<br>0012:0000h | SA22 (64KB) | | | | SA23 (64KB) | | | | ROMS6 |
| 0011:FFFFh<br>0010:0000h | SA20 (64KB) | | | | SA21 (64KB) | | | | |
| 000F:FFFFh<br>000E:0000h | SA18 (64KB) | | | | SA19 (64KB) | | | | ROMS5 |
| 000D:FFFFh<br>000C:0000h | SA16 (64KB) | | | | SA17 (64KB) | | | | ROMS4 |
| 000B:FFFFh<br>000A:0000h | SA14 (64KB) | | | | SA15 (64KB) | | | | ROMS3 |
| 0009:FFFFh<br>0008:0000h | SA12 (64KB) | | | | SA13 (64KB) | | | | ROMS2 |
| 0007:FFFFh<br>0006:0000h | SA10 (64KB) | | | | SA11 (64KB) | | | | ROMS1 |
| 0005:FFFFh<br>0004:0000h | SA8 (64KB) | | | | SA9 (64KB) | | | | ROMS0 |
| | addr+0 | addr+1 | addr+2 | addr+3 | addr+4 | addr+5 | addr+6 | addr+7 | |
| 16bit read/write | dat[31:16] | | dat[15:0] | | dat[31:16] | | dat[15:0] | | |
| 32bit read | dat[31:0] | | | | dat[31:0] | | | | |
| Legend | Memory not available in this area | | | | Memory available in this area | | | | |

### 9.3.2. Flash access timing settings in CPU mode

The following tables list all settings for a given maximum Core Frequency (through the setting of CLKB or maximum clock modulation) for Flash read and write access.

*Flash read timing settings (synchronous read)*

| Core clock (CLKB) | ATD | ALEH | EQ | WEXH | WTC | Remark |
|---|---|---|---|---|---|---|
| to 24 MHz | 0 | 0 | 0 | - | 1 | |
| to 48 MHz | 0 | 0 | 1 | - | 2 | |
| to 100 MHz | 1 | 1 | 3 | - | 4 | |

*Flash write timing settings (synchronous write)*

| Core clock (CLKB) | ATD | ALEH | EQ | WEXH | WTC | Remark |
|---|---|---|---|---|---|---|
| to 16 MHz | 0 | - | - | 0 | 3 | |
| to 32 MHz | 0 | - | - | 0 | 4 | |
| to 48 MHz | 0 | - | - | 0 | 5 | |
| to 64 MHz | 1 | - | - | 0 | 6 | |
| to 96 MHz | 1 | - | - | 0 | 7 | |
| to 100 MHz | 1 | - | - | 1 | 8 | |

### *9.3.3. Address mapping from CPU to parallel programming mode*

The following tables show the calculation from CPU addresses to flash macro addresses which are used in parallel programming.

*Address mapping MB91F465XA*

| CPU Address (addr) | Condition | Flash sectors | FA (flash address) Calculation |
|---|---|---|---|
| 14:8000h to 14:FFFFh | addr[2]==0 | SA4, SA6 (8 Kbyte) | FA := addr - addr%00:4000h + (addr%00:4000h)/2 - (addr/2)%4 + addr%4 - 0D:0000h |
| 14:8000h to 14:FFFFh | addr[2]==1 | SA5, SA7 (8 Kbyte) | FA := addr - addr%00:4000h + (addr%00:4000h)/2 + 00:2000h - (addr/2)%4 + addr%4 - 0D:0000h |
| 08:0000h to 13:FFFFh | addr[2]==0 | SA12, SA14, SA16, SA18 (64 Kbyte) | FA := addr - addr%02:0000 + (addr%02:0000h)/2 - (addr/2)%4 + addr%4 |
| 08:0000h to 13:FFFFh | addr[2]==1 | SA13, SA15, SA17, SA19 (64 Kbyte) | FA := addr - addr%02:0000h + (addr%02:0000h)/2 + 01:0000h - (addr/2)%4 + addr%4 |

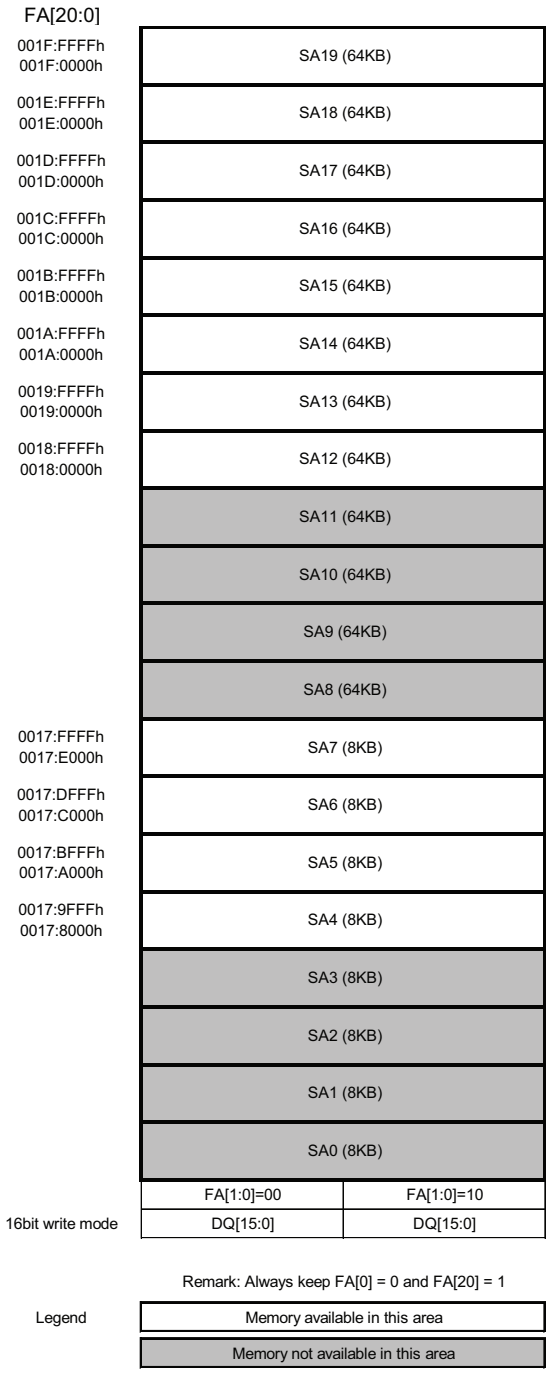
Note: FA result is without 10:0000h offset for parallel Flash programming.
Set offset by keeping FA[20] = 1 as described in section “Parallel Flash programming mode”.

## 9.4. Parallel Flash programming mode

### *9.4.1. Flash configuration in parallel Flash programming mode*

Parallel Flash programming mode (MD[2:0] = 111):

MB91F465XA




Remark: Always keep FA[0] = 0 and FA[21] = 1

### 9.4.2. Pin connections in parallel programming mode

Resetting after setting the MD[2:0] pins to [111] will halt CPU functioning. At this time, the Flash memory's interface circuit enables direct control of the Flash memory unit from external pins by directly linking some of the signals to General Purpose Ports. Please see table below for signal mapping.

In this mode, the Flash memory appears to the external pins as a stand-alone unit. This mode is generally set when writing/erasing using the parallel Flash programmer. In this mode, all operations of the 8.5 Mbits Flash memory's Auto Algorithms are available.

Correspondence between MBM29LV400TC and Flash Memory Control Signals

| MBM29LV400TC External pins | FR-CPU mode | MB91F465XA external pins | | | Comment |
|---|---|---|---|---|---|
| | | Flash memory mode | Normal function | Pin number | |
| - | INITX | - | INITX | 52 | |
| RESET | - | FRSTX | P16_7 | 53 | |
| - | - | MD2 | MD_2 | 99 | Set to '1' |
| - | - | MD1 | MD_1 | 98 | Set to '1' |
| - | - | MD0 | MD_0 | 92 | Set to '1' |
| RY/BY | FMCS:RDY bit | RY/BYX | P24_0 | 74 | |
| BYTE | Internally fixed to 'H' | BYTEX | P24_2 | 78 | |
| WE | Internal control signal + control via interface circuit | WEX | P28_3 | 29 | |
| OE | | OEX | P28_2 | 28 | |
| CE | | CEX | P28_1 | 27 | |
| - | | ATDIN | P22_1 | 73 | Set to '0' |
| - | | EQIN | P22_0 | 72 | Set to '0' |
| - | | TESTX | P24_3 | 79 | Set to '1' |
| - | | RDYI | P24_1 | 77 | Set to '0' |
| A-1 | Internal address bus | FA0 | P19_2 | 47 | Set to '0' |
| A0 to A7 | | FA1 to FA8 | P16_0 to P16_3, P27_0 to P27_3 | 16 to 23 | |
| A8 to A15 | | FA9 to FA16 | P15_0 to P15_5, P18_0, P18_1 | 68 to 71,10, 11, 57, 58 | |
| A16 to A18 | | FA17 to FA19 | P18_2, P18_4, P18_5 | 59 to 61 | |
| A19 | | FA20 | P18_6 | 62 | Set to '1' |
| DQ0 to DQ7 | Internal data bus | DQ0 to DQ7 | P17_0 to P17_7 | 48, 49, 54 to 56, 65 to 67 | |
| DQ8 to DQ15 | | DQ8 to DQ15 | P23_0, P23_1, P31_0 to P31_2, P31_4 to P31_6 | 2 to 9 | |

## 9.5. Flash Security

### *9.5.1. Vector addresses*

Two Flash Security Vectors (FSV1, FSV2) are located parallel to the Boot Security Vectors (BSV1, BSV2) controlling the protection functions of the Flash Security Module:

FSV1: 0x14:8000 BSV1: 0x14:8004
FSV2: 0x14:8008 BSV2: 0x14:800C

### *9.5.2. Security Vector FSV1*

The setting of the Flash Security Vector FSV1 is responsible for the read and write protection modes and the individual write protection of the 8 Kbytes sectors.

*FSV1 (bit31 to bit16)*

The setting of the Flash Security Vector FSV1 bits [31:16] is responsible for the read and write protection modes.

Explanation of the bits in the Flash Security Vector FSV1 [31:16]

| FSV1[31:19] | FSV1[18] Write Protection Level | FSV1[17] Write Protection | FSV1[16] Read Protection | Flash Security Mode |
|---|---|---|---|---|
| set all to “0” | set to “0” | set to “0” | set to “1” | Read Protection (all device modes, except INTVEC mode MD[2:0] = “000”) |
| set all to “0” | set to “0” | set to “1” | set to “0” | Write Protection (all device modes, without exception) |
| set all to “0” | set to “0” | set to “1” | set to “1” | Read Protection (all device modes, except INTVEC mode MD[2:0] = “000”) and Write Protection (all device modes) |
| set all to “0” | set to “1” | set to “0” | set to “1” | Read Protection (all device modes, except INTVEC mode MD[2:0] = “000”) |
| set all to “0” | set to “1” | set to “1” | set to “0” | Write Protection (all device modes, except INTVEC mode MD[2:0] = “000”) |
| set all to “0” | set to “1” | set to “1” | set to “1” | Read Protection (all device modes, except INTVEC mode MD[2:0] = “000”) and Write Protection (all device modes except INTVEC mode MD[2:0] = “000”) |

*FSV1 (bit15 to bit0)*

The setting of the Flash Security Vector FSV1 bits [15:0] is responsible for the individual write protection of the 8 Kbytes sectors. It is only evaluated if write protection bit FSV1[17] is set.

Explanation of the bits in the Flash Security Vector FSV1 [15:0]

| FSV1 bit | Sector | Enable Write Protection | Disable Write Protection | Comment |
|---|---|---|---|---|
| FSV1[0] | — | set to “0” | set to “1” | not available |
| FSV1[1] | — | set to “0” | set to “1” | not available |
| FSV1[2] | — | set to “0” | set to “1” | not available |
| FSV1[3] | — | set to “0” | set to “1” | not available |
| FSV1[4] | SA4 | set to “0” | — | Write protection is mandatory! |
| FSV1[5] | SA5 | set to “0” | set to “1” | |

| FSV1 bit | Sector | Enable Write Protection | Disable Write Protection | Comment |
|---|---|---|---|---|
| FSV1[6] | SA6 | set to “0” | set to “1” | |
| FSV1[7] | SA7 | set to “0” | set to “1” | |
| FSV1[8] | — | set to “0” | set to “1” | not available |
| FSV1[9] | — | set to “0” | set to “1” | not available |
| FSV1[10] | — | set to “0” | set to “1” | not available |
| FSV1[11] | — | set to “0” | set to “1” | not available |
| FSV1[12] | — | set to “0” | set to “1” | not available |
| FSV1[13] | — | set to “0” | set to “1” | not available |
| FSV1[14] | — | set to “0” | set to “1” | not available |
| FSV1[15] | — | set to “0” | set to “1” | not available |

Note : It is mandatory to always set the sector where the Flash Security Vectors FSV1 and FSV2 are located to write protected (here sector SA4). Otherwise it is possible to overwrite the Security Vector to a setting where it is possible to either read out the Flash content or manipulate data by writing.
See section “Flash access in CPU mode” for an overview about the sector organization of the Flash Memory.

### *9.5.3. Security Vector FSV2*

The setting of the Flash Security Vector FSV2 bits [31:0] is responsible for the individual write protection of the 64 Kbytes sectors. It is only evaluated if write protection bit FSV1 [17] is set.

Explanation of the bits in the Flash Security Vector FSV2[31:0]

| FSV2 bit | Sector | Enable Write Protection | Disable Write Protection | Comment |
|---|---|---|---|---|
| FSV2[0] | — | set to “0” | set to “1” | not available |
| FSV2[1] | — | set to “0” | set to “1” | not available |
| FSV2[2] | — | set to “0” | set to “1” | not available |
| FSV2[3] | — | set to “0” | set to “1” | not available |
| FSV2[4] | SA12 | set to “0” | set to “1” | |
| FSV2[5] | SA13 | set to “0” | set to “1” | |
| FSV2[6] | SA14 | set to “0” | set to “1” | |
| FSV2[7] | SA15 | set to “0” | set to “1” | |
| FSV2[8] | SA16 | set to “0” | set to “1” | |
| FSV2[9] | SA17 | set to “0” | set to “1” | |
| FSV2[10] | SA18 | set to “0” | set to “1” | |
| FSV2[11] | SA19 | set to “0” | set to “1” | |
| FSV2[31:12] | — | set to “0” | set to “1” | not available |

Note : See section “Flash access in CPU mode” for an overview about the sector organization of the Flash Memory.

### 9.6. Notes About Flash Memory CRC Calculation

The Flash Security macro contains a feature to calculate the 32-bit checksum over addresses located in the Flash Memory address space. This feature is described in the MB91460 Series Hardware Manual, chapter 55.4.1 “Flash Security Control Register”.

Additional notes are given here:

The CRC calculation runs on the internal RC clock. It is recommended to switch the RC clock frequency to 2 MHz for shortening the calculation time. However, the CPU clock (CLKB) must be faster then RC clock, otherwise the CRC calculation may not start correctly.

# 10. Memory Space

The FR family has 4 Gbytes of logical address space ($2^{32}$ addresses) available to the CPU by linear access.

### Direct addressing area

The following address space area is used for I/O.

This area is called direct addressing area, and the address of an operand can be specified directly in an instruction.

The size of directly addressable area depends on the length of the data being accessed as shown below.

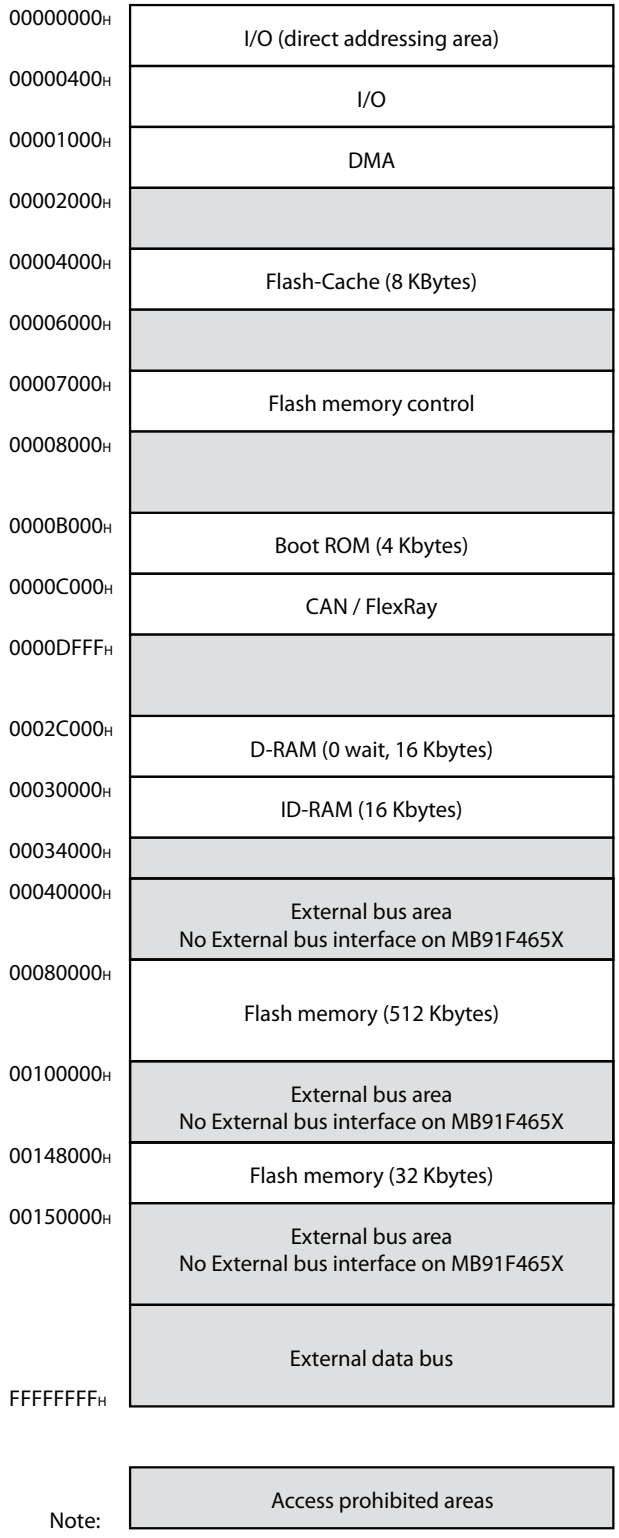
Byte data access : $000_H$ to $0FF_H$

Half word access : $000_H$ to $1FF_H$

Word data access : $000_H$ to $3FF_H$

# 11. Memory Maps

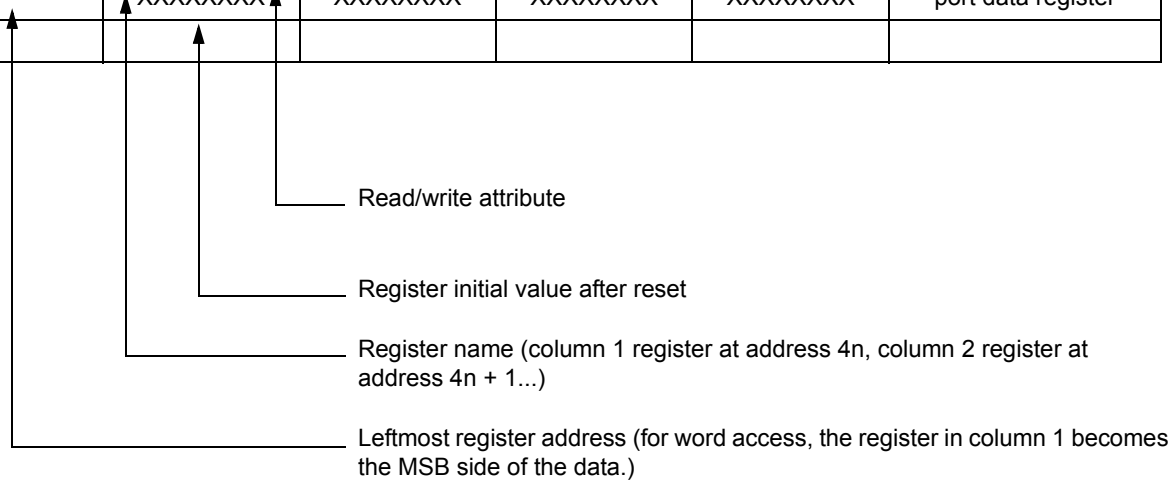
## 11.1. MB91F465XA



# 12. I/O Map

## 12.1. MB91F465XA

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | + 0 | + 1 | + 2 | + 3 | |
| $000000_H$ | PDR0 [R/W]<br>XXXXXXXX | PDR1 [R/W]<br>XXXXXXXX | PDR2 [R/W]<br>XXXXXXXX | PDR3 [R/W]<br>XXXXXXXX | T-unit<br>port data register |
| | | | | | |




Note : Initial values of register bits are represented as follows:

”1” : Initial value “1”

”0” : Initial value “0”

”X” : Initial value “undefined”

”-” : No physical register at this location

Access is barred with an undefined data access attribute.

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $000000_H$ to $000008_H$ | Reserved | | | | Reserved |
| $00000C_H$ | Reserved | Reserved | PDR14 [R/W] XXXXXXXX | PDR15 [R/W] - - XXXXXX | R-bus Port Data Register |
| $000010_H$ | PDR16 [R/W] X - - - XXXX | PDR17 [R/W] XXXXXXXX | PDR18 [R/W] - XXX - XXX | PDR19 [R/W] - - - - - XXX | |
| $000014_H$ | Reserved | Reserved | PDR22 [R/W] - - XX - - XX | PDR23 [R/W] - - - - - - XX | |
| $000018_H$ | PDR24 [R/W] XXXXXXXX | Reserved | Reserved | PDR27 [R/W] - - - - XXXX | |
| $00001C_H$ | PDR28 [R/W] - - - XXXXX | PDR29 [R/W] XXXXXXXX | Reserved | PDR31 [R/W] - XXX - XXX | |
| $000020_H$ to $00002C_H$ | Reserved | | | | Reserved |
| $000030_H$ | EIRR0 [R/W] XXXXXXXX | ENIR0 [R/W] 00000000 | ELVR0 [R/W] 00000000 00000000 | | External interrupt (INT 0 to INT 7) NMI |
| $000034_H$ | EIRR1 [R/W] XXXXXXXX | ENIR1 [R/W] 00000000 | ELVR1 [R/W] 00000000 00000000 | | External interrupt (INT 8 to INT 15) |
| $000038_H$ | DICR [R/W] - - - - - - - 0 | HRCL [R/W] 0 - - 11111 | RBSYNC | | Delay interrupt |
| $00003C_H$ | Reserved | | | | Reserved |
| $000040_H$ to $00005C_H$ | Reserved | | | | Reserved |
| $000060_H$ | SCR04 [R/W,W] 00000000 | SMR04 [R/W,W] 00000000 | SSR04 [R/W,R] 00001000 | RDR04/TDR04 [R/W] 00000000 | LIN-USART 4 with FIFO |
| $000064_H$ | ESCR04 [R/W] 00000X00 | ECCR04 [R/W,R,W] -00000XX | FSR04 [R] - - - 00000 | FCR04 [R/W] 0001 - 000 | |
| $000068_H$ to $00006C_H$ | Reserved | | | | Reserved |
| $000070_H$ | SCR06 [R/W,W] 00000000 | SMR06 [R/W,W] 00000000 | SSR06 [R/W,R] 00001000 | RDR06/TDR06 [R/W] 00000000 | LIN-USART 6 with FIFO |
| $000074_H$ | ESCR06 [R/W] 00000X00 | ECCR06 [R/W,R,W] -00000XX | FSR06 [R] - - - 00000 | FCR06 [R/W] 0001 - 000 | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $000078_H$ | SCR07 [R/W,W]<br>00000000 | SMR07 [R/W,W]<br>00000000 | SSR07 [R/W,R]<br>00001000 | RDR07/TDR07<br>[R/W]<br>00000000 | LIN-USART 7<br>with FIFO |
| $00007C_H$ | ESCR07 [R/W]<br>00000X00 | ECCR07<br>[R/W,R,W]<br>-00000XX | FSR07 [R]<br>- - - 00000 | FCR07 [R/W]<br>0001 - 000 | |
| $000080_H$<br>to<br>$000084_H$ | Reserved | | | | Reserved |
| $000088_H$ | BGR104 [R/W]<br>00000000 | BGR004 [R/W]<br>00000000 | Reserved | Reserved | Baud rate<br>Generator<br>LIN-USART 4, 6 to 7 |
| $00008C_H$ | BGR106 [R/W]<br>00000000 | BGR006 [R/W]<br>00000000 | BGR107 [R/W]<br>00000000 | BGR007 [R/W]<br>00000000 | |
| $000090_H$<br>to<br>$0000CC_H$ | Reserved | | | | Reserved |
| $0000D0_H$ | IBCR0 [R/W]<br>00000000 | IBSR0 [R]<br>00000000 | ITBAH0 [R/W]<br>- - - - - - 00 | ITBAL0 [R/W]<br>00000000 | $I^2C$ 0 |
| $0000D4_H$ | ITMKH0 [R/W]<br>00 - - - - 11 | ITMKL0 [R/W]<br>11111111 | ISMK0 [R/W]<br>01111111 | ISBA0 [R/W]<br>- 0000000 | |
| $0000D8_H$ | Reserved | IDAR0 [R/W]<br>00000000 | ICCR0 [R/W]<br>- 0011111 | Reserved | |
| $0000DC_H$<br>to<br>$0000FC_H$ | Reserved | | | | Reserved |
| $000100_H$ | GCN10 [R/W]<br>00110010 00010000 | | Reserved | GCN20 [R/W]<br>- - - - 0000 | PPG Control<br>0 to 3 |
| $000104_H$ | GCN11 [R/W]<br>00110010 00010000 | | Reserved | GCN21 [R/W]<br>- - - - 0000 | PPG Control<br>4 to 7 |
| $000108_H$ | GCN12 [R/W]<br>00110010 00010000 | | Reserved | GCN22 [R/W]<br>- - - - 0000 | PPG Control<br>8 to 11 |
| $000110_H$ | PTMR00 [R]<br>11111111 11111111 | | PCSR00 [W]<br>XXXXXXXX XXXXXXXX | | PPG 0 |
| $000114_H$ | PDUT00 [W]<br>XXXXXXXX XXXXXXXX | | PCNH00 [R/W]<br>0000000 - | PCNL00 [R/W]<br>000000 - 0 | |
| $000118_H$ | PTMR01 [R]<br>11111111 11111111 | | PCSR01 [W]<br>XXXXXXXX XXXXXXXX | | PPG 1 |
| $00011C_H$ | PDUT01 [W]<br>XXXXXXXX XXXXXXXX | | PCNH01 [R/W]<br>0000000 - | PCNL01 [R/W]<br>000000 - 0 | |
| $000120_H$ | PTMR02 [R]<br>11111111 11111111 | | PCSR02 [W]<br>XXXXXXXX XXXXXXXX | | PPG 2 |
| $000124_H$ | PDUT02 [W]<br>XXXXXXXX XXXXXXXX | | PCNH02 [R/W]<br>0000000 - | PCNL02 [R/W]<br>000000 - 0 | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $000128_H$ | PTMR03 [R]<br>11111111 11111111 | | PCSR03 [W]<br>XXXXXXXX XXXXXXXX | | PPG 3 |
| $00012C_H$ | PDUT03 [W]<br>XXXXXXXX XXXXXXXX | | PCNH03 [R/W]<br>0000000 - | PCNL03 [R/W]<br>000000 - 0 | |
| $000130_H$ | PTMR04 [R]<br>11111111 11111111 | | PCSR04 [W]<br>XXXXXXXX XXXXXXXX | | PPG 4 |
| $000134_H$ | PDUT04 [W]<br>XXXXXXXX XXXXXXXX | | PCNH04 [R/W]<br>0000000 - | PCNL04 [R/W]<br>000000 - 0 | |
| $000138_H$ | PTMR05 [R]<br>11111111 11111111 | | PCSR05 [W]<br>XXXXXXXX XXXXXXXX | | PPG 5 |
| $00013C_H$ | PDUT05 [W]<br>XXXXXXXX XXXXXXXX | | PCNH05 [R/W]<br>0000000 - | PCNL05 [R/W]<br>000000 - 0 | |
| $000140_H$ | PTMR06 [R]<br>11111111 11111111 | | PCSR06 [W]<br>XXXXXXXX XXXXXXXX | | PPG 6 |
| $000144_H$ | PDUT06 [W]<br>XXXXXXXX XXXXXXXX | | PCNH06 [R/W]<br>0000000 - | PCNL06 [R/W]<br>000000 - 0 | |
| $000148_H$ | PTMR07 [R]<br>11111111 11111111 | | PCSR07 [W]<br>XXXXXXXX XXXXXXXX | | PPG 7 |
| $00014C_H$ | PDUT07 [W]<br>XXXXXXXX XXXXXXXX | | PCNH07 [R/W]<br>0000000 - | PCNL07 [R/W]<br>000000 - 0 | |
| $000150_H$ | PTMR08 [R]<br>11111111 11111111 | | PCSR08 [W]<br>XXXXXXXX XXXXXXXX | | PPG 8 |
| $000154_H$ | PDUT08 [W]<br>XXXXXXXX XXXXXXXX | | PCNH08 [R/W]<br>0000000 - | PCNL08 [R/W]<br>000000 - 0 | |
| $000158_H$ | PTMR09 [R]<br>11111111 11111111 | | PCSR09 [W]<br>XXXXXXXX XXXXXXXX | | PPG 9 |
| $00015C_H$ | PDUT09 [W]<br>XXXXXXXX XXXXXXXX | | PCNH09 [R/W]<br>0000000 - | PCNL09 [R/W]<br>000000 - 0 | |
| $000160_H$ | PTMR10 [R]<br>11111111 11111111 | | PCSR10 [W]<br>XXXXXXXX XXXXXXXX | | PPG 10 |
| $000164_H$ | PDUT10 [W]<br>XXXXXXXX XXXXXXXX | | PCNH10 [R/W]<br>0000000 - | PCNL10 [R/W]<br>000000 - 0 | |
| $000168_H$ | PTMR11 [R]<br>11111111 11111111 | | PCSR11 [W]<br>XXXXXXXX XXXXXXXX | | PPG 11 |
| $00016C_H$ | PDUT11 [W]<br>XXXXXXXX XXXXXXXX | | PCNH11 [R/W]<br>0000000 - | PCNL11 [R/W]<br>000000 - 0 | |
| $000170_H$ to $00017C_H$ | Reserved | | | | Reserved |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $000180_H$ | Reserved | ICS01 [R/W]<br>00000000 | Reserved | ICS23 [R/W]<br>00000000 | Input Capture 0 to 3 |
| $000184_H$ | IPCP0 [R]<br>XXXXXXXX XXXXXXXX | | IPCP1 [R]<br>XXXXXXXX XXXXXXXX | | |
| $000188_H$ | IPCP2 [R]<br>XXXXXXXX XXXXXXXX | | IPCP3 [R]<br>XXXXXXXX XXXXXXXX | | |
| $00018C_H$ | OCS01 [R/W]<br>- - - 0 - - 00 0000 - - 00 | | OCS23 [R/W]<br>- - - 0 - - 00 0000 - - 00 | | Output Compare 0 to 3 |
| $000190_H$ | OCCP0 [R/W]<br>XXXXXXXX XXXXXXXX | | OCCP1 [R/W]<br>XXXXXXXX XXXXXXXX | | |
| $000194_H$ | OCCP2 [R/W]<br>XXXXXXXX XXXXXXXX | | OCCP3 [R/W]<br>XXXXXXXX XXXXXXXX | | |
| $000198_H$ to $00019C_H$ | Reserved | | | | Reserved |
| $0001A0_H$ | ADERH [R/W]<br>00000000 00000000 | | ADERL [R/W]<br>00000000 00000000 | | A/D Converter |
| 0001A4 | ADCS1 [R/W]<br>00000000 | ADCS0 [R/W]<br>00000000 | ADCR1 [R]<br>000000XX | ADCR0 [R]<br>XXXXXXXX | |
| $0001A8_H$ | ADCT1 [R/W]<br>00010000 | ADCT0 [R/W]<br>00101100 | ADSCH [R/W]<br>- - - 00000 | ADECH [R/W]<br>- - - 00000 | |
| $0001AC_H$ | Reserved | | | | Reserved |
| $0001B0_H$ | TMRLR0 [W]<br>XXXXXXXX XXXXXXXX | | TMR0 [R]<br>XXXXXXXX XXXXXXXX | | Reload Timer 0 (PPG 0, PPG 1) |
| $0001B4_H$ | Reserved | | TMCSRH0 [R/W]<br>- - - 00000 | TMCSRL0 [R/W]<br>0 - 000000 | |
| $0001B8_H$ | TMRLR1 [W]<br>XXXXXXXX XXXXXXXX | | TMR1 [R]<br>XXXXXXXX XXXXXXXX | | Reload Timer 1 (PPG 2, PPG 3) |
| $0001BC_H$ | Reserved | | TMCSRH1 [R/W]<br>- - - 00000 | TMCSRL1 [R/W]<br>0 - 000000 | |
| $0001C0_H$ | TMRLR2 [W]<br>XXXXXXXX XXXXXXXX | | TMR2 [R]<br>XXXXXXXX XXXXXXXX | | Reload Timer 2 (PPG 4, PPG 5) |
| $0001C4_H$ | Reserved | | TMCSRH2 [R/W]<br>- - - 00000 | TMCSRL2 [R/W]<br>0 - 000000 | |
| $0001C8_H$ | TMRLR3 [W]<br>XXXXXXXX XXXXXXXX | | TMR3 [R]<br>XXXXXXXX XXXXXXXX | | Reload Timer 3 (PPG 6,PPG 7) |
| $0001CC_H$ | Reserved | | TMCSRH3 [R/W]<br>- - - 00000 | TMCSRL3 [R/W]<br>0 - 000000 | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $0001D0_H$ | TMRLR4 [W]<br>XXXXXXXX XXXXXXXX | | TMR4 [R]<br>XXXXXXXX XXXXXXXX | | Reload Timer 4<br>(PPG 8, PPG 9) |
| $0001D4_H$ | Reserved | | TMCSRH4<br>[R/W]<br>- - - 00000 | TMCSRL4<br>[R/W]<br>0 - 000000 | |
| $0001D8_H$ | TMRLR5 [W]<br>XXXXXXXX XXXXXXXX | | TMR5 [R]<br>XXXXXXXX XXXXXXXX | | Reload Timer 5<br>(PPG 10, PPG 11) |
| $0001DC_H$ | Reserved | | TMCSRH5<br>[R/W]<br>- - - 00000 | TMCSRL5<br>[R/W]<br>0 - 000000 | |
| $0001E0_H$ | TMRLR6 [W]<br>XXXXXXXX XXXXXXXX | | TMR6 [R]<br>XXXXXXXX XXXXXXXX | | Reload Timer 6 |
| $0001E4_H$ | Reserved | | TMCSRH6<br>[R/W]<br>- - - 00000 | TMCSRL6<br>[R/W]<br>0 - 000000 | |
| $0001E8_H$ | TMRLR7 [W]<br>XXXXXXXX XXXXXXXX | | TMR7 [R]<br>XXXXXXXX XXXXXXXX | | Reload Timer 7<br>(A/D Convertor) |
| $0001EC_H$ | Reserved | | TMCSRH7<br>[R/W]<br>- - - 00000 | TMCSRL7<br>[R/W]<br>0 - 000000 | |
| $0001F0_H$ | TCDT0 [R/W]<br>XXXXXXXX XXXXXXXX | | Reserved | TCCS0 [R/W]<br>00000000 | Free Running Timer 0<br>(ICU 0, ICU 1) |
| $0001F4_H$ | TCDT1 [R/W]<br>XXXXXXXX XXXXXXXX | | Reserved | TCCS1 [R/W]<br>00000000 | Free Running Timer 1<br>(ICU 2, ICU 3) |
| $0001F8_H$ | TCDT2 [R/W]<br>XXXXXXXX XXXXXXXX | | Reserved | TCCS2 [R/W]<br>00000000 | Free Running Timer 2<br>(OCU 0, OCU 1) |
| $0001FC_H$ | TCDT3 [R/W]<br>XXXXXXXX XXXXXXXX | | Reserved | TCCS3 [R/W]<br>00000000 | Free Running Timer 3<br>(OCU 2, OCU 3) |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $000200_H$ | DMACA0 [R/W]<br>00000000 0000XXXX XXXXXXXX XXXXXXXX | | | | DMAC |
| $000204_H$ | DMACB0 [R/W]<br>00000000 00000000 XXXXXXXX XXXXXXXX | | | | |
| $000208_H$ | DMACA1 [R/W]<br>00000000 0000XXXX XXXXXXXX XXXXXXXX | | | | |
| $00020C_H$ | DMACB1 [R/W]<br>00000000 00000000 XXXXXXXX XXXXXXXX | | | | |
| $000210_H$ | DMACA2 [R/W]<br>00000000 0000XXXX XXXXXXXX XXXXXXXX | | | | |
| $000214_H$ | DMACB2 [R/W]<br>00000000 00000000 XXXXXXXX XXXXXXXX | | | | |
| $000218_H$ | DMACA3 [R/W]<br>00000000 0000XXXX XXXXXXXX XXXXXXXX | | | | |
| $00021C_H$ | DMACB3 [R/W]<br>00000000 00000000 XXXXXXXX XXXXXXXX | | | | |
| $000220_H$ | DMACA4 [R/W]<br>00000000 0000XXXX XXXXXXXX XXXXXXXX | | | | |
| $000224_H$ | DMACB4 [R/W]<br>00000000 00000000 XXXXXXXX XXXXXXXX | | | | |
| $000228_H$ to $00023C_H$ | Reserved | | | | Reserved |
| $000240_H$ | DMACR [R/W]<br>00 - - 0000 | Reserved | | | Reserved |
| $000244_H$ to $0002CC_H$ | Reserved | | | | Reserved |
| $0002D0_H$ | Reserved | ICS045 [R/W]<br>00000000 | Reserved | ICS67 [R/W]<br>00000000 | Input Capture 4 to 7 |
| $0002D4_H$ | IPCP4 [R]<br>XXXXXXXX XXXXXXXX | | IPCP5 [R]<br>XXXXXXXX XXXXXXXX | | |
| $0002D8_H$ | IPCP6 [R]<br>XXXXXXXX XXXXXXXX | | IPCP7 [R]<br>XXXXXXXX XXXXXXXX | | |
| $0002DC_H$ | OCS45 [R/W]<br>- - - 0 - - 00 0000 - - 00 | | Reserved | | Output Compare 4 to 5 |
| $0002E0_H$ | OCCP4 [R/W]<br>XXXXXXXX XXXXXXXX | | OCCP5 [R/W]<br>XXXXXXXX XXXXXXXX | | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $0002E4_H$ to $0002EC_H$ | Reserved | | | | Reserved |
| $0002F0_H$ | TCDT4 [R/W]<br>XXXXXXXX XXXXXXXX | | Reserved | TCCS4 [R/W]<br>00000000 | Free Running Timer 4<br>(ICU 4, ICU 5) |
| $0002F4_H$ | TCDT5 [R/W]<br>XXXXXXXX XXXXXXXX | | Reserved | TCCS5 [R/W]<br>00000000 | Free Running Timer 5<br>(ICU 6, ICU 7) |
| $0002F8_H$ | TCDT6 [R/W]<br>XXXXXXXX XXXXXXXX | | Reserved | TCCS6 [R/W]<br>00000000 | Free Running Timer 6<br>(OCU 4-, OCU ) |
| $0002FC_H$ | TCDT7 [R/W]<br>XXXXXXXX XXXXXXXX | | Reserved | TCCS7 [R/W]<br>00000000 | Free Running Timer 7 |
| $000300_H$ to $00038C_H$ | Reserved | | | | Reserved |
| $000390_H$ | ROMS [R]<br>11111111 01000011 | | Reserved | | ROM Select Register |
| $000394_H$ to $0003EC_H$ | Reserved | | | | Reserved |
| $0003F0_H$ | BSD0 [W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | Bit Search Module |
| $0003F4_H$ | BSD1 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $0003F8_H$ | BSDC [W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $0003FC_H$ | BSRR [R]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $000400_H$ to $00043C_H$ | Reserved | | | | Reserved |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $000440_H$ | ICR00 [R/W] ---11111 | ICR01 [R/W] ---11111 | ICR02 [R/W] ---11111 | ICR03 [R/W] ---11111 | Interrupt Control Unit |
| $000444_H$ | ICR04 [R/W] ---11111 | ICR05 [R/W] ---11111 | ICR06 [R/W] ---11111 | ICR07 [R/W] ---11111 | |
| $000448_H$ | ICR08 [R/W] ---11111 | ICR09 [R/W] ---11111 | ICR10 [R/W] ---11111 | ICR11 [R/W] ---11111 | |
| $00044C_H$ | ICR12 [R/W] ---11111 | ICR13 [R/W] ---11111 | ICR14 [R/W] ---11111 | ICR15 [R/W] ---11111 | |
| $000450_H$ | ICR16 [R/W] ---11111 | ICR17 [R/W] ---11111 | ICR18 [R/W] ---11111 | ICR19 [R/W] ---11111 | |
| $000454_H$ | ICR20 [R/W] ---11111 | ICR21 [R/W] ---11111 | ICR22 [R/W] ---11111 | ICR23 [R/W] ---11111 | |
| $000458_H$ | ICR24 [R/W] ---11111 | ICR25 [R/W] ---11111 | ICR26 [R/W] ---11111 | ICR27 [R/W] ---11111 | |
| $00045C_H$ | ICR28 [R/W] ---11111 | ICR29 [R/W] ---11111 | ICR30 [R/W] ---11111 | ICR31 [R/W] ---11111 | |
| $000460_H$ | ICR32 [R/W] ---11111 | ICR33 [R/W] ---11111 | ICR34 [R/W] ---11111 | ICR35 [R/W] ---11111 | |
| $000464_H$ | ICR36 [R/W] ---11111 | ICR37 [R/W] ---11111 | ICR38 [R/W] ---11111 | ICR39 [R/W] ---11111 | |
| $000468_H$ | ICR40 [R/W] ---11111 | ICR41 [R/W] ---11111 | ICR42 [R/W] ---11111 | ICR43 [R/W] ---11111 | |
| $00046C_H$ | ICR44 [R/W] ---11111 | ICR45 [R/W] ---11111 | ICR46 [R/W] ---11111 | ICR47 [R/W] ---11111 | |
| $000470_H$ | ICR48 [R/W] ---11111 | ICR49 [R/W] ---11111 | ICR50 [R/W] ---11111 | ICR51 [R/W] ---11111 | |
| $000474_H$ | ICR52 [R/W] ---11111 | ICR53 [R/W] ---11111 | ICR54 [R/W] ---11111 | ICR55 [R/W] ---11111 | |
| $000478_H$ | ICR56 [R/W] ---11111 | ICR57 [R/W] ---11111 | ICR58 [R/W] ---11111 | ICR59 [R/W] ---11111 | |
| $00047C_H$ | ICR60 [R/W] ---11111 | ICR61 [R/W] ---11111 | ICR62 [R/W] ---11111 | ICR63 [R/W] ---11111 | |
| $000480_H$ | RSRR [R/W] 10000000 | STCR [R/W] 00110011 | TBCR [R/W] X0000X00 | CTBR [W] XXXXXXXX | Clock Control Unit |
| $000484_H$ | CLKR [R/W] 00000000 | WPR [W] XXXXXXXX | DIVR0 [R/W] 00000011 | DIVR1 [R/W] 00000000 | |
| $000488_H$ | Reserved | | | | Reserved |
| $00048C_H$ | PLLDIVM [R/W] - - - - 0000 | PLLDIVN [R/W] - - 000000 | PLLDIVG [R/W] - - - - 0000 | PLLMULG [R/W] 00000000 | PLL Clock Gear Unit |
| $000490_H$ | PLLCTRL [R/W] - - - - 0000 | Reserved | Reserved | Reserved | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $000494_H$ | OSCC1 [R/W]<br>- - - - - 010 | OSCS1 [R/W]<br>00001111 | OSCC2 [R/W]<br>- - - - - 010 | OSCS2 [R/W]<br>00001111 | Main/Sub Oscillator Control |
| $000498_H$ | PORTEN [R/W]<br>- - - - - - 00 | Reserved | Reserved | Reserved | Port Input Enable Control |
| $0004A0_H$ | Reserved | WTCER [R/W]<br>- - - - - - 00 | WTCR [R/W]<br>00000000 000 - 00 - 0 | | Real Time Clock (Watch Timer) |
| $0004A4_H$ | Reserved | WTBR [R/W]<br>- - - XXXXX XXXXXXXX XXXXXXXX | | | |
| $0004A8_H$ | WTHR [R/W]<br>- - - 00000 | WTMR [R/W]<br>- - 000000 | WTSR [R/W]<br>- - 000000 | Reserved | |
| $0004AC_H$ | CSVTR [R/W]<br>- - - 00010 | CSVCR [R/W]<br>00011100 | CSCFG [R/W]<br>0X000000 | Reserved | Clock-Supervisor / Selector |
| $0004B0_H$ | CUCR [R/W]<br>- - - - - - - - - - - - 0 - - 00 | | CUTD [R/W]<br>10000000 00000000 | | Calibration Unit of Sub Oscillation |
| $0004B4_H$ | CUTR1 [R]<br>- - - - - - - - 00000000 | | CUTR2 [R]<br>00000000 00000000 | | |
| $0004B8_H$ | CMPR [R/W]<br>- - 000010 11111101 | | Reserved | CMCR [R/W]<br>- 001 - - 00 | Clock Modulator |
| $0004BC_H$ | CMT1 [R/W]<br>00000000 1 - - - 0000 | | CMT2 [R/W]<br>- - 000000 - - 000000 | | |
| $0004C0_H$ | CANPRE [R/W]<br>- - 00000 | CANCKD [R/W]<br>- - - 0 - - - 0[*1] | Reserved | Reserved | CAN Clock Control |
| $0004C4_H$ | LVSEL [R/W]<br>00000111 | LVDET [R/W]<br>00000-00 | HWWDE [R/W]<br>- - - - - - 00 | HWWD [R/W,W]<br>00011000 | LV Detection / Hardware-Watchdog |
| $0004C8_H$ | OSCRH [R/W]<br>000 - - 001 | OSCRL [R/W]<br>- - - - - 000 | WPCRH [R/W]<br>00 - - - 000 | WPCRL [R/W]<br>- - - - - - 00 | Main/Sub-Oscillation Stabilization Timer |
| $0004CC_H$ | OSCCR [R/W]<br>- - - - - - 00 | Reserved | REGSEL [R/W]<br>- - 000110 | REGCTR [R/W]<br>- - - 0 - - 00 | Main/Sub-Oscillation Standby Control / Main/Sub Regulator Control |
| $0004D0_H$ to $0004D8_H$ | Reserved | | | | Reserved |
| $0004DC_H$ | PLL2DIVM [R/W]<br>- - - - 0000 | PLL2DIVN [R/W]<br>- - 000000 | PLL2DIVG [R/W]<br>- - - - 0000 | PLL2MULG [R/W]<br>00000000 | PLL2 Clock Control (FlexRay) |
| $0004E0_H$ | PLL2CTRL [R/W]<br>- - - - 0000 | Reserved | CLKR2 [R/W]<br>- - - 00000 | Reserved | |
| $0004E4_H$ to $000BFC_H$ | Reserved | | | | Reserved |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $000C00_H$ | Reserved | Reserved | Reserved | IOS [R/W]<br>- - - - - - - 0 | I-Unit |
| $000C04_H$ to $000D08_H$ | Reserved | | | | Reserved |
| $000D0C_H$ | Reserved | Reserved | PDRD14 [R]<br>XXXXXXXX | PDRD15 [R]<br>- - XXXXXX | R-bus Port Data Direct Read Register |
| $000D10$ | PDRD16 [R]<br>X - - - XXXX | PDRD17 [R]<br>XXXXXXXX | PDRD18 [R]<br>- XXX - XXX | PDRD19 [R]<br>- - - - - XXX | |
| $000D14_H$ | Reserved | Reserved | PDRD22 [R]<br>- - XX - - XX | PDRD23 [R]<br>- - - - - - XX | |
| $000D18_H$ | PDRD24 [R]<br>XXXXXXXX | Reserved | Reserved | PDRD27 [R]<br>- - - - XXXX | |
| $000D1C_H$ | PDRD28 [R]<br>- - - XXXXX | PDRD29 [R]<br>XXXXXXXX | Reserved | PDRD31 [R]<br>- XXX - XXX | |
| $000D20_H$ to $000D48_H$ | Reserved | | | | Reserved |
| $000D4C_H$ | Reserved | Reserved | DDR14 [R/W]<br>00000000 | DDR15 [R/W]<br>- - 000000 | R-bus Port Direction Register |
| $000D50_H$ | DDR16 [R/W]<br>0 - - - 0000 | DDR17 [R/W]<br>00000000 | DDR18 [R/W]<br>- 000 - 000 | DDR19 [R/W]<br>- - - - - 000 | |
| $000D54_H$ | Reserved | Reserved | DDR22 [R/W]<br>- - 00 - - 00 | DDR23 [R/W]<br>- - - - - - 00 | |
| $000D58_H$ | DDR24 [R/W]<br>00000000 | Reserved | Reserved | DDR27 [R/W]<br>- - - - 0000 | |
| $000D5C_H$ | DDR28 [R/W]<br>- - - 00000 | DDR29 [R/W]<br>00000000 | Reserved | DDR31 [R/W]<br>- 000 - 000 | |
| $000D60_H$ to $000D88_H$ | Reserved | | | | Reserved |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $000D8C_H$ | Reserved | Reserved | PFR14 [R/W]<br>00000000 | PFR15 [R/W]<br>- - 000000 | R-bus Port Function Register |
| $000D90_H$ | PFR16 [R/W]<br>0 - - - 0000 | PFR17 [R/W]<br>00000000 | PFR18 [R/W]<br>- 000 - 000 | PFR19 [R/W]<br>- - - - - 000 | |
| $000D94_H$ | Reserved | Reserved | PFR22 [R/W]<br>- - 00 - - 00 | PFR23 [R/W]<br>- - - - - - 00 | |
| $000D98_H$ | PFR24 [R/W]<br>00000000 | Reserved | Reserved | PFR27 [R/W]<br>- - - - 0000 | |
| $000D9C_H$ | PFR28 [R/W]<br>- - - 00000 | PFR29 [R/W]<br>00000000 | Reserved | PFR31 [R/W]<br>- 000 - 000 | |
| $000DA0_H$ to $000DC8_H$ | Reserved | | | | Reserved |
| $000DCC_H$ | Reserved | Reserved | EPFR14 [R/W]<br>00000000 | EPFR15 [R/W]<br>- - 000000 | R-bus Port Extra Function Register |
| $000DD0_H$ | EPFR16 [R/W]<br>0 - - - - - - - | Reserved | EPFR18 [R/W]<br>- 0 - - - 0 - - | EPFR19 [R/W]<br>- - - - - 0 - - | |
| $000DD4_H$ | Reserved | Reserved | Reserved | Reserved | |
| $000DD8_H$ | Reserved | Reserved | Reserved | EPFR27 [R/W]<br>- - - - 0000 | |
| $000DDC_H$ | Reserved | Reserved | Reserved | EPFR31 [R/W]<br>- 000 - 000 | |
| $000DE0_H$ to $000E08_H$ | Reserved | | | | Reserved |
| $000E0C_H$ | Reserved | Reserved | PODR14 [R/W]<br>00000000 | PODR15 [R/W]<br>- - 000000 | R-bus Port Output Drive Select Register |
| $000E10_H$ | PODR16 [R/W]<br>0 - - - 0000 | PODR17 [R/W]<br>00000000 | PODR18 [R/W]<br>- 000 - 000 | PODR19 [R/W]<br>- - - - - 000 | |
| $000E14_H$ | Reserved | Reserved | PODR22 [R/W]<br>- - 00 - - 00 | PODR23 [R/W]<br>- - - - - - 00 | |
| $000E18_H$ | PODR24 [R/W]<br>00000000 | Reserved | Reserved | PODR27 [R/W]<br>- - - - 0000 | |
| $000E1C_H$ | PODR28 [R/W]<br>- - - 00000 | PODR29 [R/W]<br>00000000 | Reserved | PODR31 [R/W]<br>- 000 - 000 | |
| $000E20_H$ to $000E48_H$ | Reserved | | | | Reserved |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $000E4C_H$ | Reserved | Reserved | PILR14 [R/W]<br>00000000 | PILR15 [R/W]<br>- - 000000 | R-bus Port Input Level Select Register |
| $000E50_H$ | PILR16 [R/W]<br>0 - - - 0000 | PILR17 [R/W]<br>00000000 | PILR18 [R/W]<br>- 000 - 000 | PILR19 [R/W]<br>- - - - - 000 | |
| $000E54_H$ | Reserved | Reserved | PILR22 [R/W]<br>- - 00 - - 00 | PILR23 [R/W]<br>- - - - - - 00 | |
| $000E58_H$ | PILR24 [R/W]<br>00000000 | Reserved | Reserved | PILR27 [R/W]<br>- - - - 0000 | |
| $000E5C_H$ | PILR28 [R/W]<br>- - - 00000 | PILR29 [R/W]<br>00000000 | Reserved | PILR31 [R/W]<br>- 000 - 000 | |
| $000E60_H$ to $000E88_H$ | Reserved | | | | Reserved |
| $000E8C_H$ | Reserved | Reserved | EPILR14 [R/W]<br>00000000 | EPILR15 [R/W]<br>- - 000000 | R-bus Port Extra Input Level Select Register |
| $000E90_H$ | EPILR16 [R/W]<br>0 - - - 0000 | EPILR17 [R/W]<br>00000000 | EPILR18 [R/W]<br>- 000 - 000 | EPILR19 [R/W]<br>- - - - - 000 | |
| $000E94_H$ | Reserved | Reserved | EPILR22 [R/W]<br>- - 00 - - 00 | EPILR23 [R/W]<br>- - - - - - 00 | |
| $000E98_H$ | EPILR24 [R/W]<br>00000000 | Reserved | Reserved | EPILR27 [R/W]<br>- - - - 0000 | |
| $000E9C_H$ | EPILR28 [R/W]<br>- - - 00000 | EPILR29 [R/W]<br>00000000 | Reserved | EPILR31 [R/W]<br>- 000 - 000 | |
| $000EA0_H$ to $000EC8_H$ | Reserved | | | | Reserved |
| $000ECC_H$ | Reserved | Reserved | PPER14 [R/W]<br>00000000 | PPER15 [R/W]<br>- - 000000 | R-bus Port Pull-Up/Down Enable Register |
| $000ED0_H$ | PPER16 [R/W]<br>0 - - - 0000 | PPER17 [R/W]<br>00000000 | PPER18 [R/W]<br>- 000 - 000 | PPER19 [R/W]<br>- - - - - 000 | |
| $000ED4_H$ | Reserved | Reserved | PPER22 [R/W]<br>- - 00 - - 00 | PPER23 [R/W]<br>- - - - - - 00 | |
| $000ED8_H$ | PPER24 [R/W]<br>00000000 | Reserved | Reserved | PPER27 [R/W]<br>- - - - 0000 | |
| $000EDC_H$ | PPER28 [R/W]<br>- - - 00000 | PPER29 [R/W]<br>00000000 | Reserved | PPER31 [R/W]<br>- 000 - 000 | |
| $000EE0_H$ to $000F08_H$ | Reserved | | | | Reserved |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $000F0C_H$ | Reserved | Reserved | PPCR14 [R/W]<br>11111111 | PPCR15 [R/W]<br>- - 111111 | R-bus Port Pull-Up/Down Control Register |
| $000F10_H$ | PPCR16 [R/W]<br>1 - - - 1111 | PPCR17 [R/W]<br>11111111 | PPCR18 [R/W]<br>- 111 - 111 | PPCR19 [R/W]<br>- - - - - 111 | |
| $000F14_H$ | Reserved | Reserved | PPCR22 [R/W]<br>- - 11 - - 11 | PPCR23 [R/W]<br>- - - - - - 11 | |
| $000F18_H$ | PPCR24 [R/W]<br>11111111 | Reserved | Reserved | PPCR27 [R/W]<br>- - - - 1111 | |
| $000F1C_H$ | PPCR28 [R/W]<br>- - - 11111 | PPCR29 [R/W]<br>11111111 | Reserved | PPCR31 [R/W]<br>- 111 – 111 | |
| $000F20_H$ to $000F3C_H$ | Reserved | | | | Reserved |
| $001000_H$ | DMASA0 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | DMAC |
| $001004_H$ | DMADA0 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $001008_H$ | DMASA1 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00100C_H$ | DMADA1 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $001010_H$ | DMASA2 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $001014_H$ | DMADA2 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $001018_H$ | DMASA3 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00101C_H$ | DMADA3 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $001020_H$ | DMASA4 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $001024_H$ | DMADA4 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $001028_H$ to $006FFC_H$ | Reserved | | | | Reserved |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $007000_H$ | FMCS [R/W]<br>01101000 | FMCR [R]<br>- - - 00000 | FCHCR [R/W]<br>- - - - - - 00 10000011 | | Flash Memory/ Cache Control Register |
| $007004_H$ | FMWT [R/W]<br>11111111 11111111 | | FMWT2 [R]<br>- 001 - - - - | FMPS [R/W]<br>- - - - - 000 | |
| $007008_H$ | FMAC [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00700C_H$ to $007FFC_H$ | Reserved | | | | Reserved |
| $008000_H$ to $00BFFC_H$ | MB91F465XA Boot-ROM size is 4KB : $00B000_H$ to $00BFFC_H$<br>(instruction access is 1 waitcycle, data access is 1 waitcycle) | | | | Boot ROM area |
| $00C000_H$ | CTRLR0 [R/W]<br>00000000 00000001 | | STATR0 [R/W]<br>00000000 00000000 | | CAN 0 Control Register |
| $00C004_H$ | ERRCNT0 [R]<br>00000000 00000000 | | BTR0 [R/W]<br>00100011 00000001 | | |
| $00C008_H$ | INTR0 [R]<br>00000000 00000000 | | TESTR0 [R/W]<br>00000000 X0000000 | | |
| $00C00C_H$ | BRPE0 [R/W]<br>00000000 00000000 | | CBSYNC0 | | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $00C010_H$ | IF1CREQ0 [R/W]<br>00000000 00000001 | | IF1CMSK0 [R/W]<br>00000000 00000000 | | CAN 0<br>IF 1 Register |
| $00C014_H$ | IF1MSK20 [R/W]<br>11111111 11111111 | | IF1MSK10 [R/W]<br>11111111 11111111 | | |
| $00C018_H$ | IF1ARB20 [R/W]<br>00000000 00000000 | | IF1ARB10 [R/W]<br>00000000 00000000 | | |
| $00C01C_H$ | IF1MCTR0 [R/W]<br>00000000 00000000 | | Reserved | | |
| $00C020_H$ | IF1DTA10 [R/W]<br>00000000 00000000 | | IF1DTA20 [R/W]<br>00000000 00000000 | | |
| $00C024_H$ | IF1DTB10 [R/W]<br>00000000 00000000 | | IF1DTB20 [R/W]<br>00000000 00000000 | | |
| $00C028_H$ to $00C02C_H$ | Reserved | | | | |
| $00C030_H$ | IF1DTA20 [R/W]<br>00000000 00000000 | | IF1DTA10 [R/W]<br>00000000 00000000 | | |
| $00C034_H$ | IF1DTB20 [R/W]<br>00000000 00000000 | | IF1DTB10 [R/W]<br>00000000 00000000 | | |
| $00C038_H$ to $00C03C_H$ | Reserved | | | | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $00C040_H$ | IF2CREQ0 [R/W]<br>00000000 00000001 | | IF2CMSK0 [R/W]<br>00000000 00000000 | | CAN 0<br>IF 2 Register |
| $00C044_H$ | IF2MSK20 [R/W]<br>11111111 11111111 | | IF2MSK10 [R/W]<br>11111111 11111111 | | |
| $00C048_H$ | IF2ARB20 [R/W]<br>00000000 00000000 | | IF2ARB10 [R/W]<br>00000000 00000000 | | |
| $00C04C_H$ | IF2MCTR0 [R/W]<br>00000000 00000000 | | Reserved | | |
| $00C050_H$ | IF2DTA10 [R/W]<br>00000000 00000000 | | IF2DTA20 [R/W]<br>00000000 00000000 | | |
| $00C054_H$ | IF2DTB10 [R/W]<br>00000000 00000000 | | IF2DTB20 [R/W]<br>00000000 00000000 | | |
| $00C058_H$ to $00C05C_H$ | Reserved | | | | |
| $00C060_H$ | IF2DTA20 [R/W]<br>00000000 00000000 | | IF2DTA10 [R/W]<br>00000000 00000000 | | |
| $00C064_H$ | IF2DTB20 [R/W]<br>00000000 00000000 | | IF2DTB10 [R/W]<br>00000000 00000000 | | |
| $00C068_H$ to $00C07C_H$ | Reserved | | | | |
| $00C080_H$ | TREQR20 [R]<br>00000000 00000000 | | TREQR10 [R]<br>00000000 00000000 | | CAN 0<br>Status Flags |
| $00C084_H$ to $00C08C_H$ | Reserved | | | | |
| $00C090_H$ | NEWDT20 [R]<br>00000000 00000000 | | NEWDT10 [R]<br>00000000 00000000 | | |
| $00C094_H$ to $00C09C_H$ | Reserved | | | | |
| $00C0A0_H$ | INTPND20 [R]<br>00000000 00000000 | | INTPND10 [R]<br>00000000 00000000 | | |
| $00C0A4_H$ to $00C0AC_H$ | Reserved | | | | |
| $00C0B0_H$ | MSGVAL20 [R]<br>00000000 00000000 | | MSGVAL10 [R]<br>00000000 00000000 | | |
| $00C0B4_H$ to $00C3FC_H$ | Reserved | | | | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $00C400_H$ | CTRLR4 [R/W]<br>00000000 00000001 | | STATR4 [R/W]<br>00000000 00000000 | | CAN 4 Control Register |
| $00C404_H$ | ERRCNT4 [R]<br>00000000 00000000 | | BTR4 [R/W]<br>00100011 00000001 | | |
| $00C408_H$ | INTR4 [R]<br>00000000 00000000 | | TESTR4 [R/W]<br>00000000 X0000000 | | |
| $00C40C_H$ | BRPE4 [R/W]<br>00000000 00000000 | | CBSYNC4 | | |
| $00C410_H$ | IF1CREQ4 [R/W]<br>00000000 00000001 | | IF1CMSK4 [R/W]<br>00000000 00000000 | | CAN 4 IF 1 Register |
| $00C414_H$ | IF1MSK24 [R/W]<br>11111111 11111111 | | IF1MSK14 [R/W]<br>11111111 11111111 | | |
| $00C418_H$ | IF1ARB24 [R/W]<br>00000000 00000000 | | IF1ARB14 [R/W]<br>00000000 00000000 | | |
| $00C41C_H$ | IF1MCTR4 [R/W]<br>00000000 00000000 | | Reserved | | |
| $00C420_H$ | IF1DTA14 [R/W]<br>00000000 00000000 | | IF1DTA24 [R/W]<br>00000000 00000000 | | |
| $00C424_H$ | IF1DTB14 [R/W]<br>00000000 00000000 | | IF1DTB24 [R/W]<br>00000000 00000000 | | |
| $00C428_H$ to $00C42C_H$ | Reserved | | | | |
| $00C430_H$ | IF1DTA24 [R/W]<br>00000000 00000000 | | IF1DTA14 [R/W]<br>00000000 00000000 | | |
| $00C434_H$ | IF1DTB24 [R/W]<br>00000000 00000000 | | IF1DTB14 [R/W]<br>00000000 00000000 | | |
| $00C438_H$ to $00C43C_H$ | Reserved | | | | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $00C440_H$ | IF2CREQ4 [R/W]<br>00000000 00000001 | | IF2CMSK4 [R/W]<br>00000000 00000000 | | CAN 4<br>IF 2 Register |
| $00C444_H$ | IF2MSK24 [R/W]<br>11111111 11111111 | | IF2MSK14 [R/W]<br>11111111 11111111 | | |
| $00C448_H$ | IF2ARB24 [R/W]<br>00000000 00000000 | | IF2ARB14 [R/W]<br>00000000 00000000 | | |
| $00C44C_H$ | IF2MCTR4 [R/W]<br>00000000 00000000 | | Reserved | | |
| $00C450_H$ | IF2DTA14 [R/W]<br>00000000 00000000 | | IF2DTA24 [R/W]<br>00000000 00000000 | | |
| $00C454_H$ | IF2DTB14 [R/W]<br>00000000 00000000 | | IF2DTB24 [R/W]<br>00000000 00000000 | | |
| $00C458_H$ to $00C45C_H$ | Reserved | | | | |
| $00C460_H$ | IF2DTA24 [R/W]<br>00000000 00000000 | | IF2DTA14 [R/W]<br>00000000 00000000 | | |
| $00C464_H$ | IF2DTB24 [R/W]<br>00000000 00000000 | | IF2DTB14 [R/W]<br>00000000 00000000 | | |
| $00C468_H$ to $00C47C_H$ | Reserved | | | | |
| $00C480_H$ | TREQR24 [R]<br>00000000 00000000 | | TREQR14 [R]<br>00000000 00000000 | | CAN 4<br>Status Flags |
| $00C484_H$ to $00C48C_H$ | Reserved | | | | |
| $00C490_H$ | NEWDT24 [R]<br>00000000 00000000 | | NEWDT14 [R]<br>00000000 00000000 | | |
| $00C494_H$ to $00C49C_H$ | Reserved | | | | |
| $00C4A0_H$ | INTPND24 [R]<br>00000000 00000000 | | INTPND14 [R]<br>00000000 00000000 | | |
| $00C4A4_H$ to $00C4AC_H$ | Reserved | | | | |
| $00C4B0_H$ | MSGVAL24 [R]<br>00000000 00000000 | | MSGVAL14 [R]<br>00000000 00000000 | | |
| $00C4B4_H$ to $00CFFC_H$ | Reserved | | | | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $00D000_H$ | CIF0 [R]<br>00000100 11111111 01011011 11111111 | | | | FlexRay CIF |
| $00D004_H$ | CIF1 [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D008_H$ to $00D00C_H$ | Reserved (2) | | | | Reserved |
| $00D010_H$ | TEST1 [R/W]<br>00000000 00000000 00000011 00000000 | | | | FlexRay GIF |
| $00D014_H$ | TEST2 [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D018_H$ | Reserved (1) | | | | |
| $00D01C_H$ | LCK [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D020_H$ | EIR [R/W]<br>00000000 00000000 00000000 00000000 | | | | FlexRay INT |
| $00D024_H$ | SIR [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D028_H$ | EILS [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D02C_H$ | SILS [R/W]<br>00000011 00000011 11111111 11111111 | | | | |
| $00D030_H$ | EIES [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D034_H$ | EIER [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D038_H$ | SIES [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D03C_H$ | SIER [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D040_H$ | ILE [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D044_H$ | T0C [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D048_H$ | T1C [R/W]<br>00000000 00000010 00000000 00000000 | | | | |
| $00D04C_H$ | STPW1 [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D050_H$ | STPW2 [R/W]<br>00000000 00000000 00000000 00000000 | | | | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $00D050_H$ to $00D07C_H$ | Reserved (11) | | | | Reserved |
| $00D080_H$ | SUCC1 [R/W]<br>00001100 01000000 00010000 00000000 | | | | FlexRay SUC |
| $00D084_H$ | SUCC2 [R/W]<br>00000001 00000000 00000101 00000100 | | | | |
| $00D088_H$ | SUCC3 [R/W]<br>00000000 00000000 00000000 00010001 | | | | |
| $00D08C_H$ | NEMC [R/W]<br>00000000 00000000 00000000 00000000 | | | | FlexRay NEM |
| $00D090_H$ | PRTC1 [R/W]<br>00001000 01001100 00000110 00110011 | | | | FlexRay PRT |
| $00D094_H$ | PRTC2 [R/W]<br>00001111 00101101 00001010 00001110 | | | | |
| $00D098_H$ | MHDC [R/W]<br>00000000 00000000 00000000 00000000 | | | | FlexRay MHD |
| $00D09C_H$ | Reserved (1) | | | | Reserved |
| $00D0A0_H$ | GTUC1 [R/W]<br>00000000 00000000 00000010 10000000 | | | | FlexRay GTU |
| $00D0A4_H$ | GTUC2 [R/W]<br>00000000 00000010 00000000 00001010 | | | | |
| $00D0A8_H$ | GTUC3 [R/W]<br>00000010 00000010 00000000 00000000 | | | | |
| $00D0AC_H$ | GTUC4 [R/W]<br>00000000 00001000 00000000 00000111 | | | | |
| $00D0B0_H$ | GTUC5 [R/W]<br>00001110 00000000 00000000 00000000 | | | | |
| $00D0B4_H$ | GTUC6 [R/W]<br>00000000 00000010 00000000 00000000 | | | | |
| $00D0B8_H$ | GTUC7 [R/W]<br>00000000 00000010 00000000 00000100 | | | | |
| $00D0BC_H$ | GTUC8 [R/W]<br>00000000 00000000 00000000 00000010 | | | | |
| $00D0C0_H$ | GTUC9 [R/W]<br>00000000 00000000 0000001 00000001 | | | | |
| $00D0C4_H$ | GTUC10 [R/W]<br>00000000 00000010 00000000 00000101 | | | | |
| $00D0C8_H$ | GTUC11 [R/W]<br>00000000 00000000 00000000 00000000 | | | | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $00D0CC_H$ to $00D0FC_H$ | Reserved (11) | | | | Reserved |
| $00D100_H$ | CCSV [R]<br>00000000 00010000 01000000 00000000 | | | | FlexRay SUC |
| $00D104_H$ | CCEV [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D108_H$ to $00D10C_H$ | Reserved (2) | | | | Reserved |
| $00D110_H$ | SCV [R]<br>00000000 00000000 00000000 00000000 | | | | FlexRay GTU |
| $00D114_H$ | MTCCV [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D118_H$ | RCV [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D11C_H$ | OCV [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D120_H$ | SFS [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D124_H$ | SWNIT [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D128_H$ | ACS [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D12C_H$ | Reserved (1) | | | | |
| $00D130_H$- $00D168_H$ | ESIDn[1-15] [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D16C_H$ | Reserved (1) | | | | |
| $00D170_H$ $00D1A8_H$ | OSIDn[1-15] [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D1AC_H$ | Reserved (1) | | | | Reserved |
| $00D1B0_H$ $00D1B8_H$ | NMVn[1-3] [R]<br>00000000 00000000 00000000 00000000 | | | | FlexRay NEM |
| $00D1BC_H$- $00D2FC_H$ | Reserved (81) | | | | Reserved |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $00D300_H$ | MRC [R/W]<br>00000001 10000000 00000000 00000000 | | | | FlexRay MHD |
| $00D304_H$ | FRF [R/W]<br>00000001 10000000 00000000 00000000 | | | | |
| $00D308_H$ | FRFM [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D30C_H$ | FCL [R/W]<br>00000000 00000000 0000000 10000000 | | | | |
| $00D310_H$ | MHDS [R/W]<br>00000000 00000000 0000000 10000000 | | | | |
| $00D314_H$ | LDTS [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D318_H$ | FSR [R]<br>00000000 00000000 00000000 00000000 | | | | FlexRay MHD |
| $00D31C_H$ | MHDF [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D320_H$ | TXRQ1 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D324_H$ | TXRQ2 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D328_H$ | TXRQ3 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D32C_H$ | TXRQ4 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D330_H$ | NDAT1 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D334_H$ | NDAT2 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D338_H$ | NDAT3 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D33C_H$ | NDAT4 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D340_H$ | MBSC1 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D344_H$ | MBSC2 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D348_H$ | MBSC3 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D34C_H$ | MBSC4 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D350_H$ to $00D3EC_H$ | Reserved (40) | | | | Reserved |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $00D3F0_H$ | CREL [R]<br>00010000 00000110 00000101 00011001 | | | | FlexRay GIF |
| $00D3F4_H$ | ENDN [R]<br>10000111 01100101 0100011 00100001 | | | | |
| $00D3F8_H$ to $00D3FC_H$ | Reserved (2) | | | | Reserved |
| $00D400_H$ to $00D4FC_H$ | WRDSn[1-64] [R/W]<br>00000000 00000000 0000000 00000000 | | | | FlexRay IBF |
| $00D500_H$ | WRHS1 [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D504_H$ | WRHS2 [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D508_H$ | WRHS3 [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D50C_H$ | Reserved (1) | | | | |
| $00D510_H$ | IBCM [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D514_H$ | IBCR [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D518_H$ to $00D5FC_H$ | Reserved (58) | | | | Reserved |
| $00D600_H$ to $00D6FC_H$ | RDDSn[1-64] [R]<br>00000000 00000000 00000000 00000000 | | | | FlexRay OBF |
| $00D700_H$ | RDHS1 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D704_H$ | RDHS2 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D708_H$ | RDHS3 [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D70C_H$ | MBS [R]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D710_H$ | OBCM [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D714_H$ | OBCR [R/W]<br>00000000 00000000 00000000 00000000 | | | | |
| $00D718_H$ to $00D7FC_H$ | Reserved (58) | | | | Reserved |
| $00D800_H$ to $00EFFC_H$ | Reserved | | | | Reserved |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $00F000_H$ | BCTRL [R/W]<br>-------- -------- 11111100 00000000 | | | | EDSU / MPU |
| $00F004_H$ | BSTAT [R/W]<br>-------- -----000 00000000 10--0000 | | | | |
| $00F008_H$ | BIAC [R]<br>-------- -------- 00000000 00000000 | | | | |
| $00F00C_H$ | BOAC [R]<br>-------- -------- 00000000 00000000 | | | | |
| $00F010_H$ | BIRQ [R/W]<br>-------- -------- 00000000 00000000 | | | | |
| $00F014_H$ to $00F01C_H$ | Reserved | | | | |
| $00F020_H$ | BCR0 [R/W]<br>-------- 00000000 00000000 00000000 | | | | |
| $00F024_H$ | BCR1 [R/W]<br>-------- 00000000 00000000 00000000 | | | | |
| $00F028_H$ | BCR2 [R/W]<br>-------- 00000000 00000000 00000000 | | | | |
| $00F02C_H$ | BCR3 [R/W]<br>-------- 00000000 00000000 00000000 | | | | |
| $00F030_H$ to $00F07C_H$ | Reserved | | | | |
| $00F080_H$ | BAD0 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F084_H$ | BAD1 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F088_H$ | BAD2 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F08C_H$ | BAD3 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F090_H$ | BAD4 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F094_H$ | BAD5 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F098_H$ | BAD6 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F09C_H$ | BAD7 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F0A0_H$ | BAD8 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F0A4_H$ | BAD9 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |

| Address | Register | | | | Block |
|---|---|---|---|---|---|
| | +0 | +1 | +2 | +3 | |
| $00F0A8_H$ | BAD10 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | EDSU / MPU |
| $00F0AC_H$ | BAD11 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F0B0_H$ | BAD12 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F0B4_H$ | BAD13 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F0B8_H$ | BAD14 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F0BC_H$ | BAD15 [R/W]<br>XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX | | | | |
| $00F0C0_H$ - $01FFFC_H$ | Reserved | | | | |
| $020000_H$ to $02FFFC_H$ | MB91F465XA D-RAM size is 16 Kbytes: $02C000_H$ to $02FFFC_H$<br>(data access is 0 waitcycles) | | | | D-RAM area |
| $030000_H$ to $03FFFC_H$ | MB91F465XA ID-RAM size is 16 Kbytes: $030000_H$ to $033FFC_H$<br>(instruction access is 0 waitcycles, data access is 1 waitcycle) | | | | ID-RAM area |

*1 : depends on the number of available CAN channels.

## 12.2. Flash memory and external bus area

| 32bit read/write | dat[31:0] | | | | dat[31:0] | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 16bit read/write | dat[31:16] | | dat[15:0] | | dat[31:16] | | dat[15:0] | | |
| Address | Register | | | | | | | | Block |
| | + 0 | + 1 | + 2 | + 3 | + 4 | + 5 | + 6 | + 7 | |
| $040000_H$ to $05FFFF_H$ | Reserved | | | | | | | | ROMS0 |
| $060000_H$ to $07FFFF_H$ | Reserved | | | | | | | | ROMS1 |
| $080000_H$ to $09FFFF_H$ | SA12 (64KB) | | | | SA13 (64KB) | | | | ROMS2 |
| $0A0000_H$ to $0BFFFC_H$ | SA14 (64KB) | | | | SA15 (64KB) | | | | ROMS3 |
| $0C0000_H$ to $0DFFFC_H$ | SA16 (64KB) | | | | SA17 (64KB) | | | | ROMS4 |
| $0E0000_H$ to $0FFFF4_H$ | SA18 (64KB) | | | | SA19 (64KB) | | | | ROMS5 |
| $0FFFF8_H$ to $0FFFFC_H$ | FMV [R] $06\ 00\ 00\ 00_H$ | | | | FRV [R] $00\ 00\ BF\ F8_H$ | | | | |
| $100000_H$ to $11FFFF_H$ | Reserved | | | | | | | | ROMS6 |
| $120000_H$ to $13FFFF_H$ | Reserved | | | | | | | | |

| 32bit read/write | dat[31:0] | | | | dat[31:0] | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 16bit read/write | dat[31:16] | | dat[15:0] | | dat[31:16] | | dat[15:0] | | |
| Address | Register | | | | | | | | Block |
| | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 | |
| $140000_H$ to $143FFF_H$ | Reserved | | | | | | | | ROMS7 |
| $144000_H$ to $147FFF_H$ | Reserved | | | | | | | | |
| $148000_H$ to $14BFFF_H$ | SA4 (8KB) | | | | SA5 (8KB) | | | | |
| $14C000_H$ to $14FFFF_H$ | SA6 (8KB) | | | | SA7 (8KB) | | | | |
| $150000_H$ to $17FFFF_H$ | Reserved | | | | | | | | |
| $180000_H$ to $FFFFFF_H$ | Reserved | | | | | | | | |

Notes: Write operations to address $0FFFF8_H$ and $0FFFFC_H$ are not possible. When reading these addresses, the values shown above will be read.

## 13. Interrupt Vector Table

| Interrupt | Interrupt number | | Interrupt level*1 | | Interrupt vector*2 | | Resource number |
|---|---|---|---|---|---|---|---|
| | Decimal | Hexa-decimal | Setting Register | Register address | Offset | Default Vector address | |
| Reset | 0 | 00 | — | — | $3FC_H$ | 000FFFFC | |
| Mode vector | 1 | 01 | — | — | $3F8_H$ | 000FFFF8 | |
| System reserved | 2 | 02 | — | — | $3F4_H$ | 000FFFF4 | |
| System reserved | 3 | 03 | — | — | $3F0_H$ | 000FFFF0 | |
| System reserved | 4 | 04 | — | — | $3EC_H$ | 000FFFEC | |
| CPU supervisor mode (INT #5 instruction) *2 | 5 | 05 | — | — | $3E8_H$ | 000FFFE8 | |
| Memory Protection exception *2 | 6 | 06 | — | — | $3E4_H$ | 000FFFE4 | |
| System reserved | 7 | 07 | — | — | $3E0_H$ | 000FFFE0 | |
| System reserved | 8 | 08 | — | — | $3DC_H$ | 000FFFDC | |
| System reserved | 9 | 09 | — | — | $3D8_H$ | 000FFFD8 | |
| System reserved | 10 | 0A | — | — | $3D4_H$ | 000FFFD4 | |
| System reserved | 11 | 0B | — | — | $3D0_H$ | 000FFFD0 | |
| System reserved | 12 | 0C | — | — | $3CC_H$ | 000FFFCC | |
| System reserved | 13 | 0D | — | — | $3C8_H$ | 000FFFC8 | |
| Undefined instruction exception | 14 | 0E | — | — | $3C4_H$ | 000FFFC4 | |
| NMI request | 15 | 0F | $F_H$ fixed | | $3C0_H$ | 000FFFC0 | |
| External Interrupt 0 | 16 | 10 | ICR00 | $440_H$ | $3BC_H$ | 000FFFBC | 0, 16 |
| External Interrupt 1 | 17 | 11 | | | $3B8_H$ | 000FFFB8 | 1, 17 |
| External Interrupt 2 | 18 | 12 | ICR01 | $441_H$ | $3B4_H$ | 000FFFB4 | 2, 18 |
| External Interrupt 3 | 19 | 13 | | | $3B0_H$ | 000FFFB0 | 3, 19 |
| External Interrupt 4 | 20 | 14 | ICR02 | $442_H$ | $3AC_H$ | 000FFFAC | 20 |
| External Interrupt 5 | 21 | 15 | | | $3A8_H$ | 000FFFA8 | 21 |
| External Interrupt 6 | 22 | 16 | ICR03 | $443_H$ | $3A4_H$ | 000FFFA4 | 22 |
| External Interrupt 7 | 23 | 17 | | | $3A0_H$ | 000FFFA0 | 23 |
| External Interrupt 8 | 24 | 18 | ICR04 | $444_H$ | $39C_H$ | 000FFF9C | |
| System reserved | 25 | 19 | | | $398_H$ | 000FFF98 | |
| System reserved | 26 | 1A | ICR05 | $445_H$ | $394_H$ | 000FFF94 | |
| System reserved | 27 | 1B | | | $390_H$ | 000FFF90 | |
| External Interrupt 12 | 28 | 1C | ICR06 | $446_H$ | $38C_H$ | 000FFF8C | |
| System reserved | 29 | 1D | | | $388_H$ | 000FFF88 | |
| External Interrupt 14 | 30 | 1E | ICR07 | $447_H$ | $384_H$ | 000FFF84 | |
| System reserved | 31 | 1F | | | $380_H$ | 000FFF80 | |
| Reload Timer 0 | 32 | 20 | ICR08 | $448_H$ | $37C_H$ | 000FFF7C | 4, 32 |
| Reload Timer 1 | 33 | 21 | | | $378_H$ | 000FFF78 | 5, 33 |

| Interrupt | Interrupt number | | Interrupt level[*1] | | Interrupt vector[*2] | | Resource number |
|---|---|---|---|---|---|---|---|
| | Decimal | Hexa-decimal | Setting Register | Register address | Offset | Default Vector address | |
| Reload Timer2 | 34 | 22 | ICR09 | $449_H$ | $374_H$ | 000FFF74 | 34 |
| Reload Timer 3 | 35 | 23 | | | $370_H$ | 000FFF70 | 35 |
| Reload Timer 4 | 36 | 24 | ICR10 | $44A_H$ | $36C_H$ | 000FFF6C | 36 |
| Reload Timer 5 | 37 | 25 | | | $368_H$ | 000FFF68 | 37 |
| Reload Timer 6 | 38 | 26 | ICR11 | $44B_H$ | $364_H$ | 000FFF64 | 38 |
| Reload Timer 7 | 39 | 27 | | | $360_H$ | 000FFF60 | 39 |
| Free Run Timer 0 | 40 | 28 | ICR12 | $44C_H$ | $35C_H$ | 000FFF5C | 40 |
| Free Run Timer 1 | 41 | 29 | | | $358_H$ | 000FFF58 | 41 |
| Free Run Timer 2 | 42 | 2A | ICR13 | $44D_H$ | $354_H$ | 000FFF54 | 42 |
| Free Run Timer 3 | 43 | 2B | | | $350_H$ | 000FFF50 | 43 |
| Free Run Timer 4 | 44 | 2C | ICR14 | $44E_H$ | $34C_H$ | 000FFF4C | 44 |
| Free Run Timer 5 | 45 | 2D | | | $348_H$ | 000FFF48 | 45 |
| Free Run Timer 6 | 46 | 2E | ICR15 | $44F_H$ | $344_H$ | 000FFF44 | 46 |
| Free Run Timer 7 | 47 | 2F | | | $340_H$ | 000FFF40 | 47 |
| CAN 0 | 48 | 30 | ICR16 | $450_H$ | $33C_H$ | 000FFF3C | |
| System reserved | 49 | 31 | | | $338_H$ | 000FFF38 | |
| System reserved | 50 | 32 | ICR17 | $451_H$ | $334_H$ | 000FFF34 | |
| System reserved | 51 | 33 | | | $330_H$ | 000FFF30 | |
| CAN 4 | 52 | 34 | ICR18 | $452_H$ | $32C_H$ | 000FFF2C | |
| System reserved | 53 | 35 | | | $328_H$ | 000FFF28 | |
| System reserved | 54 | 36 | ICR19 | $453_H$ | $324_H$ | 000FFF24 | 6, 48 |
| System reserved | 55 | 37 | | | $320_H$ | 000FFF20 | 7, 49 |
| System reserved | 56 | 38 | ICR20 | $454_H$ | $31C_H$ | 000FFF1C | 8, 50 |
| System reserved | 57 | 39 | | | $318_H$ | 000FFF18 | 9, 51 |
| System reserved | 58 | 3A | ICR21 | $455_H$ | $314_H$ | 000FFF14 | 52 |
| System reserved | 59 | 3B | | | $310_H$ | 000FFF10 | 53 |
| System reserved | 60 | 3C | ICR22 | $456_H$ | $30C_H$ | 000FFF0C | 54 |
| System reserved | 61 | 3D | | | $308_H$ | 000FFF08 | 55 |
| System reserved | 62 | 3E | ICR23 [*4] | $457_H$ | $304_H$ | 000FFF04 | |
| Delayed Interrupt | 63 | 3F | | | $300_H$ | 000FFF00 | |
| System reserved [*3] | 64 | 40 | (ICR24) | $(458)_H$ | $2FC_H$ | 000FFEFC | |
| System reserved [*3] | 65 | 41 | | | $2F8_H$ | $000FFEF8_H$ | |
| LIN-USART (FIFO) 4 RX | 66 | 42 | ICR25 | $459_H$ | $2F4_H$ | $000FFEF4_H$ | 10, 56 |
| LIN-USART (FIFO) 4 TX | 67 | 43 | | | $2F0_H$ | $000FFEF0_H$ | 11, 57 |
| System reserved | 68 | 44 | ICR26 | $45A_H$ | $2EC_H$ | $000FFEEC_H$ | 12, 58 |
| System reserved | 69 | 45 | | | $2E8_H$ | $000FFEE8_H$ | 13, 59 |

| Interrupt | Interrupt number | | Interrupt level*1 | | Interrupt vector*2 | | Resource number |
|---|---|---|---|---|---|---|---|
| | Decimal | Hexa-decimal | Setting Register | Register address | Offset | Default Vector address | |
| LIN-USART (FIFO) 6 RX | 70 | 46 | ICR27 | $45B_H$ | $2E4_H$ | $000FFEE4_H$ | 60 |
| LIN-USART (FIFO) 6 TX | 71 | 47 | | | $2E0_H$ | $000FFEE0_H$ | 61 |
| LIN-USART (FIFO) 7 RX | 72 | 48 | ICR28 | $45C_H$ | $2DC_H$ | $000FFEDC_H$ | 62 |
| LIN-USART (FIFO) 7 TX | 73 | 49 | | | $2D8_H$ | $000FFED8_H$ | 63 |
| $I^2C$ 0 | 74 | 4A | ICR29 | $45D_H$ | $2D4_H$ | $000FFED4_H$ | |
| System reserved | 75 | 4B | | | $2D0_H$ | $000FFED0_H$ | |
| System reserved | 76 | 4C | ICR30 | $45E_H$ | $2CC_H$ | $000FFECC_H$ | 64 |
| System reserved | 77 | 4D | | | $2C8_H$ | $000FFEC8_H$ | 65 |
| System reserved | 78 | 4E | ICR31 | $45F_H$ | $2C4_H$ | $000FFEC4_H$ | 66 |
| System reserved | 79 | 4F | | | $2C0_H$ | $000FFEC0_H$ | 67 |
| System reserved | 80 | 50 | ICR32 | $460_H$ | $2BC_H$ | $000FFEBC_H$ | 68 |
| System reserved | 81 | 51 | | | $2B8_H$ | $000FFEB8_H$ | 69 |
| System reserved | 82 | 52 | ICR33 | $461_H$ | $2B4_H$ | $000FFEB4_H$ | 70 |
| System reserved | 83 | 53 | | | $2B0_H$ | $000FFEB0_H$ | 71 |
| FlexRay 0 | 84 | 54 | ICR34 | $462_H$ | $2AC_H$ | $000FFEAC_H$ | 72<br>116 (IBF)<br>117 (OBF) |
| FlexRay Timer 0 | 85 | 55 | | | $2A8_H$ | $000FFEA8_H$ | 73 |
| FlexRay 1 | 86 | 56 | ICR35 | $463_H$ | $2A4_H$ | $000FFEA4_H$ | 74<br>116 (IBF)<br>117 (OBF) |
| FlexRay Timer 1 | 87 | 57 | | | $2A0_H$ | $000FFEA0_H$ | 75 |
| System reserved | 88 | 58 | ICR36 | $464_H$ | $29C_H$ | $000FFE9C_H$ | 76 |
| System reserved | 89 | 59 | | | $298_H$ | $000FFE98_H$ | 77 |
| System reserved | 90 | 5A | ICR37 | $465_H$ | $294_H$ | $000FFE94_H$ | 78 |
| System reserved | 91 | 5B | | | $290_H$ | $000FFE90_H$ | 79 |
| Input Capture 0 | 92 | 5C | ICR38 | $466_H$ | $28C_H$ | $000FFE8C_H$ | 80 |
| Input Capture 1 | 93 | 5D | | | $288_H$ | $000FFE88_H$ | 81 |
| Input Capture 2 | 94 | 5E | ICR39 | $467_H$ | $284_H$ | $000FFE84_H$ | 82 |
| Input Capture 3 | 95 | 5F | | | $280_H$ | $000FFE80_H$ | 83 |
| Input Capture 4 | 96 | 60 | ICR40 | $468_H$ | $27C_H$ | $000FFE7C_H$ | 84 |
| Input Capture 5 | 97 | 61 | | | $278_H$ | 000FFE78 | 85 |
| Input Capture 6 | 98 | 62 | ICR41 | $469_H$ | $274_H$ | 000FFE74 | 86 |
| Input Capture 7 | 99 | 63 | | | $270_H$ | 000FFE70 | 87 |
| Output Compare 0 | 100 | 64 | ICR42 | $46A_H$ | $26C_H$ | 000FFE6C | 88 |
| Output Compare 1 | 101 | 65 | | | $268_H$ | 000FFE68 | 89 |
| Output Compare 2 | 102 | 66 | ICR43 | $46B_H$ | $264_H$ | 000FFE64 | 90 |
| Output Compare 3 | 103 | 67 | | | $260_H$ | 000FFE60 | 91 |

| Interrupt | Interrupt number | | Interrupt level*1 | | Interrupt vector*2 | | Resource number |
|---|---|---|---|---|---|---|---|
| | Decimal | Hexa-decimal | Setting Register | Register address | Offset | Default Vector address | |
| Output Compare 4 | 104 | 68 | ICR44 | $46C_H$ | $25C_H$ | 000FFE5C | 92 |
| Output Compare 5 | 105 | 69 | | | $258_H$ | 000FFE58 | 93 |
| System reserved | 106 | 6A | ICR45 | $46D_H$ | $254_H$ | 000FFE54 | 94 |
| System reserved | 107 | 6B | | | $250_H$ | 000FFE50 | 95 |
| System reserved | 108 | 6C | ICR46 | $46E_H$ | $24C_H$ | 000FFE4C | |
| System reserved | 109 | 6D | | | $248_H$ | 000FFE48 | |
| System reserved | 110 | 6E | ICR47 *4 | $46F_H$ | $244_H$ | 000FFE44 | |
| System reserved | 111 | 6F | | | $240_H$ | 000FFE40 | |
| PPG0 | 112 | 70 | ICR48 | $470_H$ | $23C_H$ | 000FFE3C | 15, 96 |
| PPG1 | 113 | 71 | | | $238_H$ | 000FFE38 | 97 |
| PPG2 | 114 | 72 | ICR49 | $471_H$ | $234_H$ | 000FFE34 | 98 |
| PPG3 | 115 | 73 | | | $230_H$ | 000FFE30 | 99 |
| PPG4 | 116 | 74 | ICR50 | $472_H$ | $22C_H$ | 000FFE2C | 100 |
| PPG5 | 117 | 75 | | | $228_H$ | 000FFE28 | 101 |
| PPG6 | 118 | 76 | ICR51 | $473_H$ | $224_H$ | 000FFE24 | 102 |
| PPG7 | 119 | 77 | | | $220_H$ | 000FFE20 | 103 |
| PPG8 | 120 | 78 | ICR52 | $474_H$ | $21C_H$ | 000FFE1C | 104 |
| PPG9 | 121 | 79 | | | $218_H$ | 000FFE18 | 105 |
| PPG10 | 122 | 7A | ICR53 | $475_H$ | $214_H$ | 000FFE14 | 106 |
| PPG11 | 123 | 7B | | | $210_H$ | 000FFE10 | 107 |
| System reserved | 124 | 7C | ICR54 | $476_H$ | $20C_H$ | 000FFE0C | 108 |
| System reserved | 125 | 7D | | | $208_H$ | 000FFE08 | 109 |
| System reserved | 126 | 7E | ICR55 | $477_H$ | $204_H$ | 000FFE04 | 110 |
| System reserved | 127 | 7F | | | $200_H$ | 000FFE00 | 111 |
| System reserved | 128 | 80 | ICR56 | $478_H$ | $1FC_H$ | 000FFDFC | |
| System reserved | 129 | 81 | | | $1F8_H$ | 000FFDF8 | |
| System reserved | 130 | 82 | ICR57 | $479_H$ | $1F4_H$ | 000FFDF4 | |
| System reserved | 131 | 83 | | | $1F0_H$ | 000FFDF0 | |
| Real Time Clock | 132 | 84 | ICR58 | $47A_H$ | $1EC_H$ | 000FFDEC | |
| Calibration Unit | 133 | 85 | | | $1E8_H$ | 000FFDE8 | |
| A/D Converter 0 | 134 | 86 | ICR59 | $47B_H$ | $1E4_H$ | 000FFDE4 | 14, 112 |
| System reserved | 135 | 87 | | | $1E0_H$ | 000FFDE0 | |
| System reserved | 136 | 88 | ICR60 | $47C_H$ | $1DC_H$ | 000FFDDC | |
| System reserved | 137 | 89 | | | $1D8_H$ | 000FFDD8 | |
| Low Voltage Detection | 138 | 8A | ICR61 | $47D_H$ | $1D4_H$ | 000FFDD4 | |
| System reserved | 139 | 8B | | | $1D0_H$ | 000FFDD0 | |

| Interrupt | Interrupt number | | Interrupt level*1 | | Interrupt vector*2 | | Resource number |
|---|---|---|---|---|---|---|---|
| | Decimal | Hexa-decimal | Setting Register | Register address | Offset | Default Vector address | |
| Timebase Overflow | 140 | 8C | ICR62 | $47E_H$ | $1CC_H$ | 000FFDCC | |
| PLL Clock Gear / PLL2 Clock Gear (FlexRay) | 141 | 8D | | | $1C8_H$ | 000FFDC8 | |
| DMA Controller | 142 | 8E | ICR63 | $47F_H$ | $1C4_H$ | 000FFDC4 | |
| Main/Sub OSC stability wait | 143 | 8F | | | $1C0_H$ | 000FFDC0 | |
| Security vector | 144 | 90 | — | — | $1BC_H$ | 000FFDBC | |
| Used by the INT instruction. | 145 to 255 | 91 to FF | — | — | $1B8_H$ to $000_H$ | 000FFDB8 to 000FFC00 | |

*1 : The ICRs are located in the interrupt controller and set the interrupt level for each interrupt request. An ICR is provided for each interrupt request.

*2 : The vector address for each EIT (exception, interrupt or trap) is calculated by adding the listed offset to the table base register value (TBR) . The TBR specifies the top of the EIT vector table. The addresses listed in the table are for the default TBR value ($000FFC00_H$) . The TBR is initialized to this value by a reset. The TBR is set to $000FFC00_H$ after the internal boot ROM is executed.

*3 : ICR23 and ICR47 can be exchanged by setting the REALOS compatibility bit (addr $0C03_H$ : IOS[0])

*4 : Used by REALOS

# 14. Recommended Settings

## 14.1. PLL and Clockgear settings

Please note that for MB91F465XA the core base clock frequencies are valid in the 1.8V operation mode of the Main regulator and Flash.

### Recommended PLL divider and clockgear settings

| PLL Input (CLK) [MHz] | Frequency Parameter | | Clockgear Parameter | | PLL Output (X) [MHz] | Core Base Clock [MHz] | Remarks |
|---|---|---|---|---|---|---|---|
| | DIVM | DIVN | DIVG | MULG | MULG | | |
| 4 | 2 | 25 | 16 | 24 | 200 | 100 | |
| 4 | 2 | 24 | 16 | 24 | 192 | 96 | |
| 4 | 2 | 23 | 16 | 24 | 184 | 92 | |
| 4 | 2 | 22 | 16 | 24 | 176 | 88 | |
| 4 | 2 | 21 | 16 | 20 | 168 | 84 | |
| 4 | 2 | 20 | 16 | 20 | 160 | 80 | |
| 4 | 2 | 19 | 16 | 20 | 152 | 76 | |
| 4 | 2 | 18 | 16 | 20 | 144 | 72 | |
| 4 | 2 | 17 | 16 | 16 | 136 | 68 | |
| 4 | 2 | 16 | 16 | 16 | 128 | 64 | |
| 4 | 2 | 15 | 16 | 16 | 120 | 60 | |
| 4 | 2 | 14 | 16 | 16 | 112 | 56 | |
| 4 | 2 | 13 | 16 | 12 | 104 | 52 | |
| 4 | 2 | 12 | 16 | 12 | 96 | 48 | |
| 4 | 2 | 11 | 16 | 12 | 88 | 44 | |
| 4 | 4 | 10 | 16 | 24 | 160 | 40 | |
| 4 | 4 | 9 | 16 | 24 | 144 | 36 | |
| 4 | 4 | 8 | 16 | 24 | 128 | 32 | |
| 4 | 4 | 7 | 16 | 24 | 112 | 28 | |
| 4 | 6 | 6 | 16 | 24 | 144 | 24 | |
| 4 | 8 | 5 | 16 | 28 | 160 | 20 | |
| 4 | 10 | 4 | 16 | 32 | 160 | 16 | |
| 4 | 12 | 3 | 16 | 32 | 144 | 12 | |

## 14.2. Clock Modulator settings

The following table shows all possible settings for the Clock Modulator in a base clock frequency range from 32MHz up to 88MHz.

The Flash access time settings need to be adjusted according to Fmax while the PLL and clockgear settings should be set according to base clock frequency.

**Clock Modulator settings, frequency range and supported supply voltage**

| Modulation Degree (k) | Random No (N) | CMPR [hex] | Baseclk [MHz] | Fmin [MHz] | Fmax [MHz] | Remarks |
|---|---|---|---|---|---|---|
| 1 | 3 | 026F | 88 | 79.5 | 98.5 | |
| 1 | 3 | 026F | 84 | 76.1 | 93.8 | |
| 1 | 3 | 026F | 80 | 72.6 | 89.1 | |
| 1 | 5 | 02AE | 80 | 68.7 | 95.8 | |
| 2 | 3 | 046E | 80 | 68.7 | 95.8 | |
| 1 | 3 | 026F | 76 | 69.1 | 84.5 | |
| 1 | 5 | 02AE | 76 | 65.3 | 90.8 | |
| 1 | 7 | 02ED | 76 | 62 | 98.1 | |
| 2 | 3 | 046E | 76 | 65.3 | 90.8 | |
| 3 | 3 | 066D | 76 | 62 | 98.1 | |
| 1 | 3 | 026F | 72 | 65.5 | 79.9 | |
| 1 | 5 | 02AE | 72 | 62 | 85.8 | |
| 1 | 7 | 02ED | 72 | 58.8 | 92.7 | |
| 2 | 3 | 046E | 72 | 62 | 85.8 | |
| 3 | 3 | 066D | 72 | 58.8 | 92.7 | |
| 1 | 3 | 026F | 68 | 62 | 75.3 | |
| 1 | 5 | 02AE | 68 | 58.7 | 80.9 | |
| 1 | 7 | 02ED | 68 | 55.7 | 87.3 | |
| 1 | 9 | 032C | 68 | 53 | 95 | |
| 2 | 3 | 046E | 68 | 58.7 | 80.9 | |
| 2 | 5 | 04AC | 68 | 53 | 95 | |
| 3 | 3 | 066D | 68 | 55.7 | 87.3 | |
| 4 | 3 | 086C | 68 | 53 | 95 | |
| 1 | 3 | 026F | 64 | 58.5 | 70.7 | |
| 1 | 5 | 02AE | 64 | 55.3 | 75.9 | |
| 1 | 7 | 02ED | 64 | 52.5 | 82 | |
| 1 | 9 | 032C | 64 | 49.9 | 89.1 | |
| 1 | 11 | 036B | 64 | 47.6 | 97.6 | |
| 2 | 3 | 046E | 64 | 55.3 | 75.9 | |
| 2 | 5 | 04AC | 64 | 49.9 | 89.1 | |
| 3 | 3 | 066D | 64 | 52.5 | 82 | |
| 4 | 3 | 086C | 64 | 49.9 | 89.1 | |
| 5 | 3 | 0A6B | 64 | 47.6 | 97.6 | |

| Modulation Degree (k) | Random No (N) | CMPR [hex] | Baseclk [MHz] | Fmin [MHz] | Fmax [MHz] | Remarks |
|---|---|---|---|---|---|---|
| 1 | 3 | 026F | 60 | 54.9 | 66.1 | |
| 1 | 5 | 02AE | 60 | 51.9 | 71 | |
| 1 | 7 | 02ED | 60 | 49.3 | 76.7 | |
| 1 | 9 | 032C | 60 | 46.9 | 83.3 | |
| 1 | 11 | 036B | 60 | 44.7 | 91.3 | |
| 2 | 3 | 046E | 60 | 51.9 | 71 | |
| 2 | 5 | 04AC | 60 | 46.9 | 83.3 | |
| 3 | 3 | 066D | 60 | 49.3 | 76.7 | |
| 4 | 3 | 086C | 60 | 46.9 | 83.3 | |
| 5 | 3 | 0A6B | 60 | 44.7 | 91.3 | |
| 1 | 3 | 026F | 56 | 51.4 | 61.6 | |
| 1 | 5 | 02AE | 56 | 48.6 | 66.1 | |
| 1 | 7 | 02ED | 56 | 46.1 | 71.4 | |
| 1 | 9 | 032C | 56 | 43.8 | 77.6 | |
| 1 | 11 | 036B | 56 | 41.8 | 84.9 | |
| 1 | 13 | 03AA | 56 | 39.9 | 93.8 | |
| 2 | 3 | 046E | 56 | 48.6 | 66.1 | |
| 2 | 5 | 04AC | 56 | 43.8 | 77.6 | |
| 2 | 7 | 04EA | 56 | 39.9 | 93.8 | |
| 3 | 3 | 066D | 56 | 46.1 | 71.4 | |
| 3 | 5 | 06AA | 56 | 39.9 | 93.8 | |
| 4 | 3 | 086C | 56 | 43.8 | 77.6 | |
| 5 | 3 | 0A6B | 56 | 41.8 | 84.9 | |
| 6 | 3 | 0C6A | 56 | 39.9 | 93.8 | |
| 1 | 3 | 026F | 52 | 47.8 | 57 | |
| 1 | 5 | 02AE | 52 | 45.2 | 61.2 | |
| 1 | 7 | 02ED | 52 | 42.9 | 66.1 | |
| 1 | 9 | 032C | 52 | 40.8 | 71.8 | |
| 1 | 11 | 036B | 52 | 38.8 | 78.6 | |
| 1 | 13 | 03AA | 52 | 37.1 | 86.8 | |
| 1 | 15 | 03E9 | 52 | 35.5 | 96.9 | |
| 2 | 3 | 046E | 52 | 45.2 | 61.2 | |
| 2 | 5 | 04AC | 52 | 40.8 | 71.8 | |
| 2 | 7 | 04EA | 52 | 37.1 | 86.8 | |
| 3 | 3 | 066D | 52 | 42.9 | 66.1 | |
| 3 | 5 | 06AA | 52 | 37.1 | 86.8 | |
| 4 | 3 | 086C | 52 | 40.8 | 71.8 | |
| 5 | 3 | 0A6B | 52 | 38.8 | 78.6 | |

| Modulation Degree (k) | Random No (N) | CMPR [hex] | Baseclk [MHz] | Fmin [MHz] | Fmax [MHz] | Remarks |
|---|---|---|---|---|---|---|
| 6 | 3 | 0C6A | 52 | 37.1 | 86.8 | |
| 7 | 3 | 0E69 | 52 | 35.5 | 96.9 | |
| 1 | 3 | 026F | 48 | 44.2 | 52.5 | |
| 1 | 5 | 02AE | 48 | 41.8 | 56.4 | |
| 1 | 7 | 02ED | 48 | 39.6 | 60.9 | |
| 1 | 9 | 032C | 48 | 37.7 | 66.1 | |
| 1 | 11 | 036B | 48 | 35.9 | 72.3 | |
| 1 | 13 | 03AA | 48 | 34.3 | 79.9 | |
| 1 | 15 | 03E9 | 48 | 32.8 | 89.1 | |
| 2 | 3 | 046E | 48 | 41.8 | 56.4 | |
| 2 | 5 | 04AC | 48 | 37.7 | 66.1 | |
| 2 | 7 | 04EA | 48 | 34.3 | 79.9 | |
| 3 | 3 | 066D | 48 | 39.6 | 60.9 | |
| 3 | 5 | 06AA | 48 | 34.3 | 79.9 | |
| 4 | 3 | 086C | 48 | 37.7 | 66.1 | |
| 5 | 3 | 0A6B | 48 | 35.9 | 72.3 | |
| 6 | 3 | 0C6A | 48 | 34.3 | 79.9 | |
| 7 | 3 | 0E69 | 48 | 32.8 | 89.1 | |
| 1 | 3 | 026F | 44 | 40.6 | 48.1 | |
| 1 | 5 | 02AE | 44 | 38.4 | 51.6 | |
| 1 | 7 | 02ED | 44 | 36.4 | 55.7 | |
| 1 | 9 | 032C | 44 | 34.6 | 60.4 | |
| 1 | 11 | 036B | 44 | 33 | 66.1 | |
| 1 | 13 | 03AA | 44 | 31.5 | 73 | |
| 1 | 15 | 03E9 | 44 | 30.1 | 81.4 | |
| 2 | 3 | 046E | 44 | 38.4 | 51.6 | |
| 2 | 5 | 04AC | 44 | 34.6 | 60.4 | |
| 2 | 7 | 04EA | 44 | 31.5 | 73 | |
| 2 | 9 | 0528 | 44 | 28.9 | 92.1 | |
| 3 | 3 | 066D | 44 | 36.4 | 55.7 | |
| 3 | 5 | 06AA | 44 | 31.5 | 73 | |
| 4 | 3 | 086C | 44 | 34.6 | 60.4 | |
| 4 | 5 | 08A8 | 44 | 28.9 | 92.1 | |
| 5 | 3 | 0A6B | 44 | 33 | 66.1 | |
| 6 | 3 | 0C6A | 44 | 31.5 | 73 | |
| 7 | 3 | 0E69 | 44 | 30.1 | 81.4 | |
| 8 | 3 | 1068 | 44 | 28.9 | 92.1 | |
| 1 | 3 | 026F | 40 | 37 | 43.6 | |

| Modulation Degree (k) | Random No (N) | CMPR [hex] | Baseclk [MHz] | Fmin [MHz] | Fmax [MHz] | Remarks |
|---|---|---|---|---|---|---|
| 1 | 5 | 02AE | 40 | 34.9 | 46.8 | |
| 1 | 7 | 02ED | 40 | 33.1 | 50.5 | |
| 1 | 9 | 032C | 40 | 31.5 | 54.8 | |
| 1 | 11 | 036B | 40 | 30 | 59.9 | |
| 1 | 13 | 03AA | 40 | 28.7 | 66.1 | |
| 1 | 15 | 03E9 | 40 | 27.4 | 73.7 | |
| 2 | 3 | 046E | 40 | 34.9 | 46.8 | |
| 2 | 5 | 04AC | 40 | 31.5 | 54.8 | |
| 2 | 7 | 04EA | 40 | 28.7 | 66.1 | |
| 2 | 9 | 0528 | 40 | 26.3 | 83.3 | |
| 3 | 3 | 066D | 40 | 33.1 | 50.5 | |
| 3 | 5 | 06AA | 40 | 28.7 | 66.1 | |
| 3 | 7 | 06E7 | 40 | 25.3 | 95.8 | |
| 4 | 3 | 086C | 40 | 31.5 | 54.8 | |
| 4 | 5 | 08A8 | 40 | 26.3 | 83.3 | |
| 5 | 3 | 0A6B | 40 | 30 | 59.9 | |
| 6 | 3 | 0C6A | 40 | 28.7 | 66.1 | |
| 7 | 3 | 0E69 | 40 | 27.4 | 73.7 | |
| 8 | 3 | 1068 | 40 | 26.3 | 83.3 | |
| 9 | 3 | 1267 | 40 | 25.3 | 95.8 | |
| 1 | 3 | 026F | 36 | 33.3 | 39.2 | |
| 1 | 5 | 02AE | 36 | 31.5 | 42 | |
| 1 | 7 | 02ED | 36 | 29.9 | 45.3 | |
| 1 | 9 | 032C | 36 | 28.4 | 49.2 | |
| 1 | 11 | 036B | 36 | 27.1 | 53.8 | |
| 1 | 13 | 03AA | 36 | 25.8 | 59.3 | |
| 1 | 15 | 03E9 | 36 | 24.7 | 66.1 | |
| 2 | 3 | 046E | 36 | 31.5 | 42 | |
| 2 | 5 | 04AC | 36 | 28.4 | 49.2 | |
| 2 | 7 | 04EA | 36 | 25.8 | 59.3 | |
| 2 | 9 | 0528 | 36 | 23.7 | 74.7 | |
| 3 | 3 | 066D | 36 | 29.9 | 45.3 | |
| 3 | 5 | 06AA | 36 | 25.8 | 59.3 | |
| 3 | 7 | 06E7 | 36 | 22.8 | 85.8 | |
| 4 | 3 | 086C | 36 | 28.4 | 49.2 | |
| 4 | 5 | 08A8 | 36 | 23.7 | 74.7 | |
| 5 | 3 | 0A6B | 36 | 27.1 | 53.8 | |
| 6 | 3 | 0C6A | 36 | 25.8 | 59.3 | |

| Modulation Degree (k) | Random No (N) | CMPR [hex] | Baseclk [MHz] | Fmin [MHz] | Fmax [MHz] | Remarks |
|---|---|---|---|---|---|---|
| 7 | 3 | 0E69 | 36 | 24.7 | 66.1 | |
| 8 | 3 | 1068 | 36 | 23.7 | 74.7 | |
| 9 | 3 | 1267 | 36 | 22.8 | 85.8 | |
| 1 | 3 | 026F | 32 | 29.7 | 34.7 | |
| 1 | 5 | 02AE | 32 | 28 | 37.3 | |
| 1 | 7 | 02ED | 32 | 26.6 | 40.2 | |
| 1 | 9 | 032C | 32 | 25.3 | 43.6 | |
| 1 | 11 | 036B | 32 | 24.1 | 47.7 | |
| 1 | 13 | 03AA | 32 | 23 | 52.5 | |
| 1 | 15 | 03E9 | 32 | 22 | 58.6 | |
| 2 | 3 | 046E | 32 | 28 | 37.3 | |
| 2 | 5 | 04AC | 32 | 25.3 | 43.6 | |
| 2 | 7 | 04EA | 32 | 23 | 52.5 | |
| 2 | 9 | 0528 | 32 | 21.1 | 66.1 | |
| 2 | 11 | 0566 | 32 | 19.5 | 89.1 | |
| 3 | 3 | 066D | 32 | 26.6 | 40.2 | |
| 3 | 5 | 06AA | 32 | 23 | 52.5 | |
| 3 | 7 | 06E7 | 32 | 20.3 | 75.9 | |
| 4 | 3 | 086C | 32 | 25.3 | 43.6 | |
| 4 | 5 | 08A8 | 32 | 21.1 | 66.1 | |
| 5 | 3 | 0A6B | 32 | 24.1 | 47.7 | |
| 5 | 5 | 0AA6 | 32 | 19.5 | 89.1 | |
| 6 | 3 | 0C6A | 32 | 23 | 52.5 | |
| 7 | 3 | 0E69 | 32 | 22 | 58.6 | |
| 8 | 3 | 1068 | 32 | 21.1 | 66.1 | |
| 9 | 3 | 1267 | 32 | 20.3 | 75.9 | |
| 10 | 3 | 1466 | 32 | 19.5 | 89.1 | |

## 14.3. FlexRay PLL, Clock and Port settings

| | | | | | |
|---|---|---|---|---|---|
| $0004DC_H$ | PLL2DIVM [R/W]<br>- - - - 0000 | PLL2DIVN [R/W]<br>- - 000000 | PLL2DIVG [R/W]<br>- - - - 0000 | PLL2MULG [R/W]<br>00000000 | PLL2 Clock Control (FlexRay) |
| $0004E0_H$ | PLL2CTRL [R/W]<br>- - - - 0000 | Reserved | CLKR2 [R/W]<br>- - - 00000 | Reserved | |

### *14.3.1. Recommended FlexRay PLL divider and clockgear settings*

| PLL Input (CK) [MHz] | Frequency Parameter | | Clockgear Parameter | | PLL2 Output (X) [MHz] | FlexRay SCLK Clock [MHz] |
|---|---|---|---|---|---|---|
| | DIVM2 | DIVN2 | DIVG2 | MULG2 | | |
| 4 | 2 | 20 | 0 | 0 | 160 | 80 |

Note: It is recommended to follow the same PLL startup procedure and lock wait time as recommended for the main PLL1.

### *14.3.2. Recommended FlexRay clock settings*

Specification of the CLKR2 registers is as follows:

REGISTER CLKR2 addr 0x04E2

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | bit |
|---|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| CLKR2 | | | | | CKDBL | PLL2EN | CLKS1 | CLKS0 | |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Initial |
| | | | | | R/W | R/W | R/W | R/W | Attribute |

Note: It is recommended to follow the same PLL startup procedure and lock wait time as recommended for the main PLL1.

**CLKS1, CLKS0:** FlexRay SCLK Source Selection

CLKS[1:0] = 00 : SCLK is operated with CLKB (core clock)
CLKS[1:0] = 01 : SCLK is operated with Main PLL (base clock)
CLKS[1:0] = 10 : SCLK is operated with FlexRay PLL **<- recommended setting**
CLKS[1:0] = 11 : Test mode only, do not set !

**PLL2EN:** FlexRay PLL enable

PLL2EN = 0 : FlexRay PLL is disabled
PLL2EN = 1 : FlexRay PLL is enabled

**CKDBL:** FlexRay Clock disable (BCLK and SCLK)

CKDBL = 0 : FlexRay clocks are enabled
CKDBL = 1 : FlexRay clocks are disabled

*14.3.3. Recommended FlexRay port settings*

**Channel A**

**TXDA:** FlexRay channel A transmit (P31_0)
To enable TXDA set PFR31_0 = 1 and EPFR31_0 = 1

**TXENA:** FlexRay channel A enable (P31_1)
To enable TXENA set PFR31_1 = 1 and EPFR31_1 = 1

**RXDA:** FlexRay channel A receive (P31_2)
To enable RXDA set PFR31_2 = 1 and EPFR31_2 = 1

**Channel B**

**TXDB:** FlexRay channel B transmit (P31_4)
To enable TXDB set PFR31_4 = 1 and EPFR31_4 = 1

**TXENB:** FlexRay channel B enable (P31_5)
To enable TXENB set PFR31_5 = 1 and EPFR31_5 = 1

**RXDB:** FlexRay channel B receive (P31_6)
To enable RXDB set PFR31_6 = 1 and EPFR31_6 = 1

# 15. Electrical Characteristics

## 15.1. Absolute maximum ratings

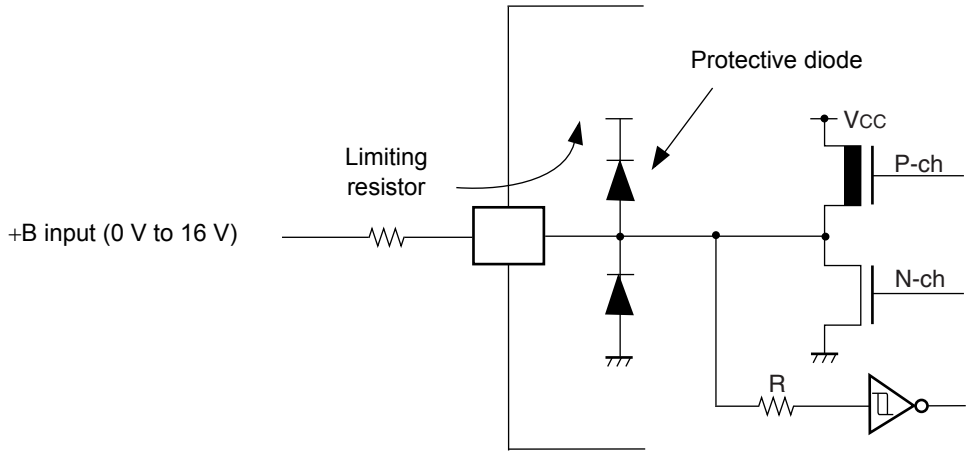
| Parameter | Symbol | Rating | | Unit | Remarks |
|---|---|---|---|---|---|
| | | Min | Max | | |
| Power supply slew rate | — | — | 50 | V/ms | |
| Power supply voltage 1*1 | $V_{DD}5R$ | − 0.3 | + 6.0 | V | |
| Power supply voltage 2*1 | $V_{DD}5$ | − 0.3 | + 6.0 | V | |
| Relationship of the supply voltages | $AV_{CC}5$ | $V_{DD}5$-0.3 | $V_{DD}5$+0.3 | V | At least one pin of the Ports 27 to 29 (ANn) is used as digital input or output. |
| | | $V_{SS}5$-0.3 | $V_{DD}5$+0.3 | V | All pins of the Ports 27 to 29 (ANn) follow the condition of $V_{IA}$ |
| Analog power supply voltage*1 | $AV_{CC}5$ | − 0.3 | + 6.0 | V | *2 |
| Analog reference power supply voltage*1 | AVRH5 | − 0.3 | + 6.0 | V | *2 |
| Input voltage 1*1 | $V_{I1}$ | Vss5 − 0.3 | $V_{DD}5$ + 0.3 | V | |
| Analog pin input voltage*1 | $V_{IA}$ | AVss5 − 0.3 | AVcc5 + 0.3 | V | |
| Output voltage 1*1 | $V_{O1}$ | Vss5 − 0.3 | $V_{DD}5$ + 0.3 | V | |
| Maximum clamp current | $I_{CLAMP}$ | − 4.0 | + 4.0 | mA | *3 |
| Total maximum clamp current | $\Sigma \lvert I_{CLAMP} \rvert$ | — | 20 | mA | *3 |
| “L” level maximum output current*4 | $I_{OL}$ | — | 10 | mA | |
| “L” level average output current*5 | $I_{OLAV}$ | — | 8 | mA | |
| “L” level total maximum output current | $\Sigma I_{OL}$ | — | 100 | mA | |
| “L” level total average output current*6 | $\Sigma I_{OLAV}$ | — | 50 | mA | |
| “H” level maximum output current*4 | $I_{OH}$ | — | − 10 | mA | |
| “H” level average output current*5 | $I_{OHAV}$ | — | − 4 | mA | |
| “H” level total maximum output current | $\Sigma I_{OH}$ | — | − 100 | mA | |
| “H” level total average output current*6 | $\Sigma I_{OHAV}$ | — | − 25 | mA | |
| Power consumption | $P_D$ | — | 1000 | mW | |
| Operating temperature | $T_A$ | − 40 | + 105 | °C | |
| Storage temperature | Tstg | − 55 | + 150 | °C | |

*1 : The parameter is based on $V_{SS}5 = HV_{SS}5 = AV_{SS}5 = 0.0$ V.

*2 : $AV_{CC}5$ and AVRH5 must not exceed $V_{DD}5 + 0.3$ V.

*3 : • Use within recommended operating conditions.
- Use with DC voltage (current).
- + B signals are input signals that exceed the $V_{DD}5$ voltage. +B signals should always be applied by connecting a limiting resistor between the +B signal and the microcontroller.
- The value of the limiting resistor should be set so that the current input to the microcontroller pin does not exceed the rated value at any time, either instantaneously or for an extended period, when the +B signal is input.
- Note that when the microcontroller drive current is low, such as in the low power consumption modes, the +B input potential can increase the potential at the power supply pin via a protective diode, possibly affecting other devices.
- Note that if the +B signal is input when the microcontroller is off (not fixed at 0 V), power is supplied through the +B input pin; therefore, the microcontroller may partially operate.
- Note that if the +B signal is input at power-on, since the power is supplied through the pin, the power-on reset may not function in the power supply voltage.
- Do not leave +B input pins open.
- Example of recommended circuit:

**Input/output equivalent circuit**




*4 : Maximum output current is defined as the value of the peak current flowing through any one of the corresponding pins.

*5 : Average output current is defined as the value of the average current flowing through any one of the corresponding pins for a 100 ms period.

*6 : Total average output current is defined as the value of the average current flowing through all of the corresponding pins for a 100 ms period.

WARNING: Semiconductor devices can be permanently damaged by application of stress (voltage, current, temperature, etc.) in excess of absolute maximum ratings. Do not exceed these ratings.

## 15.2. Recommended operating conditions

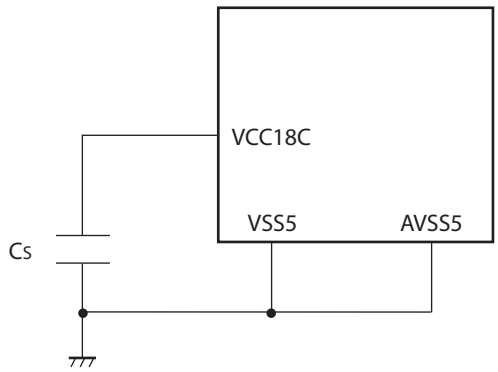
($V_{SS}5 = AV_{SS}5 = 0.0$ V)

| Parameter | Symbol | Value | | | Unit | Remarks |
|---|---|---|---|---|---|---|
| | | Min | Typ | Max | | |
| Power supply voltage | $V_{DD}5$ | 3.0 | — | 5.5 | V | |
| | $V_{DD}5R$ | 3.0 | — | 5.5 | V | Internal regulator |
| | $AV_{CC}5$ | 3.0 | — | 5.5 | V | A/D converter |
| Smoothing capacitor at VCC18C pin | $C_S$ | — | 4.7 | — | µF | Use a X7R ceramic capacitor or a capacitor that has similar frequency characteristics. |
| Power supply slew rate | | — | — | 50 | V/ms | |
| Operating temperature | $T_A$ | − 40 | — | + 105 | °C | |
| Main Oscillation Stabilization time | | 10 | | | ms | |
| Lock-up time PLL (4 MHz ->16 ...100MHz) | | | | 0.6 | ms | |
| ESD Protection (Human body model) | $V_{surge}$ | 2 | | | kV | $R_{discharge}$ = 1.5kΩ<br>$C_{discharge}$ = 100pF |
| RC Oscillator | $f_{RC100kHz}$<br>$f_{RC2MHz}$ | 50<br>1 | 100<br>2 | 200<br>4 | kHz<br>MHz | $VDD_{CORE}$ ≥ 1.65V |

WARNING: The recommended operating conditions are required in order to ensure the normal operation of the semiconductor device. All of the device's electrical characteristics are warranted when the device is operated within these ranges.

Always use semiconductor devices within their recommended operating condition ranges. Operation outside these ranges may adversely affect reliability and could result in device failure.

No warranty is made with respect to uses, operating conditions, or combinations not represented on the data sheet. Users considering application outside the listed conditions are advised to contact their representatives beforehand.



### 15.3. DC characteristics

Note: In the following tables, "$V_{SS}$" means $V_{SS}5$ for the other pins.

($V_{DD}5 = AV_{CC}5 = 3.0$ V to 5.5 V, $V_{SS}5 = AV_{SS}5 = 0$ V, $T_A = -40°C$ to $+105°C$)

| Parameter | Symbol | Pin name | Condition | Value | | | Unit | Remarks |
|---|---|---|---|---|---|---|---|---|
| | | | | Min | Typ | Max | | |
| Input "H" voltage | $V_{IH}$ | — | Port inputs if CMOS Hysteresis 0.8/0.2 input is selected | $0.8 \times V_{DD}$ | — | $V_{DD} + 0.3$ | V | CMOS hysteresis input |
| | | — | Port inputs if CMOS Hysteresis 0.7/0.3 input is selected | $0.7 \times V_{DD}$ | — | $V_{DD} + 0.3$ | V | 4.5 V ≤ $V_{DD}$ ≤ 5.5 V |
| | | | | $0.74 \times V_{DD}$ | — | $V_{DD} + 0.3$ | V | 3 V ≤ $V_{DD}$ < 4.5 V |
| | | — | AUTOMOTIVE Hysteresis input is selected | $0.8 \times V_{DD}$ | — | $V_{DD} + 0.3$ | V | |
| | | — | Port inputs if TTL input is selected | 2.0 | — | $V_{DD} + 0.3$ | V | |
| | $V_{IHR}$ | INITX | — | $0.8 \times V_{DD}$ | — | $V_{DD} + 0.3$ | V | INITX input pin (CMOS Hysteresis) |
| | $V_{IHM}$ | MD_2 to MD_0 | — | $V_{DD} - 0.3$ | — | $V_{DD} + 0.3$ | V | Mode input pins |
| | $V_{IHX0S}$ | X0, X0A | — | 2.5 | — | $V_{DD} + 0.3$ | V | External clock in "Oscillation mode" |
| | $V_{IHX0F}$ | X0 | — | $0.8 \times V_{DD}$ | — | $V_{DD} + 0.3$ | V | External clock in "Fast Clock Input mode" |
| Input "L" voltage | $V_{IL}$ | — | Port inputs if CMOS Hysteresis 0.8/0.2 input is selected | $V_{SS} - 0.3$ | — | $0.2 \times V_{DD}$ | V | |
| | | — | Port inputs if CMOS Hysteresis 0.7/0.3 input is selected | $V_{SS} - 0.3$ | — | $0.3 \times V_{DD}$ | V | |
| | | — | Port inputs if AUTOMOTIVE Hysteresis input is selected | $V_{SS} - 0.3$ | — | $0.5 \times V_{DD}$ | V | 4.5 V ≤ $V_{DD}$ ≤ 5.5 V |
| | | | | $V_{SS} - 0.3$ | — | $0.46 \times V_{DD}$ | V | 3 V ≤ $V_{DD}$ < 4.5 V |
| | | — | Port inputs if TTL input is selected | $V_{SS} - 0.3$ | — | 0.8 | V | |
| | $V_{ILR}$ | INITX | — | $V_{SS} - 0.3$ | — | $0.2 \times V_{DD}$ | V | INITX input pin (CMOS Hysteresis) |
| | $V_{ILM}$ | MD_2 to MD_0 | — | $V_{SS} - 0.3$ | — | $V_{SS} + 0.3$ | V | Mode input pins |
| | $V_{ILXDS}$ | X0, X0A | — | $V_{SS} - 0.3$ | — | 0.5 | V | External clock in "Oscillation mode" |
| | $V_{ILXDF}$ | X0 | — | $V_{SS} - 0.3$ | — | $0.2 \times V_{DD}$ | V | External clock in "Fast Clock Input mode" |

| Parameter | Symbol | Pin name | Condition | Value | | | Unit | Remarks |
|---|---|---|---|---|---|---|---|---|
| | | | | Min | Typ | Max | | |
| Output “H” voltage | $V_{OH2}$ | Normal outputs | $4.5V \le V_{DD} \le 5.5V$, $I_{OH} = -2mA$<br>$3.0V \le V_{DD} \le 4.5V$, $I_{OH} = -1.6mA$ | $V_{DD} - 0.5$ | — | — | V | Driving strength set to 2 mA |
| | $V_{OH5}$ | Normal outputs | $4.5V \le V_{DD} \le 5.5V$, $I_{OH} = -5mA$<br>$3.0V \le V_{DD} \le 4.5V$, $I_{OH} = -3mA$ | $V_{DD} - 0.5$ | — | — | V | Driving strength set to 5 mA |
| | $V_{OH3}$ | $I^2C$ outputs | $3.0V \le V_{DD} \le 5.5V$, $I_{OH} = -3mA$ | $V_{DD} - 0.5$ | — | — | V | |

($V_{DD}5 = AV_{CC}5 = 3.0$ V to 5.5 V, $V_{SS}5 = AV_{SS}5 = 0$ V, $T_A = -40°C$ to $+105°C$)

| Parameter | Symbol | Pin name | Condition | Value | | | Unit | Remarks |
|---|---|---|---|---|---|---|---|---|
| | | | | Min | Typ | Max | | |
| Output "L" voltage | $V_{OL2}$ | Normal outputs | $4.5V \le V_{DD} \le 5.5V$, $I_{OH} = +2mA$ | — | — | 0.4 | V | Driving strength set to 2 mA |
| | | | $3.0V \le V_{DD} \le 4.5V$, $I_{OH} = +1.6mA$ | | | | | |
| | $V_{OL5}$ | Normal outputs | $4.5V \le V_{DD} \le 5.5V$, $I_{OH} = +5mA$ | — | — | 0.4 | V | Driving strength set to 5 mA |
| | | | $3.0V \le V_{DD} \le 4.5V$, $I_{OH} = +3mA$ | | | | | |
| | $V_{OL3}$ | $I^2C$ outputs | $3.0V \le V_{DD} \le 5.5V$, $I_{OH} = +3mA$ | — | — | 0.4 | V | |
| Input leakage current | $I_{IL}$ | Pnn_m *1 | $3.0V \le V_{DD} \le 5.5V$ $V_{SS}5 < V_I < V_{DD}$ $T_A$=25°C | −1 | — | +1 | µA | |
| | | | $3.0V \le V_{DD} \le 5.5V$ $V_{SS}5 < V_I < V_{DD}$ $T_A$=105°C | −3 | — | +3 | | |
| Analog input leakage current | $I_{AIN}$ | ANn *3 | $3.0V \le V_{DD} \le 5.5V$ $V_{SS}5 < V_I < V_{DD}$ $T_A$=25°C | −1 | — | +1 | µA | |
| | | | $3.0V \le V_{DD} \le 5.5V$ $V_{SS}5 < V_I < V_{DD}$ $T_A$=105°C | −3 | — | +3 | | |
| Pull-up resistance | $R_{UP}$ | Pnn_m*1 , INITX | $3.0V \le V_{DD} \le 3.6V$ | 40 | 100 | 160 | kΩ | |
| | | | $4.5V \le V_{DD} \le 5.5V$ | 25 | 50 | 100 | | |
| Pull-down resistance | $R_{DOWN}$ | Pnn_m*1 | $3.0V \le V_{DD} \le 3.6V$ | 40 | 100 | 180 | kΩ | |
| | | | $4.5V \le V_{DD} \le 5.5V$ | 25 | 50 | 100 | | |
| Input capacitance | $C_{IN}$ | All except $V_{DD}5$, $V_{DD}5R$, $V_{SS}5$, $AV_{CC}5$, $AV_{SS}$, AVRH5 | f = 1 MHz | — | 5 | 15 | pF | |

| Parameter | Symbol | Pin name | Condition | Value | | | Unit | Remarks |
|---|---|---|---|---|---|---|---|---|
| | | | | Min | Typ | Max | | |
| Power supply current MB91F465XA | $I_{CC}$ | $V_{DD}5R$ | MB91F465XA<br>CLKB: 100 MHz<br>CLKP: 50 MHz<br>CLKT: 50 MHz<br>CLKCAN: 50 MHz | — | 125 | 155 | mA | Code fetch from Flash |
| | $I_{CCH}$ | $V_{DD}5R$ | $T_A$ = + 25°C | — | 30 | 150 | µA | At stop mode *2 |
| | | | $T_A$ = + 105°C | — | 300 | 2000 | µA | |
| | | | $T_A$ = + 25°C | — | 100 | 500 | µA | RTC : 4 MHz mode *2 |
| | | | $T_A$ = + 105°C | — | 500 | 2400 | µA | |
| | | | $T_A$ = + 25°C | — | 50 | 250 | µA | RTC : 100 kHz mode *2 |
| | | | $T_A$ = + 105°C | — | 400 | 2200 | µA | |
| | $I_{LVE}$ | $V_{DD}5$ | — | — | 70 | 150 | µA | External low voltage detection |
| | $I_{LVI}$ | $V_{DD}5R$ | — | — | 50 | 100 | µA | Internal low voltage detection |
| | $I_{OSC}$ | $V_{DD}5$ | — | — | 250 | 500 | µA | Main clock (4 MHz) |
| | | | — | — | 20 | 40 | µA | Sub clock (32 kHz) |

*1: Pnn_m includes all pins unless the pins, which include analog inputs.

*2: Main regulator OFF, sub regulator set to 1.2V, Low voltage detection disabled.

*3: ANn includes all pins where AN channels are enabled.

## 15.4. A/D converter characteristics

($V_{DD}5 = AV_{CC}5 = 3.0$ V to 5.5 V, $V_{SS}5 = AV_{SS}5 = 0$ V, $T_A = -40°C$ to $+105°C$)

| Parameter | Symbol | Pin name | Value | | | Unit | Remarks |
|---|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | | |
| Resolution | — | — | — | — | 10 | bit | |
| Total error | — | — | − 3 | — | + 3 | LSB | |
| Nonlinearity error | — | — | − 2.5 | — | + 2.5 | LSB | |
| Differential nonlinearity error | — | — | − 1.9 | — | + 1.9 | LSB | |
| Zero reading voltage | $V_{OT}$ | ANn | AVRL - 1.5 LSB | AVRL + 0.5 LSB | AVRL + 2.5 LSB | V | |
| Full scale reading voltage | $V_{FST}$ | ANn | AVRH - 3.5 LSB | AVRH - 1.5 LSB | AVRH + 0.5 LSB | V | |
| Compare time | $T_{comp}$ | — | 0.6 | — | 16,500 | µs | 4.5 V ≤ $AV_{CC}5$ ≤ 5.5 V |
| | | | 2.0 | — | — | µs | 3.0 V ≤ $AV_{CC}5$ ≤ 4.5 V |
| Sampling time | $T_{samp}$ | — | 0.4 | — | — | µs | 4.5 V ≤ $AV_{CC}5$ ≤ 5.5 V, $R_{EXT}$ < 2 kΩ |
| | | | 1.0 | — | — | µs | 3.0 V ≤ $AV_{CC}5$ ≤ 4.5 V, $R_{EXT}$ < 1 kΩ |
| Conversion time | $T_{conv}$ | — | 1.0 | — | — | µs | 4.5 V ≤ $AV_{CC}5$ ≤ 5.5 V |
| | | | 3.0 | — | — | µs | 3.0 V ≤ $AV_{CC}5$ ≤ 4.5 V |
| Input capacitance | $C_{IN}$ | ANn | — | — | 11 | pF | |
| Input resistance | $R_{IN}$ | ANn | — | — | 2.6 | kΩ | 4.5 V ≤ $AV_{CC}5$ ≤ 5.5 V |
| | | | — | — | 12.1 | kΩ | 3.0 V ≤ $AV_{CC}5$ ≤ 4.5 V |
| Analog input leakage current | $I_{AIN}$ | ANn | − 1 | — | + 1 | µA | $T_A$ = + 25°C |
| | | | − 3 | — | + 3 | µA | $T_A$ = + 105°C |
| Analog input voltage range | $V_{AIN}$ | ANn | AVRL | — | AVRH | V | |
| Offset between input channels | — | ANn | — | — | 4 | LSB | |

*(Continued)*

Note : The accuracy gets worse as AVRH - AVRL becomes smaller

*(Continued)*

| Parameter | Symbol | Pin name | Value | | | Unit | Remarks |
|---|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | | |
| Reference voltage range | AVRH | AVRH5 | 0.75 × $AV_{CC}5$ | — | $AV_{CC}5$ | V | |
| | AVRL | $AV_{SS}5$ | $AV_{SS}5$ | — | $AV_{CC}5 \times 0.25$ | V | |
| Power supply current | $I_A$ | $AV_{CC}5$ | — | 2.5 | 5 | mA | A/D Converter active |
| | $I_{AH}$ | $AV_{CC}5$ | — | — | 5 | µA | A/D Converter not operated *1 |
| Reference voltage current | $I_R$ | AVRH5 | — | 0.7 | 1 | mA | A/D Converter active |
| | $I_{RH}$ | AVRH5 | — | — | 5 | µA | A/D Converter not operated *2 |

*1 : Supply current at $AV_{CC}5$, if A/D converter and ALARM comparator are not operating, ($V_{DD}5 = AV_{CC}5 = AVRH = 5.0$ V)

*2 : Input current at AVRH5, if A/D converter is not operating, ($V_{DD}5 = AV_{CC}5 = AVRH = 5.0$ V)

Sampling Time Calculation

$T_{samp} = (2.6\ kOhm + R_{EXT}) \times 11pF \times 7$; for $4.5V \le AV_{CC}5 \le 5.5V$

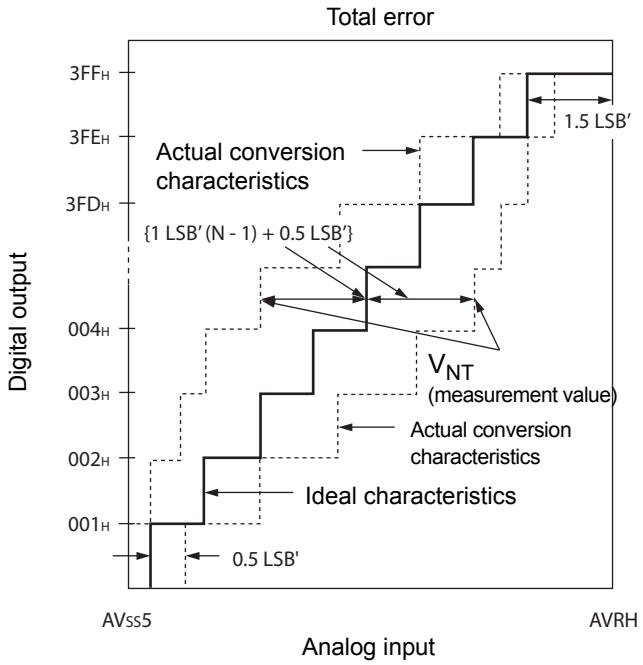
$T_{samp} = (12.1\ kOhm + R_{EXT}) \times 11pF \times 7$; for $3.0V \le AV_{CC}5 \le 4.5V$

Conversion Time Calculation

$T_{conv} = T_{samp} + T_{comp}$

### Definition of A/D converter terms

- Resolution
  Analog variation that is recognizable by the A/D converter.
- Linearity error
  Deviation between actual conversion characteristics and a straight line connecting the zero transition point ($00\ 0000\ 0000_B \leftrightarrow 00\ 0000\ 0001_B$) and the full scale transition point ($11\ 1111\ 1110_B \leftrightarrow 11\ 1111\ 1111_B$).
- Differential nonlinearity error
  Deviation of the input voltage from the ideal value that is required to change the output code by 1 LSB.
- Total error
  This error indicates the difference between actual and theoretical values, including the zero transition error, full scale transition error, and nonlinearity error.



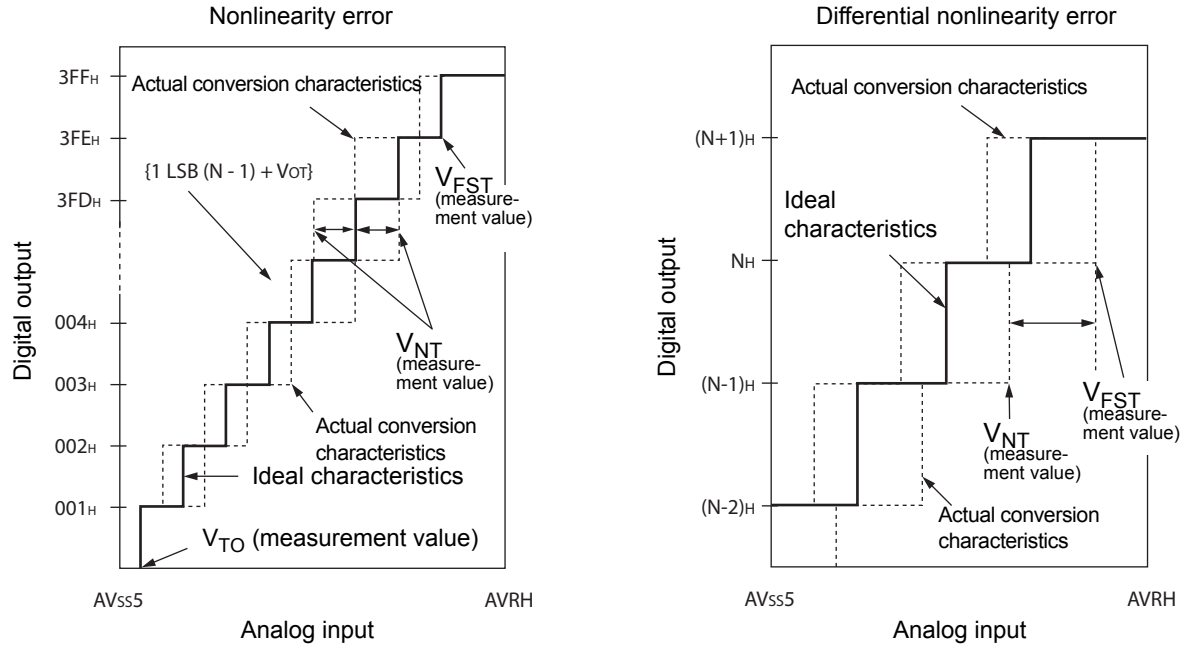

$$1LSB' \text{ (ideal value)} = \frac{AVRH - AV_{SS}5}{1024} [V]$$

$$\text{Total error of digital output N} = \frac{V_{NT} - \{1\ LSB' \times (N - 1) + 0.5\ LSB'\}}{1\ LSB'}$$

N : A/D converter digital output value
$V_{OT}'$ (ideal value) = $AV_{SS}5 + 0.5$ LSB' [V]
$V_{FST}'$ (ideal value) = AVRH – 1.5 LSB' [V]
$V_{NT}$ : Voltage at which the digital output changes from $(N + 1)_H$ to $N_H$

*(Continued)*

*(Continued)*




$$\text{Nonlinearity error of digital output N} = \frac{V_{NT} - \{1LSB \times (N - 1) + V_{OT}\}}{1LSB} \text{ [LSB]}$$

$$\text{Differential nonlinearity error of digital output N} = \frac{V_{(N+1)T} - V_{NT}}{1LSB} - 1 \text{ [LSB]}$$

$$1LSB = \frac{V_{FST} - V_{OT}}{1022} \text{ [V]}$$

N : A/D converter digital output value
$V_{OT}$ : Voltage at which the digital output changes from $000_H$ to $001_H$.
$V_{FST}$ : Voltage at which the digital output changes from $3FE_H$ to $3FF_H$.

## 15.5. FLASH memory program/erase characteristics

### *15.5.1. MB91F465XA*

($T_A$ = 25°C, Vcc = 5.0V)

| Parameter | Value | | | Unit | Remarks |
|---|---|---|---|---|---|
| | Min | Typ | Max | | |
| Sector erase time | - | 0.9 | 3.6 | s | Erasure programming time not included |
| Chip erase time | - | n*0.9 | n*3.6 | s | n is the number of Flash sector of the device |
| Word (16-bit width) programming time | - | 23 | 370 | µs | System overhead time not included |
| Programme/Erase cycle | 10 000 | | | cycle | |
| Flash data retention time | 20 | | | year | *1 |

*1:This value was converted from the results of evaluating the reliability of the technology (using Arrhenius equation to convert high temperature measurements into normalized value at 85°C).

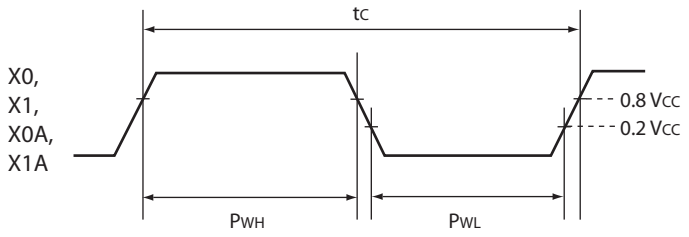
## 15.6. AC characteristics

### *15.6.1. Clock timing*

($V_{DD}5$ = 3.0 V to 5.5 V, Vss5 = AVss5 = 0 V, $T_A$ = –40°C to + 105°C)

| Parameter | Symbol | Pin name | Value | | | Unit | Condition |
|---|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | | |
| Clock frequency | $f_C$ | X0<br>X1 | 3.5 | 4 | 16 | MHz | Opposite phase external supply or crystal |
| | | X0A<br>X1A | 32 | 32.768 | 100 | kHz | |

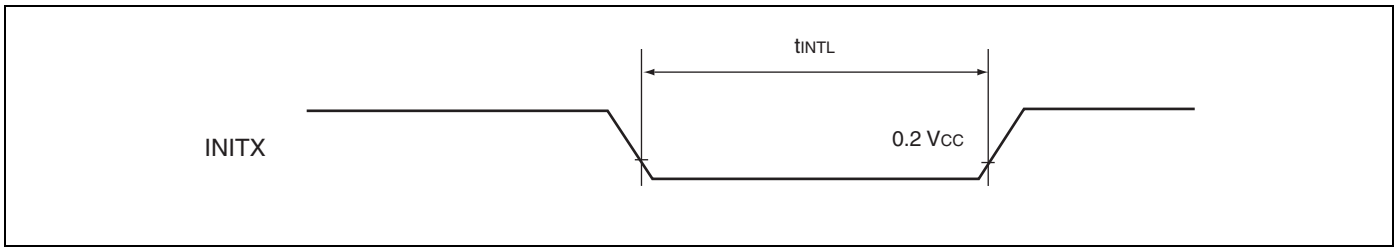
**Clock timing condition**



### 15.6.2. Reset input ratings

($V_{DD}5$ = 3.0 V to 5.5 V, $V_{SS}5$ = $AV_{SS}5$ = 0 V, $T_A$ = –40°C to + 105°C)

| Parameter | Symbol | Pin name | Condition | Value | | Unit |
|---|---|---|---|---|---|---|
| | | | | Min | Max | |
| INITX input time (at power-on) | $t_{INTL}$ | INITX | — | 8 | — | ms |
| INITX input time (other than the above) | | | | 20 | — | µs |



*15.6.3. LIN-USART Timings at $V_{DD}5$ = 3.0 to 5.5 V*

- Conditions during AC measurements
- All AC tests were measured under the following conditions:
- - $IO_{drive}$ = 5 mA
- - $V_{DD}5$ = 3.0 V to 5.5 V, $I_{load}$ = 3 mA
- - $V_{SS}5$ = 0 V
- - $T_a$ = -40°C to +105°C
- - $C_l$ = 50 pF (load capacity value of pins when testing)
- - VOL = 0.2 x $V_{DD}5$
- - VOH = 0.8 x $V_{DD}5$
- - EPILR = 0, PILR = 1 (Automotive Level = worst case)

($V_{DD}5$ = 3.0 V to 5.5 V, $V_{SS}5$ = $AV_{SS}5$ = 0 V, $T_A$ = –40°C to + 105°C)

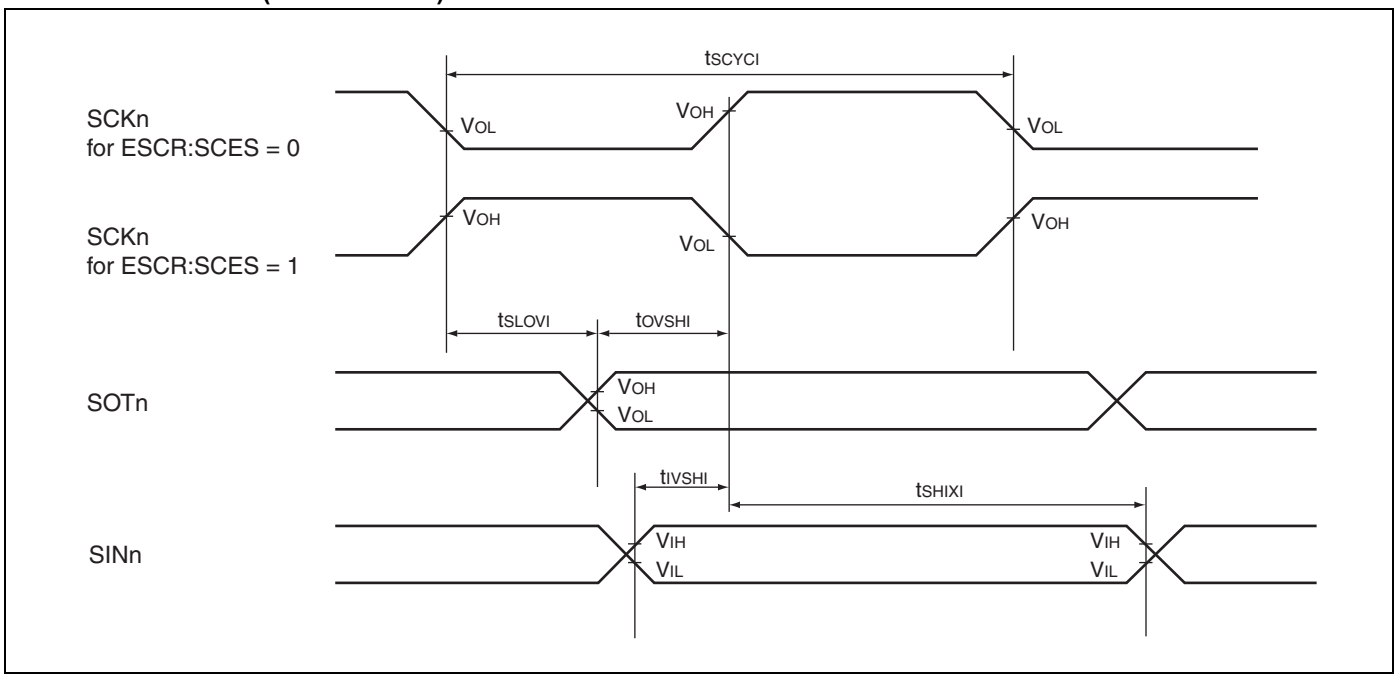
| Parameter | Symbol | Pin name | Condition | $V_{DD}5$ = 3.0 V to 4.5 V | | $V_{DD}5$ = 4.5 V to 5.5 V | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | Min | Max | Min | Max | |
| Serial clock cycle time | $t_{SCYCI}$ | SCKn | Internal clock operation (master mode) | 4 $t_{CLKP}$ | — | 4 $t_{CLKP}$ | — | ns |
| SCK ↓ → SOT delay time | $t_{SLOVI}$ | SCKn SOTn | | – 30 | 30 | – 20 | 20 | ns |
| SOT → SCK ↓ delay time | $t_{OVSHI}$ | SCKn SOTn | | m × $t_{CLKP}$ – 30* | — | m × $t_{CLKP}$ – 20* | — | ns |
| Valid SIN → SCK ↑ setup time | $t_{IVSHI}$ | SCKn SINn | | $t_{CLKP}$ + 55 | — | $t_{CLKP}$ + 45 | — | ns |
| SCK ↑ → valid SIN hold time | $t_{SHIXI}$ | SCKn SINn | | 0 | — | 0 | — | ns |
| Serial clock “H” pulse width | $t_{SHSLE}$ | SCKn | External clock operation (slave mode) | $t_{CLKP}$ + 10 | — | $t_{CLKP}$ + 10 | — | ns |
| Serial clock “L” pulse width | $t_{SLSHE}$ | SCKn | | $t_{CLKP}$ + 10 | — | $t_{CLKP}$ + 10 | — | ns |
| SCK ↓ → SOT delay time | $t_{SLOVE}$ | SCKn SOTn | | — | 2 $t_{CLKP}$ + 55 | — | 2 $t_{CLKP}$ + 45 | ns |
| Valid SIN → SCK ↑ setup time | $t_{IVSHE}$ | SCKn SINn | | 10 | — | 10 | — | ns |
| SCK ↑ → valid SIN hold time | $t_{SHIXE}$ | SCKn SINn | | $t_{CLKP}$ + 10 | — | $t_{CLKP}$ + 10 | — | ns |
| SCK rising time | $t_{FE}$ | SCKn | | — | 20 | — | 20 | ns |
| SCK falling time | $t_{RE}$ | SCKn | | — | 20 | — | 20 | ns |

* : Parameter m depends on $t_{SCYCI}$ and can be calculated as :

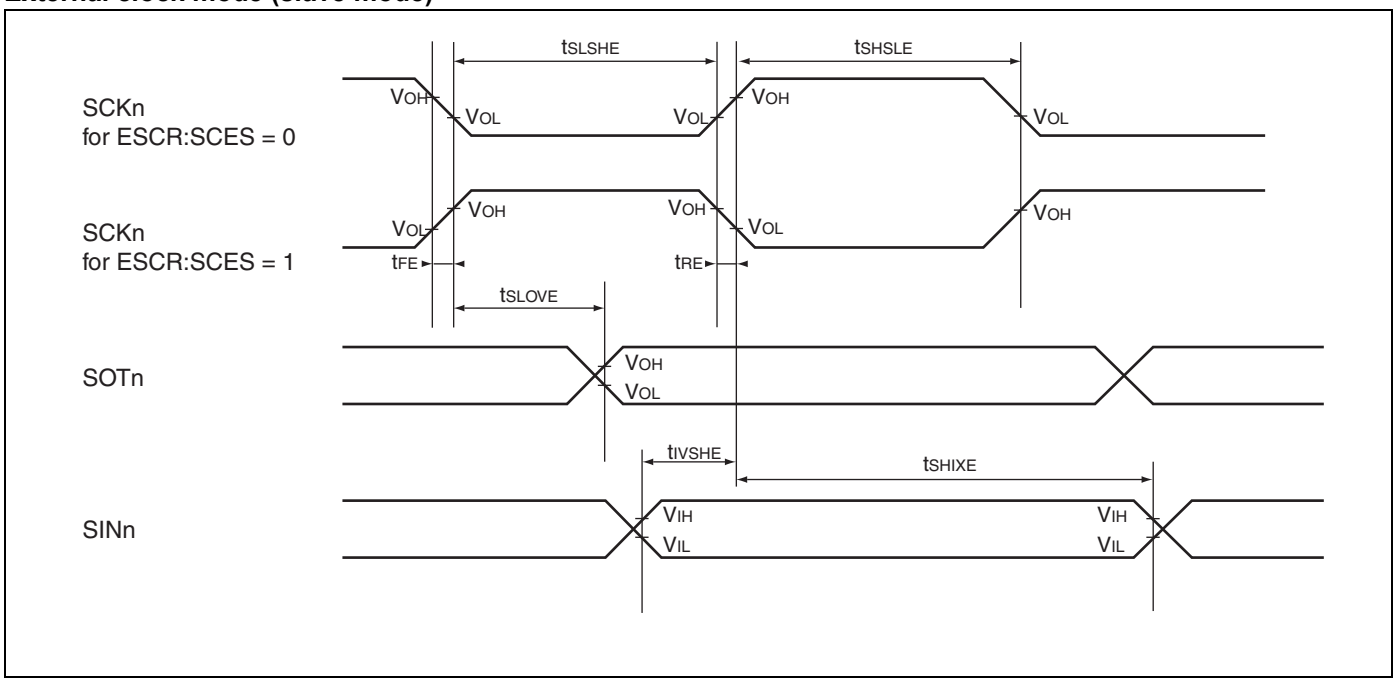
- if $t_{SCYCI}$ = 2*k*$t_{CLKP}$, then m = k, where k is an integer > 2
- if $t_{SCYCI}$ = (2*k + 1)*$t_{CLKP}$, then m = k + 1, where k is an integer > 1

Notes : • The above values are AC characteristics for CLK synchronous mode.
• $t_{CLKP}$ is the cycle time of the peripheral clock.

### Internal clock mode (master mode)




### External clock mode (slave mode)



*15.6.4. $I^2C$ AC Timings at $V_{DD}5$ = 3.0 to 5.5 V*

- Conditions during AC measurements
  All AC tests were measured under the following conditions:
  - $IO_{drive}$ = 3 mA
  - $V_{DD}5$ = 3.0 V to 5.5 V, $I_{load}$ = 3 mA ($V_{DD}$ = 4.5 V to 5.5 V for MB91F465XA)
  - $V_{SS}5$ = 0 V
  - Ta = − 40°C to + 105°C
  - $C_l$ = 50 pF
  - VOL = 0.3 × $V_{DD}5$
  - VOH = 0.7 × $V_{DD}5$
  - EPILR = 0, PILR = 0 (CMOS Hysteresis 0.3 × $V_{DD}5$/0.7 × $V_{DD}5$)

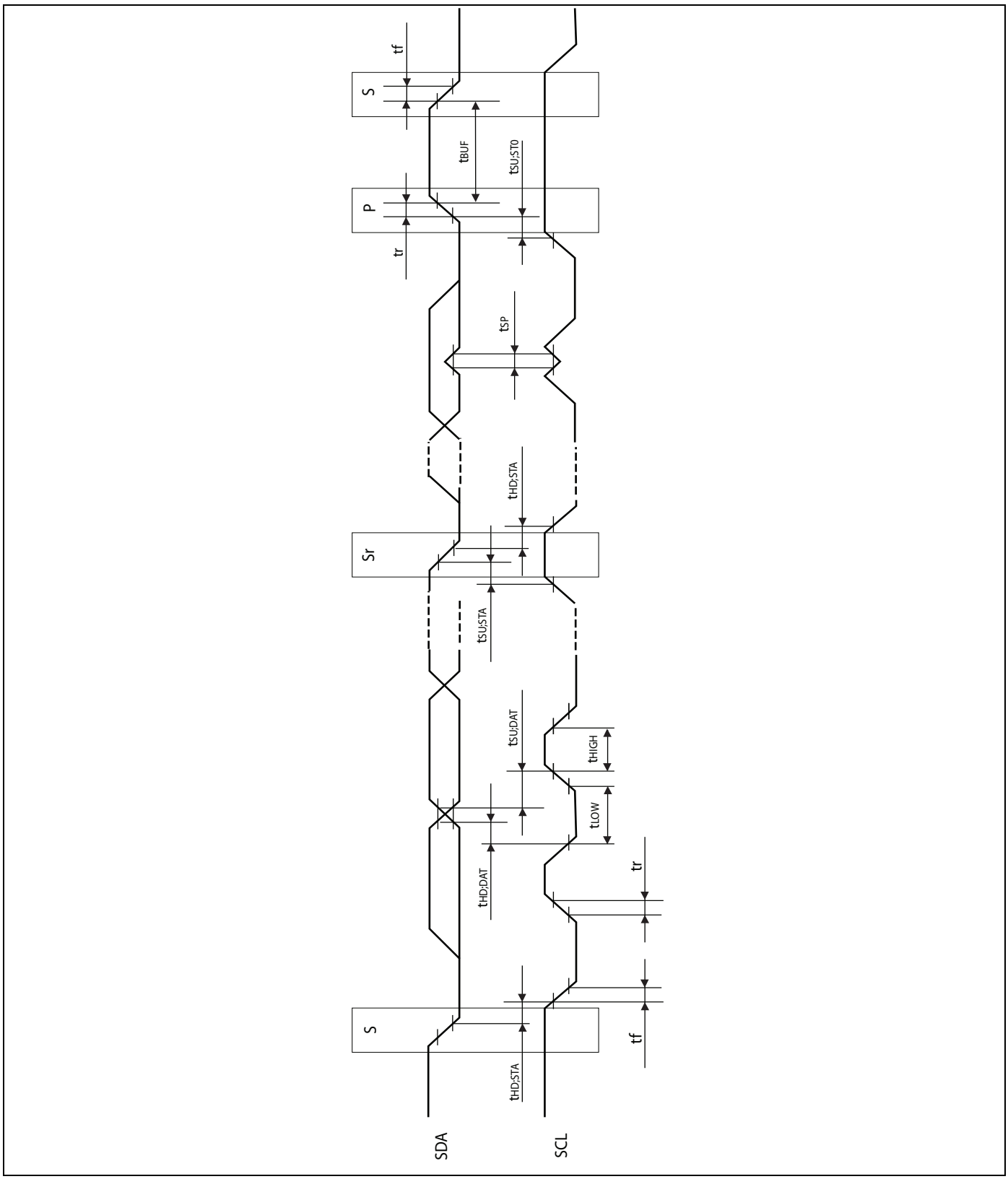
Fast mode:

($V_{DD}5$ = 3.5 V to 5.5 V, $V_{SS}5$ = $AV_{SS}5$ = 0 V, $T_A$ = –40°C to + 105°C)

| Parameter | Symbol | Pin name | Value | | Unit | Remark |
|---|---|---|---|---|---|---|
| | | | Min | Max | | |
| SCL clock frequency | $f_{SCL}$ | SCLn | 0 | 400 | kHz | |
| Hold time (repeated) START condition. After this period, the first clock pulse is generated | $t_{HD;STA}$ | SCLn, SDAn | 0.6 | — | µs | |
| LOW period of the SCL clock | $t_{LOW}$ | SCLn | 1.3 | — | µs | |
| HIGH period of the SCL clock | $t_{HIGH}$ | SCLn | 0.6 | — | µs | |
| Setup time for a repeated START condition | $t_{SU;STA}$ | SCLn, SDAn | 0.6 | — | µs | |
| Data hold time for $I^2C$-bus devices | $t_{HD;DAT}$ | SCLn, SDAn | 0 | 0.9 | µs | |
| Data setup time | $t_{SU;DAT}$ | SCLn SDAn | 100 | — | ns | |
| Rise time of both SDA and SCL signals | $t_r$ | SCLn, SDAn | 20 + 0.1Cb | 300 | ns | |
| Fall time of both SDA and SCL signals | $t_f$ | SCLn, SDAn | 20 + 0.1Cb | 300 | ns | |
| Setup time for STOP condition | $t_{SU;STO}$ | SCLn, SDAn | 0.6 | — | µs | |
| Bus free time between a STOP and START condition | $t_{BUF}$ | SCLn, SDAn | 1.3 | — | µs | |
| Capacitive load for each bus line | $C_b$ | SCLn, SDAn | — | 400 | pF | |
| Pulse width of spike suppressed by input filter | $t_{SP}$ | SCLn, SDAn | 0 | (1..1.5) × $t_{CLKP}$ | ns | *1 |

*1 : The noise filter will suppress single spikes with a pulse width of 0ns and between (1 to 1.5) cycles of peripheral clock, depending on the phase relationship between $I^2C$ signals (SDA, SCL) and peripheral clock.

Note: $t_{CLKP}$ is the cycle time of the peripheral clock.

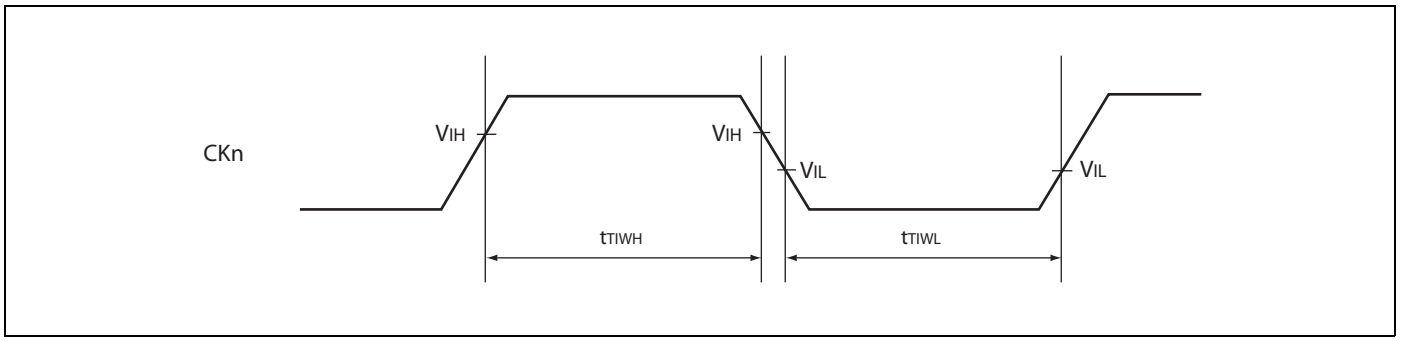

tf
S
tBUF
tSU;STO
P
tr
tSP
tHD;STA
Sr
tSU;STA
tSU;DAT
tHIGH
tLOW
tHD;DAT
tr
tf
S
tHD;STA
SDA
SCL

### 15.6.5. Free-run timer clock

($V_{DD}5$ = 3.0 V to 5.5 V, $V_{SS}5$ = $AV_{SS}5$ = 0 V, $T_A$ = –40°C to + 105°C)

| Parameter | Symbol | Pin name | Condition | Value | | Unit |
|---|---|---|---|---|---|---|
| | | | | Min | Max | |
| Input pulse width | $t_{TIWH}$<br>$t_{TIWL}$ | CKn | — | $4t_{CLKP}$ | — | ns |

Note : $t_{CLKP}$ is the cycle time of the peripheral clock.

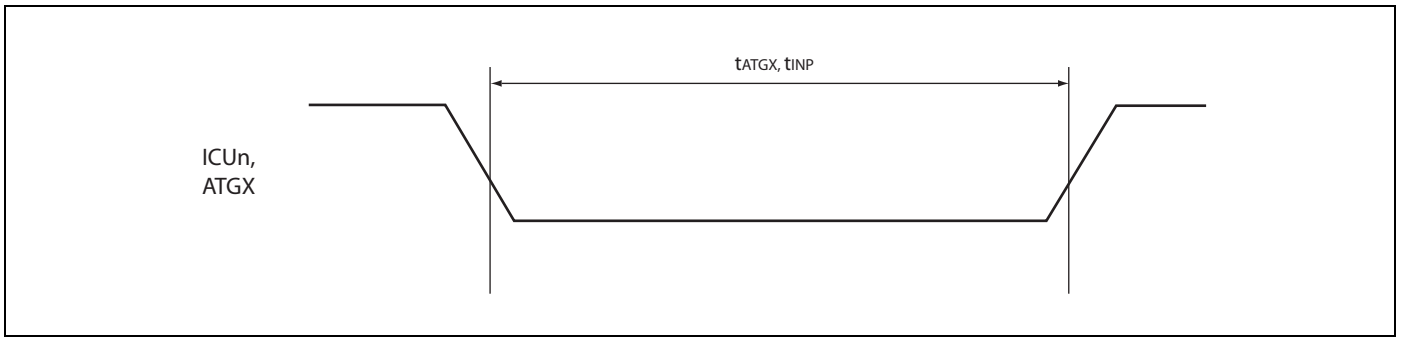



### 15.6.6. Trigger input timing

($V_{DD}5$ = 3.0 V to 5.5 V, $V_{SS}5$ = $AV_{SS}5$ = 0 V, $T_A$ = –40°C to + 105°C)

| Parameter | Symbol | Pin name | Condition | Value | | Unit |
|---|---|---|---|---|---|---|
| | | | | Min | Max | |
| Input capture input trigger | $t_{INP}$ | ICUn | — | $5t_{CLKP}$ | — | ns |
| A/D converter trigger | $t_{ATGX}$ | ATGX | — | $5t_{CLKP}$ | — | ns |

Note : $t_{CLKP}$ is the cycle time of the peripheral clock.



# 16. E-Ray Overview

The E-Ray module is a FlexRay IP-module that can be integrated as stand-alone device or as part of an ASIC. It is described in VHDL on RTL level, prepared for synthesis. The E-Ray IP-module performs communication according to the FlexRay protocol specification v2.1. With maximum specified sample clock the bitrate is 10 MBits. Additional bus driver (BD) hardware is required for connection to the physical layer.

For communication on a FlexRay network, individual message buffers with up to 254 data bytes are configurable. The message storage consists of a single-ported Message RAM that holds up to 128 message buffers. All functions concerning the handling of messages are implemented in the Message Handler. Those functions are the acceptance filtering, the transfer of messages between the two FlexRay Channel Protocol Controllers and the Message RAM, maintaining the transmission schedule as well as providing message status information.

The register set of the E-Ray IP-module can be accessed directly by an external Host via the module's Host interface. These registers are used to control/configure/monitor the FlexRay Channel Protocol Controllers, Message Handler, Global Time Unit, System Universal Control, Frame and Symbol Processing, Network Management, Interrupt Control, and to access the Message RAM via Input / Output Buffer.

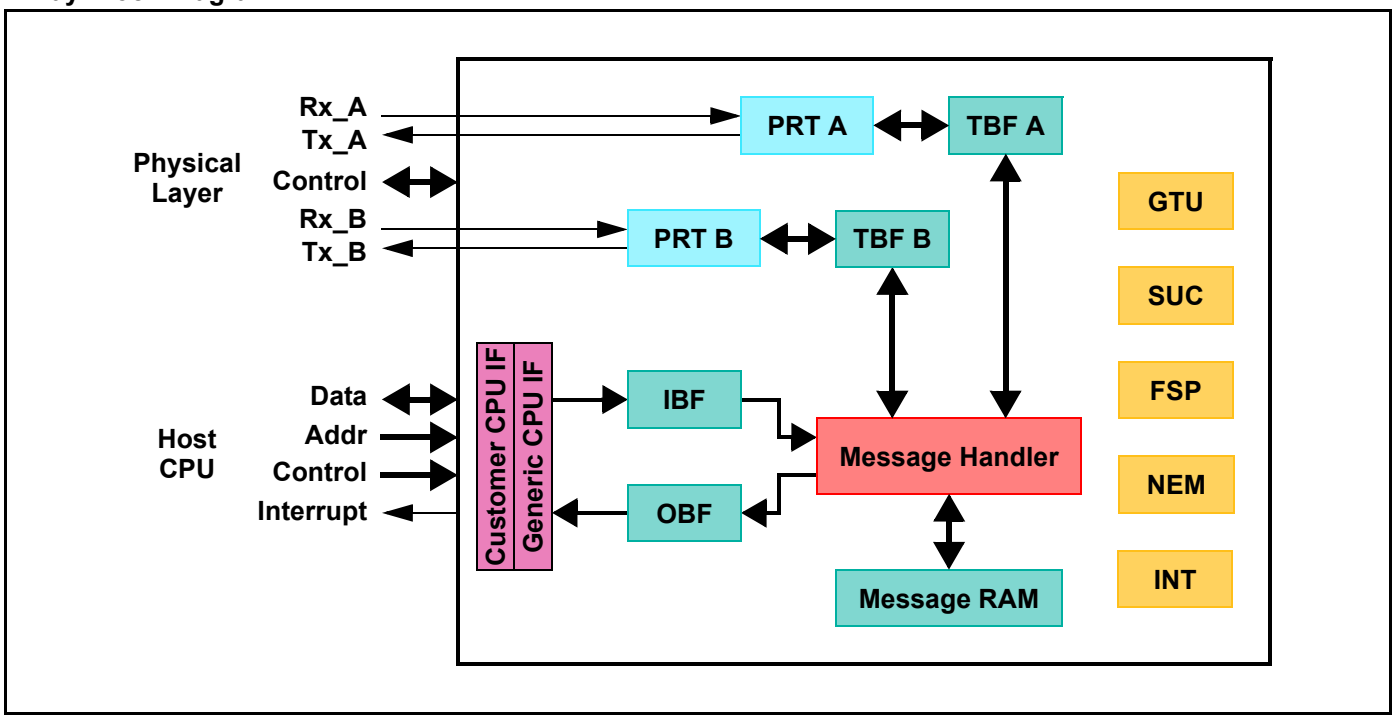
The E-Ray IP-module can be connected to a wide range of customer-specific Host CPUs via its 8/16/32-bit Generic CPU Interface.

The E-Ray IP-module supports the following features:

- Conformance with FlexRay protocol specification v2.1
- Data rates of up to 10 Mbit/s on each channel
- Up to 128 message buffers configurable
- 8 Kbyte of Message RAM for storage of e.g. 128 message buffers with max. 48 byte data section or up to 30 message buffers with 254 byte data section
- Configuration of message buffers with different payload lengths possible
- One configurable receive FIFO
- Each message buffer can be configured as receive buffer, as transmit buffer or as part of the receive FIFO
- Host access to message buffers via Input and Output Buffer
  Input Buffer: Holds message to be transferred to the Message RAM
  Output Buffer: Holds message read from the Message RAM
- Filtering for slot counter, cycle counter, and channel
- Maskable module interrupts
- Network Management supported
- 8/16/32-bit Generic CPU Interface, connectable to a wide range of customer-specific Host CPUs

## 16.1. Block Diagram

### E-Ray Block Diagram




### Customer CPU Interface (CIF)

Connects a customer specific Host CPU to the E-Ray IP-module via the Generic CPU Interface.

### Generic CPU Interface (GIF)

The E-Ray IP-module is provided with an 8/16/32-bit Generic CPU Interface prepared for the connection to a wide range of customer-specific Host CPUs. Configuration registers, status registers, and interrupt registers are attached to the respective blocks and can be accessed via the Generic CPU Interface.

### Input Buffer (IBF)

For write access to the message buffers configured in the Message RAM, the Host can write the header and data section for a specific message buffer to the Input Buffer. The Message Handler then transfers the data from the Input Buffer to the selected message buffer in the Message RAM.

### Output Buffer (OBF)

For read access to a message buffer configured in the Message RAM the Message Handler transfers the selected message buffer to the Output Buffer. After the transfer has completed, the Host can read the header and data section of the transferred message buffer from the Output Buffer.

### Message Handler (MHD)

The E-Ray Message Handler controls data transfers between the following components:

- Input / Output Buffer and Message RAM
- Transient Buffer RAMs of the two FlexRay Protocol Controllers and Message RAM

### Message RAM (MRAM)

The Message RAM consists of a single-ported RAM that stores up to 128 FlexRay message buffers together with the related configuration data (header and data partition).

### Transient Buffer RAM (TBF A/B)

Stores the data section of two complete messages.

### FlexRay Channel Protocol Controller (PRT A/B)

The FlexRay Channel Protocol Controllers consist of shift register and FlexRay protocol FSM. They are connected to the Transient Buffer RAMs for intermediate message storage and to the physical layer via bus driver BD.

They perform the following functionality:

- Control and check of bit timing
- Reception / transmission of FlexRay frames and symbols
- Check of header CRC
- Generation / check of frame CRC
- Interfacing to bus driver

The FlexRay Channel Protocol Controllers have interfaces to:

- Physical Layer (bus driver)
- Transient Buffer RAM
- Message Handler
- Global Time Unit
- System Universal Control
- Frame and Symbol Processing
- Network Management
- Interrupt Control

### Global Time Unit (GTU)

The Global Time Unit performs the following functions:

- Generation of microtick
- Generation of macrotick
- Fault tolerant clock synchronization by FTM algorithm
  - rate correction
  - offset correction
- Cycle counter
- Timing control of static segment
- Timing control of dynamic segment (minislotting)
- Support of external clock correction

### System Universal Control (SUC)

The System Universal Control controls the following functions:

- Configuration
- Wakeup
- Startup
- Normal Operation
- Passive Operation
- Monitor Mode

### Frame and Symbol Processing (FSP)

The Frame and Symbol Processing controls the following functions:

- Checks the correct timing of frames and symbols
- Tests the syntactical and semantical correctness of received frames
- Sets the slot status flags

### Network Management (NEM)

Handles the network management vector.

### Interrupt Control (INT)

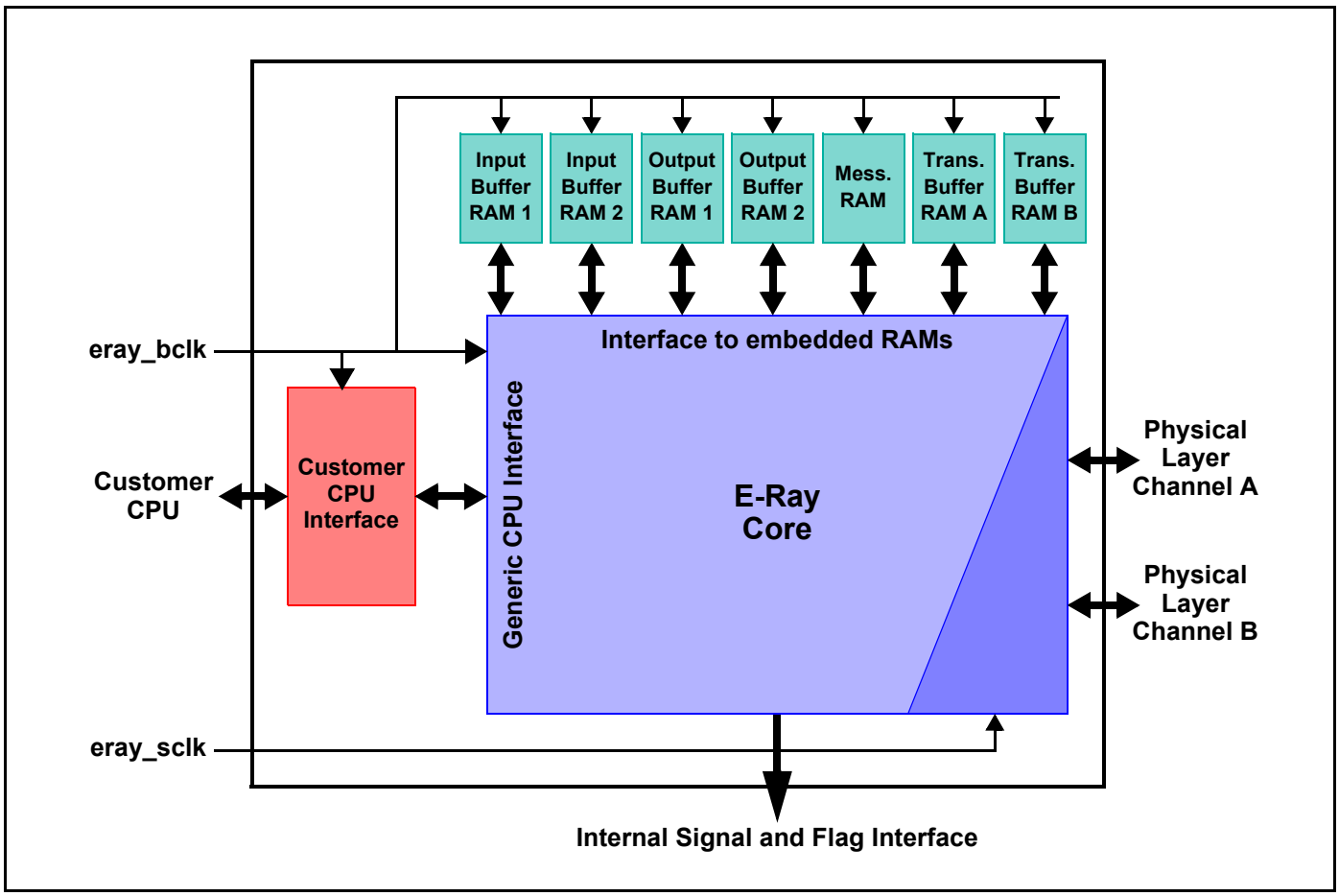
The Interrupt Controller performs the following functions:

- Provides error and status interrupt flags
- Enable / disable interrupt sources
- Assignment of interrupt sources to one of the two module interrupt lines
- Enable / disable module interrupt lines
- Manages the two interrupt timers
- Stop watch time capturing

## 17. Generic Interface

The Generic Interface encapsulates the synthesizable code of the E-Ray design (E-Ray core). All customer specific components like Customer CPU Interfaces and RAM blocks are connected to the Generic Interface. The following figure shows the connection of the E-Ray core to the external world via its Generic Interface.

**Generic interface of E-Ray core**




The Generic Interface consists of the Generic CPU Interface, the interfaces to the embedded RAMs, the Internal Signal and Flag Interface, and the Physical Layer Interface.

## 17.1. Generic CPU Interface

The Generic CPU Interface connects the E-Ray module to a customer specific Host CPU via the Customer CPU Interface. The Generic CPU Interface was designed for connection of the E-Ray module to a wide range of customer-specific CPUs. It supports 8/16/32-bit accesses.

**Generic CPU interface**

| Signal | Direction | Description |
|---|---|---|
| eray_sclk | | Sample Clock, 80 MHz |
| eray_bclk | | Bus Clock |
| eray_reset | | Module Reset, configurable via constant reset_active_c, default is LOW active |
| eray_select | | Module Select |
| eray_addr[10:0] | | Address Input |
| eray_byten[3:0] | | Byte Enable |
| eray_write | | Write/Read Control: ’1’ = write, ’0’ = read |
| eray_wdata[31:0] | | Write Data Input |
| eray_stpwt | | Stop Watch Trigger Input |
| eray_scanmode | | Scan Mode Enable Input |
| eray_wrdy | 0 | Write Ready Output |
| eray_rrdy | 0 | Read Ready Output |
| eray_rdata[31:0] | 0 | Read Data Output |
| eray_int0 | 0 | Interrupt Line 0 Output, HIGH active |
| eray_int1 | 0 | Interrupt Line 1 Output, HIGH active |
| eray_tint0 | 0 | Timer Interrupt 0 Output, HIGH active |
| eray_tint1 | 0 | Timer Interrupt 1 Output, HIGH active |
| eray_ibusy | 0 | Transfer Input Buffer RAM to Message RAM busy, active while bit IBCR.IBSYH is set |
| eray_obusy | 0 | Transfer Message RAM to Output Buffer RAM busy, active while bit OBCR.OBSYS is set |

### *17.1.1. Reset Timing*

To perform a reset of the E-Ray module, signal **eray_reset** needs to be active for at least:

- Two **eray_bclk** cycles, clock period of **eray_bclk** ≥ **eray_sclk**
- Two **eray_sclk** cycles, clock period of **eray_bclk < eray_sclk**

When leaving hard reset, an internal procedure is started that initializes the seven module-internal RAM blocks to zero. The module-internal RAMs can also be cleared by CHI command CLEAR_RAMS **(SUCC1.CMD[3:0] = "1100")** when the CC is in DEFAULT_CONFIG or CONFIG state. The initialization of the E-Ray internal RAM blocks requires 2048 **eray_bclk** cycles.

No Host access to IBF or OBF is possible during initialization of the internal RAM blocks after hard reset or after assertion of CHI command CLEAR_RAMS. Access to the configuration and status registers is possible during execution of CHI command CLEAR_RAMS.

After hard reset all registers hold the reset values as summarized in the table E-Ray register map.

### *17.1.2. Write Access*

The byte position(s) for a write access is defined by the byte-enable signals **eray_byten[3:0]**.

For 8-bit write accesses only one of the four byte-enable signals may be active:

- **eray_byten[3:0]** = "0001": Update register / RAM bits 7 downto 0 from **eray_wdata[7:0]**
- **eray_byten[3:0]** = "0010": Update register / RAM bits 15 downto 8 from **eray_wdata[15:8]**
- **eray_byten[3:0]** = "0100": Update register / RAM bits 23 downto 16 from **eray_wdata[23:16]**
- **eray_byten[3:0]** = "1000": Update register / RAM bits 31 downto 24 from **eray_wdata[31:24]**

For 16-bit write accesses either the two lower or the two higher byte-enable signals have to be active:

- **eray_byten[3:0]** = "0011": Update register / RAM bits 15 downto 0 from **eray_wdata[15:0]**
- **eray_byten[3:0]** = "1100": Update register / RAM bits 31 downto 16 from **eray_wdata[31:16]**

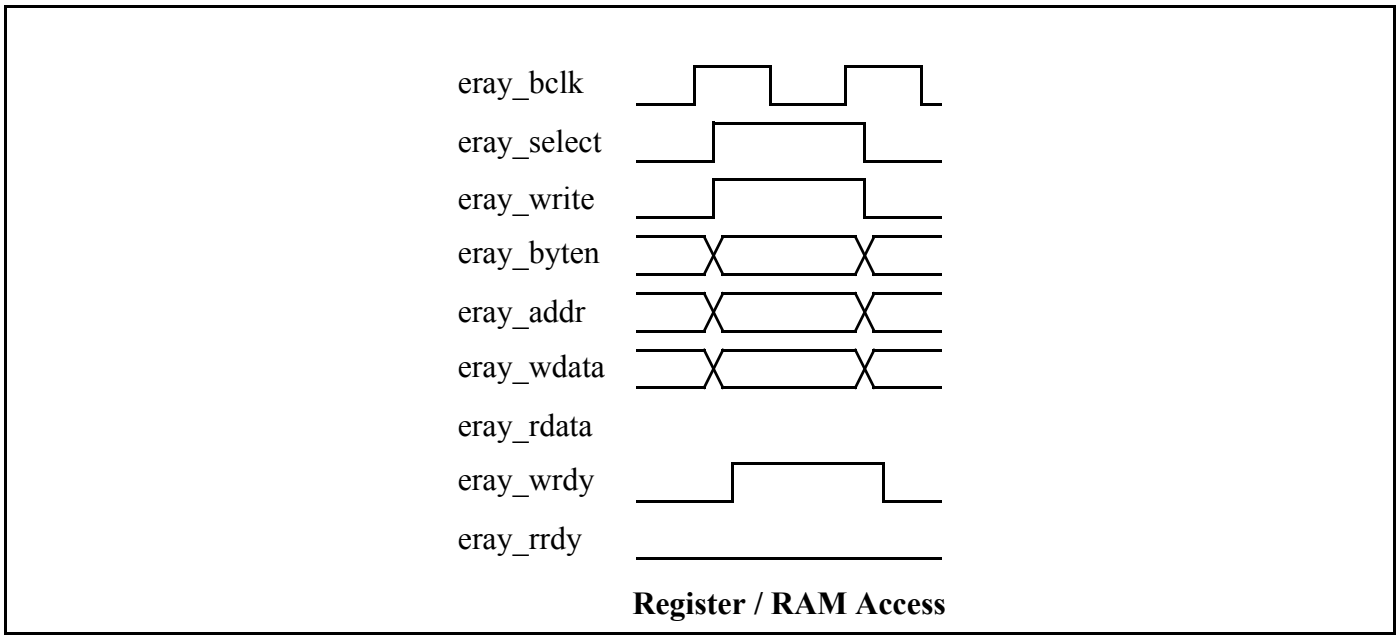
For a 32-bit write access all four byte-enable signals have to be active:

- **eray_byten[3:0]** = "1111": Update register / RAM bits 31 downto 0 from **eray_wdata[31:0]**

  Signals **eray_addr[1:0]** are not used.

When writing to Write Data Section [1...64] of the Input Buffer (see 18.10.1.Write Data Section [1…64] (WRDSn)), each 32-bit word has to be filled up by one 32-bit access OR two consecutive 16-bit accesses OR four consecutive 8-bit accesses before the transfer from the Input Buffer to the Message RAM is started by writing the number of the target message buffer in the Message RAM to register IBCR. If not all bytes of a 32-bit word have been written by the Host (8/16-bit access only), partly old data is transferred to the Message RAM.

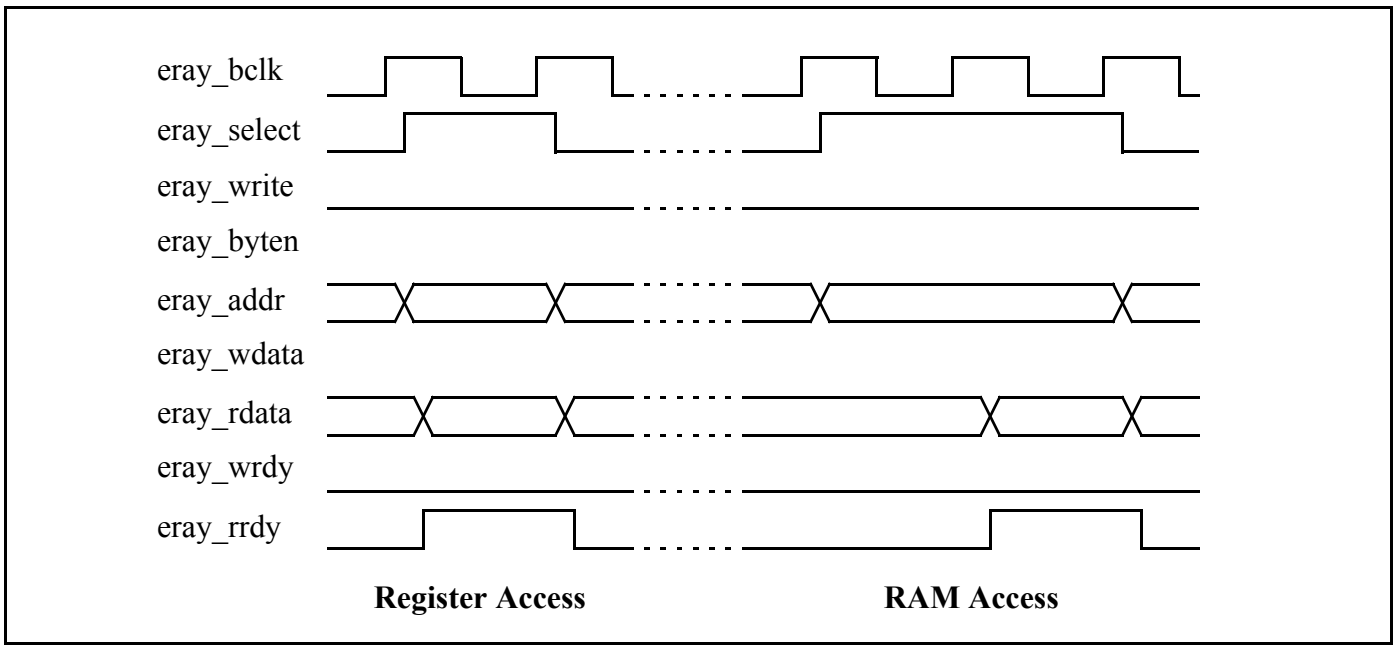
**Write access to E-Ray registers and Input Buffer RAM**




Write accesses to register and RAM locations take one **eray_bclk** cycle each.

*17.1.3. Read Access*

**Read access to E-Ray registers and Input / Output Buffer RAM**




A read access from the internal RAM blocks takes two **eray_bclk** cycles (because the RAM is synchronous), while data from registers is valid within one **eray_bclk** cycle. For read accesses signal **eray_byten[3:0]** is ignored.

Note**:** With 8/16-bit read accesses the register contents may change in between two read accesses (non-atomic read access).

*17.1.4. Timing of IBF / OBF Transfer Busy Signals*

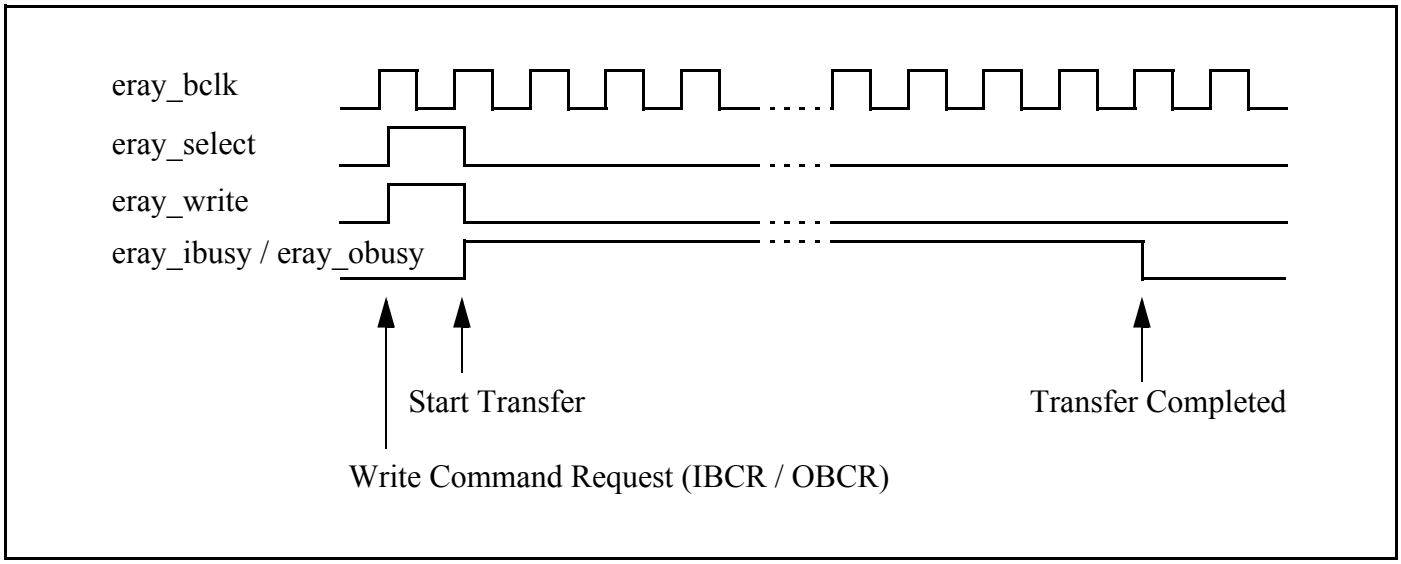
A data transfer from Input Buffer (IBF) to the Message RAM or from Message RAM to Output Buffer (OBF) is initiated by a write access to the respective command request register (IBCR / OBCR).

Signal **eray_ibusy** is activated when a write access to **IBCR.IBRH[6:0]** occurs while a transfer between Input Buffer Shadow and Message RAM is ongoing. This will also set bit **IBCR.IBSYH** to ’1’. After completion of the ongoing transfer, Input Buffer Host and Input Buffer Shadow are swapped, **IBCR.IBSYH** is reset to ’0’, and **eray_ibusy** is deactivated.

Signal **eray_obusy** is active as long as a transfer between Message RAM and Output Buffer Shadow is ongoing. This is signalled by bit **OBCR.OBSYS**.

In addition the status interrupt flags **SIR.TIBC** and **SIR.TOBC** are set whenever a data transfer has completed. If enabled, an interrupt is generated.

**Data transfer between IBF Shadow / OBF Shadow and Message RAM**




The delay time until the respective busy signal **(eray_ibusy or eray_obusy)** is reset depends on the **eray_bclk** frequency, the length of the data section of the message buffer to be accessed, and the actual state of the Message Handler. A formula for calculation of this delay can be found in the Addendum to E-Ray FlexRay IP-Module Specification.

## 17.2. Internal Signal and Flag Interface

The Internal Signal and Flag Interface is intended for E-Ray licensees who want to enhance their customer interface by additional functionality. The usage of these signals is optional. Signals (nets, ports) which are not connected to module-external logic will be removed by synthesis.

### Internal Signals

All internal signals which belong to the internal signal and flag interface are directly routed from their source to the top level of the E-Ray core. Signals with suffix '_sclk' origin from the sclk domain while all other signals origin from the bclk domain. The signals are active (HIGH) for one eray_bclk cycle resp. one eray_sclk cycle.

### Internal Flags

The bits of selected registers are routed to the internal signal and flag interface to make status information from these registers directly accessible for further processing. The assignment to bit positions and the behaviour with respect to set and reset is the same as for the respective register bits. The signals origin from the bclk domain.

### Internal signal and flag interface

| Signal | Direction | Description |
|---|---|---|
| **Internal Signals** | | |
| eray_cycs | O | Cycle Start |
| eray_cycs_sclk | O | Cycle Start, sclk domain |
| eray_mt | O | Macrotick Start |
| eray_mt_sclk | O | Macrotick Start, sclk domain |
| eray_sds | O | Start of Dynamic Segment |
| eray_mbsu_mbn1[6:0] | O | Message Buffer Status Updated for Message Buffer Number (0…127) on Channel A. Valid if either eray_mbsu_tx1 or eray_mbsu_rx1 is ’1’. |
| eray_mbsu_tx1 | O | Message Buffer Status Updated for Transmit Buffer Channel A |
| eray_mbsu_rx1 | O | Message Buffer Status Updated for Receive Buffer Channel A |
| eray_mbsu_mbn2[6:0] | O | Message Buffer Status Updated for Message Buffer Number (0…127) on Channel B. Valid if either eray_mbsu_tx2 or eray_mbsu_rx2 is ’1’. |
| eray_mbsu_tx2 | O | Message Buffer Status Updated for Transmit Buffer Channel B |
| eray_mbsu_rx2 | O | Message Buffer Status Updated for Receive Buffer Channel B |
| eray_mbsu_mbs[31:0] | O | Message Buffer Status Vector written to the Message Buffer referenced by eray_mbsu_mbn1,2[6:0]. Valid if either eray_mbsu_tx1,2 or eray_mbsu_rx1,2 is ’1’. Assignment to bit positions same as for register MBS. |
| eray_mbsu_txo[1:0] | O | Transmission of data frame Occurred on Channel A (bit #0) and/or B (bit #1) in the actual cycle. Valid if either eray_mbsu_tx1,2 or eray_mbsu_rx1,2 is ’1’. |
| eray_eir[31:0] | O | Error Interrupt Flags |
| eray_sir[31:0] | O | Status Interrupt Flags |
| eray_ccsv[31:0] | O | CC Status Vector |
| eray_ccev[31:0] | O | CC Error Vector |
| eray_scv[31:0] | O | Slot Counter Value |
| eray_mtccv[31:0] | O | Macrotick and Cycle Counter Value |
| eray_mrc[31:0] | O | Message RAM Configuration |
| eray_mhds[31:0] | O | Message Handler Status |
| eray_txrq1[31:0] | O | Transmission Request 1 |
| eray_txrq2[31:0] | O | Transmission Request 2 |
| eray_txrq3[31:0] | O | Transmission Request 3 |
| eray_txrq4[31:0] | O | Transmission Request 4 |
| eray_ndat1[31:0] | O | New Data 1 |
| eray_ndat2[31:0] | O | New Data 2 |
| eray_ndat3[31:0] | O | New Data 3 |
| eray_ndat4[31:0] | O | New Data 4 |
| eray_mbsc1[31:0] | O | Message Buffer Status Changed 1 |
| eray_mbsc2[31:0] | O | Message Buffer Status Changed 2 |
| eray_mbsc3[31:0] | O | Message Buffer Status Changed 3 |
| eray_mbsc4[31:0] | O | Message Buffer Status Changed 4 |

## 17.3. Physical Layer Interface

The physical layer interface connects the E-Ray module to the bus drivers.

### Physical layer interface

| Signal | Direction | Description |
|---|---|---|
| **Channel A** | | |
| eray_rxd1 | I | Data Receiver Input |
| eray_txd1 | O | Data Transmitter Output |
| eray_txen1_n | O | Transmit Enable signal,<br>HIGH = transmission inactive, LOW = transmission active |
| **Channel B** | | |
| eray_rxd2 | I | Data Receiver Input |
| eray_txd2 | O | Data Transmitter Output |
| eray_txen2_n | O | Transmit Enable signal,<br>HIGH = transmission inactive, LOW = transmission active |

For each of the two channels a separate bus driver device is required.

## 17.4. Interface to embedded RAM Blocks

The seven embedded RAM blocks used by the E-Ray module are connected to the E-Ray core via the interfaces described below. The E-Ray module is designed for connection to single-ported RAM with synchronous RD/WR. The width for all RAM blocks is 33 bit; 32 data bits and one parity bit.

### *17.4.1. Input Buffer Interface*

The Input Buffer RAM 1 interface has the following ports:

**Interface to Input Buffer RAM 1**

| Signal | Direction | Description |
|---|---|---|
| eray_bclk | O | Module Clock |
| eray_ibf1_addr[5:0] | O | Address Output |
| eray_ibf1_cen | O | RAM Select |
| eray_ibf1_wren | O | Write Control |
| eray_ibf1_data[32:0] | O | Write Data Output |
| eray_ibf1_q[32:0] | I | Read Data Input |

The Input Buffer RAM 2 interface has the following ports:

**Interface to Input Buffer RAM 2**

| Signal | Direction | Description |
|---|---|---|
| eray_bclk | O | Module Clock |
| eray_ibf2_addr[5:0] | O | Address Output |
| eray_ibf2_cen | O | RAM Select |
| eray_ibf2_wren | O | Write Control |
| eray_ibf2_data[32:0] | O | Write Data Output |
| eray_ibf2_q[32:0] | I | Read Data Input |

*17.4.2. Output Buffer Interface*

The Output Buffer RAM 1 interface has the following ports:

**Interface to Output Buffer RAM 1**

| Signal | Direction | Description |
|---|---|---|
| eray_bclk | O | Module Clock |
| eray_obf1_addr[5:0] | O | Address Output |
| eray_obf1_cen | O | RAM Select |
| eray_obf1_wren | O | Write Control |
| eray_obf1_data[32:0] | O | Write Data Output |
| eray_obf1_q[32:0] | I | Read Data Input |

The Output Buffer RAM 2 interface has the following ports:

**Interface to Output Buffer RAM 2**

| Signal | Direction | Description |
|---|---|---|
| eray_bclk | O | Module Clock |
| eray_obf2_addr[5:0] | O | Address Output |
| eray_obf2_cen | O | RAM Select |
| eray_obf2_wren | O | Write Control |
| eray_obf2_data[32:0] | O | Write Data Output |
| eray_obf2_q[32:0] | I | Read Data Input |

*17.4.3. Message RAM Interface*

The Message RAM stores header and data section of up to 128 message buffers. The Message RAM interface has the following ports:

**Interface to Message RAM**

| Signal | Direction | Description |
|---|---|---|
| eray_bclk | O | Module Clock |
| eray_mbf_addr[10:0] | O | Address Output |
| eray_mbf_cen | O | RAM Select |
| eray_mbf_wren | O | Write Control |
| eray_mbf_data[32:0] | O | Write Data Output |
| eray_mbf_q[32:0] | I | Read Data Input |

*17.4.4. Transient Buffer RAM Interface*

Each of the two FlexRay channels has a Transient Buffer RAM for intermediate message storage associated. The Transient Buffer RAM interface of channel A has the following ports:

### Interface to Transient Buffer RAM A

| Signal | Direction | Description |
|---|---|---|
| eray_bclk | O | Module Clock |
| eray_tbf1_addr[6:0] | O | Address Output |
| eray_tbf1_cen | O | RAM Select |
| eray_tbf1_wren | O | Write Control |
| eray_tbf1_data[32:0] | O | Write Data Output |
| eray_tbf1_q[32:0] | I | Read Data Input |

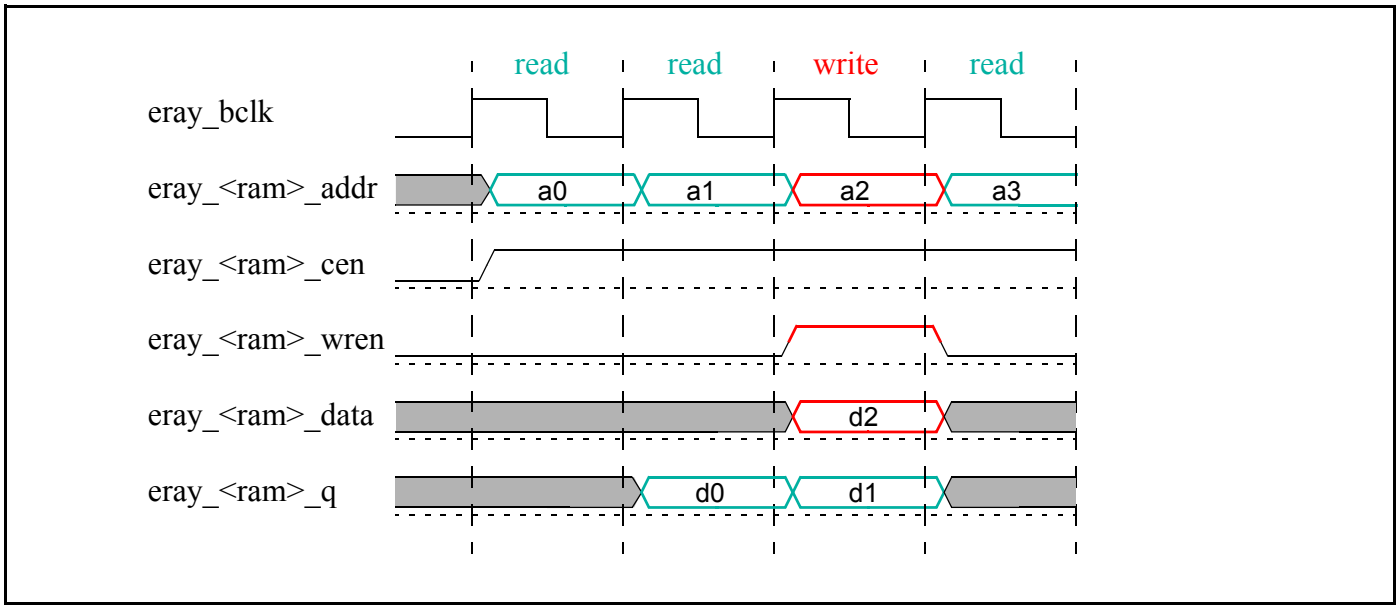
The Transient Buffer RAM interface of channel B has the following ports:

### Interface to Transient Buffer RAM B

| Signal | Direction | Description |
|---|---|---|
| eray_bclk | O | Module Clock |
| eray_tbf2_addr[6:0] | O | Address Output |
| eray_tbf2_cen | O | RAM Select |
| eray_tbf2_wren | O | Write Control |
| eray_tbf2_data[32:0] | O | Write Data Output |
| eray_tbf2_q[32:0] | I | Read Data Input |

*17.4.5. Read / write access to embedded RAM Blocks*

**Synchronous read/write access to embedded RAM blocks**



# 18. Programmer's Model

## 18.1. Register Map

The E-Ray module allocates an address space of 2 Kbytes (0x0000 to 0x07FF). The registers are organized as 32-bit registers. 8/16-bit accesses are also supported. Host access to the Message RAM is done via the Input and Output Buffers. They buffer data to be transferred to and from the Message RAM under control of the Message Handler, avoiding conflicts between Host accesses and message reception / transmission. Addresses 0x0000 to 0x000F are reserved for customer specific purposes. All functions related to these addresses are located in the Customer CPU Interface. The test registers located on address 0x0010 and 0x0014 are writable only under the conditions described in 18.3.Special Registers.

The assignment of the message buffers is done according to the scheme shown in Table  below. The number N of available message buffers depends on the payload length of the configured message buffers. The maximum number of message buffers is 128. The maximum payload length supported is 254 bytes.

The message buffers are separated into three consecutive groups:

- **Static Buffers:** Transmit / receive buffers assigned to static segment
- **Static + Dynamic Buffers:** Transmit / receive buffers assigned to static or dynamic segment
- **FIFO:** Receive FIFO

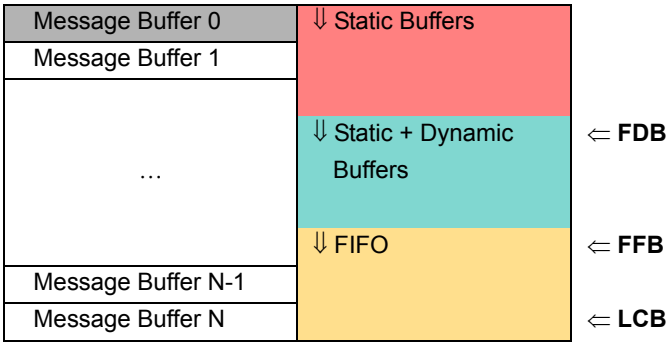
The message buffer separation configuration can be changed only in DEFAULT_CONFIG or CONFIG state only by programming register MRC (see 18.7.1.Message RAM Configuration (MRC)).

The first group starts with message buffer 0 and consists of static message buffers only. Message buffer 0 is dedicated to hold the startup / sync frame or the single slot frame, if the node transmits one, as configured by **SUCC1.TXST**, **SUCC1.TXSY**, and **SUCC1.TSM**. In addition, message buffer 1 may be used for sync frame transmission in case that sync frames or single-slot frames should have different payloads on the two channels. In this case bit **MRC.SPLM** has to be programmed to '1' and message buffers 0 and 1 have to be configured with the key slot ID and can be (re)configured in DEFAULT_CONFIG or CONFIG state only.

The second group consists of message buffers assigned to the static or to the dynamic segment. Message buffers belonging to this group may be reconfigured during run time from dynamic to static or vice versa depending on the state of **MRC.SEC[1:0]**.

The message buffers belonging to the third group are concatenated to a single receive FIFO.

### Assignment of message buffers



## E-Ray register map

| Address | Symbol | Name | Page | Reset | Acc | Block |
|---|---|---|---|---|---|---|
| **Customer Registers** | | | | | | |
| 0x0000<br>0x0004<br>0x0008<br>0x000C | see Customer CPU Interface Specification | | | | | CIF |
| **Special Registers** | | | | | | |
| 0x0010 | TEST1 | Test Register 1 | 119 | 0000 0300 | r/w | GIF |
| 0x0014 | TEST2 | Test Register 2 | 122 | 0000 0000 | r/w | |
| 0x0018 | | reserved (1) | | 0000 0000 | r | |
| 0x001C | LCK | Lock Register | 124 | 0000 0000 | r/w | GIF |
| **Interrupt Registers** | | | | | | |
| 0x0020 | EIR | Error Interrupt Register | 125 | 0000 0000 | r/w | INT |
| 0x0024 | SIR | Status Interrupt Register | 127 | 0000 0000 | r/w | |
| 0x0028 | EILS | Error Interrupt Line Select | 130 | 0000 0000 | r/w | |
| 0x002C | SILS | Status Interrupt Line Select | 131 | 0303 FFFF | r/w | |
| 0x0030 | EIES | Error Interrupt Enable Set | 132 | 0000 0000 | r/w | |
| 0x0034 | EIER | Error Interrupt Enable Reset | 132 | 0000 0000 | r/w | |
| 0x0038 | SIES | Status Interrupt Enable Set | 133 | 0000 0000 | r/w | |
| 0x003C | SIER | Status Interrupt Enable Reset | 133 | 0000 0000 | r/w | |
| 0x0040 | ILE | Interrupt Line Enable | 134 | 0000 0000 | r/w | |
| 0x0044 | T0C | Timer 0 Configuration | 135 | 0000 0000 | r/w | |
| 0x0048 | T1C | Timer 1 Configuration | 136 | 0002 0000 | r/w | |
| 0x004C | STPW1 | Stop Watch Register 1 | 137 | 0000 0000 | r/w | |
| 0x0050 | STPW2 | Stop Watch Register 2 | 138 | 0000 0000 | r/w | |
| 0x0054 to 0x007C | | reserved (11) | | 0000 0000 | r | |
| **CC Control Registers** | | | | | | |
| 0x0080 | SUCC1 | SUC Configuration Register 1 | 139 | 0C40 1080 | r/w | SUC |
| 0x0084 | SUCC2 | SUC Configuration Register 2 | 143 | 0100 0504 | r/w | |
| 0x0088 | SUCC3 | SUC Configuration Register 3 | 143 | 0000 0011 | r/w | |
| 0x008C | NEMC | NEM Configuration Register | 144 | 0000 0000 | r/w | NEM |
| 0x0090 | PRTC1 | PRT Configuration Register 1 | 145 | 084C 0633 | r/w | PRT |
| 0x0094 | PRTC2 | PRT Configuration Register 2 | 146 | 0F2D 0A0E | r/w | |
| 0x0098 | MHDC | MHD Configuration Register | 147 | 0000 0000 | r/w | MHD |
| 0x009C | | reserved (1) | | 0000 0000 | r | |
| 0x00A0 | GTUC1 | GTU Configuration Register 1 | 147 | 0000 0280 | r/w | GTU |
| 0x00A4 | GTUC2 | GTU Configuration Register 2 | 148 | 0002 000A | r/w | |
| 0x00A8 | GTUC3 | GTU Configuration Register 3 | 148 | 0202 0000 | r/w | |
| 0x00AC | GTUC4 | GTU Configuration Register 4 | 149 | 0008 0007 | r/w | |
| 0x00B0 | GTUC5 | GTU Configuration Register 5 | 149 | 0E00 0000 | r/w | |
| 0x00B4 | GTUC6 | GTU Configuration Register 6 | 150 | 0002 0000 | r/w | |
| 0x00B8 | GTUC7 | GTU Configuration Register 7 | 150 | 0002 0004 | r/w | |
| 0x00BC | GTUC8 | GTU Configuration Register 8 | 151 | 0000 0002 | r/w | |
| 0x00C0 | GTUC9 | GTU Configuration Register 9 | 151 | 0000 0101 | r/w | |
| 0x00C4 | GTUC10 | GTU Configuration Register 10 | 152 | 0002 0005 | r/w | |
| 0x00C8 | GTUC11 | GTU Configuration Register 11 | 152 | 0000 0000 | r/w | |

| Address | Symbol | Name | Page | Reset | Acc | Block |
|---|---|---|---|---|---|---|
| 0x00CC to 0x00FC | | reserved (13) | | 0000 0000 | r | |
| **CC Status Registers** | | | | | | |
| 0x0100 | CCSV | CC Status Vector | 153 | 0010 4000 | r | SUC |
| 0x0104 | CCEV | CC Error Vector | 155 | 0000 0000 | r | |
| 0x0108 to 0x010C | | reserved (2) | | 0000 0000 | r | |
| 0x0110 | SCV | Slot Counter Value | 155 | 0000 0000 | r | GTU |
| 0x0114 | MTCCV | Macrotick and Cycle Counter Value | 156 | 0000 0000 | r | |
| 0x0118 | RCV | Rate Correction Value | 156 | 0000 0000 | r | |
| 0x011C | OCV | Offset Correction Value | 157 | 0000 0000 | r | |
| 0x0120 | SFS | Sync Frame Status | 157 | 0000 0000 | r | |
| 0x0124 | SWNIT | Symbol Window and NIT Status | 158 | 0000 0000 | r | |
| 0x0128 | ACS | Aggregated Channel Status | 159 | 0000 0000 | r/w | |
| 0x012C | | reserved (1) | | 0000 0000 | r | |
| 0x0130 to 0x0168 | ESIDn | Even Sync ID [1…15] | 161 | 0000 0000 | r | |
| 0x016C | | reserved (1) | | 0000 0000 | r | |
| 0x0170 to 0x01A8 | OSIDn | Odd Sync ID [1…15] | 161 | 0000 0000 | r | |
| 0x01AC | | reserved (1) | | 0000 0000 | r | |
| 0x01B0 to 0x01B8 | NMVn | Network Management Vector [1…3] | 162 | 0000 0000 | r | NEM |
| 0x01BC to 0x02FC | | reserved (81) | | 0000 0000 | r | |
| **Message Buffer Control Registers** | | | | | | |
| 0x0300 | MRC | Message RAM Configuration | 163 | 0180 0000 | r/w | MHD |
| 0x0304 | FRF | FIFO Rejection Filter | 164 | 0180 0000 | r/w | |
| 0x0308 | FRFM | FIFO Rejection Filter Mask | 165 | 0000 0000 | r/w | |
| 0x030C | FCL | FIFO Critical Level | 165 | 0000 0080 | r/w | |

| Address | Symbol | Name | Page | Reset | Acc | Block |
|---|---|---|---|---|---|---|
| **Message Buffer Status Registers** | | | | | | |
| 0x0310 | MHDS | Message Handler Status | 166 | 0000 0080 | r/w | MHD |
| 0x0314 | LDTS | Last Dynamic Transmit Slot | 167 | 0000 0000 | r | |
| 0x0318 | FSR | FIFO Status Register | 168 | 0000 0000 | r | |
| 0x031C | MHDF | Message Handler Constraints Flags | 169 | 0000 0000 | r/w | |
| 0x0320 | TXRQ1 | Transmission Request 1 | 170 | 0000 0000 | r | |
| 0x0324 | TXRQ2 | Transmission Request 2 | 170 | 0000 0000 | r | |
| 0x0328 | TXRQ3 | Transmission Request 3 | 170 | 0000 0000 | r | |
| 0x032C | TXRQ4 | Transmission Request 4 | 170 | 0000 0000 | r | |
| 0x0330 | NDAT1 | New Data 1 | 171 | 0000 0000 | r | |
| 0x0334 | NDAT2 | New Data 2 | 171 | 0000 0000 | r | |
| 0x0338 | NDAT3 | New Data 3 | 171 | 0000 0000 | r | |
| 0x033C | NDAT4 | New Data 4 | 171 | 0000 0000 | r | |
| 0x0340 | MBSC1 | Message Buffer Status Changed 1 | 172 | 0000 0000 | r | |
| 0x0344 | MBSC2 | Message Buffer Status Changed 2 | 172 | 0000 0000 | r | |
| 0x0348 | MBSC3 | Message Buffer Status Changed 3 | 172 | 0000 0000 | r | |
| 0x034C | MBSC4 | Message Buffer Status Changed 4 | 172 | 0000 0000 | r | |
| 0x0350 to 0x03EC | | reserved (40) | | 0000 0000 | r | |
| **Identification Registers** | | | | | | |
| 0x03F0 | CREL | Core Release Register | 173 | [release info] | r | GIF |
| 0x03F4 | ENDN | Endian Register | 173 | 8765 4321 | r | |
| 0x03F8 to 0x03FC | | reserved (2) | | 0000 0000 | r | |
| **Input Buffer** | | | | | | |
| 0x0400 to 0x04FC | WRDSn | Write Data Section [1…64] | 174 | 0000 0000 | r/w | IBF |
| 0x0500 | WRHS1 | Write Header Section 1 | 175 | 0000 0000 | r/w | |
| 0x0504 | WRHS2 | Write Header Section 2 | 176 | 0000 0000 | r/w | |
| 0x0508 | WRHS3 | Write Header Section 3 | 176 | 0000 0000 | r/w | |
| 0x050C | | reserved (1) | | 0000 0000 | r/w | |
| 0x0510 | IBCM | Input Buffer Command Mask | 177 | 0000 0000 | r/w | |
| 0x0514 | IBCR | Input Buffer Command Request | 178 | 0000 0000 | r/w | |
| 0x0518 to 0x05FC | | reserved (58) | | 0000 0000 | r | |
| **Output Buffer** | | | | | | |
| 0x0600 to 0x06FC | RDDSn | Read Data Section [1…64] | 179 | 0000 0000 | r | OBF |
| 0x0700 | RDHS1 | Read Header Section 1 | 179 | 0000 0000 | r | |
| 0x0704 | RDHS2 | Read Header Section 2 | 180 | 0000 0000 | r | |
| 0x0708 | RDHS3 | Read Header Section 3 | 181 | 0000 0000 | r | |
| 0x070C | MBS | Message Buffer Status | 182 | 0000 0000 | r | |
| 0x0710 | OBCM | Output Buffer Command Mask | 184 | 0000 0000 | r/w | |
| 0x0714 | OBCR | Output Buffer Command Request | 185 | 0000 0000 | r/w | |
| 0x0718 to 0x07FC | | reserved (58) | | 0000 0000 | r | |

## 18.2. Customer Registers

The address space from 0x0000 to 0x000F is reserved for customer-specific registers. These registers, if implemented, are located in the Customer CPU Interface block. A description can be found in the specific Customer CPU Interface specification document.
Specification of the CIF registers is as follows:

```
CIF0 = 0xD000, CIF1 = 0xD004, CIF2 = 0xD008 and CIF3 = 0xD00C
```

### Customer Interface Logic

| | 31 | | | | | | | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| **CIF0** | VERSION | | | | | | | | |
| | R | | | | | | | | |
| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | bit |
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| **CIF1** | DREQO | DLVLO | DMODO | DENBO | DREQI | DLVLI | DMODI | DENBI | |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Initial |
| | R/W(RM1) | R/W | R/W | R/W | R/W(RM1) | R/W | R/W | R/W | Attribute |
| | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | bit |
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| **CIF1** | | | | MASK4 | MASK3 | MASK2 | MASK1 | MASK0 | |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Initial |
| | R0/W0 | R0/W0 | R0/W0 | R/W | R/W(RM1) | R/W | R/W | R/W | Attribute |
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | bit |
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| **CIF1** | RTEST | | | SWAP | TREQ1 | TENB1 | TREQ0 | TENB0 | |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Initial |
| | R/W | R0/W0 | R0/W0 | R/W | R/W(RM1) | R/W | R/W(RM1) | R/W | Attribute |

| | |
|---|---|
| R | Read only register |
| R/W | Read/write register |
| R/W(RM1) | Read/write register (outputs '1' on RMW) |
| R0/W0 | Read only register (always outputs '0', write '0' is recommended) |

All other CIF registers (CIF2 and CIF3) not mentioned here are R0/W0.

parameter VERSION = `32'h04_FF_5B_FF` ;
0x04: Cypress
0xFF: see Boot-ROM Device-ID
0x5B: FR:91(0x5B), FX:96(0x60)
0xFF: see E-Ray-ID

*18.2.1. CIF register functions*

**DENBI**: DMA request enable on IBF
**DENBO**: DMA request enable on OBF

DENBx = 0 : DMA request is disabled
DENBx = 1 : DMA request is enabled

**DMODI**: DMA request mode on IBF
**DMODO**:DMA request mode on OBF

DMODx = 0 : DMA request mode is **eray_ibusy/eray_obusy** level mode
DMODx = 1 : DMA request mode is **eray_ibusy/eray_obusy** edge mode

**DLVLI**: DMA level/edge selector on IBF
**DLVLO**: DMA level/edge selector on OBF

with DMODx = 0
DLVLx = 0 : DMA request level is non-inverted **eray_ibusy/eray_obusy**
DLVLx = 1 : DMA request level is inverted **eray_ibusy/eray_obusy**

with DMODx = 1
DLVLx = 0 : DMA request is negative edge of **eray_ibusy/eray_obusy**
DLVLx = 1 : DMA request is positive edge of **eray_ibusy/eray_obusy**

**DREQI**: DMA request flag on IBF
**DREQO**: DMA request flag on OBF

with DMODx = 0
DREQx = Read only, displays the **eray_ibusy/eray_obusy** level
- displays the modified **eray_ibusy/eray_obusy** level if changed with DLVLx
- On a read-modify-write bit-operation '1' is read

with DMODx = 1
DREQx = 0 : DMA request is inactive
DREQx = 1 : DMA request is active
Request is automatically cleared to '0' if a DMA transfer has started
- Possible to write '0' to clear the DMA request by the CPU
- On a read-modify-write bit-operation '1' is read

**MASK0**: DMA Channel 0 Interrupt Mask (for OBF configuration)
**MASK1**: DMA Channel 1 Interrupt Mask (for OBF configuration)
**MASK2**: DMA Channel 2 Interrupt Mask (for OBF configuration)
**MASK3**: DMA Channel 3 Interrupt Mask (for OBF configuration)
**MASK4**: DMA Channel 4 Interrupt Mask (for OBF configuration)

MASKx = 0 : DMA channel x interrupt is not masked
MASKx = 1 : DMA channel x interrupt is masked while **eray_obusy = 1**

**TENB0**: Timer 0 Interrupt Enable
**TENB1**: Timer 1 Interrupt Enable

TENBx = 0 : Direct **eray_tint0/eray_tint1** signal is used for interrupt generation
TENBx = 1 : Registered TREQx flag is used for interrupt generation

**TREQ0**: Timer 0 Interrupt Request
**TREQ1**: Timer 1 Interrupt Request

with TENBx = 0
TREQx = Read only, displays the **eray_tint0/eray_tint1** level
- On a read-modify-write bit-operation '1' is read

with TENBx = 1
TREQx = 0 : Timer interrupt request is inactive
TREQx = 1 : Timer interrupt request is active
- Possible to write '0' to clear the interrupt request by the CPU
- On a read-modify-write bit-operation '1' is read

**SWAP**: IBF/OBF data swap enable

SWAP = 0 : Read and write data on IBF/OBF is not swapped
SWAP = 1 : Read and write data on IBF/OBF is swapped

| **SWAP = O** | **SWAP = 1** |
|---|---|
| MD[ 7: 0] = DW(n), byte(n-1) | MD[ 7: 0] = DW(n+1), byte(n+2) |
| MD[15: 8] = DW(n), byte(n) | MD[15: 8] = DW(n+1), byte(n+1) |
| MD[23:16] = DW(n+1), byte(n+1) | MD[23:16] = DW(n), byte(n) |
| MD[31:24] = DW(n+1), byte(n+2) | MD[31:24] = DW(n), byte(n-1) |

**RTEST**: RAM Test address range enable (ONLY FOR TESTMODE)

RTEST = 0 : Normal operation address mapping
RTEST = 1 : RAM Test operation address mapping (use when TMC[1:0]=01)

## 18.3. Special Registers

### *18.3.1. Test Register 1 (TEST1)*

The Test Register 1 holds the control bits to configure the test modes of the E-Ray module. Write access to these bits is only possible if bit **WRTEN** is set to '1'.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **TEST1** | R | CERB3 | CERB2 | CERB1 | CERB0 | CERA3 | CERA2 | CERA1 | CERA0 | 0 | 0 | TXENB | TXENA | TXB | TXA | RXB | RXA |
| 0x0010 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | AOB | AOA | 0 | 0 | TMC1 | TMC0 | 0 | 0 | ELBE | WRTEN |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**WRTEN** Write Test Register Enable

Enables write access to the test registers. To set the bit from '0' to '1' the test mode key has to be written as defined in Section 18.3.3.Lock Register (LCK). The unlock sequence is not required when **WRTEN** is kept at '1' while other bits of the register are changed. The bit can be reset to '0' at any time.

1 =Write access to test registers enabled

0 =Write access to test registers disabled

**ELBE** External Loop Back Enable

There are two possibilities to perform a loop back test. External loop back via physical layer or internal loop back for in-system self-test (default). In case of an internal loop back pins **eray_txen1,2_n** are in their inactive state, pins **eray_txd1,2** are set to HIGH, pins **eray_rxd1,2** are not evaluated. Bit **ELBE** is evaluated only when POC is in loop back mode and test mode control is in normal operation mode **TMC[1:0]** = "00".

1 =External loop back

0 =Internal loop back (default)

**TMC[1:0]** Test Mode Control

00, 11=Normal operation mode (default)

01 =RAM Test Mode - All RAM blocks of the E-Ray module are directly accessible by the Host. This mode is intended to enable testing of the embedded RAM blocks during production testing.

10 =I/O Test Mode - Output pins **eray_txd1**, **eray_txd2**, **eray_txen1_n**, **eray_txen2_n**, are driven to the values defined by bits **TXA**, **TXB**, **TXENA**, **TXENB**. The values applied to the input pins **eray_rxd1**, **eray_rxd2** can be read from register bits **RXA**, **RXB**.

**AOA** Activity on A

The channel idle condition is specified in the FlexRay protocol spec v2.1, chapter 3, BITSTRB process zChannelIdle.

1 =Activity detected, channel A not idle

0 =No activity detected, channel A idle

**AOB** Activity on B

The channel idle condition is specified in the FlexRay protocol spec v2.1, chapter 3, BITSTRB process zChannelIdle.

1 =Activity detected, channel B not idle

0 =No activity detected, channel B idle

**CERA[3:0]** Coding Error Report Channel A

Set when a coding error is detected on channel A. Reset to zero when register TEST1 is read or written. Once the **CERA[3:0]** is set it will remain unchanged until the Host accesses the TEST1 register.

0000 =No coding error detected
0001 =Header CRC error detected
0010 =Frame CRC error detected
0011 =Frame Start Sequence FSS too long
0100 =First bit of Byte Start Sequence BSS seen LOW
0101 =Second bit of Byte Start Sequence BSS seen HIGH
0110 =First bit of Frame End Sequence FES seen HIGH
0111 =Second bit of Frame End Sequence FES seen LOW
1000 =CAS / MTS symbol seen too short
1001 =CAS / MTS symbol seen too long
1010...1111 =reserved

**CERB[3:0]** Coding Error Report Channel B

Set when a coding error is detected on channel B. Reset to zero when register TEST1 is read or written. Once the **CERB[3:0]** is set it will remain unchanged until the Host accesses the TEST1 register.

0000 =No coding error detected
0001 =Header CRC error detected
0010 =Frame CRC error detected
0011 =Frame Start Sequence FSS too long
0100 =First bit of Byte Start Sequence BSS seen LOW
0101 =Second bit of Byte Start Sequence BSS seen HIGH
0110 =First bit of Frame End Sequence FES seen HIGH
0111 =Second bit of Frame End Sequence FES seen LOW
1000 =CAS / MTS symbol seen too short
1001 =CAS / MTS symbol seen too long
1010...1111 =reserved

Note: Coding errors are also signalled when the CC is in MONITOR_MODE.
The error codes regarding CAS / MTS symbols concern only the monitored bit pattern, irrelevant whether those bit patterns are seen in the symbol window or elsewhere.

Note: The following TEST1 bits are used to test the interface to the physical layer (connectivity test) by driving / reading the respective pins.

**RXA** Monitor Channel A Receive Pin
0 =**eray_rxd1** = ’0’
1 =**eray_rxd1** = ’1’

**RXB** Monitor Channel B Receive Pin
0 =**eray_rxd2** = ’0’
1 =**eray_rxd2** = ’1’

**TXA** Control of Channel A Transmit Pin
0 =**eray_txd1** pin drives a ’0’
1 =**eray_txd1** pin drives a ’1’

**TXB** Control of Channel B Transmit Pin
0 =**eray_txd2** pin drives a ’0’
1 =**eray_txd2** pin drives a ’1’

**TXENA** Control of Channel A Transmit Enable Pin
0 =**eray_txen1_n** pin drives a ’0’
1 =**eray_txen1_n** pin drives a ’1’

**TXENB** Control of Channel B Transmit Enable Pin
0 =**eray_txen2_n** pin drives a ’0’
1 =**eray_txen2_n** pin drives a ’1’

*Asynchronous Transmit Mode (ATM)*

The asynchronous transmit mode is entered by writing **SUCC1.CMD[3:0]** = "1110" while the CC is in CONFIG state and bit **TEST1.WRTEN** is set to '1'. This write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key (unlock sequence). When called in any other state or when bit **TEST1.WRTEN** is not set, **SUCC1.CMD[3:0]** will be reset to "0000" = command_not_accepted. Reading **CCSV.POCS[5:0]** will return "00 1110" while the E-Ray module is in ATM mode. Asynchronous transmit mode can be left by writing **SUCC1.CMD[3:0]** = "0001" (CHI command: CONFIG).

In ATM mode transmission of a FlexRay frame is triggered by writing the number of the respective message buffer to **IBCR.IBRH[6:0]** while **IBCM.STXR** is set to '1'. In this mode wakeup, startup, and clock synchronization are bypassed. The CHI command SEND_MTS results in the immediate transmission of an MTS symbol.

The cycle counter value of frames send in ATM mode can be programmed via **MTCCV.CCV[5.0]** (writable in ATM and loop back mode only).

*Loop Back Mode*

The loop back mode is entered by writing **SUCC1.CMD[3:0]** = "1111" while the CC is in CONFIG state and bit **TEST1.WRTEN** is set to '1'. This write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key (unlock sequence). When called in any other state or when **TEST1.WRTEN** is not set, **SUCC1.CMD[3:0]** will be reset to "0000" = command_not_accepted. Reading **CCSV.POCS[5:0]** will return "00 1001" while the E-Ray module is in loop back mode.

Loop back mode can be left by writing **SUCC1.CMD[3:0]** = "0001" (CHI command: CONFIG).

The loop back mode is intended to check the module's internal data paths. Normal, time triggered operation is not possible in loop back mode.

There are two possibilities to perform a loop back test. External loop back via physical layer (**TEST1.ELBE** = '1') or internal loop back for in-system self-test (**TEST1.ELBE** = '0'). In case of an internal loop back pins **eray_txen1,2_n** are in their inactive state, pins **eray_txd1,2** are set to HIGH, pins **eray_rxd1,2** are not evaluated.

A loop back test is started by the Host configuring the E-Ray module and then writing a message to the Input Buffer and requesting the transmission by writing to register IBCR. The Message Handler will transfer the message into the Message RAM and then into the Transient Buffer of the selected channel. The Channel Protocol Controller (PRT) will read (in 32-bit words) the message from the transmit part of the Transient Buffer and load it into its
Rx / Tx shift register. The serial transmission is looped back into the shift register; its content is written into the receive part of the channels's Transient Buffer before the next word is loaded.

The PRT and the Message Handler will then treat this transmitted message like a received message, perform an acceptance filtering on frame ID and receive channel, and store the message into the Message RAM if it passed acceptance filtering. The loop back test ends with the Host requesting this received message from the Message RAM and then checking the contents of the Output Buffer.

Each FlexRay channel is tested separately. The E-Ray cannot receive messages from the FlexRay bus while it is in the loop back mode.

The cycle counter value of frames used in loop back mode can be programmed via **MTCCV.CCV[5.0]** (writable in ATM and loop back mode only).

Note that in case of an odd payload the last two bytes of the looped-back payload will be shifted by 16 bits to the right inside the last 32-bit data word.

### *18.3.2. Test Register 2 (TEST2)*

The Test Register 2 holds all bits required for the RAM test of the seven embedded RAM blocks of the E-Ray module. Write access to this register is only possible when **TEST1.WRTEN** is set to '1'.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **TEST2** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0014 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | RDPB | WRPB | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSEL2 | SSEL1 | SSEL0 | 0 | RS2 | RS1 | RS0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**RS[2:0]** RAM Select

In RAM Test mode the RAM blocks selected by **RS[2:0]** are mapped to module address 0x400 to 7FF(1024 byte addresses).

000 =Input Buffer RAM 1 (IBF1)
001 =Input Buffer RAM 2 (IBF2)
010 =Output Buffer RAM 1 (OBF1)
011 =Output Buffer RAM 2 (OBF2)
100 =Transient Buffer RAM A (TBF1)
101 =Transient Buffer RAM B (TBF2)
110 =Message RAM (MBF)
111 =unused

**SSEL[2:0]** Segment Select

To enable access to the complete Message RAM (8192 byte addresses) the Message RAM is segmented.

000 =access to RAM bytes 0000h to 03FFh enabled
001 =access to RAM bytes 0400h to 07FFh enabled
010 =access to RAM bytes 0800h to 0BFFh enabled
011 =access to RAM bytes 0C00h to 0FFFh enabled
100 =access to RAM bytes 1000h to 13FFh enabled
101 =access to RAM bytes 1400h to 17FFh enabled
110 =access to RAM bytes 1800h to 1BFFh enabled
111 =access to RAM bytes 1C00h to 1FFFh enabled

**WRPB** Write Parity Bit

Value of parity bit to be written to bit 32 of the addressed RAM word.

**RDPB** Read Parity Bit

Value of parity bit read from bit 32 of the addressed RAM word.

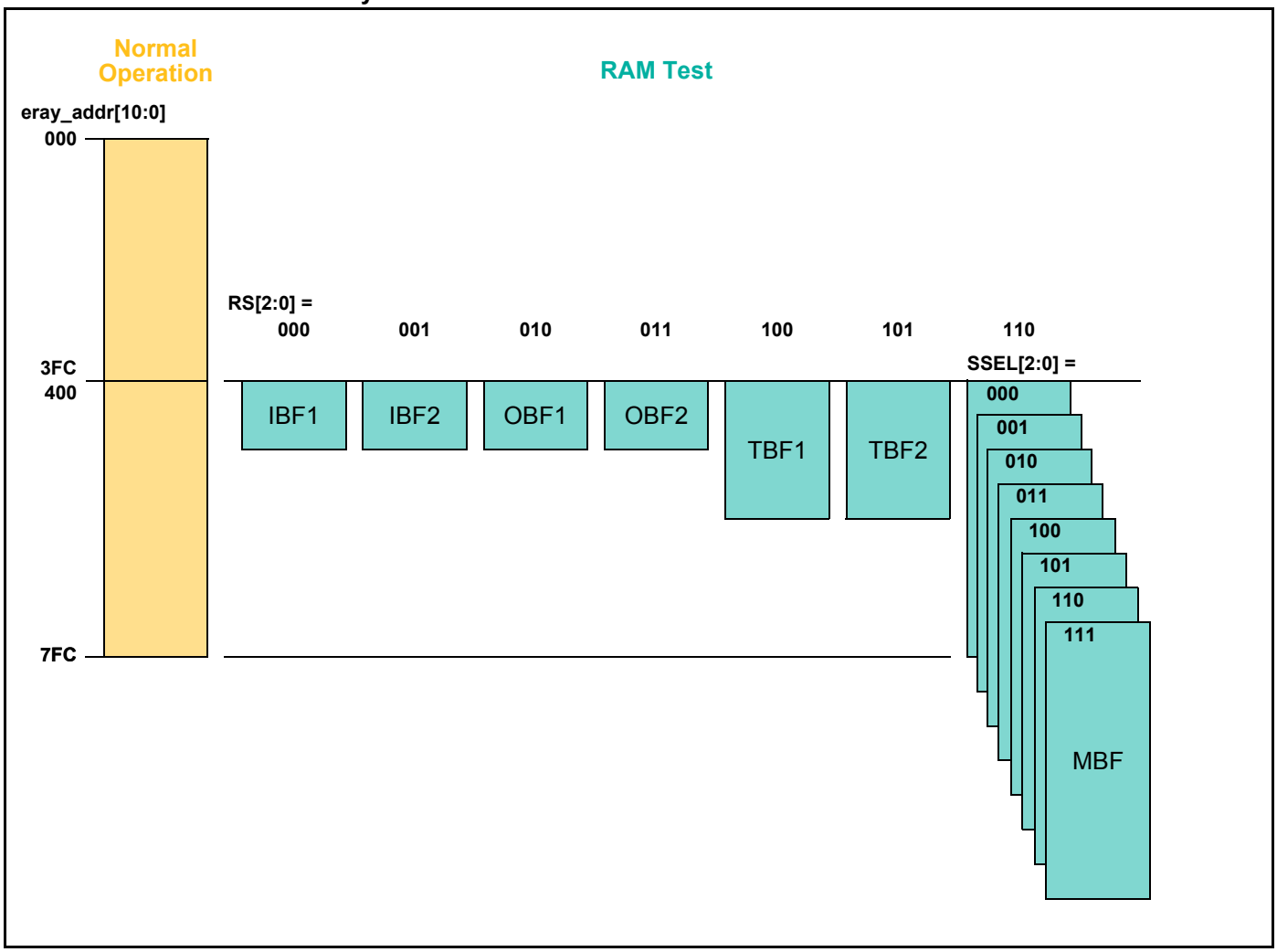
*RAM Test Mode*

In RAM test mode (**TEST1.TMC[1:0]** = "01"), one of the seven RAM blocks can be selected for direct RD/WR access by programming **TEST2.RS[2:0]**.

For external access the selected RAM block is mapped to address space 400h to 7FF (1024 byte addresses or 256 word addresses).

Because the length of the Message RAM exceeds the available address space, the Message RAM is segmented into segments of 1024 bytes. The segments can be selected by programming **TEST2.SSEL[2:0]**.

**RAM test mode access to E-Ray RAM blocks**



### *18.3.3. Lock Register (LCK)*

The Lock Register is write-only. Reading the register will return 0x0000 0000.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **LCK** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x001C | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | W | TMK7 | TMK6 | TMK5 | TMK4 | TMK3 | TMK2 | TMK1 | TMK0 | CLK7 | CLK6 | CLK5 | CLK4 | CLK3 | CLK2 | CLK1 | CLK0 |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**CLK[7:0]** Configuration Lock Key

To leave CONFIG state by writing **SUCC1.CMD[3:0]** (commands READY, MONITOR_MODE, ATM, LOOP_BACK), the write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key (unlock sequence). If this write sequence is interrupted by other write accesses, the CC remains in CONFIG state and the sequence has to be repeated.

First write:**LCK.CLK[7:0]**= "1100 1110" (0xCE)
Second write:**LCK.CLK[7:0]**= "0011 0001" (0x31)
Third write:**SUCC1.CMD[3:0]**

**TMK[7:0]** Test Mode Key

To write bit **TEST1.WRTEN**, the write operation has to be directly preceded by two consecutive write accesses to the Test Mode Key (unlock sequence). If this write sequence is interrupted by other write accesses, **TEST1.WRTEN** is not set to '1' and the sequence has to be repeated.

First write:**LCK.TMK[7:0]**= "0111 0101" (0x75)
Second write:**LCK.TMK[7:0]**= "1000 1010" (0x8A)
Third write:**TEST1.WRTEN**= '1'

Note: In case that the Host uses 8/16-bit accesses to write the listed bit fields, the programmer has to ensure that no "dummy accesses" e.g. to the remaining register bytes / words are inserted by the compiler.

## 18.4. Interrupt Registers

### *18.4.1. Error Interrupt Register (EIR)*

The flags are set when the CC detects one of the listed error conditions. The flags remain set until the Host clears them. A flag is cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect on the flag. A hard reset will also clear the register.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **EIR** | R | 0 | 0 | 0 | 0 | 0 | TABB | LTVB | EDB | 0 | 0 | 0 | 0 | 0 | TABA | LTVA | EDA |
| 0x0020 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | MHF | IOBA | IIBA | EFA | RFO | PERR | CCL | CCF | SFO | SFBM | CNA | PEMC |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PEMC** POC Error Mode Changed

This flag is set whenever the error mode signalled by **CCEV.ERRM[1:0]** has changed.

1 =Error mode has changed

0 =Error mode has not changed

**CNA** Command Not Accepted

The flag signals that the write access to the CHI command vector **SUCC1.CMD[3:0]** was not successful because the requested command was not valid in the actual POC state, or because the CHI command was locked (**CCL** = '1').

1 =CHI command not accepted

0 =CHI command accepted

**SFBM** Sync Frames Below Minimum

This flag signals that the number of sync frames received during the last communication cycle was below the limit required by the FlexRay protocol. May be set during startup and therefore should be cleared by the Host after the CC entered NORMAL_ACTIVE state.

1 =Less than the required minimum of sync frames received

0 =Sync node:1 or more sync frames received

Non-sync node:2 or more sync frames received

**SFO** Sync Frame Overflow

Set when the number of sync frames received during the last communication cycle exceeds the maximum number of sync frames as defined by **GTUC2.SNM[3:0]**.

1 =More sync frames received than configured by **GTUC2.SNM[3:0]**

0 =Number of received sync frames ≤ **GTUC2.SNM[3:0]**

**CCF** Clock Correction Failure

This flag is set at the end of the cycle whenever one of the following errors occurred:

- Missing offset and / or rate correction
- Clock correction limit reached

The clock correction status is monitored in registers CCEV and SFS. A failure may occur during startup, therefore bit **CCF** should be cleared by the Host after the CC entered NORMAL_ACTIVE state.

1 =Clock correction failed

0 =No clock correction error

**CCL** CHI Command Locked

The flag signals that the write access to the CHI command vector **SUCC1.CMD[3:0]** was not successful because it coincided with a POC state change triggered by protocol functions. In this case bit **CNA** is also set to '1'.

1 =CHI command not accepted

0 =CHI command accepted

**PERR** Parity Error

The flag signals a parity error to the Host. It is set whenever one of the flags **MHDS.PIBF**, **MHDS.POBF**, **MHDS.PMR**, **MHDS.PTBF1**, **MHDS.PTBF2** changes from ’0’ to ’1’.

1 =Parity error detected

0 =No parity error detected

**RFO** Receive FIFO Overrun

The flag is set by the CC when a receive FIFO overrun is detected. When a receive FIFO overrun occurs, the oldest message is overwritten with the actual received message. The actual state of the FIFO is monitored in register FSR.

1 =A receive FIFO overrun has been detected

0 =No receive FIFO overrun detected

**EFA** Empty FIFO Access

This flag is set by the CC when the Host requests the transfer of a message from the receive FIFO via Output Buffer while the receive FIFO is empty.

1 =Host access to empty FIFO occurred

0 =No Host access to empty FIFO occurred

**IIBA** Illegal Input Buffer Access

This flag is set by the CC when the Host wants to modify a message buffer via Input Buffer while the CC is not in CONFIG or DEFAULT_CONFIG state and one of the following conditions applies:

1) The Host writes to the Input Buffer Command Request register to modify the

- Header section of message buffer 0, 1 if configured for transmission in key slot
- Header section of static message buffers with buffer number < **MRC.FDB[7:0]** while **MRC.SEC[1:0]** = "01"
- Header section of any static or dynamic message buffer while **MRC.SEC[1:0]** = "1x"
- Header and / or data section of any message buffer belonging to the receive FIFO

2) The Host writes to any register of the Input Buffer while **IBCR.IBSYH** is set to ’1’.

1 =Illegal Host access to Input Buffer occurred

0 =No illegal Host access to Input Buffer occurred

**IOBA** Illegal Output buffer Access

This flag is set by the CC when the Host requests the transfer of a message buffer from the Message RAM to the Output Buffer while **OBCR.OBSYS** is set to ’1’.

1 =Illegal Host access to Output Buffer occurred

0 =No illegal Host access to Output Buffer occurred

**MHF** Message Handler Constraints Flag

The flag signals a Message Handler constraints violation condition. It is set whenever one of the flags **MHDF.SNUA**, **MHDF.SNUB**, **MHDF.FNFA**, **MHDF.FNFB**, **MHDF.TBFA**, **MHDF.TBFB**, **MHDF.WAHP** changes from ’0’ to ’1’.

1 =Message Handler failure detected

0 =No Message Handler failure detected

### Channel-specific error flags:

**EDA** Error Detected on Channel A

This bit is set whenever one of the flags **ACS.SEDA**, **ACS.CEDA**, **ACS.CIA**, **ACS.SBVA** changes from ’0’ to ’1’.

1 =Error detected on channel A

0 =No error detected on channel A

**LTVA** Latest Transmit Violation Channel A

The flag signals a latest transmit violation on channel A to the Host.

1 =Latest transmit violation detected on channel A

0 =No latest transmit violation detected on channel A

**TABA** Transmission Across Boundary Channel A

The flag signals to the Host that a transmission across a slot boundary occurred for channel A.

1 =Transmission across slot boundary detected on channel A

0 =No transmission across slot boundary detected on channel A

**EDB** Error Detected on Channel B

This bit is set whenever one of the flags **ACS.SEDB**, **ACS.CEDB**, **ACS.CIB**, **ACS.SBVB** changes from ’0’ to ’1’.

1 =Error detected on channel B

0 =No error detected on channel B

**LTVB** Latest Transmit Violation Channel B

The flag signals a latest transmit violation on channel B to the Host.

1 =Latest transmit violation detected on channel B

0 =No latest transmit violation detected on channel B

**TABB** Transmission Across Boundary Channel B

The flag signals to the Host that a transmission across a slot boundary occurred for channel B.

1 =Transmission across slot boundary detected on channel B

0 =No transmission across slot boundary detected on channel B

*18.4.2. Status Interrupt Register (SIR)*

The flags are set when the CC detects one of the listed events. The flags remain set until the Host clears them. A flag is cleared by writing a ’1’ to the corresponding bit position. Writing a ’0’ has no effect on the flag. A hard reset will also clear the register.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **SIR** | R | 0 | 0 | 0 | 0 | 0 | 0 | MTSB | WUPB | 0 | 0 | 0 | 0 | 0 | 0 | MTSA | WUPA |
| 0x0024 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | SDS | MBSI | SUCS | SWE | TOBC | TIBC | TI1 | TI0 | NMVC | RFCL | RFNE | RXI | TXI | CYCS | CAS | WST |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**WST** Wakeup Status

This flag is set whenever the wakeup status vector **CCSV.WSV[2:0]** has changed.

1 =Wakeup status changed

0 =Wakeup status unchanged

**CAS** Collision Avoidance Symbol

This flag is set by the CC during STARTUP state when a CAS or a potential CAS was received.

1 =Bit pattern matching the CAS symbol received

0 =No bit pattern matching the CAS symbol received

**CYCS** Cycle Start Interrupt

This flag is set by the CC when a communication cycle starts.

1 =Communication cycle started

0 =No communication cycle started

**TXI** Transmit Interrupt

This flag is set by the CC at the end of frame transmission if bit **MBI** in the respective message buffer is set to ’1’ (see tableHeader section of a message buffer in the Message RAM).

1 =At least one frame was transmitted from a transmit buffer with **MBI** = ’1’

0 =No frame transmitted from a transmit buffer with **MBI** = ’1’

**RXI** Receive Interrupt

This flag is set by the CC whenever the set condition of a message buffers ND flag is fulfilled (see 18.8.6.New Data 1/2/3/4 (NDAT1/2/3/4)), and if bit **MBI** of that message buffer is set to ’1’ (see tableHeader section of a message buffer in the Message RAM).

1 =At least one ND flag of a receive buffer with **MBI** = ’1’ has been set to ’1’

0 =No ND flag of a receive buffer with **MBI** = ’1’ has been set to ’1’

**RFNE** Receive FIFO Not Empty

This flag is set by the CC when a received valid frame was stored into the empty receive FIFO. The actual state of the receive FIFO is monitored in register FSR.

1 =Receive FIFO is not empty

0 =Receive FIFO is empty

**RFCL** Receive FIFO Critical Level

This flag is set when the receive FIFO fill level **FSR.RFFL[7:0]** is equal or greater than the critical level as configured by **FCL.CL[7:0]**.

1 =Receive FIFO critical level reached

0 =Receive FIFO below critical level

**NMVC** Network Management Vector Changed

This interrupt flag signals a change in the Network Management Vector visible to the Host.

1 =Network management vector changed

0 =No change in the network management vector

**TI0** Timer Interrupt 0

This flag is set whenever timer 0 matches the conditions configured in register T0C. A Timer Interrupt 0 is also signalled on pin **eray_tint0**.

1 =Timer Interrupt 0 occurred

0 =No Timer Interrupt 0

**TI1** Timer Interrupt 1

This flag is set whenever timer 1 matches the conditions configured in register T1C. A Timer Interrupt 1 is also signalled on pin **eray_tint1**.

1 =Timer Interrupt 1 occurred

0 =No Timer Interrupt 1

**TIBC** Transfer Input Buffer Completed

This flag is set whenever a transfer from Input Buffer to the Message RAM has completed and **IBCR.IBSYS** has been reset by the Message Handler.

1 =Transfer between Input Buffer and Message RAM completed

0 =No transfer completed

**TOBC** Transfer Output Buffer Completed

This flag is set whenever a transfer from the Message RAM to the Output Buffer has completed and **OBCR.OBSYS** has been reset by the Message Handler.

1 =Transfer between Message RAM and Output Buffer completed

0 =No transfer completed

**SWE** Stop Watch Event

If enabled by the respective control bits located in register STPW1, a rising or falling edge on pin **eray_stpwt**, an interrupt 0,1 event (rising edge on pin **eray_int0** or **eray_int1**) or a software trigger event will generate a stop watch event.

1 =Stop Watch Event occurred

0 =No Stop Watch Event

**SUCS** Startup Completed Successfully

This flag is set whenever a startup completed successfully and the CC entered NORMAL_ACTIVE state.

1 =Startup completed successfully

0 =No startup completed successfully

**MBSI** Message Buffer Status Interrupt

This flag is set by the CC when the message buffer status **MBS** has changed if bit **MBI** of that message buffer is set (see tableHeader section of a message buffer in the Message RAM).

1 =Message buffer status of at least one message buffer with **MBI** = ’1’ has changed

0 =No message buffer status change of message buffer with **MBI** = ’1’

**SDS** Start of Dynamic Segment

This flag is set by the CC when the dynamic segment starts.

1 =Dynamic segment started

0 =Dynamic segment not yet started

**Channel-specific status flags:**

**WUPA** Wakeup Pattern Channel A

This flag is set by the CC when a wakeup pattern was received on channel A. Only set when the CC is in WAKEUP, READY, or STARTUP state, or when in Monitor mode.

1 =Wakeup pattern received on channel A

0 =No wakeup pattern received on channel A

**MTSA** MTS Received on Channel A (vSS!ValidMTSA)

Media Access Test symbol received on channel A during the last symbol window. Updated by the CC for each channel at the end of the symbol window.

1 =MTS symbol received on channel A

0 =No MTS symbol received on channel A

**WUPB** Wakeup Pattern Channel B

This flag is set by the CC when a wakeup pattern was received on channel B. Only set when the CC is in WAKEUP, READY, or STARTUP state, or when in Monitor mode.

1 =Wakeup pattern received on channel B

0 =No wakeup pattern received on channel B

**MTSB** MTS Received on Channel B (vSS!ValidMTSB)

Media Access Test symbol received on channel B during the last symbol window. Updated by the CC for each channel at the end of the symbol window.

1 =MTS symbol received on channel B

0 =No MTS symbol received on channel B

### 18.4.3. Error Interrupt Line Select (EILS)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **EILS** | R | 0 | 0 | 0 | 0 | 0 | TABBL | LTVBL | EDBL | 0 | 0 | 0 | 0 | 0 | TABAL | LTVAL | EDAL |
| 0x0028 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | MHFL | IOBAL | IIBAL | EFAL | RFOL | PERRL | CCLL | CCFL | SFOL | SFBML | CNAL | PEMCL |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The Error Interrupt Line Select register assigns an interrupt generated by a specific error interrupt flag from register EIR to one of the two module interrupt lines:

1 =Interrupt assigned to interrupt line **eray_int1**
0 =Interrupt assigned to interrupt line **eray_int0**

**PEMCL** POC Error Mode Changed Interrupt Line
**CNAL** Command Not Accepted Interrupt Line
**SFBML** Sync Frames Below Minimum Interrupt Line
**SFOL** Sync Frame Overflow Interrupt Line
**CCFL** Clock Correction Failure Interrupt Line
**CCLL** CHI Command Locked Interrupt Line
**PERRL** Parity Error Interrupt Line
**RFOL** Receive FIFO Overrun Interrupt Line
**EFAL** Empty FIFO Access Interrupt Line
**IIBAL** Illegal Input Buffer Access Interrupt Line
**IOBAL** Illegal Output Buffer Access Interrupt Line
**MHFL** Message Handler Constraints Flag Interrupt Line
**EDAL** Error Detected on Channel A Interrupt Line
**LTVAL** Latest Transmit Violation Channel A Interrupt Line
**TABAL** Transmission Across Boundary Channel A Interrupt Line
**EDBL** Error Detected on Channel B Interrupt Line
**LTVBL** Latest Transmit Violation Channel B Interrupt Line
**TABBL** Transmission Across Boundary Channel B Interrupt Line

### 18.4.4. Status Interrupt Line Select (SILS)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **SILS** | R | 0 | 0 | 0 | 0 | 0 | 0 | MTSBL | WUPBL | 0 | 0 | 0 | 0 | 0 | 0 | MTSAL | WUPAL |
| 0x002C | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | SDSL | MBSIL | SUCSL | SWEL | TOBCL | TIBCL | TI1L | TI0L | NMVCL | RFCLL | RFNEL | RXIL | TXIL | CYCSL | CASL | WSTL |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

The Status Interrupt Line Select register assign an interrupt generated by a specific status interrupt flag from register SIR to one of the two module interrupt lines:

1 =Interrupt assigned to interrupt line **eray_int1**
0 =Interrupt assigned to interrupt line **eray_int0**

**WSTL** Wakeup Status Interrupt Line
**CASL** Collision Avoidance Symbol Interrupt Line
**CYCSL** Cycle Start Interrupt Line
**TXIL** Transmit Interrupt Line
**RXIL** Receive Interrupt Line
**RFNEL** Receive FIFO Not Empty Interrupt Line
**RFCLL** Receive FIFO Critical Level Interrupt Line
**NMVCL** Network Management Vector Changed Interrupt Line
**TI0L** Timer Interrupt 0 Line
**TI1L** Timer Interrupt 1 Line
**TIBCL** Transfer Input Buffer Completed Interrupt Line
**TOBCL** Transfer Output Buffer Completed Interrupt Line
**SWEL** Stop Watch Event Interrupt Line
**SUCSL** Startup Completed Successfully Interrupt Line
**MBSIL** Message Buffer Status Interrupt Line
**SDSL** Start of Dynamic Segment Interrupt Line
**WUPAL** Wakeup Pattern Channel A Interrupt Line
**MTSAL** Media Access Test Symbol Channel A Interrupt Line
**WUPBL** Wakeup Pattern Channel B Interrupt Line
**MTSBL** Media Access Test Symbol Channel B Interrupt Line

*18.4.5. Error Interrupt Enable Set / Reset (EIES, EIER)*

The settings in the Error Interrupt Enable register determine which status changes in the Error Interrupt Register will result in an interrupt.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **EIES,R** | R | 0 | 0 | 0 | 0 | 0 | TABBE | LTVBE | EDBE | 0 | 0 | 0 | 0 | 0 | TABAE | LTVAE | EDAE |
| S:0x0030R :0x0034 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | MHFE | IOBAE | IIBAE | EFAE | RFOE | PERRE | CCLE | CCFE | SFOE | SFBME | CNAE | PEMCE |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The enable bits are set by writing to address 0x0030 and reset by writing to address 0x0034. Writing a ’1’ sets / resets the specific enable bit, writing a ’0’ has no effect. Reading from both addresses will result in the same value.

1 =Interrupt enabled

0 =Interrupt disabled

**PEMCE** POC Error Mode Changed Interrupt Enable
**CNAE** Command Not Accepted Interrupt Enable
**SFBME** Sync Frames Below Minimum Interrupt Enable
**SFOE** Sync Frame Overflow Interrupt Enable
**CCFE** Clock Correction Failure Interrupt Enable
**CCLE** CHI Command Locked Interrupt Enable
**PERRE** Parity Error Interrupt Enable
**RFOE** Receive FIFO Overrun Interrupt Enable
**EFAE** Empty FIFO Access Interrupt Enable
**IIBAE** Illegal Input Buffer Access Interrupt Enable
**IOBAE** Illegal Output Buffer Access Interrupt Enable
**MHFE** Message Handler Constraints Flag Interrupt Enable
**EDAE** Error Detected on Channel A Interrupt Enable
**LTVAE** Latest Transmit Violation Channel A Interrupt Enable
**TABAE** Transmission Across Boundary Channel A Interrupt Enable
**EDBE** Error Detected on Channel B Interrupt Enable
**LTVBE** Latest Transmit Violation Channel B Interrupt Enable
**TABBE** Transmission Across Boundary Channel B Interrupt Enable

### *18.4.6. Status Interrupt Enable Set / Reset (SIES, SIER)*

The settings in the Status Interrupt Enable register determine which status changes in the Status Interrupt Register will result in an interrupt.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **SIES,R** | R | 0 | 0 | 0 | 0 | 0 | 0 | MTSBE | WUPBE | 0 | 0 | 0 | 0 | 0 | 0 | MTSAE | WUPAE |
| S:0x0038 R:0x003C | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | SDSE | MBSIE | SUCSE | SWEE | TOBCE | TIBCE | TI1E | TI0E | NMVCE | RFCLE | RFNEE | RXIE | TXIE | CYCSE | CASE | WSTE |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The enable bits are set by writing to address 0x0038 and reset by writing to address 0x003C. Writing a ’1’ sets / resets the specific enable bit, writing a ’0’ has no effect. Reading from both addresses will result in the same value.

1 =Interrupt enabled

0 =Interrupt disabled

**WSTE** Wakeup Status Interrupt Enable
**CASE** Collision Avoidance Symbol Interrupt Enable
**CYCSE** Cycle Start Interrupt Enable
**TXIE** Transmit Interrupt Enable
**RXIE** Receive Interrupt Enable
**RFNEE** Receive FIFO Not Empty Interrupt Enable
**RFCLE** Receive FIFO Critical Level Interrupt Enable
**NMVCE** Network Management Vector Changed Interrupt Enable
**TI0E** Timer Interrupt 0 Enable
**TI1E** Timer Interrupt 1 Enable
**TIBCE** Transfer Input Buffer Completed Interrupt Enable
**TOBCE** Transfer Output Buffer Completed Interrupt Enable
**SWEE** Stop Watch Event Interrupt Enable
**SUCSE** Startup Completed Successfully Interrupt Enable
**MBSIE** Message Buffer Status Interrupt Enable
**SDSE** Start of Dynamic Segment Interrupt Enable
**WUPAE** Wakeup Pattern Channel A Interrupt Enable
**MTSAE** MTS Received on Channel A Interrupt Enable
**WUPBE** Wakeup Pattern Channel B Interrupt Enable
**MTSBE** MTS Received on Channel B Interrupt Enable

### *18.4.7. Interrupt Line Enable (ILE)*

Each of the two interrupt lines to the Host (**eray_int0**, **eray_int1**) can be enabled / disabled separately by programming bit **EINT0** and **EINT1**.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **ILE** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0040 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | EINT1 | EINT0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**EINT0** Enable Interrupt Line 0
- 1 =Interrupt line **eray_int0** enabled
- 0 =Interrupt line **eray_int0** disabled

**EINT1** Enable Interrupt Line 1
- 1 =Interrupt line **eray_int1** enabled
- 0 =Interrupt line **eray_int1** disabled

### *18.4.8. Timer 0 Configuration (T0C)*

Absolute timer. Specifies in terms of cycle count and macrotick the point in time when the timer 0 interrupt occurs. When the timer 0 interrupt is asserted, output signal **eray_tint0** is set to '1' for the duration of one macrotick and **SIR.TI0** is set to '1'.

Timer 0 can be activated as long as the POC is either in NORMAL_ACTIVE state or in NORMAL_PASSIVE state. Timer 0 is deactivated when leaving NORMAL_ACTIVE state or NORMAL_PASSIVE state except for transitions between the two states.

Before reconfiguration of the timer, the timer has to be halted first by writing bit **T0RC** to '0'.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **T0C** | R | 0 | 0 | T0MO13 | T0MO12 | T0MO11 | T0MO10 | T0MO9 | T0MO8 | T0MO7 | T0MO6 | T0MO5 | T0MO4 | T0MO3 | T0MO2 | T0MO1 | T0MO0 |
| 0x0044 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | T0CC6 | T0CC5 | T0CC4 | T0CC3 | T0CC2 | T0CC1 | T0CC0 | 0 | 0 | 0 | 0 | 0 | 0 | T0MS | T0RC |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**T0RC** Timer 0 Run Control
1 =Timer 0 running
0 =Timer 0 halted

**T0MS** Timer 0 Mode Select
1 =Continuous mode
0 =Single-shot mode

**T0CC[6:0]** Timer 0 Cycle Code
The 7-bit timer 0 cycle code determines the cycle set used for generation of the timer 0 interrupt. For details about the configuration of the cycle code see Section 19.7.2.Cycle Counter Filtering.

**T0MO[13:0]** Timer 0 Macrotick Offset
Configures the macrotick offset from the beginning of the cycle where the interrupt is to occur. The Timer 0 Interrupt occurs at this offset for each cycle of the cycle set.

Note: The configuration of timer 0 is compared against the macrotick counter value, there is no separate counter for timer 0. In case the CC leaves NORMAL_ACTIVE or NORMAL_PASSIVE state, or if timer 0 is halted by Host command, output signal **eray_tint0** is reset to '0' immediately.

### 18.4.9. Timer 1 Configuration (T1C)

Relative timer. After the specified number of macroticks has expired, the timer 1 interrupt is asserted, output signal **eray_tint1** is set to '1' for the duration of one macrotick and **SIR.TI1** is set to '1'.

Timer 1 can be activated as long as the POC is either in NORMAL_ACTIVE state or in NORMAL_PASSIVE state. Timer 1 is deactivated when leaving NORMAL_ACTIVE state or NORMAL_PASSIVE state except for transitions between the two states.

Before reconfiguration of the timer, the timer has to be halted first by writing bit **T1RC** to '0'.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **T1C** | R | 0 | 0 | T1MC 13 | T1MC 12 | T1MC 11 | T1MC 10 | T1MC9 | T1MC8 | T1MC7 | T1MC6 | T1MC5 | T1MC4 | T1MC3 | T1MC2 | T1MC1 | T1MC0 |
| 0x0048 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | T1MS | T1RC |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**T1RC** Timer 1 Run Control
1 =Timer 1 running
0 =Timer 1 halted

**T1MS** Timer 1 Mode Select
1 =Continuous mode
0 =Single-shot mode

**T1MC[13:0]** Timer 1 Macrotick Count
When the configured macrotick count is reached the timer 1 interrupt is generated. In case the configured macrotick count is not within the valid range, timer 1 will not start.
Valid values are:
2 to 16383 MT in continuous mode
1 to 16383 MT in single-shot mode

Note: In case the CC leaves NORMAL_ACTIVE or NORMAL_PASSIVE state, or if timer 1 is halted by Host command, output signal **eray_tint1** is reset to '0' immediately.

### 18.4.10. Stop Watch Register 1 (STPW1)

The stop watch is activated by a rising or falling edge on pin **eray_stpwt**, by an interrupt 0,1 event (rising edge on pin **eray_int0** or **eray_int1**) or by the Host by writing bit **SSWT** to '1'. With the macrotick counter increment following next to the stop watch activation the actual cycle counter and macrotick values are captured in register STPW1 while the slot counter values for channel A and B are captured in register STPW2.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **STPW1** | R | 0 | 0 | SMTV 13 | SMTV 12 | SMTV 11 | SMTV 10 | SMTV9 | SMTV8 | SMTV7 | SMTV6 | SMTV5 | SMTV4 | SMTV3 | SMTV2 | SMTV1 | SMTV0 |
| 0x004C | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | SCCV5 | SCCV4 | SCCV3 | SCCV2 | SCCV1 | SCCV0 | 0 | EINT1 | EINT0 | EETP | SSWT | EDGE | SWMS | ESWT |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ESWT** Enable Stop Watch Trigger

If enabled an edge on input **eray_stpwt** or an interrupt 0,1 event (rising edge on pin **eray_int0** or **eray_int1**) activates the stop watch. In single-shot mode this bit is reset to '0' after the stop watch event occurred.

1 =Stop watch trigger enabled

0 =Stop watch trigger disabled

**SWMS** Stop Watch Mode Select

1 =Continuous mode

0 =Single-shot mode

**EDGE** Stop Watch Trigger Edge Select

1 =Rising edge

0 =Falling edge

**SSWT** Software Stop Watch Trigger

When the Host writes this bit to '1' the stop watch is activated. After the actual cycle counter and macrotick value are stored in the Stop Watch register this bit is reset to '0'. The bit is only writeable while **ESWT** = '0'.

1 =Stop watch activated by software trigger

0 =Software trigger reset

**EETP** Enable External Trigger Pin

Enables stop watch trigger event via pin **eray_stpwt** if **ESWT** = '1'.

1 =Edge on pin **eray_stpwt** triggers stop watch

0 =Stop watch trigger via pin **eray_stpwt** disabled

**EINT0** Enable Interrupt 0 Trigger

Enables stop watch trigger by interrupt 0 event if **ESWT** = '1'.

1 =Interrupt 0 event triggers stop watch

0 =Stop watch trigger by interrupt 0 disabled

**EINT1** Enable Interrupt 1 Trigger

Enables stop watch trigger by interrupt 1event if **ESWT** = '1'.

1 =Interrupt 1 event triggers stop watch

0 =Stop watch trigger by interrupt 1 disabled

**SCCV[5:0]** Stop Watch Captured Cycle Counter Value

State of the cycle counter when the stop watch event occurred. Valid values are 0 to 63.

**SMTV[13:0]** Stop Watch Captured Macrotick Value

State of the macrotick counter when the stop watch event occurred. Valid values are 0 to 16000.

Bits **ESWT** and **SSWT** cannot be set to '1' simultaneously. In this case the write access is ignored, and both bits keep their previous values. Either the external stop watch trigger or the software stop watch trigger may be used.

### 18.4.11. Stop Watch Register 2 (STPW2)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **STPW2** | R | 0 | 0 | 0 | 0 | 0 | SSCVB 10 | SSCVB 9 | SSCVB 8 | SSCVB 7 | SSCVB 6 | SSCVB 5 | SSCVB 4 | SSCVB 3 | SSCVB 2 | SSCVB 1 | SSCVB 0 |
| 0x0050 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | SSCVA 10 | SSCVA 9 | SSCVA 8 | SSCVA 7 | SSCVA 6 | SSCVA 5 | SSCVA 4 | SSCVA 3 | SSCVA 2 | SSCVA 1 | SSCVA 0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**SSCVA[10:0]** Stop Watch Captured Slot Counter Value Channel A

State of the slot counter for channel A when the stop watch event occurred. Valid values are 0 to 2047.

**SSCVB[10:0]** Stop Watch Captured Slot Counter Value Channel B

State of the slot counter for channel B when the stop watch event occurred. Valid values are 0 to 2047.

## 18.5. CC Control Registers

This section describes the registers provided by the CC to allow the Host to control the operation of the CC. The FlexRay protocol specification requires the Host to write application configuration data in CONFIG state only. Please consider that the configuration registers are not locked for writing in DEFAULT_CONFIG state.

The configuration data is reset when DEFAULT_CONFIG state is entered from hard reset. To change POC state from DEFAULT_CONFIG to CONFIG state the Host has to apply CHI command CONFIG. If the Host wants the CC to leave CONFIG state, the Host has to proceed as described in Section 18.3.3.Lock Register (LCK).

All bits marked with an asterisk * can be updated in DEFAULT_CONFIG or CONFIG state only!

### *18.5.1. SUC Configuration Register 1 (SUCC1)*

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **SUCC1** | R | 0 | 0 | 0 | 0 | CCHB* | CCHA* | MTSB* | MTSA* | HCSE* | TSM* | WUCS* | PTA4* | PTA3* | PTA2* | PTA1* | PTA0* |
| 0x0080 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | CSA4* | CSA3* | CSA2* | CSA1* | CSA0* | 0 | TXSY* | TXST* | PBSY | 0 | 0 | 0 | CMD3 | CMD2 | CMD1 | CMD0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**CMD[3:0]** CHI Command Vector

The Host may write any CHI command at any time, but certain commands are enabled only in certain POC states. If a command is not enabled, it will not be executed, the CHI command vector **CMD[3:0]** will be reset to "0000" = command_not_accepted, and flag **EIR.CNA** will be set to '1'. In this case **EIR.CCL** is set to '1' together with **EIR.CNA**; the CHI command needs to be repeated. When applying a POC state change command while the CC is already in the requested POC state, this command is ignored.

0000 =command_not_accepted
0001 =CONFIG
0010 =READY
0011 =WAKEUP
0100 =RUN
0101 =ALL_SLOTS
0110 =HALT
0111 =FREEZE
1000 =SEND_MTS
1001 =ALLOW_COLDSTART
1010 =RESET_STATUS_INDICATORS
1011 =MONITOR_MODE
1100 =CLEAR_RAMS
1101 =reserved
1110 =reserved
1111 =reserved

Reading **CMD[3:0]** shows whether the last CHI command was accepted. The actual POC state is monitored by **CCSV.POCS[5:0]**. The reserved CHI commands belong to the hardware test functions.

**command_not_accepted**
**CMD[3:0]** is reset to "0000" due to one of the following conditions:
• Illegal command applied by the Host
• Host applied command to leave CONFIG state without preceding config lock key
• Host applied new command while execution of the previous Host command has not completed
• Host writes **command_not_accepted**
When **CMD[3:0]** is reset to "0000", **EIR.CNA** is set, and - if enabled - an interrupt is generated. Commands which are not accepted are not executed.

**CONFIG**
Go to POC state CONFIG when called in POC states DEFAULT_CONFIG, READY, or in MONITOR_MODE. When called in HALT state the CC transits to POC state DEFAULT_CONFIG. When called in any other state, **CMD[3:0]** will be reset to "0000" = command_not_accepted.

**READY**
Go to POC state READY when called in POC states CONFIG, NORMAL_ACTIVE, NORMAL_PASSIVE, STARTUP, or WAKEUP. When called in any other state, **CMD[3:0]** will be reset to "0000" = command_not_accepted.

**WAKEUP**
Go to POC state WAKEUP when called in POC state READY. When called in any other state, **CMD[3:0]** will be reset to "0000" = command_not_accepted.

**RUN**
Go to POC state STARTUP when called in POC state READY. When called in any other state, **CMD[3:0]** will be reset to "0000" = command_not_accepted.

**ALL_SLOTS**
Leave SINGLE slot mode after successful startup / integration at the next end of cycle when called in POC states NORMAL_ACTIVE or NORMAL_PASSIVE. When called in any other state, **CMD[3:0]** will be reset to "0000" = command_not_accepted.

**HALT**
Set halt request **CCSV.HRQ** and go to POC state HALT at the next end of cycle when called in POC states NORMAL_ACTIVE or NORMAL_PASSIVE. When called in any other state, **CMD[3:0]** will be reset to "0000" = command_not_accepted.

**FREEZE**
Set the freeze status indicator **CCSV.FSI** and go to POC state HALT immediately. Can be called from any state.

**SEND_MTS**
Send single MTS symbol during the next following symbol window on the channel configured by **MTSA**, **MTSB**, when called in POC state NORMAL_ACTIVE after CC entered ALL slot mode
(**CCSV.SLM[1:0]** = "11"). When called in any other state, or when called while a previously requested MTS has not yet been transmitted, **CMD[3:0]** will be reset to "0000" = command_not_accepted.

**ALLOW_COLDSTART**
The command resets **CCSV.CSI** to enable the node to become leading coldstarter. When called in states DEFAULT_CONFIG, CONFIG, or HALT, **CMD[3:0]** will be reset to "0000" = command_not_accepted. To become leading coldstarter it is also required that both **TXST** and **TXSY** are set.

**RESET_STATUS_INDICATORS**
Reset status flags **CCSV.FSI**, **CCSV.HRQ**, **CCSV.CSNI**, **CCSV.CSAI**, **CCSV.WSV[2:0]**, and register CCEV. May be called in any state.

**MONITOR_MODE**
Enter MONITOR_MODE when called in POC state CONFIG. In this mode the CC is able to receive FlexRay frames and CAS / MTS symbols. It is also able to detect coding errors. The temporal integrity of received frames is not checked. This mode can be used for debugging purposes, e.g. in case that the startup of a FlexRay network fails. When called in any other state, **CMD[3:0]** will be reset to "0000" = command_not_accepted. For details see 19.5.4.MONITOR_MODE.

**CLEAR_RAMS**
Sets **MHDS.CRAM** when called in DEFAULT_CONFIG or CONFIG state. When called in any other state, **CMD[3:0]** will be reset to "0000" = command_not_accepted. **MHDS.CRAM** is also set when the CC leaves hard reset. By setting **MHDS.CRAM** all internal RAM blocks are initialized to zero. During the initialization of the RAMs, **PBSY** will show POC busy. Access to the configuration and status registers is possible during execution of CHI command CLEAR_RAMS.
The initialization of the E-Ray internal RAM blocks requires 2048 **eray_bclk** cycles. There should be no Host access to IBF or OBF during initialization of the internal RAM blocks after hard reset or after assertion of CHI command CLEAR_RAMS. Before asserting CHI command CLEAR_RAMS the Host should make sure that no transfer between Message RAM and IBF / OBF or the Transient Buffer RAMs is ongoing. This command also resets the Message Buffer Status registers MHDS, LDTS, FSR, MHDF, TXRQ1/2/3/4, NDAT1/2/3/4, and MBSC1/2/3/4.

Note: All accepted commands with exception of CLEAR_RAMS and SEND_MTS will cause a change of register CCSV after at most 8 cycles of the slower of the two clocks **eray_bclk** and **eray_sclk**, counted from the falling edge of the CHI input signal **eray_select**, assumed that POC was not busy when the command was applied and that no POC state change was forced by bus activity in that time frame. Reading register CCSV will show data that is delayed by synchronization from **eray_sclk** to **eray_bclk** domain and by the Host-specific CPU interface.

**PBSY** POC Busy
Signals that the POC is busy and cannot accept a command from the Host. **CMD[3:0]** is locked against write accesses. Set to ’1’ after hard reset during initialization of internal RAM blocks.
1 =POC is busy, **CMD[3:0]** locked
0 =POC not busy, **CMD[3:0]** writeable

**TXST** Transmit Startup Frame in Key Slot (pKeySlotUsedForStartup)
Defines whether the key slot is used to transmit startup frames. The bit can be modified in DEFAULT_CONFIG or CONFIG state only.
1 =Key slot used to transmit startup frame, node is leading or following coldstarter
0 =No startup frame transmission in key slot, node is non-coldstarter

**TXSY** Transmit Sync Frame in Key Slot (pKeySlotUsedForSync)
Defines whether the key slot is used to transmit sync frames. The bit can be modified in DEFAULT_CONFIG or CONFIG state only.
1 =Key slot used to transmit sync frame, node is sync node
0 =No sync frame transmission in key slot, node is neither sync nor coldstart node
The protocol requires that both bits **TXST** and **TXSY** are set for coldstart nodes.

**CSA[4:0]** Cold Start Attempts (gColdStartAttempts)
Configures the maximum number of attempts that a cold starting node is permitted to try to start up the network without receiving any valid response from another node. It can be modified in DEFAULT_CONFIG or CONFIG state only. Must be identical in all nodes of a cluster. Valid values are 2 to 31.

**PTA[4:0]** Passive to Active (pAllowPassiveToActive)
Defines the number of consecutive even / odd cycle pairs that must have valid clock correction terms before the CC is allowed to transit from NORMAL_PASSIVE to NORMAL_ACTIVE state. If set to "00000" the CC is **not** allowed to transit from NORMAL_PASSIVE to NORMAL_ACTIVE state. It can be modified in DEFAULT_CONFIG or CONFIG state only. Valid values are 0 to 31 even / odd cycle pairs.

**WUCS** Wakeup Channel Select (pWakeupChannel)
With this bit the Host selects the channel on which the CC sends the Wakeup pattern. The CC ignores any attempt to change the status of this bit when not in DEFAULT_CONFIG or CONFIG state.
1 =Send wakeup pattern on channel B
0 =Send wakeup pattern on channel A

**TSM** Transmission Slot Mode (pSingleSlotEnabled)
Selects the initial transmission slot mode. In SINGLE slot mode the CC may only transmit in the preconfigured key slot. The key slot ID is configured in the header section of message buffer 0 respectively message buffers 0 and 1 depending on bit **MRC.SPLM**. In case **TSM** = '1', message buffer 0 respectively message buffers 0,1 can be (re)configured in DEFAULT_CONFIG or CONFIG state only. In ALL slot mode the CC may transmit in all slots. **TSM** is a configuration bit which can only be set / reset by the Host. The bit can be written in DEFAULT_CONFIG or CONFIG state only. The CC changes to ALL slot mode when the Host successfully applied the ALL_SLOTS command by writing **CMD[3:0]** = "0101" in POC states NORMAL_ACTIVE or NORMAL_PASSIVE. The actual slot mode is monitored by **CCSV.SLM[1:0]**.
1 =SINGLE Slot Mode (default after hard reset)
0 =ALL Slot Mode

**HCSE** Halt due to Clock Sync Error (pAllowHaltDueToClock)
Controls the transition to HALT state due to a clock synchronization error. The bit can be modified in DEFAULT_CONFIG or CONFIG state only.
1 =CC will enter HALT state
0 =CC will enter / remain in NORMAL_PASSIVE

**MTSA** Select Channel A for MTS Transmission
The bit selects channel A for MTS symbol transmission. The flag is reset by default and may be modified only in DEFAULT_CONFIG or CONFIG state.
1 =Channel A selected for MTS transmission
0 =Channel A disabled for MTS transmission

**MTSB** Select Channel B for MTS Transmission
The bit selects channel B for MTS symbol transmission. The flag is reset by default and may be modified only in DEFAULT_CONFIG or CONFIG state.
1 =Channel B selected for MTS transmission
0 =Channel B disabled for MTS transmission
If both bits **MTSA** and **MTSB** are set to '1' an MTS symbol will be transmitted on both channels when requested by writing **CMD[3:0]** = "1000".

**CCHA** Connected to Channel A (pChannels)
Configures whether the node is connected to channel A.
1 =Node connected to channel A (default after hard reset)
0 =Not connected to channel A

**CCHB** Connected to Channel B (pChannels)
Configures whether the node is connected to channel B.
1 =Node connected to channel B (default after hard reset)
0 =Not connected to channel B

### 18.5.2. SUC Configuration Register 2 (SUCC2)

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **SUCC2** | R | 0 | 0 | 0 | 0 | LTN3* | LTN2* | LTN1* | LTN0* | 0 | 0 | 0 | LT20* | LT19* | LT18* | LT17* | LT16* |
| 0x0084 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | LT15* | LT14* | LT13* | LT12* | LT11* | LT10* | LT9* | LT8* | LT7* | LT6* | LT5* | LT4* | LT3* | LT2* | LT1* | LT0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

**LT[20:0]** Listen Timeout (pdListenTimeout)

Configures wakeup / startup listen timeout in µT. The range for pdListenTimeout is 1284 to 1283846 µT.

**LTN[3:0]** Listen Timeout Noise (gListenNoise - 1)

Configures the upper limit for startup and wakeup listen timeout in the presence of noise expressed as a multiple of pdListenTimeout. The range for gListenNoise is 2 to 16. **LTN[3:0]** must be configured identical in all nodes of a cluster.

Note: The wakeup / startup noise timeout is calculated as follows:
pdListenTimeout • gListenNoise **= LT[20:0]** • (**LTN[3:0]** + 1)

### 18.5.3. SUC Configuration Register 3 (SUCC3)

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **SUCC3** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0088 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | WCF3* | WCF2* | WCF1* | WCF0* | WCP3* | WCP2* | WCP1* | WCP0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

**WCP[3:0]** Maximum Without Clock Correction Passive (gMaxWithoutClockCorrectionPassive)

Defines the number of consecutive even / odd cycle pairs with missing clock correction terms that will cause a transition from NORMAL_ACTIVE to NORMAL_PASSIVE state. Must be identical in all nodes of a cluster. Valid values are 1 to 15 cycle pairs.

**WCF[3:0]** Maximum Without Clock Correction Fatal (gMaxWithoutClockCorrectionFatal)

Defines the number of consecutive even / odd cycle pairs with missing clock correction terms that will cause a transition from NORMAL_ACTIVE or NORMAL_PASSIVE to HALT state. Must be identical in all nodes of a cluster. Valid values are 1 to 15 cycle pairs.

### *18.5.4. NEM Configuration Register (NEMC)*

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **NEMC** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x008C | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | NML3* | NML2* | NML1* | NML0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**NML[3:0]** Network Management Vector Length (gNetworkManagementVectorLength)

These bits configure the length of the NM vector. The configured length must be identical in all nodes of a cluster. Valid values are 0 to 12 bytes.

### *18.5.5. PRT Configuration Register 1 (PRTC1)*

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **PRTC1** | R | RWP5* | RWP4* | RWP3* | RWP2* | RWP1* | RWP0* | 0 | RXW8* | RXW7* | RXW6* | RXW5* | RXW4* | RXW3* | RXW2* | RXW1* | RXW0* |
| 0x0090 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | BRP1* | BRP0* | SPP1* | SPP0* | 0 | CASM6 | CASM5* | CASM4* | CASM3* | CASM2* | CASM1* | CASM0* | TSST3* | TSST2* | TSST1* | TSST0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |

**TSST[3:0]** Transmission Start Sequence Transmitter (gdTSSTransmitter)

Configures the duration of the Transmission Start Sequence (TSS) in terms of bit times (1 bit time = 4 µT = 100ns @ 10Mbps). Must be identical in all nodes of a cluster. Valid values are 3 to 15 bit times.

**CASM[6:0]** Collision Avoidance Symbol Max (gdCASRxLowMax)

Configures the upper limit of the acceptance window for a collision avoidance symbol (CAS). **CASM6** is fixed to '1'. Valid values are 67 to 99 bit times.

**SPP[1:0]** Strobe Point Position

Defines the sample count value for strobing. The strobed bit value is set to the voted value when the sample count is incremented to the value configured by **SPP[1:0]**.

00, 11=Sample 5 (default)

01 =Sample 4

10 =Sample 6

The current revision 2.1 of the FlexRay protocol requires that **SPP[1:0]** = "00". The alternate strobe point positions could be used to compensate for asymmetries in the physical layer.

**BRP[1:0]** Baud Rate Prescaler (gdSampleClockPeriod, pSamplesPerMicrotick)

The Baud Rate Prescaler configures the baud rate on the FlexRay bus. The baud rates listed below are valid with a sample clock **eray_sclk** = 80 MHz. One bit time always consists of 8 samples independent of the configured baud rate.

00 =10 MBit/s (default)

gdSampleClockPeriod = 12.5 ns = 1 • **eray_sclk**

pSamplesPerMicrotick = 2 (1 µT = 25 ns)

01 =5 MBit/s

gdSampleClockPeriod = 25 ns = 2 • **eray_sclk**

pSamplesPerMicrotick = 1 (1 µT = 25 ns)

10, 11 =2.5 MBit/s

gdSampleClockPeriod = 50 ns = 4 • **eray_sclk**

pSamplesPerMicrotick = 1 (1 µT = 50 ns)

**RXW[8:0]** Wakeup Symbol Receive Window Length (gdWakeupSymbolRxWindow)

Configures the number of bit times used by the node to test the duration of the received wakeup pattern. Must be identical in all nodes of a cluster. Valid values are 76 to 301 bit times.

**RWP[5:0]** Repetitions of Tx Wakeup Pattern (pWakeupPattern)

Configures the number of repetitions (sequences) of the Tx wakeup symbol. Valid values are 2 to 63.

### *18.5.6. PRT Configuration Register 2 (PRTC2)*

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **PRTC2** | R | 0 | 0 | TXL5* | TXL4* | TXL3* | TXL2* | TXL1* | TXL0* | TXI7* | TXI6* | TXI5* | TXI4* | TXI3* | TXI2* | TXI1* | TXI0* |
| 0x0094 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | RXL5* | RXL4* | RXL3* | RXL2* | RXL1* | RXL0* | 0 | 0 | RXI5* | RXI4* | RXI3* | RXI2* | RXI1* | RXI0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |

**RXI[5:0]** Wakeup Symbol Receive Idle (gdWakeupSymbolRxIdle)

Configures the number of bit times used by the node to test the duration of the idle phase of the received wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 14 to 59 bit times.

**RXL[5:0]** Wakeup Symbol Receive Low (gdWakeupSymbolRxLow)

Configures the number of bit times used by the node to test the duration of the low phase of the received wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 10 to 55 bit times.

**TXI[7:0]** Wakeup Symbol Transmit Idle (gdWakeupSymbolTxIdle)

Configures the number of bit times used by the node to transmit the idle phase of the wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 45 to 180 bit times.

**TXL[5:0]** Wakeup Symbol Transmit Low (gdWakeupSymbolTxLow)

Configures the number of bit times used by the node to transmit the low phase of the wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 15 to 60 bit times.

### 18.5.7. MHD Configuration Register (MHDC)

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **MHDC** | R | 0 | 0 | 0 | SLT12* | SLT11* | SLT10* | SLT9* | SLT8* | SLT7* | SLT6* | SLT5* | SLT4* | SLT3* | SLT2* | SLT1* | SLT0* |
| 0x0098 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SFDL6* | SFDL5* | SFDL4* | SFDL3* | SFDL2* | SFDL1* | SFDL0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**SFDL[6:0]** Static Frame Data Length (gPayloadLengthStatic)

Configures the cluster-wide payload length for all frames sent in the static segment in double bytes. The payload length must be identical in all nodes of a cluster. Valid values are 0 to 127.

**SLT[12:0]** Start of Latest Transmit (pLatestTx)

Configures the maximum minislot value allowed before inhibiting frame transmission in the dynamic segment of the cycle. There is no transmission in dynamic segment if **SLT[12:0]** is set to zero. Valid values are 0 to 7981 minislots.

### 18.5.8. GTU Configuration Register 1 (GTUC1)

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **GTUC1** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | UT19* | UT18* | UT17* | UT16* |
| 0x00A0 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | UT15* | UT14* | UT13* | UT12* | UT11* | UT10* | UT9* | UT8* | UT7* | UT6* | UT5* | UT4* | UT3* | UT2* | UT1* | UT0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**UT[19:0]** Microtick per Cycle (pMicroPerCycle)

Configures the duration of the communication cycle in microticks. Valid values are 640 to 640000 µT.

### *18.5.9. GTU Configuration Register 2 (GTUC2)*

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **GTUC2** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SNM3* | SNM2* | SNM1* | SNM0* |
| 0x00A4 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | MPC13* | MPC12* | MPC11* | MPC10* | MPC9* | MPC8* | MPC7* | MPC6* | MPC5* | MPC4* | MPC3* | MPC2* | MPC1* | MPC0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |

**MPC[13:0]** Macrotick Per Cycle (gMacroPerCycle)

Configures the duration of one communication cycle in macroticks. The cycle length must be identical in all nodes of a cluster. Valid values are 10 to 16000 MT.

**SNM[3:0]** Sync Node Max (gSyncNodeMax)

Maximum number of frames within a cluster with sync frame indicator bit **SYN** set to '1'. Must be identical in all nodes of a cluster. Valid values are 2 to 15.

### *18.5.10. GTU Configuration Register 3 (GTUC3)*

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **GTUC3** | R | 0 | MIOB6* | MIOB5* | MIOB4* | MIOB3* | MIOB2* | MIOB1* | MIOB0* | 0 | MIOA6* | MIOA5* | MIOA4* | MIOA3* | MIOA2* | MIOA1* | MIOA0* |
| 0x00A8 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | UIOB7* | UIOB6* | UIOB5* | UIOB4* | UIOB3* | UIOB2* | UIOB1* | UIOB0* | UIOA7* | UIOA6* | UIOA5* | UIOA4* | UIOA3* | UIOA2* | UIOA1* | UIOA0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**UIOA[7:0]** Microtick Initial Offset Channel A (pMicroInitialOffset[A])

Configures the number of microticks between the actual time reference point on channel A and the subsequent macrotick boundary of the secondary time reference point. The parameter depends on pDelayCompensation[A] and therefore has to be set for each channel independently. Valid values are 0 to 240 µT.

**UIOB[7:0]** Microtick Initial Offset Channel B (pMicroInitialOffset[B])

Configures the number of microticks between the actual time reference point on channel B and the subsequent macrotick boundary of the secondary time reference point. The parameter depends on pDelayCompensation[B] and therefore has to be set for each channel independently. Valid values are 0 to 240 µT.

**MIOA[6:0]** Macrotick Initial Offset Channel A (pMacroInitialOffset[A])

Configures the number of macroticks between the static slot boundary and the subsequent macrotick boundary of the secondary time reference point based on the nominal macrotick duration. Must be identical in all nodes of a cluster. Valid values are 2 to 72 MT.

**MIOB[6:0]** Macrotick Initial Offset Channel B (pMacroInitialOffset[B])

Configures the number of macroticks between the static slot boundary and the subsequent macrotick boundary of the secondary time reference point based on the nominal macrotick duration. Must be identical in all nodes of a cluster. Valid values are 2 to 72 MT.

### *18.5.11. GTU Configuration Register 4 (GTUC4)*

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only. For details about configuration of **NIT[13:0]** and **OCS[13:0]** see Section 19.1.5.Configuration of NIT Start and Offset Correction Start.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **GTUC4** | R | 0 | 0 | OCS13* | OCS12* | OCS11* | OCS10* | OCS9* | OCS8* | OCS7* | OCS6* | OCS5* | OCS4* | OCS3* | OCS2* | OCS1* | OCS0* |
| 0x00AC | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | NIT13* | NIT12* | NIT11* | NIT10* | NIT9* | NIT8* | NIT7* | NIT6* | NIT5* | NIT4* | NIT3* | NIT2* | NIT1* | NIT0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

**NIT[13:0]** Network Idle Time Start (gMacroPerCycle - gdNIT - 1)

Configures the starting point of the Network Idle Time NIT at the end of the communication cycle expressed in terms of macroticks from the beginning of the cycle. The start of NIT is recognized if
Macrotick = gMacroPerCycle - gdNIT -1 and the increment pulse of Macrotick is set. Must be identical in all nodes of a cluster.
Valid values are 7 to 15997 MT.

**OCS[13:0]** Offset Correction Start (gOffsetCorrectionStart - 1)

Determines the start of the offset correction within the NIT phase, calculated from start of cycle. Must be identical in all nodes of a cluster. For cluster consisting of E-Ray implementations only, it is sufficient to program **OCS** = **NIT** + 1. Valid values are 8 to 15998 MT.

### *18.5.12. GTU Configuration Register 5 (GTUC5)*

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **GTUC5** | R | DEC7* | DEC6* | DEC5* | DEC4* | DEC3* | DEC2* | DEC1* | DEC0* | 0 | 0 | 0 | CDD4* | CDD3* | CDD2* | CDD1* | CDD0* |
| 0x00B0 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | DCB7* | DCB6* | DCB5* | DCB4* | DCB3* | DCB2* | DCB1* | DCB0* | DCA7* | DCA6* | DCA5* | DCA4* | DCA3* | DCA2* | DCA1* | DCA0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**DCA[7:0]** Delay Compensation Channel A (pDelayCompensation[A])

Used to compensate for reception delays on the indicated channel. This covers assumed propagation delay up to cPropagationDelayMax for microticks in the range of 0.0125 to 0.05µs. In practice, the minimum of the propagation delays of all sync nodes should be applied.
Valid values are 0 to 200 µT.

**DCB[7:0]** Delay Compensation Channel B (pDelayCompensation[B])

Used to compensate for reception delays on the indicated channel. This covers assumed propagation delay up to cPropagationDelayMax for microticks in the range of 0.0125 to 0.05µs. In practice, the minimum of the propagation delays of all sync nodes should be applied.
Valid values are 0 to 200 µT.

**CDD[4:0]** Cluster Drift Damping (pClusterDriftDamping)

Configures the cluster drift damping value used in clock synchronization to minimize accumulation of rounding errors. Valid values are 0 to 20 µT.

**DEC[7:0]** Decoding Correction (pDecodingCorrection)

Configures the decoding correction value used to determine the primary time reference point. Valid values are 14 to 143 µT.

### *18.5.13. GTU Configuration Register 6 (GTUC6)*

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **GTUC6** | R | 0 | 0 | 0 | 0 | 0 | MOD 10* | MOD9* | MOD8* | MOD7* | MOD6* | MOD5* | MOD4* | MOD3* | MOD2* | MOD1* | MOD0* |
| 0x00B4 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | ASR10* | ASR9* | ASR8* | ASR7* | ASR6* | ASR5* | ASR4* | ASR3* | ASR2* | ASR1* | ASR0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ASR[10:0]** Accepted Startup Range (pdAcceptedStartupRange)

Number of microticks constituting the expanded range of measured deviation for startup frames during integration. Valid values are 0 to 1875 µT.

**MOD[10:0]** Maximum Oscillator Drift (pdMaxDrift)

Maximum drift offset between two nodes that operate with unsynchronized clocks over one communication cycle in µT. Valid values are 2 to 1923 µT.

### *18.5.14. GTU Configuration Register 7 (GTUC7)*

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **GTUC7** | R | 0 | 0 | 0 | 0 | 0 | 0 | NSS9* | NSS8* | NSS7* | NSS6* | NSS5* | NSS4* | NSS3* | NSS2* | NSS1* | NSS0* |
| 0x00B8 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | SSL9* | SSL8* | SSL7* | SSL6* | SSL5* | SSL4* | SSL3* | SSL2* | SSL1* | SSL0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

**SSL[9:0]** Static Slot Length (gdStaticSlot)

Configures the duration of a static slot in macroticks. The static slot length must be identical in all nodes of a cluster. Valid values are 4 to 659 MT.

**NSS[9:0]** Number of Static Slots (gNumberOfStaticSlots)

Configures the number of static slots in a cycle. At least 2 coldstart nodes must be configured to startup a FlexRay network. The number of static slots must be identical in all nodes of a cluster. Valid values are 2 to 1023.

### *18.5.15. GTU Configuration Register 8 (GTUC8)*

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **GTUC8** | R | 0 | 0 | 0 | NMS 12* | NMS 11* | NMS 10* | NMS9* | NMS8* | NMS7* | NMS6* | NMS5* | NMS4* | NMS3* | NMS2* | NMS1* | NMS0* |
| 0x00BC | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | MSL5* | MSL4* | MSL3* | MSL2* | MSL1* | MSL0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

**MSL[5:0]** Minislot Length (gdMinislot)

Configures the duration of a minislot in macroticks. The minislot length must be identical in all nodes of a cluster. Valid values are 2 to 63 MT.

**NMS[12:0]** Number of Minislots (gNumberOfMinislots)

Configures the number of minislots within the dynamic segment of a cycle. The number of minislots must be identical in all nodes of a cluster. Valid values are 0 to 7986.

### *18.5.16. GTU Configuration Register 9 (GTUC9)*

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **GTUC9** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DSI1* | DSI0* |
| 0x00C0 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | MAPO4* | MAPO3* | MAPO2* | MAPO1* | MAPO0* | 0 | 0 | APO5* | APO4* | APO3* | APO2* | APO1* | APO0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**APO[5:0]** Action Point Offset (gdActionPointOffset)

Configures the action point offset in macroticks within static slots and symbol window. Must be identical in all nodes of a cluster. Valid values are 1 to 63 MT.

**MAPO[4:0]** Minislot Action Point Offset (gdMinislotActionPointOffset)

Configures the action point offset in macroticks within the minislots of the dynamic segment. Must be identical in all nodes of a cluster. Valid values are 1 to 31 MT.

**DSI[1:0]** Dynamic Slot Idle Phase (gdDynamicSlotIdlePhase)

The duration of the dynamic slot idle phase has to be greater or equal than the idle detection time. Must be identical in all nodes of a cluster. Valid values are 0 to 2 Minislot.

### 18.5.17. GTU Configuration Register 10 (GTUC10)

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| GTUC10 | R | 0 | 0 | 0 | 0 | 0 | MRC 10* | MRC9* | MRC8* | MRC7* | MRC6* | MRC5* | MRC4* | MRC3* | MRC2* | MRC1* | MRC0* |
| 0x00C4 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | MOC 13* | MOC 12* | MOC 11* | MOC 10* | MOC9* | MOC8* | MOC7* | MOC6* | MOC5* | MOC4* | MOC3* | MOC2* | MOC1* | MOC0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |

**MOC[13:0]** Maximum Offset Correction (pOffsetCorrectionOut)

Holds the maximum permitted offset correction value to be applied by the internal clock synchronization algorithm (absolute value). The CC checks only the internal offset correction value against the maximum offset correction value. Valid values are 5 to 15266 µT.

**MRC[10:0]** Maximum Rate Correction (pRateCorrectionOut)

Holds the maximum permitted rate correction value to be applied by the internal clock synchronization algorithm. The CC checks only the internal rate correction value against the maximum rate correction value (absolute value). Valid values are 2 to 1923 µT.

### 18.5.18. GTU Configuration Register 11 (GTUC11)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| GTUC11 | R | 0 | 0 | 0 | 0 | 0 | ERC2* | ERC1* | ERC0* | 0 | 0 | 0 | 0 | 0 | EOC2* | EOC1* | EOC0* |
| 0x00C8 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | ERCC1 | ERCC0 | 0 | 0 | 0 | 0 | 0 | 0 | EOCC1 | EOCC0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**EOCC[1:0]** External Offset Correction Control (vExternOffsetControl)

By writing to **EOCC[1:0]** the external offset correction is enabled as specified below. Should be modified only outside NIT.

00, 01 =No external offset correction

10 =External offset correction value subtracted from calculated offset correction value

11 =External offset correction value added to calculated offset correction value

**ERCC[1:0]** External Rate Correction Control (vExternRateControl)

By writing to **ERCC[1:0]** the external rate correction is enabled as specified below. Should be modified only outside NIT.

00, 01 =No external rate correction

10 =External rate correction value subtracted from calculated rate correction value

11 =External rate correction value added to calculated rate correction value

**EOC[2:0]** External Offset Correction (pExternOffsetCorrection)

Holds the external offset correction value in microticks to be applied by the internal clock synchronization algorithm. The value is subtracted / added from / to the calculated offset correction value. The value is applied during NIT. May be modified in DEFAULT_CONFIG or CONFIG state only. Valid values are 0 to 7 µT.

**ERC[2:0]** External Rate Correction (pExternRateCorrection)

Holds the external rate correction value in microticks to be applied by the internal clock synchronization algorithm. The value is subtracted / added from / to the calculated rate correction value. The value is applied during NIT. May be modified in DEFAULT_CONFIG or CONFIG state only. Valid values are 0 to 7 µT.

## 18.6. CC Status Registers

During 8/16-bit accesses to status variables coded with more than 8/16-bit, the variable might be updated by the CC between two accesses (non-atomic read accesses). All internal counters and the CC status flags are reset when the CC transits from CONFIG to READY state.

### *18.6.1. CC Status Vector (CCSV)*

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **CCSV** | R | 0 | 0 | PSL5 | PSL4 | PSL3 | PSL2 | PSL1 | PSL0 | RCA4 | RCA3 | RCA2 | RCA1 | RCA0 | WSV2 | WSV1 | WSV0 |
| 0x0100 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | CSI | CSAI | CSNI | 0 | 0 | SLM1 | SLM0 | HRQ | FSI | POCS5 | POCS4 | POCS3 | POCS2 | POCS1 | POCS0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**POCS[5:0]** Protocol Operation Control Status

Indicates the actual state of operation of the CC Protocol Operation Control
00 0000 =DEFAULT_CONFIG state
00 0001 =READY state
00 0010 =NORMAL_ACTIVE state
00 0011 =NORMAL_PASSIVE state
00 0100 =HALT state
00 0101 =MONITOR_MODE state
00 0110…00 1110 = reserved
00 1111 =CONFIG state
Indicates the actual state of operation of the POC in the wakeup path
01 0000 =WAKEUP_STANDBY state
01 0001 =WAKEUP_LISTEN state
01 0010 =WAKEUP_SEND state
01 0011 =WAKEUP_DETECT state
01 0100…01 1111 = reserved
Indicates the actual state of operation of the POC in the startup path
10 0000 =STARTUP_PREPARE state
10 0001 =COLDSTART_LISTEN state
10 0010 =COLDSTART_COLLISION_RESOLUTION state
10 0011 =COLDSTART_CONSISTENCY_CHECK state
10 0100 =COLDSTART_GAP state
10 0101 =COLDSTART_JOIN State
10 0110 =INTEGRATION_COLDSTART_CHECK state
10 0111 =INTEGRATION_LISTEN state
10 1000 =INTEGRATION_CONSISTENCY_CHECK state
10 1001 =INITIALIZE_SCHEDULE state
10 1010 =ABORT_STARTUP state
10 1011…11 1111 = reserved

**FSI** Freeze Status Indicator (vPOC!Freeze)
Indicates that the POC has entered the HALT state due to CHI command FREEZE or due to an error condition requiring an immediate POC halt. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state.

**HRQ** Halt Request (vPOC!CHIHaltRequest)
Indicates that a request from the Host has been received to halt the POC at the end of the communication cycle. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or when entering READY state.

**SLM[1:0]** Slot Mode (vPOC!SlotMode)
Indicates the actual slot mode of the POC. Default is SINGLE. Changes to ALL, depending on **SUCC1.TSM**. In NORMAL_ACTIVE or NORMAL_PASSIVE state the CHI command ALL_SLOTS will change the slot mode from SINGLE over ALL_PENDING to ALL. When not in NORMAL_ACTIVE or NORMAL_PASSIVE state then reset by CHI command RESET_STATUS_INDICATORS to the value defined by **SUCC1.TSM**.
00 =SINGLE
01 =reserved
10 =ALL_PENDING
11 =ALL

**CSNI** Coldstart Noise Indicator (vPOC!ColdstartNoise)
Indicates that the cold start procedure occurred under noisy conditions. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or from READY to STARTUP state.

**CSAI** Coldstart Abort Indicator
Coldstart aborted. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or from READY to STARTUP state.

**CSI** Cold Start Inhibit (vColdStartInhibit)
Indicates that the node is disabled from cold starting. The flag is set whenever the POC enters READY state. The flag has to be reset under control of the Host by CHI command ALLOW_COLDSTART (**SUCC1.CMD[3:0]** = "1001").
1 =Cold starting of node disabled
0 =Cold starting of node enabled

**WSV[2:0]** Wakeup Status (vPOC!WakeupStatus)
Indicates the status of the current wakeup attempt. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or from READY to STARTUP state.
000 =UNDEFINED. No wakeup attempt since CONFIG state was left.
001 =RECEIVED_HEADER. Set when the CC finishes wakeup due to the reception of a frame header without coding violation on either channel in WAKEUP_LISTEN state.
010 =RECEIVED_WUP. Set when the CC finishes wakeup due to the reception of a valid wakeup pattern on the configured wakeup channel in WAKEUP_LISTEN state.
011 =COLLISION_HEADER. Set when the CC stops wakeup due to a detected collision during wakeup pattern transmission by receiving a valid header on either channel.
100 =COLLISION_WUP. Set when the CC stops wakeup due to a detected collision during wakeup pattern transmission by receiving a valid wakeup pattern on the configured wakeup channel.
101 =COLLISION_UNKNOWN. Set when the CC stops wakeup by leaving WAKEUP_DETECT state after expiration of the wakeup timer without receiving a valid wakeup pattern or a valid frame header.
110 =TRANSMITTED. Set when the CC has successfully completed the transmission of the wakeup pattern.
111 =reserved

**RCA[4:0]** Remaining Coldstart Attempts (vRemainingColdstartAttempts)
Indicates the number of remaining coldstart attempts. The maximum number of coldstart attempts is configured by **SUCC1.CSA[4:0]**.

**PSL[5:0]** POC Status Log
Status of POCS[5:0] immediately before entering HALT state. Set when entering HALT state. Set to HALT when FREEZE command is applied during HALT state. Reset to "00 0000" when leaving HALT state.
CHI command RESET_STATUS_INDICATORS (**SUCC1.CMD[3:0]** = "1010") resets flags **FSI**, **HRQ**, **CSNI**, **CSAI**, the slot mode **SLM[1:0]**, and the wakeup status **WSV[2:0]**.

### 18.6.2. CC Error Vector (CCEV)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **CCEV** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0104 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | PTAC4 | PTAC3 | PTAC2 | PTAC1 | PTAC0 | ERRM1 | ERRM0 | 0 | 0 | CCFC3 | CCFC2 | CCFC1 | CCFC0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or when entering READY state.

**CCFC[3:0]** Clock Correction Failed Counter (vClockCorrectionFailed)

The Clock Correction Failed Counter is incremented by one at the end of any odd communication cycle where either the missing offset correction error or missing rate correction error are active. The Clock Correction Failed Counter is reset to '0' at the end of an odd communication cycle if neither the offset correction failed nor the rate correction failed errors are active. The Clock Correction Failed Counter stops at 15.

**ERRM[1:0]** Error Mode (vPOC!ErrorMode)

Indicates the actual error mode of the POC.
00 =ACTIVE (green)
01 =PASSIVE (yellow)
10 =COMM_HALT (red)
11 =reserved

**PTAC[4:0]** Passive to Active Count (vAllowPassiveToActive)

Indicates the number of consecutive even / odd cycle pairs that have passed with valid rate and offset correction terms, while the node is waiting to transit from NORMAL_PASSIVE state to NORMAL_ACTIVE state. The transition takes place when **PTAC[4:0]** equals **SUCC1.PTA[4:0] -1**.

### 18.6.3. Slot Counter Value (SCV)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **SCV** | R | 0 | 0 | 0 | 0 | 0 | SCCB10 | SCCB9 | SCCB8 | SCCB7 | SCCB6 | SCCB5 | SCCB4 | SCCB3 | SCCB2 | SCCB1 | SCCB0 |
| 0x0110 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | SCCA10 | SCCA9 | SCCA8 | SCCA7 | SCCA6 | SCCA5 | SCCA4 | SCCA3 | SCCA2 | SCCA1 | SCCA0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**SCCA[10:0]** Slot Counter Channel A (vSlotCounter[A])

Current slot counter value on channel A. The value is incremented by the CC and reset at the start of a communication cycle. Valid values are 0 to 2047.

**SCCB[10:0]** Slot Counter Channel B (vSlotCounter[B])

Current slot counter value on channel B. The value is incremented by the CC and reset at the start of a communication cycle. Valid values are 0 to 2047.

### 18.6.4. Macrotick and Cycle Counter Value (MTCCV)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **MTCCV** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | CCV5 | CCV4 | CCV3 | CCV2 | CCV1 | CCV0 |
| 0x0114 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | MTV13 | MTV12 | MTV11 | MTV10 | MTV9 | MTV8 | MTV7 | MTV6 | MTV5 | MTV4 | MTV3 | MTV2 | MTV1 | MTV0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**MTV[13:0]** Macrotick Value (vMacrotick)

Current macrotick value. The value is incremented by the CC and reset at the start of a communication cycle. Valid values are 0 to 16000.

**CCV[5:0]** Cycle Counter Value (vCycleCounter)

Current cycle counter value. The value is incremented by the CC at the start of a communication cycle. Valid values are 0 to 63.

### 18.6.5. Rate Correction Value (RCV)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **RCV** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0118 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | RCV11 | RCV10 | RCV9 | RCV8 | RCV7 | RCV6 | RCV5 | RCV4 | RCV3 | RCV2 | RCV1 | RCV0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**RCV[11:0]** Rate Correction Value (vRateCorrection)

Rate correction value (two's complement). Calculated internal rate correction value **before** limitation. If the RCV value exceeds the limits defined by **GTUC10.MRC[10:0]**, flag **SFS.RCLR** is set to '1'.

### 18.6.6. Offset Correction Value (OCV)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **OCV** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | OCV18 | OCV17 | OCV16 |
| 0x011C | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | OCV15 | OCV14 | OCV13 | OCV12 | OCV11 | OCV10 | OCV9 | OCV8 | OCV7 | OCV6 | OCV5 | OCV4 | OCV3 | OCV2 | OCV1 | OCV0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**OCV[18:0]** Offset Correction Value (vOffsetCorrection)

Offset correction value (two's complement). Calculated internal offset correction value **before** limitation. If the OCV value exceeds the limits defined by **GTUC10.MOC[13:0]**, flag **SFS.OCLR** is set to '1'.

The external rate / offset correction value is added to the limited rate / offset correction value.

### 18.6.7. Sync Frame Status (SFS)

The maximum number of valid sync frames in a communication cycle is 15.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **SFS** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RCLR | MRCS | OCLR | MOCS |
| 0x0120 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | VSBO3 | VSBO2 | VSBO1 | VSBO0 | VSBE3 | VSBE2 | VSBE1 | VSBE0 | VSAO3 | VSAO2 | VSAO1 | VSAO0 | VSAE3 | VSAE2 | VSAE1 | VSAE0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**VSAE[3:0]** Valid Sync Frames Channel A, **even** communication cycle (vSyncFramesEvenA)

Holds the number of valid sync frames received on channel A in the even communication cycle. If transmission of sync frames is enabled by **SUCC1.TXSY** the value is incremented by one. The value is updated during the NIT of each even communication cycle.

**VSAO[3:0]** Valid Sync Frames Channel A, **odd** communication cycle (vSyncFramesOddA)

Holds the number of valid sync frames received on channel A in the odd communication cycle. If transmission of sync frames is enabled by **SUCC1.TXSY** the value is incremented by one. The value is updated during the NIT of each odd communication cycle.

**VSBE[3:0]** Valid Sync Frames Channel B, **even** communication cycle (vSyncFramesEvenB)

Holds the number of valid sync frames received on channel B in the even communication cycle. If transmission of sync frames is enabled by **SUCC1.TXSY** the value is incremented by one. The value is updated during the NIT of each even communication cycle.

**VSBO[3:0]** Valid Sync Frames Channel B, **odd** communication cycle (vSyncFramesOddB)

Holds the number of valid sync frames received on channel B in the odd communication cycle. If transmission of sync frames is enabled by **SUCC1.TXSY** the value is incremented by one. The value is updated during the NIT of each odd communication cycle.

The bit fields above are only valid if the respective channel is assigned to the CC by **SUCC1.CCHA** or **SUCC1.CCHB**.

**MOCS** Missing Offset Correction Signal

The Missing Offset Correction flag signals to the Host, that no offset correction calculation can be performed because no sync frames were received. The flag is updated by the CC at start of offset correction phase.

1 =Missing offset correction signal

0 =Offset correction signal valid

**OCLR** Offset Correction Limit Reached

The Offset Correction Limit Reached flag signals to the Host, that the offset correction value has exceeded its limit as defined by **GTUC10.MOC[13:0]**. The flag is updated by the CC at start of offset correction phase.

1 =Offset correction limit reached

0 =Offset correction below limit

**MRCS** Missing Rate Correction Signal

The Missing Rate Correction flag signals to the Host, that no rate correction calculation can be performed because no pairs of even / odd sync frames were received. The flag is updated by the CC at start of offset correction phase.

1 =Missing rate correction signal

0 =Rate correction signal valid

**RCLR** Rate Correction Limit Reached

The Rate Correction Limit Reached flag signals to the Host, that the rate correction value has exceeded its limit as defined by **GTUC10.MRC[10:0]**. The flag is updated by the CC at start of offset correction phase.

1 =Rate correction limit reached

0 =Rate correction below limit

### *18.6.8. Symbol Window and NIT Status (SWNIT)*

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **SWNIT** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0124 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | SBNB | SENB | SBNA | SENA | MTSB | MTSA | TCSB | SBSB | SESB | TCSA | SBSA | SESA |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Symbol window related status information. Updated by the CC at the end of the symbol window for each channel. During startup the status data is not updated.

**SESA** Syntax Error in Symbol Window Channel A (vSS!SyntaxErrorA)

1 =Syntax error during symbol window detected on channel A

0 =No syntax error detected

**SBSA** Slot Boundary Violation in Symbol Window Channel A (vSS!BViolationA)

1 =Slot boundary violation during symbol window detected on channel A

0 =No slot boundary violation detected

**TCSA** Transmission Conflict in Symbol Window Channel A (vSS!TxConflictA)

1 =Transmission conflict in symbol window detected on channel A

0 =No transmission conflict detected

**SESB** Syntax Error in Symbol Window Channel B (vSS!SyntaxErrorB)

1 =Syntax error during symbol window detected on channel B

0 =No syntax error detected

**SBSB** Slot Boundary Violation in Symbol Window Channel B (vSS!BViolationB)

1 =Slot boundary violation during symbol window detected on channel B

0 =No slot boundary violation detected

**TCSB** Transmission Conflict in Symbol Window Channel B (vSS!TxConflictB)

1 =Transmission conflict in symbol window detected on channel B

0 =No transmission conflict detected

**MTSA** MTS Received on Channel A (vSS!ValidMTSA)
Media Access Test symbol received on channel A during the last symbol window. Updated by the CC for each channel at the end of the symbol window. When this bit is set to '1', also interrupt flag **SIR.MTSA** is set to '1'.
1 =MTS symbol received on channel A
0 =No MTS symbol received on channel A

**MTSB** MTS Received on Channel B (vSS!ValidMTSB)
Media Access Test symbol received on channel B during the last symbol window. Updated by the CC for each channel at the end of the symbol window. When this bit is set to '1', also interrupt flag **SIR.MTSB** is set to '1'.
1 =MTS symbol received on channel B
0 =No MTS symbol received on channel B
NIT related status information. Updated by the CC at the end of the NIT for each channel:

**SENA** Syntax Error during NIT Channel A (vSS!SyntaxErrorA)
1 =Syntax error during NIT detected on channel A
0 =No syntax error detected

**SBNA** Slot Boundary Violation during NIT Channel A (vSS!BViolationA)
1 =Slot boundary violation during NIT detected on channel A
0 =No slot boundary violation detected

**SENB** Syntax Error during NIT Channel B (vSS!SyntaxErrorB)
1 =Syntax error during NIT detected on channel B
0 =No syntax error detected

**SBNB** Slot Boundary Violation during NIT Channel B (vSS!BViolationB)
1 =Slot boundary violation during NIT detected on channel B
0 =No slot boundary violation detected

### *18.6.9. Aggregated Channel Status (ACS)*

The aggregated channel status provides the Host with an accrued status of channel activity for all communication slots regardless of whether they are assigned for transmission or subscribed for reception. The aggregated channel status also includes status data from the symbol window and the network idle time. The status data is updated (set) after each slot and aggregated until it is reset by the Host. During startup the status data is not updated. A flag is cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect on the flag. A hard reset will also clear the register.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **ACS** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0128 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | SBVB | CIB | CEDB | SEDB | VFRB | 0 | 0 | 0 | SBVA | CIA | CEDA | SEDA | VFRA |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**VFRA** Valid Frame Received on Channel A (vSS!ValidFrameA)
One or more valid frames were received on channel A in any static or dynamic slot during the observation period.
1 =Valid frame(s) received on channel A
0 =No valid frame received

**SEDA** Syntax Error Detected on Channel A (vSS!SyntaxErrorA)
One or more syntax errors in static or dynamic slots, symbol window, and NIT were observed on channel A.
1 =Syntax error(s) observed on channel A
0 =No syntax error observed

**CEDA** Content Error Detected on Channel A (vSS!ContentErrorA)
One or more frames with a content error were received on channel A in any static or dynamic slot during the observation period.
1 =Frame(s) with content error received on channel A
0 =No frame with content error received

**CIA** Communication Indicator Channel A
One or more valid frames were received on channel A in slots that also contained any additional communication during the observation period, i.e. one or more slots received a valid frame AND had any combination of either syntax error OR content error OR slot boundary violation.
1 =Valid frame(s) received on channel A in slots containing any additional communication
0 =No valid frame(s) received in slots containing any additional communication

**SBVA** Slot Boundary Violation on Channel A (vSS!BViolationA)
One or more slot boundary violations were observed on channel A at any time during the observation period (static or dynamic slots, symbol window, and NIT).
1 =Slot boundary violation(s) observed on channel A
0 =No slot boundary violation observed

**VFRB** Valid Frame Received on Channel B (vSS!ValidFrameB)
One or more valid frames were received on channel B in any static or dynamic slot during the observation period. Reset under control of the Host.
1 =Valid frame(s) received on channel B
0 =No valid frame received

**SEDB** Syntax Error Detected on Channel B (vSS!SyntaxErrorB)
One or more syntax errors in static or dynamic slots, symbol window, and NIT were observed on channel B.
1 =Syntax error(s) observed on channel B
0 =No syntax error observed

**CEDB** Content Error Detected on Channel B (vSS!ContentErrorB)
One or more frames with a content error were received on channel B in any static or dynamic slot during the observation period.
1 =Frame(s) with content error received on channel B
0 =No frame with content error received

**CIB** Communication Indicator Channel B
One or more valid frames were received on channel B in slots that also contained any additional communication during the observation period, i.e. one or more slots received a valid frame AND had any combination of either syntax error OR content error OR slot boundary violation.
1 =Valid frame(s) received on channel B in slots containing any additional communication
0 =No valid frame(s) received in slots containing any additional communication

**SBVB** Slot Boundary Violation on Channel B (vSS!BViolationB)
One or more slot boundary violations were observed on channel B at any time during the observation period (static or dynamic slots, symbol window, and NIT).
1 =Slot boundary violation(s) observed on channel B
0 =No slot boundary violation observed

The set condition of flags **CIA** and **CIB** is also fulfilled if there is only one single frame in the slot and the slot boundary at the end of the slot is reached during the frames channel idle recognition phase.
When one of the flags **SEDB**, **CEDB**, **CIB**, **SBVB** changes from ‘0‘ to ‘1‘, interrupt flag **EIR.EDB** is set to ’1’.When one of the flags **SEDA**, **CEDA**, **CIA**, **SBVA** changes from ‘0‘ to ‘1‘, interrupt flag **EIR.EDA** is set to ’1’.

### *18.6.10. Even Sync ID [1…15] (ESIDn)*

Registers ESID1 to ESID15 hold the frame IDs of the sync frames received in **even** communication cycles, assorted in ascending order, with register ESID1 holding the lowest received sync frame ID. If the node transmits a sync frame in an even communication cycle by itself, register ESID1 holds the respective sync frame ID as configured in message buffer 0. The value is updated during the NIT of each even communication cycle.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **ESIDn** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0130 - 0x0168 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | RXEB | RXEA | 0 | 0 | 0 | 0 | EID9 | EID8 | EID7 | EID6 | EID5 | EID4 | EID3 | EID2 | EID1 | EID0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**EID[9:0]** Even Sync ID (vsSyncIDListA,B even)
Sync frame ID even communication cycle.

**RXEA** Received Even Sync ID on Channel A
A sync frame corresponding to the stored even sync ID was received on channel A.
1 =Sync frame received on channel A
0 =Sync frame not received on channel A

**RXEB** Received Even Sync ID on Channel B
A sync frame corresponding to the stored even sync ID was received on channel B.
1 =Sync frame received on channel B
0 =Sync frame not received on channel B

### *18.6.11. Odd Sync ID [1…15] (OSIDn)*

Registers OSID1 to OSID15 hold the frame IDs of the sync frames received in **odd** communication cycles, assorted in ascending order, with register OSID1 holding the lowest received sync frame ID. If the node transmits a sync frame in an odd communication cycle by itself, register OSID1 holds the respective sync frame ID as configured in message buffer 0. The value is updated during the NIT of each odd communication cycle.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **OSIDn** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0170 - 0x01A8 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | RXOB | RXOA | 0 | 0 | 0 | 0 | OID9 | OID8 | OID7 | OID6 | OID5 | OID4 | OID3 | OID2 | OID1 | OID0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**OID[9:0]** Odd Sync ID (vsSyncIDListA,B odd)
Sync frame ID odd communication cycle.

**RXOA** Received Odd Sync ID on Channel A
A sync frame corresponding to the stored odd sync ID was received on channel A.
1 =Sync frame received on channel A
0 =Sync frame not received on channel A

**RXOB** Received Odd Sync ID on Channel B
A sync frame corresponding to the stored odd sync ID was received on channel B.
1 =Sync frame received on channel B
0 =Sync frame not received on channel B

### *18.6.12. Network Management Vector [1…3] (NMVn)*

The three network management registers hold the accrued NM vector (configurable 0 to 12 bytes). The accrued NM vector is generated by the CC by bit-wise ORing each NM vector received (valid static frames with PPI = '1') on each channel (see 19.6.Network Management).

The CC updates the NM vector at the end of each communication cycle as long as the CC is either in NORMAL_ACTIVE or NORMAL_PASSIVE state.

NMVn-bytes exceeding the configured NM vector length are not valid.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **NMVn** | R | NM31 | NM30 | NM29 | NM28 | NM27 | NM26 | NM25 | NM24 | NM23 | NM22 | NM21 | NM20 | NM19 | NM18 | NM17 | NM16 |
| 0x01B0 - 0x01B8 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | NM15 | NM14 | NM13 | NM12 | NM11 | NM10 | NM9 | NM8 | NM7 | NM6 | NM5 | NM4 | NM3 | NM2 | NM1 | NM0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Table  below shows the assignment of the received payload's data bytes to the network management vector.

| Bit / Word | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| NMV1 | Data3 | | | | | | | | Data2 | | | | | | | | Data1 | | | | | | | | Data0 | | | | | | | |
| NMV2 | Data7 | | | | | | | | Data6 | | | | | | | | Data5 | | | | | | | | Data4 | | | | | | | |
| NMV3 | Data11 | | | | | | | | Data10 | | | | | | | | Data9 | | | | | | | | Data8 | | | | | | | |

Assignment of data bytes to network management vector

## 18.7. Message Buffer Control Registers

### *18.7.1. Message RAM Configuration (MRC)*

The Message RAM Configuration register defines the number of message buffers assigned to the static segment, dynamic segment, and FIFO. The register can be written during DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **MRC** | R | 0 | 0 | 0 | 0 | 0 | SPLM* | SEC1* | SEC0* | LCB7* | LCB6* | LCB5* | LCB4* | LCB3* | LCB2* | LCB1* | LCB0* |
| 0x0300 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | FFB7* | FFB6* | FFB5* | FFB4* | FFB3* | FFB2* | FFB1* | FFB0* | FDB7* | FDB6* | FDB5* | FDB4* | FDB3* | FDB2* | FDB1* | FDB0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**FDB[7:0]** First Dynamic Buffer
0= No group of message buffers exclusively for the static segment configured
1…127= Message buffers 0 to **FDB** - 1 reserved for static segment
≥128= No dynamic message buffers configured

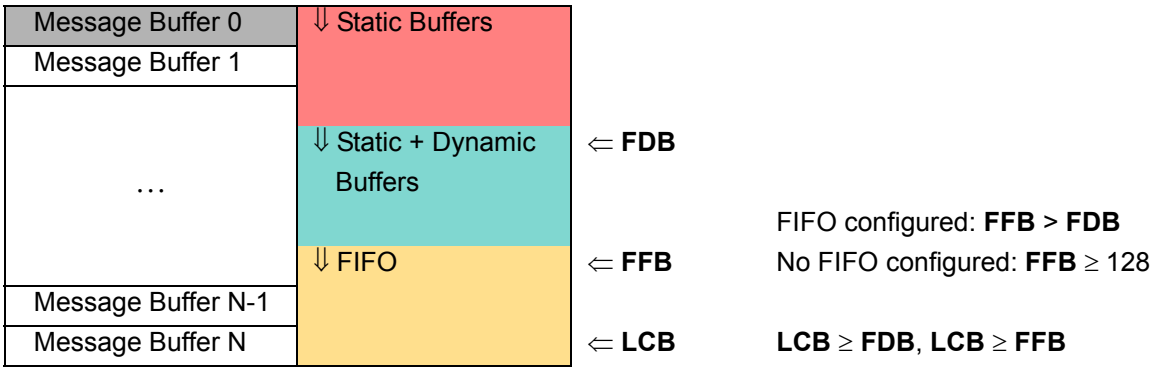
**FFB[7:0]** First Buffer of FIFO
0= All message buffers assigned to the FIFO
1…127= Message buffers from FFB to LCB assigned to the FIFO
≥128= No message buffer assigned to the FIFO

**LCB[7:0]** Last Configured Buffer
0…127= Number of message buffers is **LCB** + 1
≥128= No message buffer configured

**SEC[1:0]** Secure Buffers
Not evaluated when the CC is in DEFAULT_CONFIG or CONFIG state.
00 =Reconfiguration of message buffers enabled with numbers < **FFB** enabled
Exception: In nodes configured for sync frame transmission or for single slot mode operation message buffer 0 (and if **SPLM** = ’1’, also message buffer 1) is always locked
01 =Reconfiguration of message buffers with numbers < **FDB** and with numbers ≥ **FFB** locked
and transmission of message buffers for static segment with numbers ≥ **FDB** disabled
10 =Reconfiguration of all message buffers locked
11 =Reconfiguration of all message buffers locked
and transmission of message buffers for static segment with numbers ≥ **FDB** disabled

**SPLM** Sync Frame Payload Multiplex
This bit is only evaluated if the node is configured as sync node (**SUCC1.TXSY** = ’1’) or for single slot mode operation (**SUCC1.TSM** = ’1’). When this bit is set to ’1’ message buffers 0 and 1 are dedicated for sync frame transmission with different payload data on channel A and B. When this bit is set to ’0’, sync frames are transmitted from message buffer 0 with the same payload data on both channels. Note that the channel filter configuration for message buffer 0 resp. message buffer 1 has to be chosen accordingly.
1 =Both message buffers 0 and 1 are locked against reconfiguration
0 =Only message buffer 0 locked against reconfiguration

Note: In case the node is configured as sync node (**SUCC1.TXSY** = ’1’) or for single slot mode operation (**SUCC1.TSM** = ’1’), message buffer 0 resp. 1 is reserved for sync frames or single slot frames and have to be configured with the node-specific key slot ID. In case the node is neither configured as sync node nor for single slot operation message buffer 0 resp. 1 is treated like all other message buffers.




The programmer has to ensure that the configuration defined by **FDB[7:0]**, **FFB[7:0]**, and **LCB[7:0]** is valid.
**The CC does not check for erroneous configurations!**
The maximum number of header sections is 128. This means a maximum of 128 message buffers can be configured. The maximum length of a data section is 254 bytes. The length of the data section may be configured differently for each message buffer. For details see Section 19.12.Message RAM.
The payload length configured and the length of the data section need to be configured identical for all message buffers belonging to the FIFO via **WRHS2.PLC[6:0]** and **WRHS3.DP[10:0]**.
When the CC is not in DEFAULT_CONFIG or CONFIG state reconfiguration of message buffers belonging to the FIFO is locked.

*18.7.2. FIFO Rejection Filter (FRF)*

The FIFO Rejection Filter defines a user specified sequence of bits to which channel, frame ID, and cycle count of the incoming frames are compared. Together with the FIFO Rejection Filter Mask this register determines whether a message is rejected by the FIFO. The FRF register can be written during DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **FRF** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RNF* | RSS* | CYF6* | CYF5* | CYF4* | CYF3* | CYF2* | CYF1* | CYF0* |
| 0x0304 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | FID10* | FID9* | FID8* | FID7* | FID6* | FID5* | FID4* | FID3* | FID2* | FID1* | FID0* | CH1* | CH0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**CH[1:0]** Channel Filter
11 =no reception
10 =receive only on channel A
01 =receive only on channel B
00 =receive on both channels
If reception on both channels is configured, also in static segment always both frames (from channel A and B) are stored in the FIFO, even if they are identical.

**FID[10:0]** Frame ID Filter
A frame ID filter value of zero means that **no** frame ID is rejected.
0…2047 =Frame ID filter values

**CYF[6:0]** Cycle Counter Filter
The 7-bit cycle counter filter determines the cycle set to which frame ID and channel rejection filter are applied. In cycles **not** belonging to the cycle set specified by **CYF[6:0]**, all frames are rejected. For details about the configuration of the cycle counter filter see Section 19.7.2.Cycle Counter Filtering.

**RSS** Reject in Static Segment

If this bit is set, the FIFO is used only for the dynamic segment.

1 =Reject messages in static segment

0 =FIFO also used for static segment

**RNF** Reject Null Frames

If this bit is set, received null frames are not stored in the FIFO.

1 =Reject all null frames

0 =Null frames are stored in the FIFO

### *18.7.3. FIFO Rejection Filter Mask (FRFM)*

The FIFO Rejection Filter Mask specifies which of the corresponding frame ID filter bits are relevant for rejection filtering. If a bit is set, it indicates that the corresponding bit in the FRF register will not be considered for rejection filtering. The FRFM register can be written during DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **FRFM** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0308 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | MFID 10* | MFID 9* | MFID 8* | MFID 7* | MFID 6* | MFID 5* | MFID 4* | MFID 3* | MFID 2* | MFID 1* | MFID 0* | 0 | 0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**MFID[10:0]** Mask Frame ID Filter

1 =Ignore corresponding frame ID filter bit.

0 =Corresponding frame ID filter bit is used for rejection filtering

### *18.7.4. FIFO Critical Level (FCL)*

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **FCL** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x030C | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | CL7* | CL6* | CL5* | CL4* | CL3* | CL2* | CL1* | CL0* |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**CL[7:0]** Critical Level

When the receive FIFO fill level **FSR.RFFL[7:0]** is equal or greater than the critical level configured by **CL[7:0]**, the receive FIFO critical level flag **FSR.RFCL** is set. If **CL[7:0]** is programmed to values > 128, bit **FSR.RFCL** is never set. When **FSR.RFCL** changes from ’0’ to ’1’ bit **SIR.RFCL** is set to ’1’, and if enabled, an interrupt is generated.

## 18.8. Message Buffer Status Registers

### *18.8.1. Message Handler Status (MHDS)*

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **MHDS** | R | 0 | MBU6 | MBU5 | MBU4 | MBU3 | MBU2 | MBU1 | MBU0 | 0 | MBT6 | MBT5 | MBT4 | MBT3 | MBT2 | MBT1 | MBT0 |
| 0x0310 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | FMB6 | FMB5 | FMB4 | FMB3 | FMB2 | FMB1 | FMB0 | CRAM | MFMB | FMBD | PTBF2 | PTBF1 | PMR | POBF | PIBF |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

A flag is cleared by writing a ’1’ to the corresponding bit position. Writing a ’0’ has no effect on the flag. The register will also be cleared by hard reset or by CHI command CLEAR_RAMS.

**PIBF** Parity Error Input Buffer RAM 1,2
1 =Parity error occurred when reading Input Buffer RAM 1,2
0 =No parity error

**POBF** Parity Error Output Buffer RAM 1,2
1 =Parity error occurred when reading Output Buffer RAM 1,2
0 =No parity error

**PMR** Parity Error Message RAM
1 =Parity error occurred when reading the Message RAM
0 =No parity error

**PTBF1** Parity Error Transient Buffer RAM A
1 =Parity error occurred when reading Transient Buffer RAM A
0 =No parity error

**PTBF2** Parity Error Transient Buffer RAM B
1 =Parity error occurred when reading Transient Buffer RAM B
0 =No parity error
When one of the flags **PIBF**, **POBF**, **PMR**, **PTBF1**, **PTBF2** changes from ’0’ to ’1’ **EIR.PERR** is set to ’1’.

**FMBD** Faulty Message Buffer Detected
1 =Message buffer referenced by **FMB[6:0]** holds faulty data due to a parity error
0 =No faulty message buffer

**MFMB** Multiple Faulty Message Buffers detected
1 =Another faulty message buffer was detected while flag **FMBD** is set
0 =No additional faulty message buffer

**CRAM** Clear all internal RAM’s
Signals that execution of the CHI command CLEAR_RAMS is ongoing (all bits of all internal RAM blocks are written to ’0’). The bit is set by hard reset or by CHI command CLEAR_RAMS.
1 =Execution of the CHI command CLEAR_RAMS ongoing
0 =No execution of the CHI command CLEAR_RAMS

**FMB[6:0]** Faulty Message Buffer
Parity error occurred when reading from the message buffer or when transferring data from Input Buffer or Transient Buffer 1,2 to the message buffer referenced by **FMB[6:0]**. Value only valid when one of the flags **PIBF**, **PMR**, **PTBF1**, **PTBF2**, and flag **FMBD** is set. Updated only after the Host has reset flag **FMBD**.

**MBT[6:0]** Message Buffer Transmitted
Number of last successfully transmitted message buffer. If the message buffer is configured for single-shot mode, the respective **TXR** flag in the TXRQ1/2/3/4 registers was reset.

**MBU[6:0]** Message Buffer Updated
Number of message buffer that was updated last by the CC. For this message buffer the respective **ND** and / or **MBC** flag in the NDAT1/2/3/4 registers and the MBSC1/2/3/4 registers are also set.
**MBT[6:0]** and **MBU[6:0]** are reset when the CC leaves CONFIG state or enters STARTUP state.

### 18.8.2. Last Dynamic Transmit Slot (LDTS)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **LDTS** | R | 0 | 0 | 0 | 0 | 0 | LDTB10 | LDTB9 | LDTB8 | LDTB7 | LDTB6 | LDTB5 | LDTB4 | LDTB3 | LDTB2 | LDTB1 | LDTB0 |
| 0x0314 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | LDTA10 | LDTA9 | LDTA8 | LDTA7 | LDTA6 | LDTA5 | LDTA4 | LDTA3 | LDTA2 | LDTA1 | LDTA0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

**LDTA[10:0]** Last Dynamic Transmission Channel A

Value of vSlotCounter[A] at the time of the last frame transmission on channel A in the dynamic segment of this node. It is updated at the end of the dynamic segment and is reset to zero if no frame was transmitted during the dynamic segment.

**LDTB[10:0]** Last Dynamic Transmission Channel B

Value of vSlotCounter[B] at the time of the last frame transmission on channel B in the dynamic segment of this node. It is updated at the end of the dynamic segment and is reset to zero if no frame was transmitted during the dynamic segment.

### 18.8.3. FIFO Status Register (FSR)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **FSR** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0318 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | RFFL7 | RFFL6 | RFFL5 | RFFL4 | RFFL3 | RFFL2 | RFFL1 | RFFL0 | 0 | 0 | 0 | 0 | 0 | RFO | RFCL | RFNE |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

**RFNE** Receive FIFO Not Empty

This flag is set by the CC when a received valid frame (data or null frame depending on rejection mask) was stored in the FIFO. In addition, interrupt flag **SIR.RFNE** is set. The bit is reset after the Host has read all message from the FIFO.

1 =Receive FIFO is not empty

0 =Receive FIFO is empty

**RFCL** Receive FIFO Critical Level

This flag is set when the receive FIFO fill level **RFFL[7:0]** is equal or greater than the critical level as configured by **FCL.CL[7:0]**. The flag is cleared by the CC as soon as **RFFL[7:0]** drops below **FCL.CL[7:0]**. When **RFCL** changes from ’0’ to ’1’ bit **SIR.RFCL** is set to ’1’, and if enabled, an interrupt is generated.

1 =Receive FIFO critical level reached

0 =Receive FIFO below critical level

**RFO** Receive FIFO Overrun

The flag is set by the CC when a receive FIFO overrun is detected. When a receive FIFO overrun occurs, the oldest message is overwritten with the actual received message. In addition, interrupt flag **EIR.RFO** is set.The flag is cleared by the next FIFO read access issued by the Host.

1 =A receive FIFO overrun has been detected

0 =No receive FIFO overrun detected

**RFFL[7:0]** Receive FIFO Fill Level

Number of FIFO buffers filled with new data not yet read by the Host. Maximum value is 128.

### 18.8.4. Message Handler Constraints Flags (MHDF)

Some constraints exist for the Message Handler regarding eray_bclk frequency, Message RAM configuration, and FlexRay bus traffic (see Addendum to E-Ray FlexRay IP-Module Specification). To simplify software development, constraints violations are reported by setting flags in the MHDF.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **MHDF** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x031C | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | WAHP | 0 | 0 | TBFB | TBFA | FNFB | FNFA | SNUB | SNUA |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

A flag is cleared by writing a ’1’ to the corresponding bit position. Writing a ’0’ has no effect on the flag. A hard reset will also clear the register. The register is reset when the CC leaves CONFIG state or enters STARTUP state.

**SNUA** Status Not Updated Channel A

This flag is set by the CC when the Message Handler, due to overload condition, was not able to update a message buffer’s status MBS with respect to channel A.

1 =MBS for channel A not updated

0 =No overload condition occurred when updating MBS for channel A

**SNUB** Status Not Updated Channel B

This flag is set by the CC when the Message Handler, due to overload condition, was not able to update a message buffer’s status MBS with respect to channel B.

1 =MBS for channel B not updated

0 =No overload condition occurred when updating MBS for channel B

**FNFA** Find Sequence Not Finished Channel A

This flag is set by the CC when the Message Handler, due to overload condition, was not able to finish a find sequence (scan of Message RAM for matching message buffer) with respect to channel A.

1 =Find sequence not finished for channel A

0 =No find sequence not finished for channel A

**FNFB** Find Sequence Not Finished Channel B

This flag is set by the CC when the Message Handler, due to overload condition, was not able to finish a find sequence (scan of Message RAM for matching message buffer) with respect to channel B.

1 =Find sequence not finished for channel B

0 =No find sequence not finished for channel B

**TBFA** Transient Buffer Access Failure A

This flag is set by the CC when a read or write access to TBF A requested by PRT A could not complete within the available time.

1 =TBF A access failure

0 =No TBF A access failure

**TBFB** Transient Buffer Access Failure B

This flag is set by the CC when a read or write access to TBF B requested by PRT B could not complete within the available time.

1 =TBF B access failure

0 =No TBF B access failure

**WAHP** Write Attempt to Header Partition

This flag is set by the CC when the message handler tries to write message data into the header partition of the Message RAM due to faulty configuration of a message buffer. The write attempt is not executed, to protect the header partition from unintended write accesses.

1 =Write attempt to header partition

0 =No write attempt to header partition

When one of the flags **SNUA**, **SNUB**, **FNFA**, **FNFB**, **TBFA**, **TBFB**, **WAHP** changes from ’0’ to ’1’, interrupt flag **EIR.MHF** is set to ’1’.

### 18.8.5. Transmission Request 1/2/3/4 (TXRQ1/2/3/4)

The four registers reflect the state of the **TXR** flags of all configured message buffers. The flags are evaluated for transmit buffers only. If the number of configured message buffers is less than 128, the remaining **TXR** flags have no meaning.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **TXRQ4** | R | TXR127 | TXR126 | TXR125 | TXR124 | TXR123 | TXR122 | TXR121 | TXR120 | TXR119 | TXR118 | TXR117 | TXR116 | TXR115 | TXR114 | TXR113 | TXR112 |
| 0x032C | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | TXR111 | TXR110 | TXR109 | TXR108 | TXR107 | TXR106 | TXR105 | TXR104 | TXR103 | TXR102 | TXR101 | TXR100 | TXR99 | TXR98 | TXR97 | TXR96 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **TXRQ3** | R | TXR95 | TXR94 | TXR93 | TXR92 | TXR91 | TXR90 | TXR89 | TXR88 | TXR87 | TXR86 | TXR85 | TXR84 | TXR83 | TXR82 | TXR81 | TXR80 |
| 0x0328 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | TXR79 | TXR78 | TXR77 | TXR76 | TXR75 | TXR74 | TXR73 | TXR72 | TXR71 | TXR70 | TXR69 | TXR68 | TXR67 | TXR66 | TXR65 | TXR64 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **TXRQ2** | R | TXR63 | TXR62 | TXR61 | TXR60 | TXR59 | TXR58 | TXR57 | TXR56 | TXR55 | TXR54 | TXR53 | TXR52 | TXR51 | TXR50 | TXR49 | TXR48 |
| 0x0324 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | TXR47 | TXR46 | TXR45 | TXR44 | TXR43 | TXR42 | TXR41 | TXR40 | TXR39 | TXR38 | TXR37 | TXR36 | TXR35 | TXR34 | TXR33 | TXR32 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **TXRQ1** | R | TXR31 | TXR30 | TXR29 | TXR28 | TXR27 | TXR26 | TXR25 | TXR24 | TXR23 | TXR22 | TXR21 | TXR20 | TXR19 | TXR18 | TXR17 | TXR16 |
| 0x0320 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | TXR15 | TXR14 | TXR13 | TXR12 | TXR11 | TXR10 | TXR9 | TXR8 | TXR7 | TXR6 | TXR5 | TXR4 | TXR3 | TXR2 | TXR1 | TXR0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**TXR[127:0]** Transmission Request

If the flag is set, the respective message buffer is ready for transmission respectively transmission of this message buffer is in progress. In single-shot mode the flags are reset after transmission has completed.

### 18.8.6. New Data 1/2/3/4 (NDAT1/2/3/4)

The four registers reflect the state of the **ND** flags of all configured message buffers. **ND** flags belonging to transmit buffers have no meaning. If the number of configured message buffers is less than 128, the remaining **ND** flags have no meaning. The registers are reset when the CC leaves CONFIG state or enters STARTUP state.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **NDAT4** | R | ND127 | ND126 | ND125 | ND124 | ND123 | ND122 | ND121 | ND120 | ND119 | ND118 | ND117 | ND116 | ND115 | ND114 | ND113 | ND112 |
| 0x033C | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | ND111 | ND110 | ND109 | ND108 | ND107 | ND106 | ND105 | ND104 | ND103 | ND102 | ND101 | ND100 | ND99 | ND98 | ND97 | ND96 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **NDAT3** | R | ND95 | ND94 | ND93 | ND92 | ND91 | ND90 | ND89 | ND88 | ND87 | ND86 | ND85 | ND84 | ND83 | ND82 | ND81 | ND80 |
| 0x0338 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | ND79 | ND78 | ND77 | ND76 | ND75 | ND74 | ND73 | ND72 | ND71 | ND70 | ND69 | ND68 | ND67 | ND66 | ND65 | ND64 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **NDAT2** | R | ND63 | ND62 | ND61 | ND60 | ND59 | ND58 | ND57 | ND56 | ND55 | ND54 | ND53 | ND52 | ND51 | ND50 | ND49 | ND48 |
| 0x0334 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | ND47 | ND46 | ND45 | ND44 | ND43 | ND42 | ND41 | ND40 | ND39 | ND38 | ND37 | ND36 | ND35 | ND34 | ND33 | ND32 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **NDAT1** | R | ND31 | ND30 | ND29 | ND28 | ND27 | ND26 | ND25 | ND24 | ND23 | ND22 | ND21 | ND20 | ND19 | ND18 | ND17 | ND16 |
| 0x0330 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | ND15 | ND14 | ND13 | ND12 | ND11 | ND10 | ND9 | ND8 | ND7 | ND6 | ND5 | ND4 | ND3 | ND2 | ND1 | ND0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ND[127:0]** New Data

The flags are set when a valid received data frame matches the message buffer's filter configuration, independent of the payload length received or the payload length configured for that message buffer. The flags are not set after reception of null frames except for message buffers belonging to the receive FIFO. An **ND** flag is reset when the header section of the corresponding message buffer is reconfigured or when the data section has been transferred to the Output Buffer.

### *18.8.7. Message Buffer Status Changed 1/2/3/4 (MBSC1/2/3/4)*

The four registers reflect the state of the **MBC** flags of all configured message buffers. If the number of configured message buffers is less than 128, the remaining **MBC** flags have no meaning. The registers are reset when the CC leaves CONFIG state or enters STARTUP state.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **MBSC4** | R | MBC127 | MBC126 | MBC125 | MBC124 | MBC123 | MBC122 | MBC121 | MBC120 | MBC119 | MBC118 | MBC117 | MBC116 | MBC115 | MBC114 | MBC113 | MBC112 |
| 0x034C | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | MBC111 | MBC110 | MBC109 | MBC108 | MBC107 | MBC106 | MBC105 | MBC104 | MBC103 | MBC102 | MBC101 | MBC100 | MBC99 | MBC98 | MBC97 | MBC96 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **MBSC3** | R | MBC95 | MBC94 | MBC93 | MBC92 | MBC91 | MBC90 | MBC89 | MBC88 | MBC87 | MBC86 | MBC85 | MBC84 | MBC83 | MBC82 | MBC81 | MBC80 |
| 0x0348 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | MBC79 | MBC78 | MBC77 | MBC76 | MBC75 | MBC74 | MBC73 | MBC72 | MBC71 | MBC70 | MBC69 | MBC68 | MBC67 | MBC66 | MBC65 | MBC64 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **MBSC2** | R | MBC63 | MBC62 | MBC61 | MBC60 | MBC59 | MBC58 | MBC57 | MBC56 | MBC55 | MBC54 | MBC53 | MBC52 | MBC51 | MBC50 | MBC49 | MBC48 |
| 0x0344 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | MBC47 | MBC46 | MBC45 | MBC44 | MBC43 | MBC42 | MBC41 | MBC40 | MBC39 | MBC38 | MBC37 | MBC36 | MBC35 | MBC34 | MBC33 | MBC32 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **MBSC1** | R | MBC31 | MBC30 | MBC29 | MBC28 | MBC27 | MBC26 | MBC25 | MBC24 | MBC23 | MBC22 | MBC21 | MBC20 | MBC19 | MBC18 | MBC17 | MBC16 |
| 0x0340 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | MBC15 | MBC14 | MBC13 | MBC12 | MBC11 | MBC10 | MBC9 | MBC8 | MBC7 | MBC6 | MBC5 | MBC4 | MBC3 | MBC2 | MBC1 | MBC0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**MBC[127:0]** Message Buffer Status Changed

An **MBC** flag is set whenever the Message Handler changes one of the status flags **VFRA**, **VFRB**, **SEOA**, **SEOB**, **CEOA**, **CEOB**, **SVOA**, **SVOB**, **TCIA**, **TCIB**, **ESA**, **ESB**, **MLST**, **FTA**, **FTB** in the header section (see 18.11.5.Message Buffer Status (MBS) and 19.12.1.Header Partition, header 4) of the respective message buffer. An **MBC** flag is reset when the header section of the corresponding message buffer is reconfigured or when it has been transferred to the Output Buffer.

## 18.9. Identification Registers

### *18.9.1. Core Release Register (CREL)*

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **CREL** | R | REL3 | REL2 | REL1 | REL0 | STEP7 | STEP6 | STEP5 | STEP4 | STEP3 | STEP2 | STEP1 | STEP0 | YEAR3 | YEAR2 | YEAR1 | YEAR0 |
| 0x03F0 | W | | | | | | | | | | | | | | | | |
| Reset | | release info | | | | | | | | | | | | | | | |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | MON7 | MON6 | MON5 | MON4 | MON3 | MON2 | MON1 | MON0 | DAY7 | DAY6 | DAY5 | DAY4 | DAY3 | DAY2 | DAY1 | DAY0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | release info | | | | | | | | | | | | | | | |

**DAY[7:0]** Design Time Stamp, Day
Two digits, BCD-coded.
**MON[7:0]** Design Time Stamp, Month
Two digits, BCD-coded.
**YEAR[3:0]** Design Time Stamp, Year
One digit, BCD-coded.
**STEP[7:0]** Step of Core Release
Two digits, BCD-coded.
**REL[3:0]** Core Release
One digit, BCD-coded.

The table below shows how releases are coded in register CREL.

#### Coding for releases

| Release | Step | Sub-Step | Name |
|---|---|---|---|
| 0 | 7 | 0 | Beta2 |
| 0 | 7 | 1 | Beta2ct |
| 1 | 0 | 0 | Revision 1.0.0 |

### *18.9.2. Endian Register (ENDN)*

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **ENDN** | R | ETV31 | ETV30 | ETV29 | ETV28 | ETV27 | ETV26 | ETV25 | ETV24 | ETV23 | ETV22 | ETV21 | ETV20 | ETV19 | ETV18 | ETV17 | ETV16 |
| 0x03F4 | W | | | | | | | | | | | | | | | | |
| Reset | | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | ETV15 | ETV14 | ETV13 | ETV12 | ETV11 | ETV10 | ETV9 | ETV8 | ETV7 | ETV6 | ETV5 | ETV4 | ETV3 | ETV2 | ETV1 | ETV0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |

**ETV[31:0]** Endianness Test Value
The endianness test value is 0x87654321.

## 18.10. Input Buffer

Double buffer structure consisting of Input Buffer Host and Input Buffer Shadow. While the Host can write to Input Buffer Host, the transfer to the Message RAM is done from Input Buffer Shadow. The Input Buffer holds the header and data sections to be transferred to the selected message buffer in the Message RAM. It is used to configure the message buffers in the Message RAM and to update the data sections of transmit buffers.

When updating the header section of a message buffer in the Message RAM from the Input Buffer, the Message Buffer Status as described in Section 18.11.5.Message Buffer Status (MBS) is automatically reset to zero.

The header sections of message buffers belonging to the receive FIFO can only be (re)configured when the CC is in DEFAULT_CONFIG or CONFIG state. For those message buffers only the payload length configured and the data pointer need to be configured via **WRHS2.PLC[6.0]** and **WRHS3.DP[10:0]**. All information required for acceptance filtering is taken from the FIFO rejection filter and the FIFO rejection filter mask.

The data transfer between Input Buffer (IBF) and Message RAM is described in detail in Section Data Transfer from Input Buffer to Message RAM.

### *18.10.1. Write Data Section [1…64] (WRDSn)*

Holds the data words to be transferred to the data section of the addressed message buffer. The data words ($DW_n$) are written to the Message RAM in transmission order from $DW_1$ (byte0, byte1) to $DW_{PL}$ (PL = number of data words as defined by the payload length configured **WRHS2.PLC[6:0]**).

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **WRDSn** | R | MD31 | MD30 | MD29 | MD28 | MD27 | MD26 | MD25 | MD24 | MD23 | MD22 | MD21 | MD20 | MD19 | MD18 | MD17 | MD16 |
| 0x0400 - 0x04FC | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | MD15 | MD14 | MD13 | MD12 | MD11 | MD10 | MD9 | MD8 | MD7 | MD6 | MD5 | MD4 | MD3 | MD2 | MD1 | MD0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**MD[31:0]** Message Data
**MD[7:0]**= $DW_n$, $byte_{n-1}$
**MD[15:8]**= $DW_n$, $byte_n$
**MD[23:16]**= $DW_{n+1}$, $byte_{n+1}$
**MD[31:24]**= $DW_{n+1}$, $byte_{n+2}$

Note: DW127 is located on WRDS64.MD[15:0]. In this case WRDS64.MD[31:16] is unused (no valid data). The Input Buffer RAMs are initialized to zero when leaving hard reset or by CHI command CLEAR_RAMS.

### 18.10.2. Write Header Section 1 (WRHS1)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **WRHS1** | R | 0 | 0 | MBI | TXM | PPIT | CFG | CHB | CHA | 0 | CYC6 | CYC5 | CYC4 | CYC3 | CYC2 | CYC1 | CYC0 |
| 0x0500 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | FID10 | FID9 | FID8 | FID7 | FID6 | FID5 | FID4 | FID3 | FID2 | FID1 | FID0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**FID[10:0]** Frame ID

Frame ID of the selected message buffer. The frame ID defines the slot number for transmission / reception of the respective message. **Message buffers with frame ID = '0' are considered as not valid.**

**CYC[6:0]** Cycle Code

The 7-bit cycle code determines the cycle set used for cycle counter filtering. For details about the configuration of the cycle code see Section 19.7.2.Cycle Counter Filtering.

**CHA, CHB** Channel Filter Control

The 2-bit channel filtering field associated with each buffer serves as a filter for receive buffers, and as a control field for transmit buffers.

| CHA | CHB | **Transmit Buffer**<br>transmit frame on | **Receive Buffer**<br>store frame received from |
|---|---|---|---|
| 1 | 1 | both channels<br>(static segment only) | channel A or B<br>(store first semantically valid frame, static segment only) |
| 1 | 0 | channel A | channel A |
| 0 | 1 | channel B | channel B |
| 0 | 0 | no transmission | ignore frame |

If a message buffer is configured for the dynamic segment and both bits of the channel filtering field are set to '1', no frames are transmitted resp. received frames are ignored (same function as **CHA = CHB** = '0')

**CFG** Message Buffer Direction Configuration Bit

This bit is used to configure the corresponding buffer as transmit buffer or as receive buffer. For message buffers belonging to the receive FIFO the bit is not evaluated.

1 =The corresponding buffer is configured as **Transmit Buffer**

0 =The corresponding buffer is configured as **Receive Buffer**

**PPIT** Payload Preamble Indicator Transmit

This bit is used to control the state of the Payload Preamble Indicator in transmit frames. If the bit is set in a static message buffer, the respective message buffer holds network management information. If the bit is set in a dynamic message buffer the first two bytes of the payload segment may be used for message ID filtering by the receiver. Message ID filtering of received FlexRay frames is not supported by the E-Ray module, but can be done by the Host.

1 =Payload Preamble Indicator set

0 =Payload Preamble Indicator not set

**TXM** Transmission Mode

This bit is used to select the transmission mode (see Section 19.8.3.Transmit Buffers).

1 =Single-shot mode

0 =Continuous mode

**MBI** Message Buffer Interrupt

This bit enables the receive / transmit interrupt for the corresponding message buffer. After a dedicated receive buffer has been updated by the Message Handler, flag **SIR.RXI** and /or **SIR.MBSI** are set. After a transmission has completed flag **SIR.TXI** is set.

1 =The corresponding message buffer interrupt is enabled

0 =The corresponding message buffer interrupt is disabled

### *18.10.3. Write Header Section 2 (WRHS2)*

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **WRHS2** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PLC6 | PLC5 | PLC4 | PLC3 | PLC2 | PLC1 | PLC0 |
| 0x0504 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | CRC10 | CRC9 | CRC8 | CRC7 | CRC6 | CRC5 | CRC4 | CRC3 | CRC2 | CRC1 | CRC0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**CRC[10:0]** Header CRC (vRF!Header!HeaderCRC)

Receive Buffer: Configuration not required

Transmit Buffer: Header CRC calculated and configured by the Host

For calculation of the header CRC the payload length of the frame send on the bus has to be considered. In static segment the payload length of all frames is configured by **MHDC.SFDL[6:0]**.

**PLC[6:0]** Payload Length Configured

Length of data section (number of 2-byte words) as configured by the Host. During static segment the static frame payload length as configured by **MHDC.SFDL[6:0]** defines the payload length for all static frames. If the payload length configured by **PLC[6:0]** is shorter than this value padding bytes are inserted to ensure that frames have proper physical length. The padding pattern is logical zero.

### *18.10.4. Write Header Section 3 (WRHS3)*

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **WRHS3** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x0508 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | DP10 | DP9 | DP8 | DP7 | DP6 | DP5 | DP4 | DP3 | DP2 | DP1 | DP0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**DP[10:0]** Data Pointer

Pointer to the first 32-bit word of the data section of the addressed message buffer in the Message RAM.

### 18.10.5. Input Buffer Command Mask (IBCM)

Configures how the message buffer in the Message RAM selected by register IBCR is updated. When IBF Host and IBF Shadow are swapped, also mask bits **LHSH**, **LDSH**, and **STXRH** are swapped with bits **LHSS**, **LDSS**, and **STXRS** to keep them attached to the respective Input Buffer transfer.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **IBCM** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | STXRS | LDSS | LHSS |
| 0x0510 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | STXRH | LDSH | LHSH |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**LHSH** Load Header Section Host

1 =Header section selected for transfer from Input Buffer to the Message RAM

0 =Header section is not updated

**LDSH** Load Data Section Host

1 =Data section selected for transfer from Input Buffer to the Message RAM

0 =Data section is not updated

**STXRH** Set Transmission Request Host

If this bit is set to '1', the **TXR** flag for the selected message buffer is set in the TXRQ1/2/3/4 registers to release the message buffer for transmission. In single-shot mode the flag is cleared by the CC after transmission has completed. **TXR** is evaluated for transmit buffers only.

1 =Set **TXR** flag, transmit buffer released for transmission

0 =Reset **TXR** flag

**LHSS** Load Header Section Shadow

1 =Header section selected for transfer from Input Buffer to the Message RAM (transfer ongoing or finished)

0 =Header section is not updated

LDSS Load Data Section Shadow

1 =Data section selected for transfer from Input Buffer to the Message RAM (transfer ongoing or finished)

0 =Data section is not updated

**STXRS** Set Transmission Request Shadow

1 =Set **TXR** flag, transmit buffer released for transmission (operation ongoing or finished)

0 =Reset **TXR** flag

*18.10.6. Input Buffer Command Request (IBCR)*

When the Host writes the number of the target message buffer in the Message RAM to **IBRH[6:0]**, IBF Host and IBF Shadow are swapped. In addition the message buffer numbers stored under **IBRH[6:0]** and **IBRS[6:0]** are also swapped (see also Section Data Transfer from Input Buffer to Message RAM).

With this write operation the **IBSYS** is set to ’1’. The Message Handler then starts to transfer the contents of IBF Shadow to the message buffer in the Message RAM selected by **IBRS[6:0]**.

While the Message Handler transfers the data from IBF Shadow to the target message buffer in the Message RAM, the Host may write the next message into the IBF Host. After the transfer between IBF Shadow and the Message RAM has completed, **IBSYS** is set back to ’0’ and the next transfer to the Message RAM may be started by the Host by writing the respective target message buffer number to **IBRH[6:0]**.

If a write access to **IBRH[6:0]** occurs while **IBSYS** is ’1’, **IBSYH** is set to ’1’. After completion of the ongoing data transfer from IBF Shadow to the Message RAM, IBF Host and IBF Shadow are swapped, **IBSYH** is reset to ’0’. **IBSYS** remains set to ’1’, and the next transfer to the Message RAM is started. In addition the message buffer numbers stored under **IBRH[6:0]** and **IBRS[6:0]** are also swapped.

Any write access to an Input Buffer register while both **IBSYS** and **IBSYH** are set will cause the error flag **EIR.IIBA** to be set. In this case the Input Buffer will not be changed.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **IBCR** | R | IBSYS | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | IBRS6 | IBRS5 | IBRS4 | IBRS3 | IBRS2 | IBRS1 | IBRS0 |
| 0x0514 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | IBSYH | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | IBRH6 | IBRH5 | IBRH4 | IBRH3 | IBRH2 | IBRH1 | IBRH0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**IBRH[6:0]** Input Buffer Request Host

Selects the target message buffer in the Message RAM for data transfer from Input Buffer.
Valid values are 0x00 to 0x7F (0…127).

**IBSYH** Input Buffer Busy Host

Set to ’1’ by writing **IBRH[6:0]** while **IBSYS** is still ’1’. After the ongoing transfer between IBF Shadow and the Message RAM has completed, the **IBSYH** is set back to ’0’.
1 = Request while transfer between IBF Shadow and Message RAM in progress
0 = No request pending

**IBRS[6:0]** Input Buffer Request Shadow

Number of the target message buffer actually updated / lately updated.
Valid values are 0x00 to 0x7F (0…127).

**IBSYS** Input Buffer Busy Shadow

Set to ’1’ after writing **IBRH[6:0]**. When the transfer between IBF Shadow and the Message RAM has completed, **IBSYS** is set back to ’0’.
1 =Transfer between IBF Shadow and Message RAM in progress
0 =Transfer between IBF Shadow and Message RAM completed

## 18.11. Output Buffer

Double buffer structure consisting of Output Buffer Host and Output Buffer Shadow. Used to read out message buffers from the Message RAM. While the Host can read from Output Buffer Host, the Message Handler transfers the selected message buffer from Message RAM to Output Buffer Shadow. The data transfer between Message RAM and Output Buffer (OBF) is described in Section Data Transfer from Message RAM to Output Buffer.

### *18.11.1. Read Data Section [1…64] (RDDSn)*

Holds the data words read from the data section of the addressed message buffer. The data words ($DW_n$) are read from the Message RAM in reception order from $DW_1$ (byte0, byte1) to $DW_{PL}$ (PL = number of data words as defined by the payload length configured **RDHS2.PLC[6:0]**).

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **RDDSn** | R | MD31 | MD30 | MD29 | MD28 | MD27 | MD26 | MD25 | MD24 | MD23 | MD22 | MD21 | MD20 | MD19 | MD18 | MD17 | MD16 |
| 0x0600 - 0x06FC | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | MD15 | MD14 | MD13 | MD12 | MD11 | MD10 | MD9 | MD8 | MD7 | MD6 | MD5 | MD4 | MD3 | MD2 | MD1 | MD0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**MD[31:0]** Message Data
- **MD[7:0]**= $DW_n$, $byte_{n-1}$
- **MD[15:8]**= $DW_n$, $byte_n$
- **MD[23:16]**= $DW_{n+1}$, $byte_{n+1}$
- **MD[31:24]**= $DW_{n+1}$, $byte_{n+2}$

Note: DW127 is located on RDDS64.MD[15:0]. In this case RDDS64.MD[31:16] is unused (no valid data). The Output Buffer RAMs are initialized to zero when leaving hard reset or by CHI command CLEAR_RAMS.

### *18.11.2. Read Header Section 1 (RDHS1)*

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **RDHS1** | R | 0 | 0 | MBI | TXM | PPIT | CFG | CHB | CHA | 0 | CYC6 | CYC5 | CYC4 | CYC3 | CYC2 | CYC1 | CYC0 |
| 0x0700 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | FID10 | FID9 | FID8 | FID7 | FID6 | FID5 | FID4 | FID3 | FID2 | FID1 | FID0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Values as configured by the Host via WRHS1:

**FID[10:0]** Frame ID
**CYC[6:0]** Cycle Code
**CHA, CHB** Channel Filter Control
**CFG** Message Buffer Direction Configuration Bit
**PPIT** Payload Preamble Indicator Transmit
**TXM** Transmission Mode
**MBI** Message Buffer Interrupt

In case that the message buffer read from the Message RAM belongs to the receive FIFO, **FID[10:0]** holds the received frame ID, while **CYC[6:0]**, **CHA**, **CHB**, **CFG**, **PPIT**, **TXM**, and **MBI** are reset to '0'.

### 18.11.3. Read Header Section 2 (RDHS2)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **RDHS2** | R | 0 | PLR6 | PLR5 | PLR4 | PLR3 | PLR2 | PLR1 | PLR0 | 0 | PLC6 | PLC5 | PLC4 | PLC3 | PLC2 | PLC1 | PLC0 |
| 0x0704 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | CRC10 | CRC9 | CRC8 | CRC7 | CRC6 | CRC5 | CRC4 | CRC3 | CRC2 | CRC1 | CRC0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**CRC[10:0]** Header CRC (vRF!Header!HeaderCRC)
Receive Buffer: Header CRC updated from received data frames
Transmit Buffer: Header CRC calculated and configured by the Host

**PLC[6:0]** Payload Length Configured
Length of data section (number of 2-byte words) as configured by the Host.

**PLR[6:0]** Payload Length Received (vRF!Header!Length)
Payload length value updated from received data frames (exception: if message buffer belongs to the receive FIFO **PLR[6:0]** is also updated from received null frames)

When a message is stored into a message buffer the following behaviour with respect to payload length received and payload length configured is implemented:

**PLR[6:0]** > **PLC[6:0]**:The payload data stored in the message buffer is truncated to the payload length configured if **PLC[6:0]** even or else truncated to **PLC[6:0]** + 1.

**PLR[6:0]** ≤ **PLC[6:0]**:The received payload data is stored into the message buffers data section. The remaining data bytes of the data section as configured by **PLC[6:0]** are filled with undefined data

**PLR[6:0]** = zero:The message buffer's data section is filled with undefined data

**PLC[6:0]** = zero:Message buffer has no data section configured. No data is stored into the message buffer's data section.

Note: The Message RAM is organized in 4-byte words. When received data is stored into a message buffer's data section, the number of 2-byte data words written into the message buffer is **PLC[6:0]** rounded to the next even value. **PLC[6:0]** should be configured identical for all message buffers belonging to the receive FIFO. Header 2 is updated from data frames only.

### 18.11.4. Read Header Section 3 (RDHS3)

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **RDHS3** | R | 0 | 0 | RES | PPI | NFI | SYN | SFI | RCI | 0 | 0 | RCC5 | RCC4 | RCC3 | RCC2 | RCC1 | RCC0 |
| 0x0708 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | DP10 | DP9 | DP8 | DP7 | DP6 | DP5 | DP4 | DP3 | DP2 | DP1 | DP0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**DP[10:0]** Data Pointer

Pointer to the first 32-bit word of the data section of the addressed message buffer in the Message RAM.

**RCC[5:0]** Receive Cycle Count (vRF!Header!CycleCount)

Cycle counter value updated from received data frame.

**RCI** Received on Channel Indicator (vSS!Channel)

Indicates the channel from which the received data frame was taken to update the respective receive buffer.

1 =Frame received on channel A

0 =Frame received on channel B

**SFI** Startup Frame Indicator (vRF!Header!SuFIndicator)

A startup frame is marked by the startup frame indicator.

1 =The received frame is a startup frame

0 =The received frame is not a startup frame

**SYN** Sync Frame Indicator (vRF!Header!SyFIndicator)

A sync frame is marked by the sync frame indicator.

1 =The received frame is a sync frame

0 =The received frame is not a sync frame

**NFI** Null Frame Indicator (vRF!Header!NFIndicator)

Is set to ’1’ after storage of the first received data frame.

1 =At least one data frame has been stored into the respective message buffer

0 =Up to now no data frame has been stored into the respective message buffer

**PPI** Payload Preamble Indicator (vRF!Header!PPIndicator)

The payload preamble indicator defines whether a network management vector or message ID is contained within the payload segment of the received frame.

1 =Static segment:Network management vector in the first part of the payload

Dynamic segment:Message ID in the first part of the payload

0 =The payload segment of the received frame does not contain a network management vector nor a message ID

**RES** Reserved Bit (vRF!Header!Reserved)

Reflects the state of the received reserved bit. The reserved bit is transmitted as ’0’.

Header 3 is updated from data frames only.

### *18.11.5. Message Buffer Status (MBS)*

The message buffer status is updated by the CC with respect to the assigned channel(s) latest at the end of the slot following the slot assigned to the message buffer. The flags are updated only when the CC is in NORMAL_ACTIVE or NORMAL_PASSIVE state. If only one channel (A or B) is assigned to a message buffer, the channel-specific status flags of the other channel are written to zero. If both channels are assigned to a message buffer, the channel-specific status flags of both channels are updated. The message buffer status is updated only when the slot counter reached the configured frame ID and when the cycle counter filter matched. When the Host updates a message buffer via Input Buffer, all MBS flags are reset to zero independent of which IBCM bits are set or not. For details about receive / transmit filtering see Sections 19.7.Filtering and Masking, 19.8.Transmit Process, and 19.9.Receive Process. Whenever the Message Handler changes one of the flags **VFRA**, **VFRB**, **SEOA**, **SEOB**, **CEOA**, **CEOB**, **SVOA**, **SVOB**, **TCIA**, **TCIB**, **ESA**, **ESB**, **MLST**, **FTA**, **FTB** the respective message buffer's **MBC** flag in registers MBSC1/2/3/4 is set.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **MBS** | R | 0 | 0 | RESS | PPIS | NFIS | SYNS | SFIS | RCIS | 0 | 0 | CCS5 | CCS4 | CCS3 | CCS2 | CCS1 | CCS0 |
| 0x070C | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | FTB | FTA | 0 | MLST | ESB | ESA | TCIB | TCIA | SVOB | SVOA | CEOB | CEOA | SEOB | SEOA | VFRB | VFRA |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**VFRA** Valid Frame Received on Channel A (vSS!ValidFrameA)
A valid frame indication is set if a valid frame was received on channel A.
1 =Valid frame received on channel A
0 =No valid frame received on channel A

**VFRB** Valid Frame Received on Channel B (vSS!ValidFrameB)
A valid frame indication is set if a valid frame was received on channel B.
1 =Valid frame received on channel B
0 =No valid frame received on channel B

**SEOA** Syntax Error Observed on Channel A (vSS!SyntaxErrorA)
A syntax error was observed in the assigned slot on channel A.
1 =Syntax error observed on channel A
0 =No syntax error observed on channel A

**SEOB** Syntax Error Observed on Channel B (vSS!SyntaxErrorB)
A syntax error was observed in the assigned slot on channel B.
1 =Syntax error observed on channel B
0 =No syntax error observed on channel B

**CEOA** Content Error Observed on Channel A (vSS!ContentErrorA)
A content error was observed in the assigned slot on channel A.
1 =Content error observed on channel A
0 =No content error observed on channel A

**CEOB** Content Error Observed on Channel B (vSS!ContentErrorB)
A content error was observed in the assigned slot on channel B.
1 =Content error observed on channel B
0 =No content error observed on channel B

**SVOA** Slot Boundary Violation Observed on Channel A (vSS!BViolationA)
A slot boundary violation (channel active at the start or at the end of the assigned slot) was observed on channel A.
1 =Slot boundary violation observed on channel A
0 =No slot boundary violation observed on channel A

**SVOB** Slot Boundary Violation Observed on Channel B (vSS!BViolationB)
A slot boundary violation (channel active at the start or at the end of the assigned slot) was observed on channel B.
1 =Slot boundary violation observed on channel B
0 =No slot boundary violation observed on channel B

**TCIA** Transmission Conflict Indication Channel A (vSS!TxConflictA)
A transmission conflict indication is set if a transmission conflict has occurred on channel A.
1 =Transmission conflict occurred on channel A
0 =No transmission conflict occurred on channel A

**TCIB** Transmission Conflict Indication Channel B (vSS!TxConflictB)
A transmission conflict indication is set if a transmission conflict has occurred on channel B.
1 =Transmission conflict occurred on channel B
0 =No transmission conflict occurred on channel B

**ESA** Empty Slot Channel A
In an empty slot there is no activity detected on the bus. The condition is checked in static and dynamic slots.
1 =No bus activity detected in the assigned slot on channel A
0 =Bus activity detected in the assigned slot on channel A

**ESB** Empty Slot Channel B
In an empty slot there is no activity detected on the bus. The condition is checked in static and dynamic slots.
1 =No bus activity detected in the assigned slot on channel B
0 =Bus activity detected in the assigned slot on channel B

**MLST** Message Lost
The flag is set in case the Host did not read the message before the message buffer was updated from a received data frame. Not affected by reception of null frames except for message buffers belonging to the receive FIFO. The flag is reset by a Host write to the message buffer via IBF or when a new message is stored into the message buffer **after** the message buffers **ND** flag was reset by reading out the message buffer via OBF.
1 =Unprocessed message was overwritten
0 =No message lost

**FTA** Frame Transmitted on Channel A
Indicates that this node has transmitted a data frame in the configured slot on channel A.
1 =Data frame transmitted on channel A
0 =No data frame transmitted on channel A

**FTB** Frame Transmitted on Channel B
Indicates that this node has transmitted a data frame in the configured slot on channel B.
1 =Data frame transmitted on channel B
0 =No data frame transmitted on channel B

The FlexRay protocol specification requires that **FTA**, and **FTB** can only be reset by the Host. Therefore the Cycle Count Status **CCS[5:0]** for these bits is only valid for the cycle where the bits are set to ’1’.

**CCS[5:0]** Cycle Count Status
Actual cycle count when status was updated.
The following status bits are updated from both valid data and null frames. If no valid frame was received, the previous value is maintained.

**RCIS** Received on Channel Indicator Status (vSS!Channel)
Indicates the channel on which the frame was received.
1 =Frame received on channel A
0 =Frame received on channel B

**SFIS** Startup Frame Indicator Status (vRF!Header!SuFIndicator)
A startup frame is marked by the startup frame indicator.
1 =The received frame is a startup frame
0 =No startup frame received

**SYNS** Sync Frame Indicator Status (vRF!Header!SyFIndicator)
A sync frame is marked by the sync frame indicator.
1 =The received frame is a sync frame
0 =No sync frame received

**NFIS** Null Frame Indicator Status (vRF!Header!NFIndicator)
If set to '0' the payload segment of the received frame contains no usable data.
1 =Received frame is **not** a null frame
0 =Received frame is a null frame

**PPIS** Payload Preamble Indicator Status (vRF!Header!PPIndicator)
The payload preamble indicator defines whether a network management vector or message ID is contained within the payload segment of the received frame.
1 =Static segment: Network management vector at the beginning of the payload
Dynamic segment: Message ID at the beginning of the payload
0 =The payload segment of the received frame does not contain a network management vector or a message ID

**RESS** Reserved Bit Status (vRF!Header!Reserved)
Reflects the state of the received reserved bit. The reserved bit is transmitted as '0'.

### *18.11.6. Output Buffer Command Mask (OBCM)*

Configures how the Output Buffer is updated from the message buffer in the Message RAM selected by register OBCR. When OBF Host and OBF Shadow are swapped, also mask bits **RDSH** and **RHSH** are swapped with bits **RDSS** and **RHSS** to keep them attached to the respective Output Buffer transfer.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **OBCM** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RDSH | RHSH |
| 0x0710 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RDSS | RHSS |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**RHSS** Read Header Section Shadow
1 =Header section selected for transfer from Message RAM to Output Buffer
0 =Header section is not read

**RDSS** Read Data Section Shadow
1 =Data section selected for transfer from Message RAM to Output Buffer
0 =Data section is not read

**RHSH** Read Header Section Host
1 =Header section selected for transfer from Message RAM to Output Buffer
0 =Header section is not read

**RDSH** Read Data Section Host
1 =Data section selected for transfer from Message RAM to Output Buffer
0 =Data section is not read

Note: After the transfer of the header section from the Message RAM to OBF Shadow has completed, the message buffer status changed flag **MBC** of the selected message buffer in the MBSC1/2/3/4 registers is cleared. After the transfer of the data section from the Message RAM to OBF Shadow has completed, the new data flag **ND** of the selected message buffer in the NDAT1/2/3/4 registers is cleared.

### *18.11.7. Output Buffer Command Request (OBCR)*

The message buffer selected by **OBRS[6:0]** is transferred from the Message RAM to the Output Buffer as soon as the Host has set **REQ** to '1'. Bit **REQ** can only be set to '1' while **OBSYS** is '0' (see also Section Data Transfer from Message RAM to Output Buffer).

After setting **REQ** to '1', **OBSYS** is automatically set to '1', and the transfer of the message buffer selected by **OBRS[6:0]** from the Message RAM to OBF Shadow is started. When the transfer between the Message RAM and OBF Shadow has completed, this is signalled by setting **OBSYS** back to '0'. By setting the **VIEW** bit to '1' while **OBSYS** is '0', OBF Host and OBF Shadow are swapped. Now the Host can read the transferred message buffer from OBF Host. In parallel the Message Handler may transfer the next message from the Message RAM to OBF Shadow if **VIEW** and **REQ** are set at the same time.

Any write access to an Output Buffer register while **OBSYS** is set will cause the error flag **EIR.IOBA** to be set. In this case the Output Buffer will not be changed.

| Bit | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **OBCR** | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | OBRH6 | OBRH5 | OBRH4 | OBRH3 | OBRH2 | OBRH1 | OBRH0 |
| 0x0714 | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R | OBSYS | 0 | 0 | 0 | 0 | 0 | REQ | VIEW | 0 | OBRS6 | OBRS5 | OBRS4 | OBRS3 | OBRS2 | OBRS1 | OBRS0 |
| | W | | | | | | | | | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**OBRS[6:0]** Output Buffer Request Shadow

Number of source message buffer to be transferred from the Message RAM to OBF Shadow. Valid values are 0x00 to 0x7F (0…127). If the number of the first message buffer of the receive FIFO is written to this register the Message Handler transfers the message buffer addressed by the GET Index (GIDX, see Section 19.10.FIFO Function) to OBF Shadow.

**VIEW** View Shadow Buffer

Toggles between OBF Shadow and OBF Host. Only writable while **OBSYS** = '0'.

1 =Swap OBF Shadow and OBF Host

0 =No action

**REQ** Request Message RAM Transfer

Requests transfer of message buffer addressed by **OBRS[6:0]** from Message RAM to OBF Shadow. Only writable while **OBSYS** = '0'.

1 =Transfer to OBF Shadow requested

0 =No request

**OBSYS** Output Buffer Busy Shadow

Set to '1' after setting bit **REQ**. When the transfer between the Message RAM and OBF Shadow has completed, **OBSYS** is set back to '0'.

1 =Transfer between Message RAM and OBF Shadow in progress

0 =No transfer in progress

**OBRH[6:0]** Output Buffer Request Host

Number of message buffer currently accessible by the Host via RDHS[1…3], MBS, and RDDS[1…64]. By writing **VIEW** to '1' OBF Shadow and OBF Host are swapped and the transferred message buffer is accessible by the Host. Valid values are 0x00 to 0x7F (0…127).

# 19. Functional Description

This chapter describes the E-Ray implementation together with the related FlexRay protocol features. More information about the FlexRay protocol itself can be found in the FlexRay protocol specification v2.1.

Communication on FlexRay networks is based on frames and symbols. The wakeup symbol (WUS) and the collision avoidance symbol (CAS) are transmitted outside the communication cycle to setup the time schedule. Frames and media access test symbols (MTS) are transmitted inside the communication cycle.

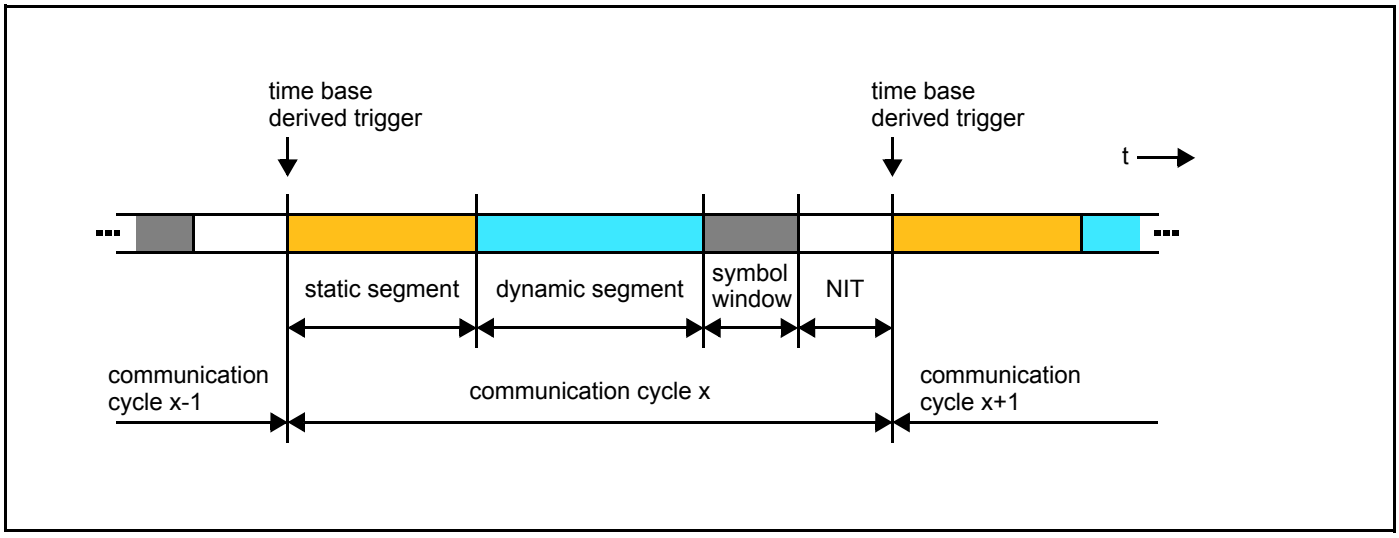
## 19.1. Communication Cycle

A FlexRay communication cycle consists of the following elements:

- Static Segment
- Dynamic Segment (optional)
- Symbol Window (optional)
- Network Idle Time (NIT)

Static segment, dynamic segment, and symbol window form the Network Communication Time (NCT). For each communication channel the slot counter starts at 1 and counts up until the end of the dynamic segment is reached. Both channels share the same arbitration grid which means that they use the same synchronized macrotick.

### Structure of communication cycle




*19.1.1. Static Segment*

The Static Segment is characterized by the following features:

- Time slots of fixed length (optionally protected by bus guardian)
- Start of frame transmission at action point of the respective static slot
- Payload length same for all frames on both channels

**Parameters**: Number of Static Slots **GTUC7.NSS[9:0]**,
Static Slot Length **GTUC7.SSL[9:0]**,
Payload Length Static **MHDC.SFDL[6:0]**,
Action Point Offset **GTUC9.APO[5:0]**

*19.1.2. Dynamic Segment*

The Dynamic Segment is characterized by the following features:

- All controllers have bus access (no bus guardian protection possible)
- Variable payload length and duration of slots, different for both channels
- Start of transmission at minislot action point

Parameters: Number of Minislots **GTUC8.NMS[12:0]**,
Minislot Length **GTUC8.MSL[5:0]**,
Minislot Action Point Offset **GTUC9.MAPO[4:0]**,
Start of Latest Transmit (last minislot) **MHDC.SLT[12:0]**

### *19.1.3. Symbol Window*

During the symbol window only one media access test symbol (MTS) may be transmitted per channel. MTS symbols are send in NORMAL_ACTIVE state to test the bus guardian.

The symbol window is characterized by the following features:

- Send single symbol
- Transmission of the MTS symbol starts at the symbol windows action point

Parameters: Symbol Window Action Point Offset **GTUC9.APO[4:0]** (same as for static slots),
Network Idle Time Start **GTUC4.NIT[13:0]**

### *19.1.4. Network Idle Time (NIT)*

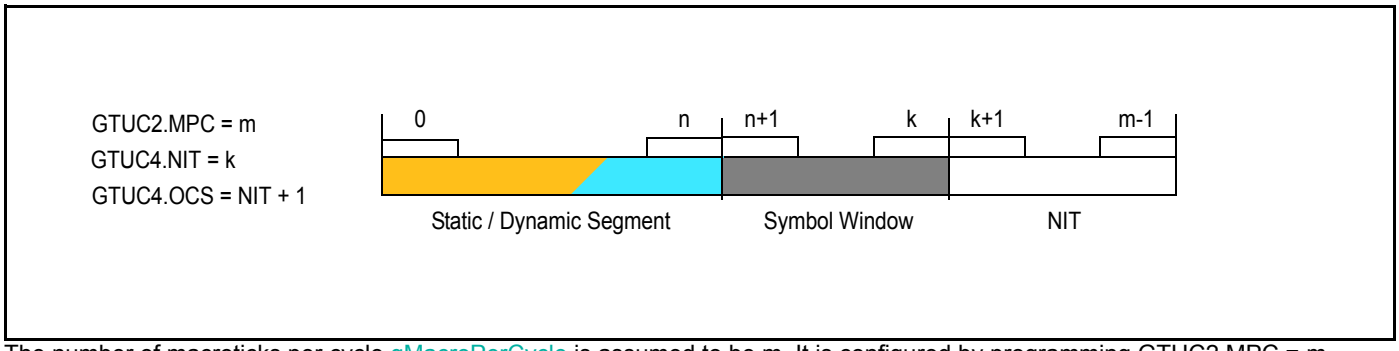
During network idle time the CC has to perform the following tasks:

- Calculate clock correction terms (offset and rate)
- Distribute offset correction over multiple macroticks after offset correction start
- Perform cluster cycle related tasks

Parameters: Network Idle Time Start **GTUC4.NIT[13:0]**,
Offset Correction Start **GTUC4.OCS[13:0]**

### *19.1.5. Configuration of NIT Start and Offset Correction Start*

**Configuration of NIT start and offset correction start**




The number of macroticks per cycle gMacroPerCycle is assumed to be m. It is configured by programming GTUC2.MPC = m.
The static / dynamic segment starts with macrotick 0 and ends with macrotick n:
n = static segment length + dynamic segment offset + dynamic segment length - 1MT
n = gNumberOfStaticSlots • gdStaticSlot + dynamic segment offset +
gNumberOfMinislots • gdMinislot - 1 MT

The static segment length is configured by **GTUC7.SSL** and **GTUC7.NSS**.
The dynamic segment length is configured by **GTUC8.MSL** and **GTUC8.NMS**.

The dynamic segment offset is:
If gdActionPointOffset ≤ gdMinislotActionPointOffset:
dynamic segment offset = 0 MT
Else if gdActionPointOffset > gdMinislotActionPointOffset:
dynamic segment offset = gdActionPointOffset - gdMinislotActionPointOffset

The NIT starts with macrotick k+1 and ends with the last macrotick of cycle m-1. It has to be configured by setting **GTUC4.NIT** = k.

For the E-Ray the offset correction start is required to be **GTUC4.OCS** ≥ **GTUC4.NIT** + 1 = k+1.

The length of symbol window results from the number of macroticks between the end of the static / dynamic segment and the beginning of the NIT. It can be calculated by k - n.

## 19.2. Communication Modes

The FlexRay Protocol Specification v2.1 defines the Time-Triggered Distributed (TT-D) mode.

### *19.2.1. Time-triggered Distributed (TT-D)*

In TT-D mode the following configurations are possible:

- **Pure static**: Minimum 2 static slots + symbol window (optional)
- **Mixed static/dynamic**: Minimum 2 static slots + dynamic segment + symbol window (optional)

A minimum of two coldstart nodes needs to be configured for distributed time-triggered operation. Two fault-free coldstart nodes are necessary for the cluster startup. Each startup frame must be a sync frame, therefore all coldstart nodes are sync nodes.

## 19.3. Clock Synchronization

In TT-D mode a distributed clock synchronization is used. Each node individually synchronizes itself to the cluster by observing the timing of received sync frames from other nodes.

### *19.3.1. Global Time*

Activities in a FlexRay node, including communication, are based on the concept of a global time, even though each individual node maintains its own view of it. It is the clock synchronization mechanism that differentiates the FlexRay cluster from other node collections with independent clock mechanisms. The global time is a vector of two values; the cycle (cycle counter) and the cycle time (macrotick counter).

**Cluster specific:**

- Macrotick (MT) = basic unit of time measurement in a FlexRay network, a macrotick consists of an integer number of microticks ($\mu$T)
- Cycle length = duration of a communication cycle in units of macroticks (MT)

### *19.3.2. Local Time*

Internally, nodes time their behaviour with microtick resolution. Microticks are time units derived from the oscillator clock tick of the specific node. Therefore microticks are controller-specific units. They may have different duration in different controllers. The precision of a node's local time difference measurements is a microtick ($\mu$T).

**Node specific:**

- Oscillator clock -> prescaler -> microtick ($\mu$T)
- mT = basic unit of time measurement in a CC, clock correction is done in units of mTs
- Cycle counter + macrotick counter = nodes local view of the global time

### *19.3.3. Synchronization Process*

Clock synchronization is performed by means of sync frames. Only preconfigured nodes (sync nodes) are allowed to send sync frames. In a two-channel cluster a sync node has to send its sync frame on both channels.

For synchronization in FlexRay the following constraints have to be considered:

- Maximum of one sync frame per node in one communication cycle
- Maximum of 15 sync frames per cluster in one communication cycle
- Every node has to use a preconfigured number of sync frames (GTUC2.SNM[3:0]) for clock synchronization
- Minimum of two sync nodes required for clock synchronization and startup

For clock synchronization the time difference between expected and observed arrival time of sync frames received during the static segment is measured. In a two channel cluster the sync node has to be configured to send sync frames on both channels. The calculation of correction terms is done during NIT (offset: every cycle, rate: every odd cycle) by using an FTM algorithm. For details see FlexRay protocol specification v2.1, chapter 8.

*Offset (phase) Correction*

- Only deviation values measured and stored in the current cycle used
- For a two channel node the smaller value will be taken
- Calculation during NIT of every communication cycle
- Offset correction value calculated in even cycles used for error checking only
- Checked against limit values
- Correction value is a signed integer number of mTs
- Correction done in odd numbered cycles, distributed over the macroticks beginning at offset correction start up to cycle end (end of NIT) to shift nodes next start of cycle (MTs lengthened / shortened)

*Rate (frequency) Correction*

- Pairs of deviation values measured and stored in even / odd cycle pair used
- For a two channel node the average of the differences from the two channels is used
- Calculated during NIT of odd numbered cycles
- Cluster drift damping is performed using global damping value
- Checked against limit values
- Correction value is a signed integer number of mTs
- Distributed over macroticks comprising the next even / odd cycle pair (MTs lengthened / shortened)

*Sync Frame Transmission*

Sync frame transmission is only possible from buffer 0 and 1. Message buffer 1 may be used for sync frame transmission in case that sync frames should have different payloads on the two channels. In this case bit **MRC.SPLM** has to be programmed to ’1’.

Message buffers used for sync frame transmission have to be configured with the key slot ID and can be (re)configured in DEFAULT_CONFIG or CONFIG state only. For nodes transmitting sync frames **SUCC1.TXSY** must be set to ’1’.

### *19.3.4. External Clock Synchronization*

During normal operation, independent clusters can drift significantly. If synchronous operation across independent clusters is desired, external synchronization is necessary; even though the nodes within each cluster are synchronized. This can be accomplished with synchronous application of host-deduced rate and offset correction terms to the clusters.

- External offset / rate correction value is a signed integer
- External offset / rate correction value is added to calculated offset / rate correction value
- Aggregated offset / rate correction term (external + internal) is not checked against configured limits

## 19.4. Error Handling

The implemented error handling concept is intended to ensure that, in case of a lower layer protocol error in one single node, communication between non-affected nodes can be maintained. In some cases, higher layer program activity is required for the CC to resume normal operation. A change of the error handling state will set **EIR.PEMC** and may trigger an interrupt to the Host if enabled. The actual error mode is signalled by **CCEV.ERRM[1:0]**.

### Error modes of the POC (degradation model)

| Error Mode | Activity |
|---|---|
| ACTIVE (green) | **Full operation**, State: NORMAL_ACTIVE<br>The CC is fully synchronized and supports the cluster wide clock synchronization. The host is informed of any error condition(s) or status change by interrupt (if enabled) or by reading the error and status interrupt flags from registers EIR and SIR. |
| PASSIVE (yellow) | **Reduced operation**, State: NORMAL_PASSIVE, CC self rescue allowed<br>The CC stops transmitting frames and symbols, but received frames are still processed. Clock synchronization mechanisms are continued based on received frames. No active contribution to the cluster wide clock synchronization. The host is informed of any error condition(s) or status change by interrupt (if enabled) or by reading the error and status interrupt flags from registers EIR and SIR. |
| COMM_HALT (red) | **Operation halted**, State: HALT, CC self rescue not allowed<br>The CC stops frame and symbol processing, clock synchronization processing, and the macrotick generation. The host has still access to error and status information by reading the error and status interrupt flags from registers EIR and SIR. The bus drivers are disabled. |

#### *19.4.1. Clock Correction Failed Counter*

When the Clock Correction Failed Counter reaches the "maximum without clock correction passive" limit defined by **SUCC3.WCP[3:0]**, the POC transits from NORMAL_ACTIVE to NORMAL_PASSIVE state. When it reaches the "maximum without clock correction fatal" limit defined by **SUCC3.WCF[3:0]**, it transits from NORMAL_ACTIVE or NORMAL_PASSIVE to HALT state.

The Clock Correction Failed Counter **CCEV.CCFC[3:0]** allows the Host to monitor the duration of the inability of a node to compute clock correction terms after the CC passed protocol startup phase. It will be incremented by one at the end of any odd communication cycle during which either the missing offset correction **SFS.MOCS** or the missing rate correction **SFS.MRCS** flag is set.

The Clock Correction Failed Counter is reset to zero at the end of an odd communication cycle if neither the missing offset correction **SFS.MOCS** nor the missing rate correction **SFS.MRCS** flag is set.

The Clock Correction Failed Counter stops incrementing when the "maximum without clock correction fatal" value **SUCC3.WCF[3:0]** is reached (i.e. incrementing the counter at its maximum value will not cause it to wrap around back to zero). The Clock Correction Failed Counter is initialized to zero when the CC enters READY state or when NORMAL_ACTIVE state is entered.

#### *19.4.2. Passive to Active Counter*

The passive to active counter controls the transition of the POC from NORMAL_PASSIVE to NORMAL_ACTIVE state. **SUCC1.PTA[4:0]** defines the number of consecutive even / odd cycle pairs that must have valid clock correction terms before the CC is allowed to transit from NORMAL_PASSIVE to NORMAL_ACTIVE state. If **SUCC1.PTA[4:0]** is set to zero the CC is not allowed to transit from NORMAL_PASSIVE to NORMAL_ACTIVE state.

#### *19.4.3. HALT Command*

In case the Host wants to stop FlexRay communication of the local node it can bring the CC into HALT state by asserting the HALT command. This can be done by writing SUCC1.CMD[3:0] = "0110". In order to shut down communication on an entire FlexRay network, a higher layer protocol is required to assure that all nodes apply the HALT command at the same time.

The POC state from which the transition to HALT state took place can be read from **CCSV.PSL[5:0]**.

When called in NORMAL_ACTIVE or NORMAL_PASSIVE state the POC transits to HALT state at the end of the current cycle. When called in any other state **SUCC1.CMD[3:0]** will be reset to "0000" = command_not_accepted and bit **EIR.CNA** is set to '1'. If enabled an interrupt to the Host is generated.

#### *19.4.4. FREEZE Command*

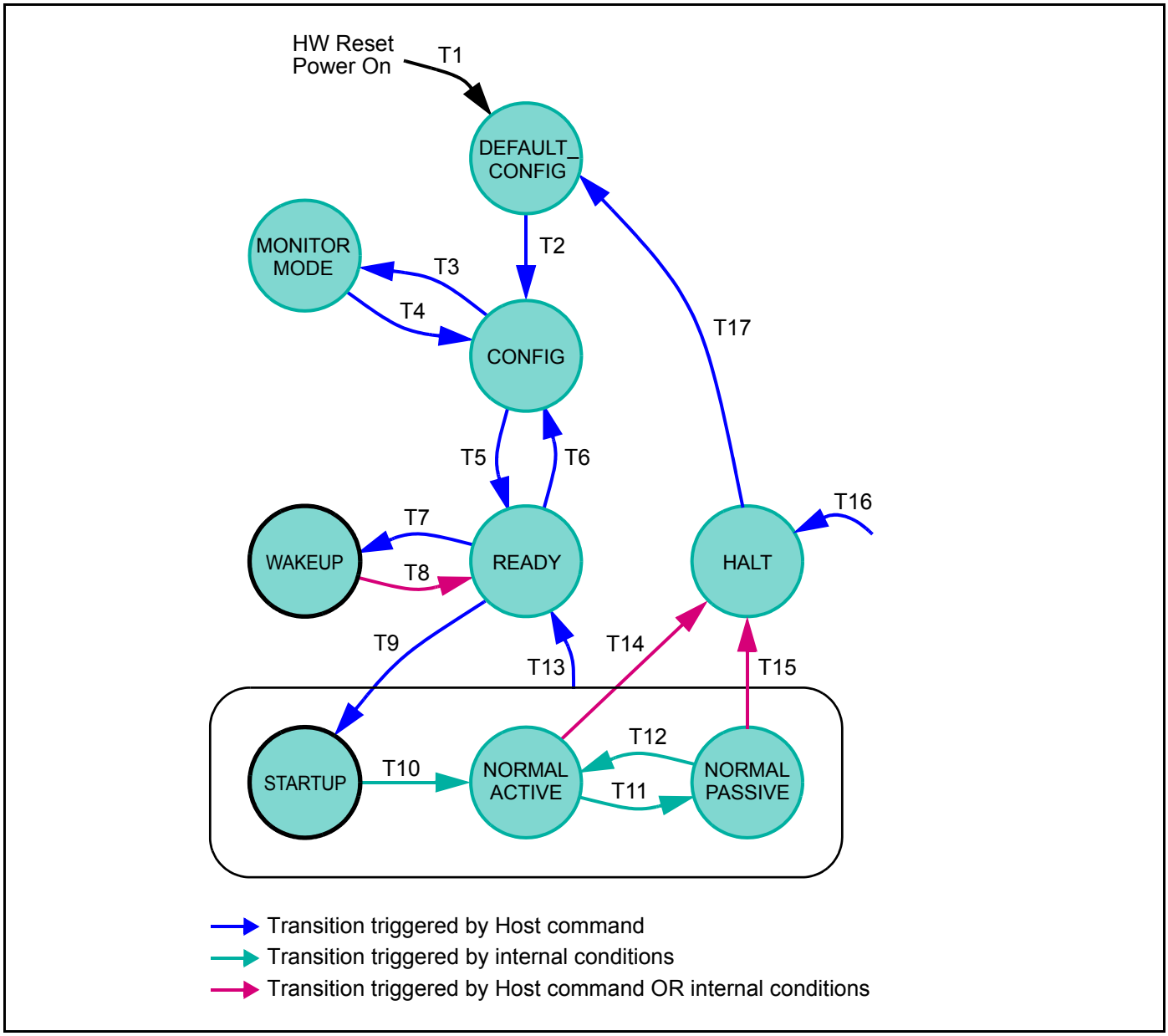
In case the Host detects a severe error condition it can bring the CC into HALT state by asserting the FREEZE command. This can be done by writing **SUCC1.CMD[3:0]** = "0111". The FREEZE command triggers the entry of the HALT state immediately regardless of the actual POC state.

The POC state from which the transition to HALT state took place can be read from **CCSV.PSL[5:0]**.

## 19.5. Communication Controller States

*19.5.1. Communication Controller State Diagram*

**Overall state diagram of E-Ray communication controller**



State transitions are controlled by externals pins **eray_reset** and **eray_rxd1**,**2**, by the POC state machine, and by the CHI Command Vector **SUCC1.CMD[3:0]**.
The CC exits from all states to HALT state after application of the FREEZE command **(SUCC1.CMD[3:0]** = "0111")
**State transitions of E-Ray overall state machine**

| T# | Condition | From | To |
|---|---|---|---|
| 1 | Hard reset | All States | DEFAULT_CONFIG |
| 2 | Command CONFIG, **SUCC1.CMD[3:0]** = "0001" | DEFAULT_CONFIG | CONFIG |
| 3 | Unlock sequence followed by command MONITOR_MODE, **SUCC1.CMD[3:0]** = "1011" | CONFIG | MONITOR_MODE |
| 4 | Command CONFIG, **SUCC1.CMD[3:0]** = "0001" | MONITOR_MODE | CONFIG |
| 5 | Unlock sequence followed by command READY, **SUCC1.CMD[3:0]** = "0010" | CONFIG | READY |
| 6 | Command CONFIG, **SUCC1.CMD[3:0]** = "0001" | READY | CONFIG |
| 7 | Command WAKEUP, **SUCC1.CMD[3:0]** = "0011" | READY | WAKEUP |
| 8 | Complete, non-aborted transmission of wakeup pattern OR received WUP OR received frame header OR wakeup collision OR command READY, **SUCC1.CMD[3:0]** = "0010" | WAKEUP | READY |
| 9 | Command RUN, **SUCC1.CMD[3:0]** = "0100" | READY | STARTUP |
| 10 | Successful startup | STARTUP | NORMAL_ACTIVE |
| 11 | Clock Correction Failed counter reached Maximum Without Clock Correction Passive limit configured by **SUCC3.WCP[3:0]** | NORMAL_ACTIVE | NORMAL_PASSIVE |
| 12 | Number of valid correction terms reached the Passive to Active limit configured by **SUCC1.PTA[4:0]** | NORMAL_PASSIVE | NORMAL_ACTIVE |
| 13 | Command READY, **SUCC1.CMD[3:0]** = "0010" | STARTUP, NORMAL_ACTIVE, NORMAL_PASSIVE | READY |
| 14 | Clock Correction Failed counter reached Maximum Without Clock Correction Fatal limit configured by **SUCC3.WCF[3:0]** AND bit **SUCC1.HCSE** set to ’1’ OR command HALT, **SUCC1.CMD[3:0]** = "0110" | NORMAL_ACTIVE | HALT |
| 15 | Clock Correction Failed counter reached Maximum Without Clock Correction Fatal limit configured by **SUCC3.WCF[3:0]** AND bit **SUCC1.HCSE** set to ’1’ OR command HALT, **SUCC1.CMD[3:0]** = "0110" | NORMAL_PASSIVE | HALT |
| 16 | Command FREEZE, **SUCC1.CMD[3:0]** = "0111" | All States | HALT |
| 17 | Command CONFIG, **SUCC1.CMD[3:0]** = "0001" | HALT | DEFAULT_CONFIG |

### 19.5.2. DEFAULT_CONFIG State

In DEFAULT_CONFIG state, the CC is stopped. All configuration registers are accessible and the pins to the physical layer are in their inactive state.

The CC enters this state

- When leaving hard reset (external reset signal **eray_reset** is deactivated)
- When exiting from HALT state

To leave DEFAULT_CONFIG state the Host has to write **SUCC1.CMD[3:0]** = "0001". The CC then transits to CONFIG state.

### 19.5.3. CONFIG State

In CONFIG state, the CC is stopped. All configuration registers are accessible and the pins to the physical layer are in their inactive state. This state is used to initialize the CC configuration.

The CC enters this state

- When exiting from DEFAULT_CONFIG state
- When exiting from MONITOR_MODE or READY state

When the state has been entered via HALT and DEFAULT_CONFIG state, the Host can analyse status information and configuration. Before leaving CONFIG state the Host has to assure that the configuration is fault-free.

To leave CONFIG state, the Host has to perform the unlock sequence as described in 18.3.3.Lock Register (LCK). Directly after unlocking the CONFIG state the Host has to write SUCC1.CMD[3:0] to enter the next state.

Internal counters and the CC status flags are reset when the CC leaves CONFIG state.

Note: Status bits MHDS[14:0], registers TXRQ1/2/3/4, and status data stored in the Message RAM are not affected by the transition of the POC from CONFIG to READY state.

When the CC is in CONFIG state it is also possible to bring the CC into a power saving mode by halting the module clocks (**eray_sclk, eray_bclk**). To do this the Host has to assure that all Message RAM transfers have finished before turning off the clocks.

### 19.5.4. MONITOR_MODE

After unlocking CONFIG state and writing **SUCC1.CMD[3:0]** = "1011" the CC enters MONITOR_MODE. In this mode the CC is able to receive FlexRay frames. The temporal integrity of received frames is not checked, and therefore cycle counter filtering is not supported. This mode can be used for debugging purposes in case e.g. that startup of a FlexRay network fails. After writing **SUCC1.CMD[3:0]** = "0001" the CC transits back to CONFIG state.

In MONITOR_MODE the pick first valid mechanism is disabled. This means that a receive message buffer may only be configured to receive on one channel. Received frames are stored into message buffers according to frame ID and receive channel. Null frames are handled like data frames. After frame reception only status bits **MBS.VFRA, MBS.VFRB, MBS.MLST, MBS.RCIS, MBS.SFIS, MBS.SYNS, MBS.NFIS, MBS.PPIS, MBS.RESS** have valid values. In MONITOR_MODE the receive FIFO is not available.

### 19.5.5. READY State

After unlocking CONFIG state and writing **SUCC1.CMD[3:0]** = "0010" the CC enters READY state. From this state the CC can transit to WAKEUP state and perform a cluster wakeup or to STARTUP state to perform a coldstart or to integrate into a running cluster.

The CC enters this state

- When exiting from CONFIG, WAKEUP, STARTUP, NORMAL_ACTIVE, or NORMAL_PASSIVE state by writing **SUCC1.CMD[3:0]** = "0010" (READY command).

The CC exits from this state

- To CONFIG state by writing **SUCC1.CMD[3:0]** = "0001" (CONFIG command)
- To WAKEUP state by writing **SUCC1.CMD[3:0]** = "0011" (WAKEUP command)
- To STARTUP state by writing **SUCC1.CMD[3:0]** = "0100" (RUN command)

Internal counters and the CC status flags are reset when the CC enters STARTUP state.

Note: Status bits MHDS[14:0], registers TXRQ1/2/3/4, and status data stored in the Message RAM are not affected by the transition of the POC from READY to STARTUP state.

### *19.5.6. WAKEUP State*

The description below is intended to help configuring wakeup for the E-Ray IP-module. A detailed description of the wakeup procedure together with the respective SDL diagrams can be found in the FlexRay protocol specification v2.1, section 7.1.

The CC enters this state

- When exiting from READY state by writing SUCC1.CMD[3:0] = "0011" (WAKEUP command).

The CC exits from this state to READY state

- After complete non-aborted transmission of wakeup pattern
- After WUP reception
- After detecting a WUP collision
- After reception of a frame header
- By writing **SUCC1.CMD[3:0]** = "0010" (READY command)

The cluster wakeup must precede the communication startup in order to ensure that all nodes in a cluster are awake. The minimum requirement for a cluster wakeup is that all bus drivers are supplied with power. A bus driver has the ability to wake up the other components of its node when it receives a wakeup pattern on its channel. At least one node in the cluster needs an **external** wakeup source.

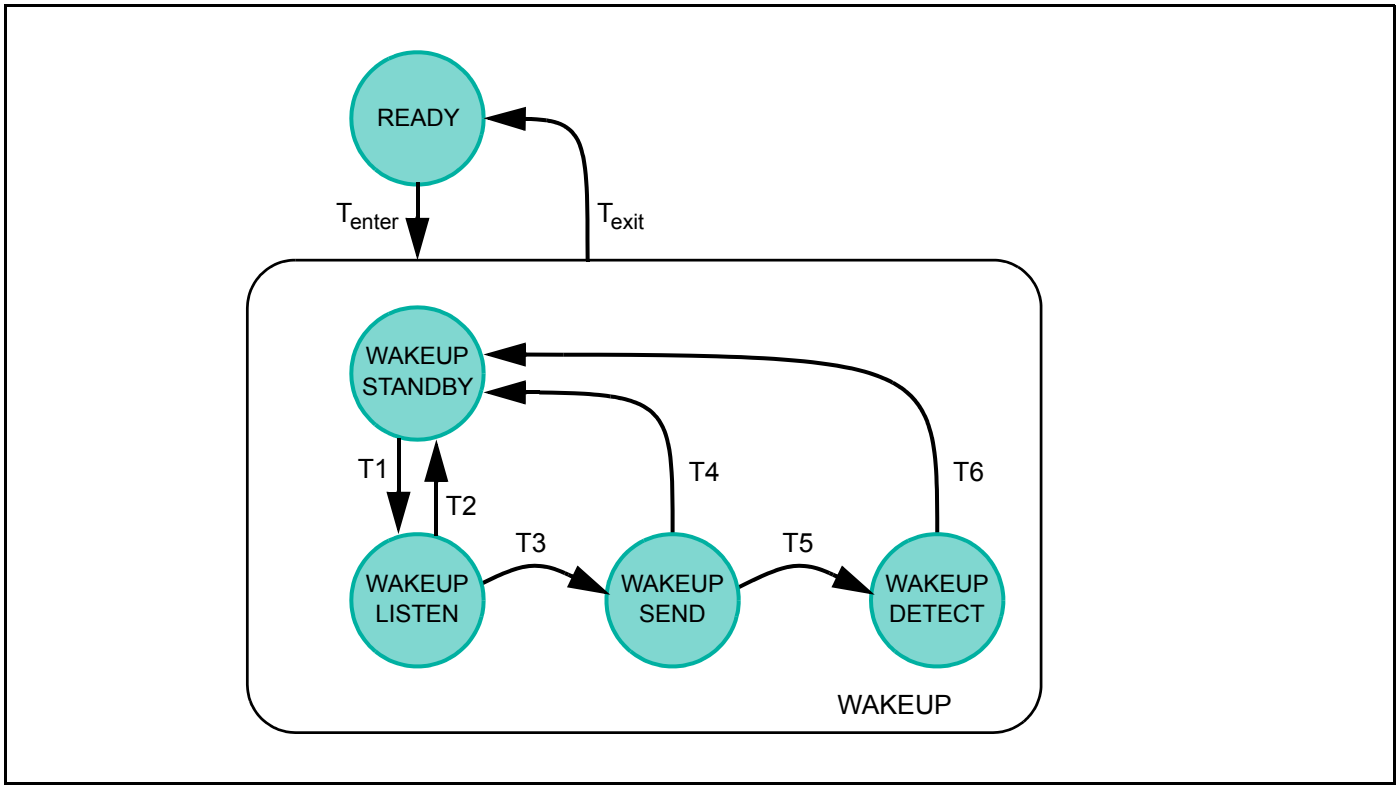
The Host completely controls the wakeup procedure. It is informed about the state of the cluster by the bus driver and the CC and configures bus guardian (if available) and CC to perform the cluster wakeup. The CC provides to the Host the ability to transmit a special wakeup pattern on each of its available channels separately. The CC needs to recognize the wakeup pattern only during WAKEUP state.

Wakeup may be performed on only one channel at a time. The Host has to configure the wakeup channel while the CC is in CONFIG state by writing SUCC1.WUCS. The CC ensures that ongoing communication on this channel is not disturbed. The CC cannot guarantee that all nodes connected to the configured channel awake upon the transmission of the wakeup pattern, since these nodes cannot give feedback until the startup phase. The wakeup procedure enables single-channel devices in a two-channel system to trigger the wakeup, by only transmitting the wakeup pattern on the single channel to which they are connected. Any coldstart node that deems a system startup necessary will then wake the remaining channel before initiating communication startup.

The wakeup procedure tolerates any number of nodes simultaneously trying to wakeup a single channel and resolves this situation such that only one node transmits the pattern. Additionally the wakeup pattern is collision resilient, so even in the presence of a fault causing two nodes to simultaneously transmit a wakeup pattern, the resulting collided signal can still wake the other nodes.

After wakeup the CC returns to READY state and signals the change of the wakeup status to the Host by setting flag **SIR.WST**. The wakeup status vector can be read from **CCSV.WSV[2:0]**. If a valid wakeup pattern was received also either flag **SIR.WUPA** or flag **SIR.WUPB** is set.

### Structure of POC state WAKEUP




### State transitions WAKEUP

| T# | Condition | From | To |
|---|---|---|---|
| enter | Host commands change to WAKEUP state by writing **SUCC1.CMD[3:0]** = "0011" (WAKEUP command) | READY | WAKEUP |
| 1 | CHI command WAKEUP triggers wakeup FSM to transit to WAKEUP_LISTEN state | WAKEUP_STANDBY | WAKEUP_LISTEN |
| 2 | Received WUP on wakeup channel selected by bit **SUCC1.WUCS** OR frame header on either available channel | WAKEUP_LISTEN | WAKEUP_STANDBY |
| 3 | Timer event | WAKEUP_LISTEN | WAKEUP_SEND |
| 4 | Complete, non-aborted transmission of wakeup pattern | WAKEUP_SEND | WAKEUP_STANDBY |
| 5 | Collision detected | WAKEUP_SEND | WAKEUP_DETECT |
| 6 | Wakeup timer expired OR WUP detected on wakeup channel selected by bit **SUCC1.WUCS** OR frame header received on either available channel | WAKEUP_DETECT | WAKEUP_STANDBY |
| exit | Wakeup completed (after T2 or T4 or T6) OR Host commands change to READY state by writing **SUCC1.CMD[3:0]** = "0010" (READY command). This command also resets the wakeup FSM to WAKEUP_STANDBY state | WAKEUP | READY |

The WAKEUP_LISTEN state is controlled by the wakeup timer and the wakeup noise timer. The two timers are controlled by the parameters listen timeout **SUCC2.LT[20:0]** and listen timeout noise **SUCC2.LTN[3:0]**. Listen timeout enables a fast cluster wakeup in case of a noise free environment, while listen timeout noise enables wakeup under more difficult conditions regarding noise interference.

In WAKEUP_SEND state the CC transmits the wakeup pattern on the configured channel and checks for collisions. After return from wakeup the Host has to bring the CC into STARTUP state by CHI command RUN.

In WAKEUP_DETECT state the CC attempts to identify the reason for the wakeup collision detected in WAKEUP_SEND state. The monitoring is bounded by the expiration of listen timeout as configured by **SUCC2.LT[20:0]**. Either the detection of a wakeup pattern indicating a wakeup attempt by another node or the reception of a frame header indicating ongoing communication, causes the direct transition to READY state. Otherwise WAKEUP_DETECT is left after expiration of listen timeout; in this case the reason for wakeup collision is unknown.

The Host has to be aware of possible failures of the wakeup and act accordingly. It is advisable to delay any potential startup attempt of the node having instigated the wakeup by the minimal time it takes another coldstart node to become awake and to be configured.

The FlexRay Protocol Specification v2.1 recommends that two different CCs shall awake the two channels.

*Host activities*

The host must coordinate the wakeup of the two channels and must decide whether, or not, to wake a specific channel. The sending of the wakeup pattern is initiated by the Host. The wakeup pattern is detected by the remote BDs and signalled to their local Host.

### Wakeup procedure controlled by Host (single-channel wakeup):

- Configure the CC in CONFIG state
- Select wakeup channel by programming bit **SUCC1.WUCS**
- Check local BDs whether a WUP was received
- Activate BD of selected wakeup channel
- Command CC to enter READY state
- Command CC to start wakeup on the configured channel by writing **SUCC1.CMD[3:0]** = "0011"
- CC enters WAKEUP
- CC returns to READY state and signals status of wakeup attempt to the Host
- Wait predefined time to allow the other nodes to wakeup and configure themselves
- Coldstart node:
- In a dual channel cluster wait for WUP on the other channel
- Reset coldstart inhibit flag **CCSV.CSI** by writing **SUCC1.CMD[3:0]** = "1001" (ALLOW_COLDSTART command)
- Command CC to enter startup by writing **SUCC1.CMD[3:0]** = "0100" (RUN command)

### Wakeup procedure triggered by BD:

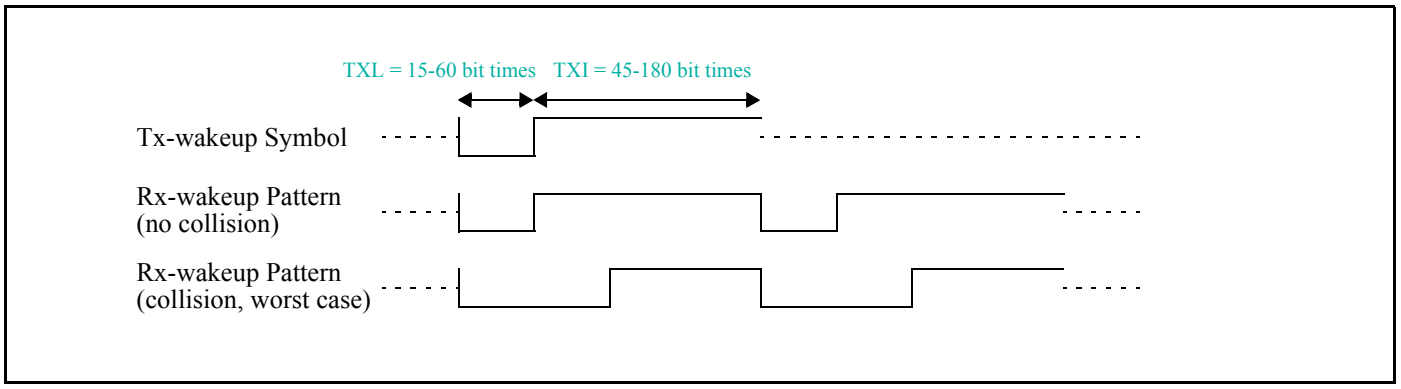
- Wakeup recognized by BD
- BD triggers power-up of Host (if required)
- BD signals wakeup event to Host
- Host configures its local CC
- If necessary, Host commands wakeup of second channel and waits predefined time to allow the other nodes to wakeup and configure themselves
- Host commands CC to enter STARTUP state by writing **SUCC1.CMD[3:0]** = "0100" (RUN command)

*Wakeup pattern (WUP)*

The wakeup pattern (WUP) is composed of at least two wakeup symbols (WUS). Wakeup symbol and wakeup pattern are configured by registers PRTC1 and PRTC2.

- Single channel wakeup, wakeup symbol may not be sent on both channels at the same time
- Wakeup symbol collision resilient for at least two sending nodes
  (two overlapping wakeup symbols always recognizable)
- Wakeup symbol must be configured identical in all nodes of a cluster
- Wakeup symbol transmit low time configured by **PRTC2.TXL[5:0]**
- Wakeup symbol idle time used to listen for activity on the bus, configured by **PRTC2.TXI[7:0]**
- A wakeup pattern composed of at least two Tx-wakeup symbols needed for wakeup
- Number of repetitions configurable by **PRTC1.RWP[5:0]** (2 to 63 repetitions)
- Wakeup symbol receive window length configured by **PRTC1.RXW[8:0]**
- Wakeup symbol receive low time configured by **PRTC2.RXL[5:0]**
- Wakeup symbol receive idle time configured by **PRTC2.RXI[5:0]**

### Timing of wakeup pattern



### 19.5.7. STARTUP State

The description below is intended to help configuring startup for the E-Ray IP-module. A detailed description of the startup procedure together with the respective SDL diagrams can be found in the FlexRay protocol specification v2.1, section 7.2.

Any node entering STARTUP state that has coldstart capability should assure that both channels attached have been awakened before initiating coldstart.

It cannot be assumed that all nodes and stars need the same amount of time to become completely awake and to be configured. Since at least two nodes are necessary to start up the cluster communication, it is advisable to delay any potential startup attempt of the node having instigated the wakeup by the minimal amount of time it takes another coldstart node to become awake, to be configured and to enter startup. It may require several hundred milliseconds (depending on the hardware used) before all nodes and stars are completely awakened and configured.

Startup is performed on all channels synchronously. During startup, a node only transmits startup frames. Startup frames are both sync frames and null frames during startup.

A fault-tolerant, distributed startup strategy is specified for initial synchronization of all nodes. In general, a node may enter NORMAL_ACTIVE state via (see figureState diagram time-triggered startup):

- Coldstart path initiating the schedule synchronization (leading coldstart node)
- Coldstart path joining other coldstart nodes (following coldstart node)
- Integration path integrating into an existing communication schedule (all other nodes)

A coldstart attempt begins with the transmission of a collision avoidance symbol (CAS). Only a coldstart node that had transmitted the CAS transmits frames in the first four cycles after the CAS, it is then joined firstly by the other coldstart nodes and afterwards by all other nodes.

A coldstart node has bits **SUCC1.TXST** and **SUCC1.TXSY** set to ’1’. Message buffer 0 holds the key slot ID which defines the slot number where the startup frame is send. In the frame header of the startup frame the startup frame indicator bit is set.

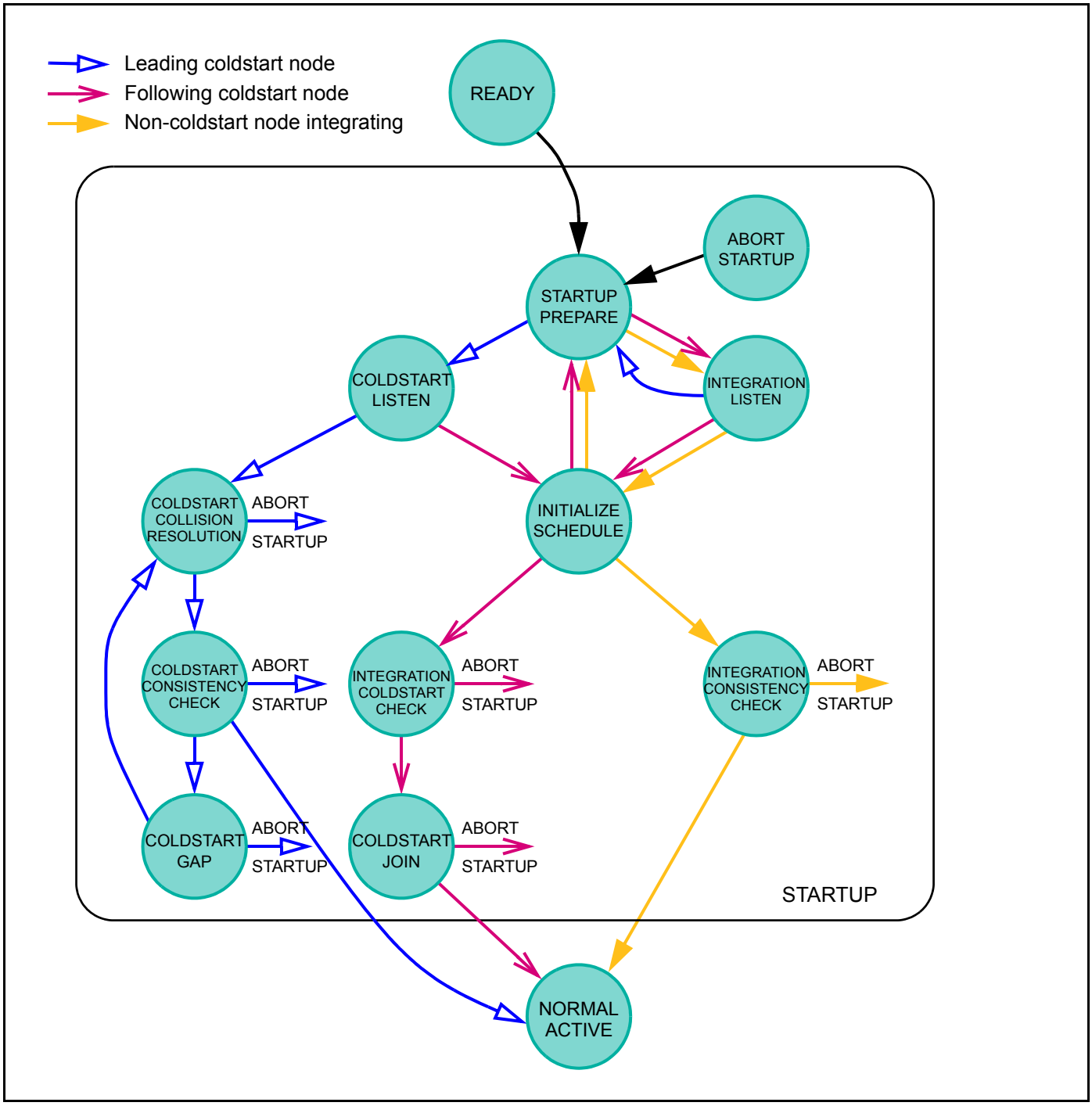
In clusters consisting of three or more nodes, at least three nodes shall be configured to be coldstart nodes. In clusters consisting of two nodes, both nodes must be coldstart nodes. At least two fault-free coldstart nodes are necessary for the cluster to startup.

Each startup frame must also be a sync frame; therefore each coldstart node will also be a sync node. The number of coldstart attempts is configured by **SUCC1.CSA[4:0]**.

A non-coldstart node requires at least two startup frames from distinct nodes for integration. It may start integration before the coldstart nodes have finished their startup. It will not finish its startup until at least two coldstart nodes have finished their startup.

Both non-coldstart nodes and coldstart nodes start passive integration via the integration path as soon as they receive sync frames from which to derive the TDMA schedule information. During integration, the node has to adapt its own clock to the global clock (rate and offset) and has to make its cycle time consistent with the global schedule observable at the network. Afterwards, these settings are checked for consistency with all available network nodes. The node can only leave the integration phase and actively participate in communication when these checks are passed.

**State diagram time-triggered startup**



*Coldstart Inhibit Mode*

In coldstart inhibit mode the node is prevented from initializing the TDMA communication schedule. If bit **CCSV.CSI** is set, the node is not allowed to initialize the cluster communication, i.e. entering the coldstart path is prohibited. The node is allowed to integrate to a running cluster or to transmit startup frames after another coldstart node started the initialization of the cluster communication.

The coldstart inhibit bit **CCSV.CSI** is set whenever the POC enters READY state. The bit has to be cleared under control of the Host by CHI command ALLOW_COLDSTART (**SUCC1.CMD[3:0]** = "1001")

*Startup Timeouts*

The CC supplies two different mT timers supporting two timeout values, startup timeout and startup noise timeout. The two timers are started when the CC enters the COLDSTART_LISTEN state. The expiration of either of these timers causes the node to leave the initial sensing phase (COLDSTART_LISTEN state) with the intention of starting up communication.

Note: The startup and startup noise timers are identical with the wakeup and wakeup noise timers and use the same configuration values **SUCC2.LT[20:0]** and **SUCC2.LTN[3:0]**.

### Startup Timeout

The startup timeout limits the listen time used by a node to determine if there is already communication between other nodes or at least one coldstart node actively requesting the integration of others. The startup timer is configured by programming **SUCC2.LT[20:0]** (see 18.5.2.SUC Configuration Register 2 (SUCC2)).

The startup timeout is: pdListenTimeout = **SUCC2.LT[20:0]**

The startup timer is restarted upon:
- Entering the COLDSTART_LISTEN state
- Both channels reaching idle state while in COLDSTART_LISTEN state

The startup timer is stopped:
- If communication channel activity is detected on one of the configured channels while the node is in the COLDSTART_LISTEN state
- When the COLDSTART_LISTEN state is left

Once the startup timeout expires, neither an overflow nor a cyclic restart of the timer is performed. The timer status is kept for further processing by the startup state machine.

### Startup Noise Timeout

At the same time the startup timer is started for the first time (transition from STARTUP_PREPARE state to COLDSTART_LISTEN state), the startup noise timer is started. This additional timeout is used to improve reliability of the startup procedure in the presence of noise. The startup noise timeout is configured by programming **SUCC2.LTN[3:0]** (see 18.5.2.SUC Configuration Register 2 (SUCC2)).

The startup noise timeout is:
pdListenTimeout • gListenNoise = **SUCC2.LT[20:0]** • **(SUCC2.LTN[3:0]** + 1)

The startup noise timer is restarted upon:
- Entering the COLDSTART_LISTEN state
- Reception of correctly decoded headers or CAS symbols while the node is in COLDSTART_LISTEN state

The startup noise timer is stopped when the COLDSTART_LISTEN state is left.

Once the startup noise timeout expires, neither an overflow nor a cyclic restart of the timer is performed. The status is kept for further processing by the startup state machine. Since the startup noise timer won't be restarted when random channel activity is sensed, this timeout defines the fall-back solution that guarantees that a node will try to start up the communication cluster even in the presence of noise.

*Path of leading Coldstart Node (initiating coldstart)*

When a coldstart node enters COLDSTART_LISTEN, it listens to its attached channels.

If no communication is detected, the node enters the COLDSTART_COLLISION_RESOLUTION state and commences a coldstart attempt. The initial transmission of a CAS symbol is succeeded by the first regular cycle. This cycle has the number zero.

From cycle zero on, the node transmits its startup frame. Since each coldstart node may perform a coldstart attempt, it may occur that several nodes simultaneously transmit the CAS symbol and enter the coldstart path. This situation is resolved during the first four cycles after CAS transmission.

As soon as a node that initiates a coldstart attempt receives a CAS symbol or a frame header during these four cycles, it re-enters the COLDSTART_LISTEN state. Thereby, only one node remains in this path. In cycle four, other coldstart nodes begin to transmit their startup frames.

After four cycles in COLDSTART_COLLISION_RESOLUTION state, the node that initiated the coldstart enters the COLDSTART_CONSISTENCY_CHECK state. It collects all startup frames from cycle four and five and performs the clock correction. If the clock correction does not deliver any errors and it has received at least one valid startup frame pair, the node leaves COLDSTART_CONSISTENCY_CHECK and enters NORMAL_ACTIVE state.

The number of coldstart attempts that a node is allowed to perform is configured by **SUCC1.CSA[4:0]**. The number of remaining coldstarts attempts can be read from **CCSV.RCA[4:0]**. The number of remaining coldstart attempts is reduced by one for each attempted coldstart. A node may enter the COLDSTART_LISTEN state only if this value is larger than one and it may enter the COLDSTART_COLLISION_RESOLUTION state only if this value is larger than zero. If the number of coldstart attempts is one, coldstart is inhibited but integration is still possible.

*Path of following Coldstart Node (responding to leading Coldstart Node)*

When a coldstart node enters the COLDSTART_LISTEN state, it tries to receive a valid pair of startup frames to derive its schedule and clock correction from the leading coldstart node.

As soon as a valid startup frame has been received the INITIALIZE_SCHEDULE state is entered. If the clock synchronization can successfully receive a matching second valid startup frame and derive a schedule from this, the INTEGRATION_COLDSTART_CHECK state is entered.

In INTEGRATION_COLDSTART_CHECK state it is assured that the clock correction can be performed correctly and that the coldstart node from which this node has initialized its schedule is still available. The node collects all sync frames and performs clock correction in the following double-cycle. If clock correction does not signal any errors and if the node continues to receive sufficient frames from the same node it has integrated on, the COLDSTART_JOIN state is entered.

In COLDSTART_JOIN state following coldstart nodes begin to transmit their own startup frames and continue to do so in subsequent cycles. Thereby, the leading coldstart node and the nodes joining it can check if their schedules agree with each other. If the clock correction signals any error, the node aborts the integration attempt. If a node in this state sees at least one valid startup frame during all even cycles in this state and at least one valid startup frame pair during all double cycles in this state, the node leaves COLDSTART_JOIN state and enters NORMAL_ACTIVE state. Thereby it leaves STARTUP at least one cycle after the node that initiated the coldstart.

*Path of Non-coldstart Node*

When a non-coldstart node enters the INTEGRATION_LISTEN state, it listens to its attached channels.

As soon as a valid startup frame has been received, the INITIALIZE_SCHEDULE state is entered. If the clock synchronization can successfully receive a matching second valid startup frame and derive a schedule from this, the INTEGRATION_CONSISTENCY_CHECK state is entered.

In INTEGRATION_CONSISTENCY_CHECK state the node verifies that the clock correction can be performed correctly and that enough coldstart nodes (at least 2) are sending startup frames that agree with the node's own schedule. Clock correction is activated, and if any errors are signalled, the integration attempt is aborted.

During the first even cycle in this state, either two valid startup frames or the startup frame of the node that this node has integrated on must be received; otherwise the node aborts the integration attempt.

During the first double-cycle in this state, either two valid startup frame pairs or the startup frame pair of the node that this node has integrated on must be received; otherwise the node aborts the integration attempt.

If after the first double-cycle less than two valid startup frames are received within an even cycle, or less than two valid startup frame pairs are received within a double-cycle, the startup attempt is aborted.

Nodes in this state need to see two valid startup frame pairs for two consecutive double-cycles each to be allowed to leave STARTUP and enter NORMAL_OPERATION. Consequently, they leave startup at least one double-cycle after the node that initiated the coldstart and only at the end of a cycle with an odd cycle number.

### *19.5.8. NORMAL_ACTIVE State*

As soon as the node that transmitted the first CAS symbol (resolving the potential access conflict and entering STARTUP via coldstart path) and one additional node have entered the NORMAL_ACTIVE state, the startup phase for the cluster has finished. In the NORMAL_ACTIVE state, all configured messages are scheduled for transmission. This includes all data frames as well as the sync frames. Rate and offset measurement is started in all even cycles (even / odd cycle pairs required).

In NORMAL_ACTIVE state the CC supports regular communication functions

- The CC performs transmissions and reception on the FlexRay bus as configured
- Clock synchronization is running
- The Host interface is operational

The CC exits from that state to

- HALT state by writing **SUCC1.CMD[3:0]** = "0110"
- (HALT command, at the end of the current cycle)
- HALT state by writing **SUCC1.CMD[3:0]** = "0111" (FREEZE command, immediately)
- HALT state due to change of the error state from ACTIVE to COMM_HALT
- NORMAL_PASSIVE state due to change of the error state from ACTIVE to PASSIVE
- READY state by writing **SUCC1.CMD[3:0]** = "0010" (READY command)

### *19.5.9. NORMAL_PASSIVE State*

NORMAL_PASSIVE state is entered from NORMAL_ACTIVE state when the error state changes from ACTIVE to PASSIVE.

In NORMAL_PASSIVE state, the node is able to receive all frames (node is fully synchronized and performs clock synchronization). Contrary to the NORMAL_ACTIVE state, the node does not actively participate in communication, i.e. neither symbols nor frames are transmitted.

In NORMAL_PASSIVE state

- The CC performs reception on the FlexRay bus
- The CC does not transmit any frames or symbols on the FlexRay bus
- Clock synchronization is running
- The Host interface is operational

The CC exits from this state to

- HALT state by writing **SUCC1.CMD[3:0]** = "0110"
  (HALT command, at the end of the current cycle)
- HALT state by writing **SUCC1.CMD[3:0]** = "0111" (FREEZE command, immediately)
- HALT state due to change of the error state from PASSIVE to COMM_HALT
- NORMAL_ACTIVE state due to change of the error state from PASSIVE to ACTIVE.
  The transition takes place when **CCEV.PTAC[4:0]** equals **SUCC1.PTA[4:0] - 1**
- To READY state by writing **SUCC1.CMD[3:0]** = "0010" (READY command)

### *19.5.10. HALT State*

In this state all communication (reception and transmission) is stopped.

The CC enters this state

- By writing **SUCC1.CMD[3:0]** = "0110" (HALT command) while the CC is in NORMAL_ACTIVE or NORMAL_PASSIVE state
- By writing **SUCC1.CMD[3:0]** = "0111" (FREEZE command) from all states
- When exiting from NORMAL_ACTIVE state because the clock correction failed counter reached the "maximum without clock correction fatal" limit
- When exiting from NORMAL_PASSIVE state because the clock correction failed counter reached the "maximum without clock correction fatal" limit

The CC exits from this state to DEFAULT_CONFIG state

- By writing **SUCC1.CMD[3:0]** = "0001" (CONFIG command)

When the CC enters HALT state, all configuration and status data is maintained for analyzing purposes.

When the Host writes **SUCC1.CMD[3:0]** = "0110" (HALT command), the CC sets bit **CCSV.HRQ** and enters HALT state after the current communication cycle has finished.

When the Host writes **SUCC1.CMD[3:0]** = "0111" (FREEZE command), the CC enters HALT state immediately and sets bit **CCSV.FSI**.

The POC state from which the transition to HALT state took place can be read from **CCSV.PSL[5:0].**

### 19.6. Network Management

The accrued Network Management (NM) vector can be read from registers NMV1...3. The CC performs a bit-wise OR operation over all NM vectors out of all received valid NM frames with the Payload Preamble Indicator (**PPI**) bit set. Only static frames may be configured to hold NM information. The CC updates the NM vector at the end of each cycle.

The length of the NM vector can be configured from 0 to 12 bytes by **NEMC.NML[3:0]**. The NM vector length must be configured identically in all nodes of a cluster.

To configure a transmit buffer to send FlexRay frames with the **PPI** bit set, bit **PPIT** in the header section of the respective transmit buffer has to be set via **WRHS1.PPIT**. In addition the Host has to write the NM information to the data section of the respective transmit buffer.

The evaluation of the NM vector has to be done by the application running on the Host.

Note: In case a message buffer is configured for transmission / reception of network management frames, the payload length configured in header 2 of that message buffer should be equal or greater than the length of the NM vector configured by **NEMC.NML[3:0]**.

## 19.7. Filtering and Masking

Filtering is done by comparison of the configuration of assigned message buffers against actual slot and cycle counter values and channel ID (channel A, B). A message buffer is only updated / transmitted if the required matches occur.

Filtering is done on:

- Slot Counter
- Cycle Counter
- Channel ID

The following filter combinations for acceptance / transmit filtering are allowed:

- Slot Counter + Channel ID
- Slot Counter + Cycle Counter + Channel ID

All configured filters must match in order to store a received message in a message buffer.

Note: For the FIFO the acceptance filter is configured by the FIFO Rejection Filter and the FIFO Rejection Filter Mask.

A message will be transmitted in the time slot corresponding to the configured frame ID on the configured channel(s). If cycle counter filtering is enabled the configured cycle filter value must also match.

### *19.7.1. Slot Counter Filtering*

Every transmit and receive buffer contains a frame ID stored in the header section. This frame ID is compared against the actual slot counter value in order to assign receive and transmit buffers to the corresponding slot.

If two or more message buffers are configured with the same frame ID, and if they have a matching cycle counter filter value for the same slot, then the message buffer with the **lowest** message buffer number is used.

### *19.7.2. Cycle Counter Filtering*

Cycle counter filtering is based on the notion of a cycle set. For filtering purposes, a match is detected if any one of the elements of the cycle set is matched. The cycle set is defined by the cycle code field in header section 1 of each message buffer.

If message buffer 0 resp. 1 is configured to hold the startup / sync frame or the single slot frame by bits **SUCC1.TXST**, **SUCC1.TXSY**, and **SUCC1.TSM**, cycle counter filtering for message buffer 0 resp. 1 must be disabled.

Note: Sharing of a static time slot via cycle counter filtering between different nodes of a FlexRay network is not allowed.

The set of cycle numbers belonging to a cycle set is determined as described in the following table.

#### Definition of cycle set

| Cycle Code | Matching Cycle Counter Values | | |
|---|---|---|---|
| 0b000000x | all Cycles | | |
| 0b00000**1**c | every second Cycle | at (Cycle Count)mod2 | = c |
| 0b0000**1**cc | every fourth Cycle | at (Cycle Count)mod4 | = cc |
| 0b000**1**ccc | every eighth Cycle | at (Cycle Count)mod8 | = ccc |
| 0b00**1**cccc | every sixteenth Cycle | at (Cycle Count)mod16 | = cccc |
| 0b0**1**ccccc | every thirty-second Cycle | at (Cycle Count)mod32 | = ccccc |
| 0b**1**cccccc | every sixty-fourth Cycle | at (Cycle Count)mod64 | = cccccc |

The following table gives some examples for valid cycle sets to be used for cycle counter filtering:

### Examples for valid cycle sets

| Cycle Code | Matching Cycle Counter Values |
|---|---|
| 0b0000011 | 1-3-5-7- …. -63 ↵ |
| 0b0000100 | 0-4-8-12- …. -60 ↵ |
| 0b0001110 | 6-14-22-30- …. -62 ↵ |
| 0b0011000 | 8-24-40-56 ↵ |
| 0b0100011 | 3-35 ↵ |
| 0b1001001 | 9 ↵ |

The received message is stored only if the cycle counter value of the cycle during which the message is received matches an element of the receive buffer's cycle set. Other filter criteria must also be met.

The content of a transmit buffer is transmitted on the configured channel(s) when an element of the cycle set matches the current cycle counter value. Other filter criteria must also be met.

*19.7.3. Channel ID Filtering*

There is a 2-bit channel filtering field (**CHA**, **CHB**) located in the header section of each message buffer in the Message RAM. It serves as a filter for receive buffers, and as a control field for transmit buffers (See the following table).

### Channel filtering configuration

| CHA | CHB | Transmit Buffer<br>transmit frame | Receive Buffer<br>store valid receive frame |
|---|---|---|---|
| 1 | 1 | on both channels<br>(static segment only) | received on channel A or B<br>(store first semantically valid frame,<br>static segment only) |
| 1 | 0 | on channel A | received on channel A |
| 0 | 1 | on channel B | received on channel B |
| 0 | 0 | no transmission | ignore frame |

The contents of a transmit buffer is transmitted on the channels specified in the channel filtering field when the slot counter filtering and cycle counter filtering criteria are also met. Only in static segment a transmit buffer may be set up for transmission on both channels (**CHA** and **CHB** set).

Valid received frames are stored if they are received on the channels specified in the channel filtering field when the slot counter filtering and cycle counter filtering criteria are also met. Only in static segment a receive buffer may be setup for reception on both channels (**CHA** and **CHB** set).

Note: If a message buffer is configured for the dynamic segment and both bits of the channel filtering field are set to '1', no frames are transmitted resp. received frames are ignored (same function as **CHA = CHB** = '0').

*19.7.4. FIFO Filtering*

For FIFO filtering there is one rejection filter and one rejection filter mask available. The FIFO filter consists of channel filter **FRF.CH[1:0]**, frame ID filter **FRF.FID[10:0]**, and cycle counter filter **FRF.CYF[6:0]**. Registers FRF and FRFM can be configured in DEFAULT_CONFIG or CONFIG state only. The filter configuration in the header section of message buffers belonging to the FIFO is ignored.

The 7-bit cycle counter filter determines the cycle set to which frame ID and channel rejection filter are applied. In cycles not belonging to the cycle set specified by **FRF.CYF[6:0]**, all frames are rejected.

A valid received frame is stored in the FIFO if channel ID, frame ID, and cycle counter are not rejected by the configured rejection filter and rejection filter mask, and if there is no matching dedicated receive buffer.

## 19.8. Transmit Process

### *19.8.1. Static Segment*

For the static segment, if there are several messages pending for transmission, the message with the frame ID corresponding to the next sending slot is selected for transmission.

The data section of transmit buffers assigned to the static segment can be updated until the end of the preceding time slot. This means that a transfer from the Input Buffer has to be started by writing to the Input Buffer Command Request register latest at this time.

### *19.8.2. Dynamic Segment*

In the dynamic segment, if several messages are pending, the message with the highest priority (lowest frame ID) is selected next. In the dynamic segment different slot counter sequences on channel A and channel B are possible (concurrent sending of different frame IDs on both channels).

The data section of transmit buffers assigned to the dynamic segment can be updated until the end of the preceding slot. This means that a transfer from the Input Buffer has to be started by writing to the Input Buffer Command Request register latest at this time.

The start of latest transmit configured by **MHDC.SLT[12:0]** defines the maximum minislot value allowed before inhibiting new frame transmission in the dynamic segment of the current cycle.

### *19.8.3. Transmit Buffers*

E-Ray message buffers can be configured as transmit buffers by programming bit CFG in the header section of the respective message buffer to ’1’ via WRHS1.

There exist the following possibilities to assign a transmit buffer to the CC channels:

- Static segment: channel A **or** channel B,
  channel A **and** channel B
- Dynamic segment: channel A **or** channel B

Message buffer 0 resp. 1 is dedicated to hold the startup frame, the sync frame, or the designated single slot frame as configured by **SUCC1.TXST**, **SUCC1.TXSY**, and **SUCC1.TSM**. In this case, it can be reconfigured in DEFAULT_CONFIG or CONFIG state only. This ensures that any node transmits at most one startup / sync frame per communication cycle. Transmission of startup / sync frames from other message buffers is not possible.

All other message buffers configured for transmission in static or dynamic segment are reconfigurable during runtime depending on the configuration of **MRC.SEC[1:0]** (see 19.11.1.Reconfiguration of Message Buffers. Due to the organization of the data partition in the Message RAM (reference by data pointer), reconfiguration of the configured payload length and the data pointer in the header section of a message buffer may lead to erroneous configurations.

If a message buffer is reconfigured (header section updated) during runtime, it may happen that this message buffer is not send out in the respective communication cycle.

The CC does not have the capability to calculate the header CRC. The Host is supposed to provide the header CRCs for all transmit buffers. If network management is required, the Host has to set the **PPIT** bit in the header section of the respective message buffer to ’1’ and write the network management information to the data section of the message buffer (see 19.6.Network Management).

The payload length field configures the payload length in 2-byte words. If the configured payload length of a static transmit buffer is shorter than the payload length configured for the static segment by **MHDC.SFDL[6:0]**, the CC generates padding bytes to ensure that frames have proper physical length. The padding pattern is logical zero.

Each transmit buffer provides a transmission mode flag **TXM** that allows the Host to configure the transmission mode for the transmit buffer. If this bit is set, the transmitter operates in the single-shot mode. If this bit is cleared, the transmitter operates in the continuous mode.

In **single-shot mode** the CC resets the respective **TXR** flag after transmission has completed. Now the Host may update the transmit buffer.

In **continuous mode**, the CC does not reset the respective transmission request flag **TXR** after successful transmission. In this case a frame is sent out each time the filter criteria match. The **TXR** flag can be reset by the Host by writing the respective message buffer number to the IBCR register while bit **IBCM.STXRH** is set to ’0’.

If two or more transmit buffers meet the filter criteria simultaneously, the transmit buffer with the lowest message buffer number will be transmitted in the respective slot.

### *19.8.4. Frame Transmission*

The following steps are required to prepare a message buffer for transmission:

- Configure the transmit buffer in the Message RAM via WRHS1, WRHS2, and WRHS3
- Write the data section of the transmit buffer via WRDSn
- Transfer the configuration and message data from Input Buffer to the Message RAM by writing the number of the target message buffer to register IBCR
- If configured in register IBCM, the transmission request flag **TXR** for the respective message buffer will be set as soon as the transfer has completed, and the message buffer is ready for transmission.
- Check whether the message buffer has been transmitted by checking the respective **TXR** bit (TXR = ’0’) in the TRXQ1/2/3/4 registers (single-shot mode only).

After transmission has completed, the respective **TXR** flag in the TXRQ1/2/3/4 register is reset (single-shot mode), and, if bit **MBI** in the header section of the message buffer is set, flag **SIR.TXI** is set to ’1’. If enabled, an interrupt is generated.

### *19.8.5. Null Frame Transmission*

If in static segment the Host does not set the transmission request flag before transmit time, and if there is no other transmit buffer with matching filter criteria, the CC transmits a null frame with the null frame indication bit set to ’0’ and the payload data **set to zero**.

In the following cases the CC transmits a null frame:

- If the message buffer with the lowest message buffer number matching the filter criteria does not have its transmission request flag set (**TXR** = ’0’).
- No transmit buffer configured for the slot has a cycle counter filter that matches the current cycle. In this case, no message buffer status MBS is updated.

Null frames are not transmitted in the dynamic segment.

## 19.9. Receive Process

### *19.9.1. Dedicated Receive Buffers*

A portion of the E-Ray message buffers can be configured as dedicated receive buffers by programming bit **CFG** in the header section of the respective message buffer to ’0’ via WRHS1.

The following possibilities exist to assign a receive buffer to the CC channels:

- Static segment: channel A **or** channel B,
  channel A **and** channel B (the CC stores the first semantically valid frame)
- Dynamic segment: channel A **or** channel B

The CC transfers the payload data of valid received messages from the shift register of the FlexRay channel protocol controller (channel A or B) to the receive buffer with the matching filter configuration. A receive buffer stores all frame elements except the frame CRC.

All message buffers configured for reception in static or dynamic segment are reconfigurable during runtime depending on the configuration of **MRC.SEC[1:0]** (see 19.11.1.Reconfiguration of Message Buffers). If a message buffer is reconfigured (header section updated) during runtime it may happen that in the respective communication cycle a received message is lost.

If two or more receive buffers meet the filter criteria simultaneously, the receive buffer with the lowest message buffer number is updated with the received message.

### *19.9.2. Frame Reception*

The following steps are required to prepare a dedicated message buffer for reception:

- Configure the receive buffer in the Message RAM via WRHS1, WRHS2, and WRHS3
- Transfer the configuration from Input Buffer to the Message RAM by writing the number of the target message buffer to register IBCR

Once these steps are performed, the message buffer functions as an active receive buffer and participates in the internal acceptance filtering process which takes place every time the CC receives a message. The first matching receive buffer is updated from the received message.

If a valid payload segment was stored in the data section of a message buffer, the respective **ND** flag in the NDAT1/2/3/4 registers is set, and, if bit **MBI** in the header section of that message buffer is set, flag **SIR.RXI** is set to ’1’. If enabled, an interrupt is generated.

In case that bit **ND** was already set when the Message Handler updates the message buffer, bit **MBS.MLST** of the respective message buffer is set and the unprocessed message data is lost.

If no frame, a null frame, or a corrupted frame was received in a slot, the data section of the message buffer configured for this slot is not updated. In this case only the respective message buffer status MBS is updated.

When the Message Handler changed the message buffer status MBS in the header section of a message buffer, the respective **MBC** flag in the MBSC1/2/3/4 registers is set, and if bit **MBI** in the header section of that message buffer is set, flag **SIR.MBSI** is set to ’1’. If enabled an interrupt is generated.

If the payload length of a received frame **PLR[6:0]** is longer than the value programmed by **PLC[6:0]** in the header section of the respective message buffer, the data field stored in the message buffer is truncated to that length.

To read a receive buffer from the Message RAM via the Output Buffer, proceed as described in Data Transfer from Message RAM to Output Buffer.

Note: The **ND** and **MBC** flags are automatically cleared by the Message Handler when the payload data and the header of a received message have been transferred to the Output Buffer, respectively.

### *19.9.3. Null Frame Reception*

The payload segment of a received null frame is not copied into the matching dedicated receive buffer. If a null frame has been received, only the message buffer status MBS of the matching message buffer is updated from the received null frame. All bits in header 2 and 3 of the matching message buffer remain unchanged. They are updated from received data frames only.

When the Message Handler changed the message buffer status MBS in the header section of a message buffer, the respective **MBC** flag in the MBSC1/2/3/4 register is set, and if bit **MBI** in the header section of that message buffer is set, flag **SIR.MBSI** is set to ’1’. If enabled, an interrupt is generated.

## 19.10. FIFO Function

### *19.10.1. Description*

A group of the message buffers can be configured as a cyclic First-In-First-Out (FIFO) buffer. The group of message buffers belonging to the FIFO is contiguous in the register map starting with the message buffer referenced by **MRC.FFB[7:0]** and ending with the message buffer referenced by **MRC.LCB[7:0]**. Up to 128 message buffers can be assigned to the FIFO.

Every **valid** incoming message not matching with any dedicated receive buffer but passing the programmable FIFO filter is stored into the FIFO. In this case frame ID, payload length, receive cycle count, and the message buffer status MBS of the addressed FIFO message buffer are overwritten with frame ID, payload length, receive cycle count, and the status from the received frame. Bit **SIR.RFNE** shows that the FIFO is not empty, bit **SIR.RFCL** is set when the receive FIFO fill level **FSR.RFFL[7:0]** is equal or greater than the critical level as configured by **FCL.CL[7:0]**, bit **EIR.RFO** shows that a FIFO overrun has been detected. If enabled, interrupts are generated.

If null frames are not rejected by the FIFO rejection filter, the null frames will be treated like data frames when they are stored into the FIFO.

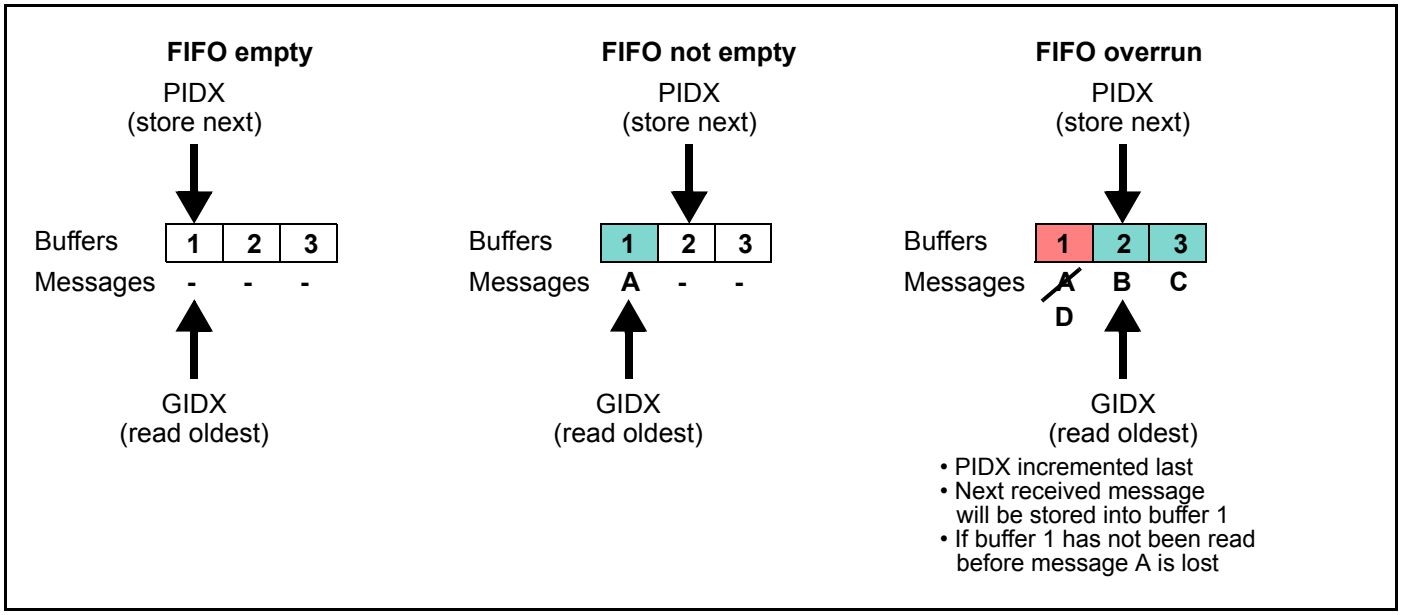
There are two index registers associated with the FIFO. The PUT Index Register (PIDX) is an index to the next available location in the FIFO. When a new message has been received it is written into the message buffer addressed by the PIDX register. The PIDX register is then incremented and addresses the next available message buffer. If the PIDX register is incremented past the highest numbered message buffer of the FIFO, the PIDX register is loaded with the number of the first (lowest numbered) message buffer in the FIFO chain. The GET Index Register (GIDX) is used to address the next message buffer of the FIFO to be read. The GIDX register is incremented after transfer of the contents of a message buffer belonging to the FIFO to the Output Buffer. The PUT Index Register and the GET Index Register are not accessible by the Host.
The FIFO is completely filled when the PUT index (PIDX) reaches the value of the GET index (GIDX). When the next message is written to the FIFO before the oldest message has been read, both PUT index and GET index are incremented and the new message overwrites the oldest message in the FIFO. This will set FIFO overrun flag **EIR.RFO**.

A FIFO non empty status is detected when the PUT index (PIDX) differs from the GET index (GIDX). In this case flag **SIR.RFNE** is set. This indicates that there is at least one received message in the FIFO. The FIFO empty, FIFO not empty, and the FIFO overrun states are explained in the figure “FIFO status: empty, not empty, overrun” for a three message buffer FIFO.

The programmable FIFO Rejection Filter (FRF) defines a filter pattern for messages to be rejected. The FIFO filter consists of channel filter, frame ID filter, and cycle counter filter. If bit **FRF.RSS** is set to ’1’ (default), all messages received in the static segment are rejected by the FIFO. If bit **FRF.RNF** is set to ’1’ (default), received null frames are not stored in the FIFO.

The FIFO Rejection Filter Mask (FRFM) specifies which bits of the frame ID filter in the FIFO Rejection Filter register are marked ’don’t care’ for rejection filtering.

**FIFO status: empty, not empty, overrun**



#### 19.10.2. Configuration of the FIFO

(Re)configuration of message buffers belonging to the FIFO is only possible when the CC is in DEFAULT_CONFIG or CONFIG state. While the CC is in DEFAULT_CONFIG or CONFIG state, the FIFO function is not available.

For all message buffers belonging to the FIFO the payload length configured should be programmed to the same value via **WRHS2.PLC[6:0]**. The data pointer to the first 32-bit word of the data section of the respective message buffer in the Message RAM has to be configured via **WRHS3.DP[10:0]**.

All information required for acceptance filtering is taken from the FIFO rejection filter and the FIFO rejection filter mask. The values configured in the header sections of the message buffers belonging to the FIFO are, with exception of DP and PLC, irrelevant.

Note: If the payload length of a received frame is longer than the value programmed by **WRHS2.PLC[6:0]** in the header section of the respective message buffer, the data field stored in a message buffer of the FIFO is truncated to that length.

#### 19.10.3. Access to the FIFO

For FIFO access outside DEFAULT_CONFIG and CONFIG state, the Host has to trigger a transfer from the Message RAM to the Output Buffer by writing the number of the first message buffer of the FIFO (referenced by **MRC.FFB[7:0]**) to the register OBCR. The Message Handler then transfers the message buffer addressed by the GET Index Register (GIDX) to the Output Buffer. After this transfer the GET Index Register (GIDX) is incremented.

### 19.11. Message Handling

The Message Handler controls data transfers between the Input / Output Buffer and the Message RAM and between the Message RAM and the two Transient Buffer RAMs. All accesses to the internal RAMs are 32+1 bit accesses. The additional bit is used for parity checking.

Access to the message buffers stored in the Message RAM is done under control of the Message Handler state machine. This avoids conflicts between accesses of the two FlexRay channel protocol controllers and the Host to the Message RAM.

Frame IDs of message buffers assigned to the static segment have to be in the range from 1 to **GTUC7.NSS[9:0]**. Frame IDs of message buffers assigned to the dynamic segment have to be in the range from **GTUC7.NSS[9:0]** + 1 to 2047.

Received messages with no matching dedicated receive buffer (static or dynamic segment) are stored in the receive FIFO (if configured) if they pass the FIFO rejection filter.

#### 19.11.1. Reconfiguration of Message Buffers

In case that an application needs to operate with more than 128 different messages, static and dynamic message buffers may be reconfigured during FlexRay operation. This is done by updating the header section of the respective message buffer via Input Buffer registers WRHS1....3.

Reconfiguration has to be enabled via control bits **MRC.SEC[1:0]** in the Message RAM Configuration register.

If a message buffer has not been transmitted / updated from a received frame before reconfiguration starts, the respective message is lost.

The point in time when a reconfigured message buffer is ready for transmission / reception according to the reconfigured frame ID depends on the actual state of the slot counter when the update of the header section has completed. Therefore it may happen that a reconfigured message buffer is not transmitted / updated from a received frame in the cycle where it was reconfigured.

The Message RAM is scanned according to the table below:

**Scan of Message RAM**

| Start of Scan in Slot | Scan for Slots |
|---|---|
| 1 | 2…15, 1 (next cycle) |
| 8 | 16…23, 1 (next cycle) |
| 16 | 24…31, 1 (next cycle) |
| 24 | 32…39, 1 (next cycle) |
| … | … |

A Message RAM scan is terminated with the start of NIT regardless whether it has completed or not. The scan of the Message RAM for slots 2 to 15 starts at the beginning of slot 1 of the actual cycle. The scan of the Message RAM for slot 1 is done in the cycle before by checking in parallel to each scan of the Message RAM whether there is a message buffer configured for slot 1 of the next cycle.

The number of the first dynamic message buffer is configured by **MRC.FDB[7:0]**. In case a Message RAM scan starts while the CC is in dynamic segment, the scan starts with the message buffer number configured by **MRC.FDB[7:0]**.

In case a message buffer should be reconfigured to be used in slot 1 of the next cycle, the following has to be considered:

- If the message buffer to be reconfigured for slot 1 is part of the "Static Buffers", it will only be found if it is reconfigured before the last Message RAM scan in the static segment of the actual cycle evaluates this message buffer.
- If the message buffer to be reconfigured for slot 1 is part of the "Static + Dynamic Buffers", it will be found if it is reconfigured before the last Message RAM scan in the actual cycle evaluates this message buffer.
- The start of NIT terminates the Message RAM scan. In case the Message RAM scan has not evaluated the reconfigured message buffer until this point in time, the message buffer will not be considered for the next cycle.

Note: Reconfiguration of message buffers may lead to the loss of messages and therefore has to be used very carefully. In worst case (reconfiguration in consecutive cycles) it may happen that a message buffer is never transmitted / updated from a received frame.

### *19.11.2. Host access to Message RAM*

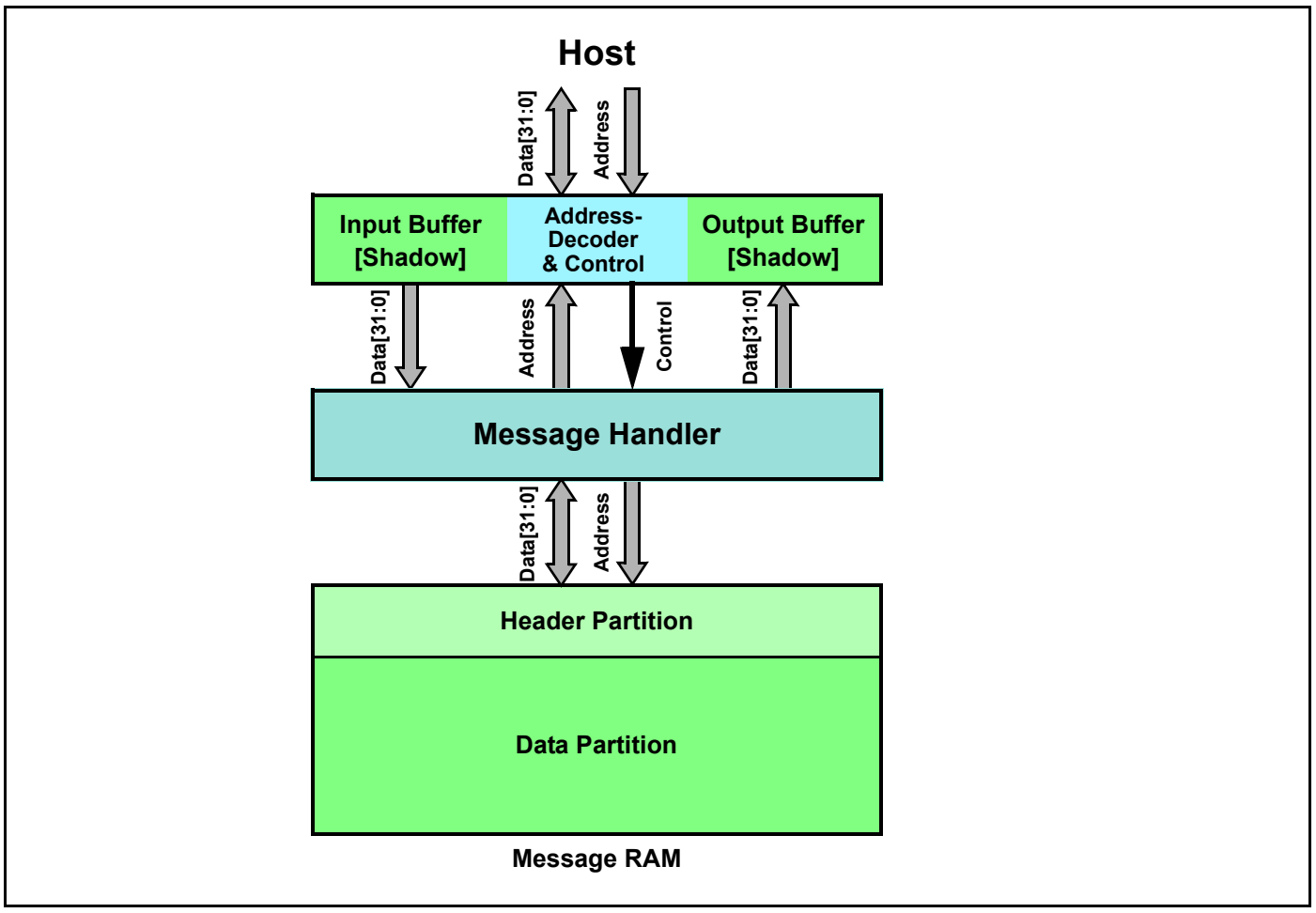
The message transfer between Input Buffer and Message RAM as well as between Message RAM and Output Buffer is triggered by the Host by writing the number of the target / source message buffer to be accessed to IBCR or OBCR register.

The IBCM and OBCM registers can be used to write / read header and data section of the selected message buffer separately.

If bit **IBCM.STXR** is set to = ’1’, the transmission request flag **TXR** of the selected message buffer is automatically set after the message buffer has been updated. If bit I**BCM.STXR** is reset to ’0’, the transmission request flag TXR of the selected message buffer is reset. This can be used to stop transmission from message buffers operated in continuous mode.

Input Buffer (IBF) and Output Buffer (OBF) are build up as a double buffer structure. One half of this double buffer structure is accessible by the Host (IBF Host / OBF Host), while the other half (IBF Shadow / OBF Shadow) is accessed by the Message Handler for data transfers between IBF / OBF and Message RAM.

**Host access to Message RAM**

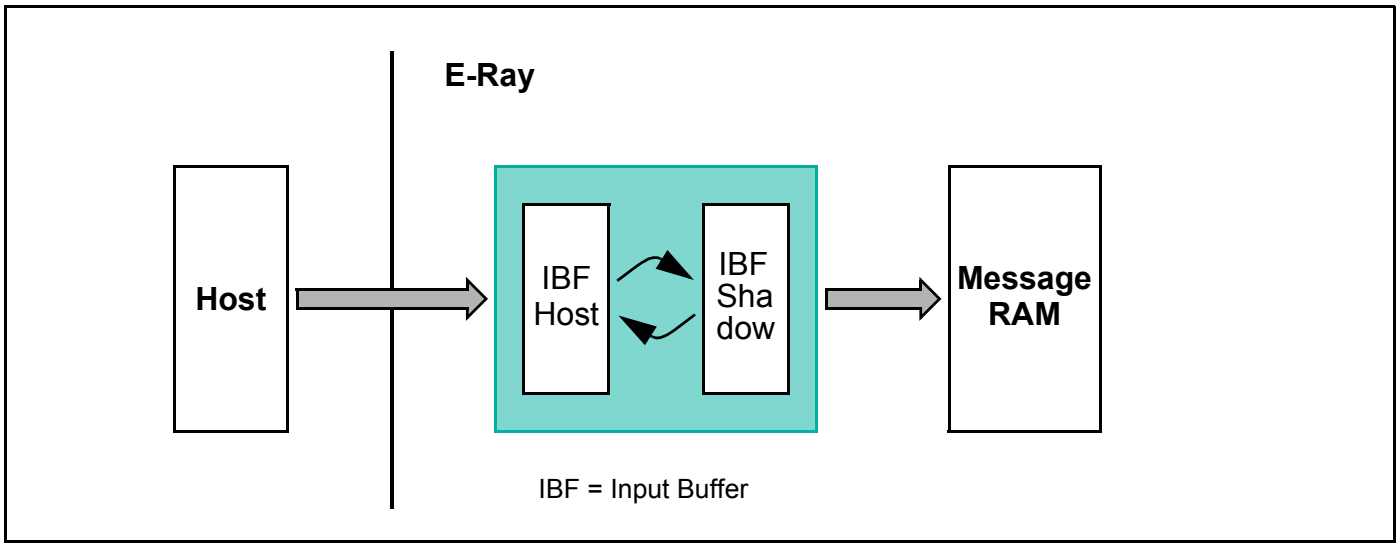


*Data Transfer from Input Buffer to Message RAM*

To configure / update a message buffer in the Message RAM, the Host has to write the data to WRDSn and the header to WRHS1...3. The specific action is selected by configuring the Input Buffer Command Mask IBCM.

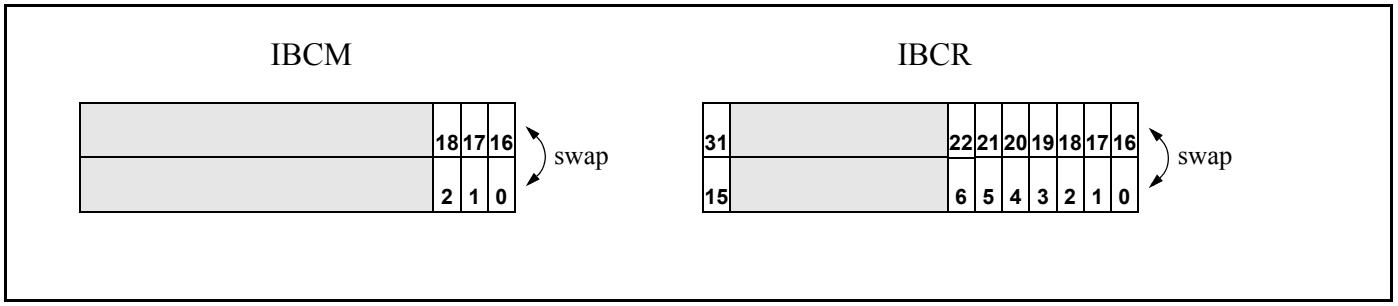
When the Host writes the number of the target message buffer in the Message RAM to **IBCR.IBRH[6:0]**, IBF Host and IBF Shadow are swapped (see the figure below).

**Double buffer structure Input Buffer**




In addition the bits in the IBCM and IBCR registers are also swapped to keep them attached to the respective IBF section (see the figure below).

**Swapping of IBCM and IBCR bits**




With this write operation bit **IBCR.IBSYS** is set to '1'. The Message Handler then starts to transfer the contents of IBF Shadow to the message buffer in the Message RAM selected by **IBCR.IBRS[6:0]**.

While the Message Handler transfers the data from IBF Shadow to the target message buffer in the Message RAM, the Host may write the next message to IBF Host. After the transfer between IBF Shadow and the Message RAM has completed, bit **IBCR.IBSYS** is set back to '0' and the next transfer to the Message RAM may be started by the Host by writing the respective target message buffer number to **IBCR.IBRH[6:0]**.

If a write access to **IBCR.IBRH[6:0]** occurs while **IBCR.IBSYS** is '1', **IBCR.IBSYH** is set to '1'. After completion of the ongoing data transfer from IBF Shadow to the Message RAM, IBF Host and IBF Shadow are swapped, **IBCR.IBSYH** is reset to '0', **IBCR.IBSYS** remains set to '1', and the next transfer to the Message RAM is started. In addition the message buffer numbers stored under **IBCR.IBRH[6:0]** and **IBCR.IBRS[6:0]** and the command mask flags are also swapped.

### Example of a 8/16/32-bit Host access sequence:

Configure / update n-th message buffer via IBF

- Wait until **IBCR.IBSYH** is reset
- Write data section to WRDSn
- Write header section to WRHS1...3
- Write Command Mask: write **IBCM.STXRH**, **IBCM.LDSH**, **IBCM.LHSH**
- Demand data transfer to target message buffer: write **IBCR.IBRH[6:0]**

Configure / update (n+1)th message buffer via IBF

- Wait until IBCR.IBSYH is reset
- Write data section to WRDSn
- Write header section to WRHS1...3
- Write Command Mask: write IBCM.STXRH, IBCM.LDSH, IBCM.LHSH
- Demand data transfer to target message buffer: write IBCR.IBRH[6:0]

Note: Any write access to IBF while **IBCR.IBSYH** is '1' will set error flag **EIR.IIBA** to '1'. In this case the write access has no effect.

### Assignment of IBCM bits

| Pos. | Access | Bit | Function |
|---|---|---|---|
| 18 | r | STXRS | Set Transmission Request Shadow ongoing or finished |
| 17 | r | LDSS | Load Data Section Shadow ongoing or finished |
| 16 | r | LHSS | Load Header Section Shadow ongoing or finished |
| 2 | r/w | STXRH | Set Transmission Request Host |
| 1 | r/w | LDSH | Load Data Section Host |
| 0 | r/w | LHSH | Load Header Section Host |

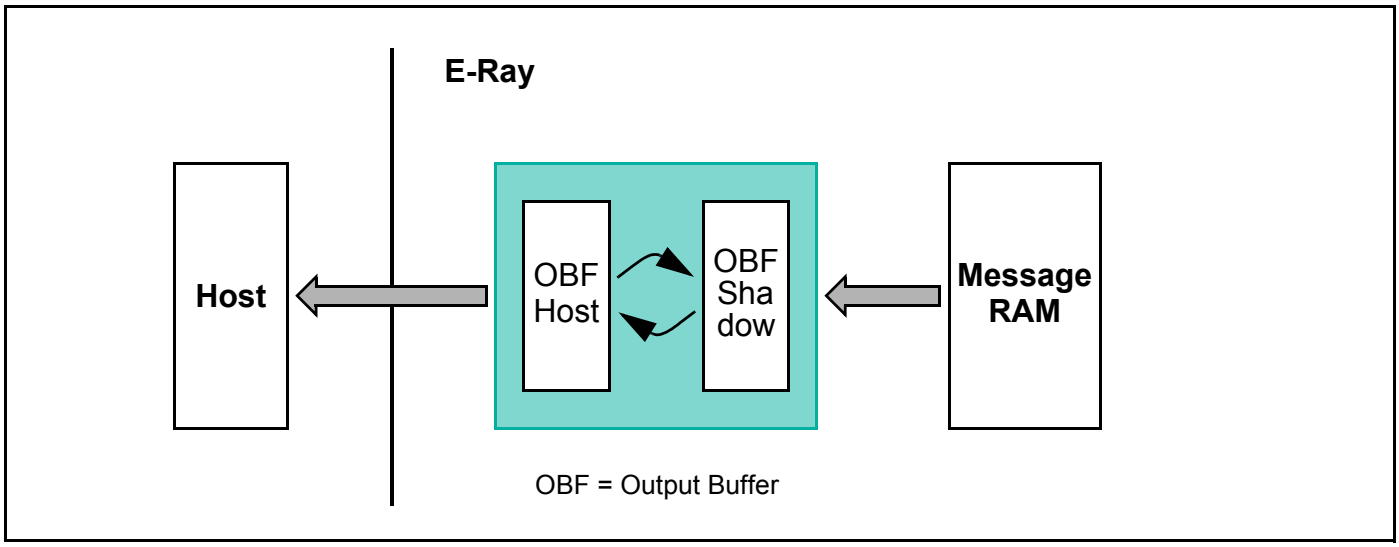
### Assignment of IBCR bits

| Pos. | Access | Bit | Function |
|---|---|---|---|
| 31 | r | IBSYS | IBF Busy Shadow,<br>signals ongoing transfer from IBF Shadow to Message RAM |
| 22…16 | r | IBRS[6:0] | IBF Request Shadow,<br>number of message buffer currently / lately updated |
| 15 | r | IBSYH | IBF Busy Host,<br>transfer request pending for message buffer referenced by IBRH[6:0] |
| 6…0 | r/w | IBRH[6:0] | IBF Request Host,<br>number of message buffer to be updated next |

*Data Transfer from Message RAM to Output Buffer*

To read a message buffer from the Message RAM, the Host has to write to register OBCR to trigger the data transfer as configured in OBCM. After the transfer has completed, the Host can read the transferred data from RDDSn, RDHS1...3, and MBS.

### Double buffer structure Output Buffer

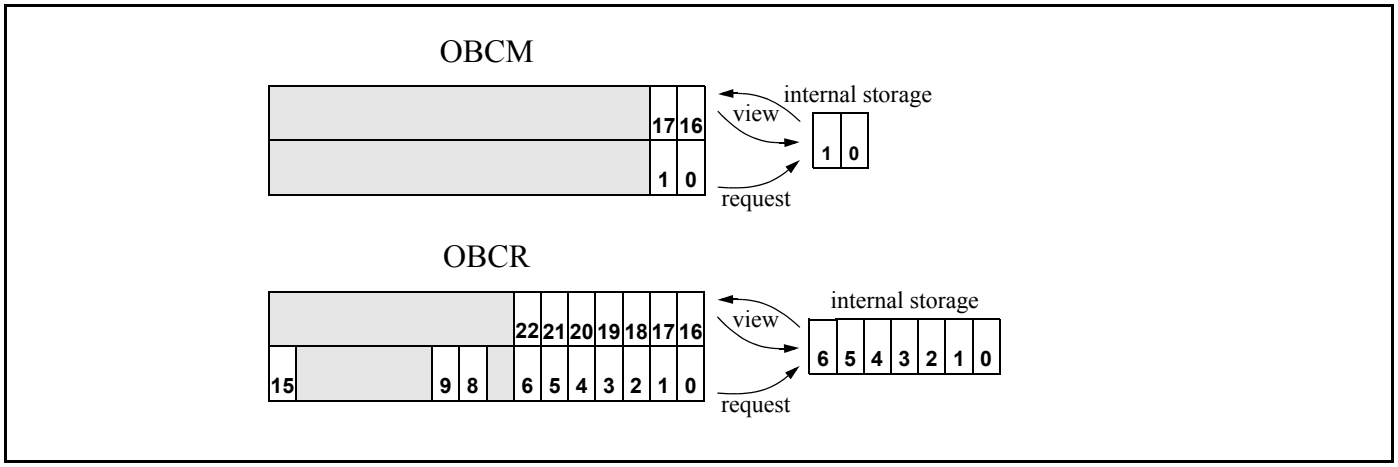



OBF Host and OBF Shadow as well as bits **OBCM.RHSS**, **OBCM.RDSS**, **OBCM.RHSH**, **OBCM.RDSH** and bits **OBCR.OBRS[6:0]**, **OBCR.OBRH[6:0]** are swapped under control of bits **OBCR.VIEW** and **OBCR.REQ**.

Writing bit **OBCR.REQ** to ’1’ copies bits **OBCM.RHSS**, **OBCM.RDSS** and bits **OBCR.OBRS[6:0]** to an internal storage (see figure “Swapping of OBCM and OBCR bits”).

After setting **OBCR.REQ** to ’1’, **OBCR.OBSYS** is set to ’1’, and the transfer of the message buffer selected by **OBCR.OBRS[6:0]** from the Message RAM to OBF Shadow is started. After the transfer between the Message RAM and OBF Shadow has completed, the **OBCR.OBSYS** bit is set back to ’0’. Bits **OBCR.REQ** and **OBCR.VIEW** can only be set to ’1’ while **OBCR.OBSYS** is ’0’.

### Swapping of OBCM and OBCR bits




OBF Host and OBF Shadow are swapped by setting bit **OBCR.VIEW** to ’1’ while bit **OBCR.OBSYS** is ’0’ (see figure “Double buffer structure Output Buffer”).

In addition bits **OBCR.OBRH[6:0]** and bits **OBCM.RHSH**, **OBCM.RDSH** are swapped with the registers internal storage thus assuring that the message buffer number stored in **OBCR.OBRH[6:0]** and the mask configuration stored in **OBCM.RHSH**, **OBCM.RDSH** matches the transferred data stored in OBF Host (see figure “Swapping of OBCM and OBCR bits”).

Now the Host can read the transferred message buffer from OBF Host while the Message Handler may transfer the next message from the Message RAM to OBF Shadow.

### Example of an 8/16/32-bit Host access to a single message buffer:

If a single message buffer has to be read out, two separate write accesses to **OBCR.REQ** and **OBCR.VIEW** are necessary:

- Wait until **OBCR.OBSYS** is reset
- Write Output Buffer Command Mask **OBCM.RHSS**, **OBCM.RDSS**
- Request transfer of message buffer to OBF Shadow by writing **OBCR.OBRS[6:0]** and **OBCR.REQ** (in case of and 8-bit Host interface, **OBCR.OBRS[6:0]** has to be written before **OBCR.REQ**).
- Wait until **OBCR.OBSYS** is reset
- Toggle OBF Shadow and OBF Host by writing **OBCR.VIEW** = '1'
- Read out transferred message buffer by reading RDDSn, RDHS1...3, and MBS

### Example of an 8/16/32-bit Host access sequence:

Request transfer of 1st message buffer to OBF Shadow

- Wait until **OBCR.OBSYS** is reset
- Write Output Buffer Command Mask **OBCM.RHSS**, **OBCM.RDSS** for 1st message buffer
- Request transfer of 1st message buffer to OBF Shadow by writing **OBCR.OBRS[6:0]** and **OBCR.REQ** (in case of an 8-bit Host interface, **OBCR.OBRS[6:0]** has to be written **before OBCR.REQ)**.

Toggle OBF Shadow and OBF Host to read out 1st transferred message buffer and request transfer of 2nd message buffer:

- Wait until **OBCR.OBSYS** is reset
- Write Output Buffer Command Mask **OBCM.RHSS**, **OBCM.RDSS** for 2nd message buffer
- Toggle OBF Shadow and OBF Host and start transfer of 2nd message buffer to OBF Shadow simultaneously by writing **OBCR.OBRS[6:0]** of 2nd message buffer, **OBCR.REQ**, and **OBCR.VIEW** (in case of and 8-bit Host interface, **OBCR.OBRS[6:0]** has to be written **before OBCR.REQ** and **OBCR.VIEW**).
- Read out 1st transferred message buffer by reading RDDSn, RDHS1...3, and MBS

Demand access to last requested message buffer without request of another message buffer:

- Wait until **OBCR.OBSYS** is reset
- Demand access to last transferred message buffer by writing **OBCR.VIEW**
- Read out last transferred message buffer by reading RDDSn, RDHS1...3, and MBS

### Assignment of OBCM bits

| Pos. | Access | Bit | Function |
|---|---|---|---|
| 17 | r | RDSH | Data Section available for Host access |
| 16 | r | RHSH | Header Section available for Host access |
| 1 | r/w | RDSS | Read Data Section Shadow |
| 0 | r/w | RHSS | Read Header Section Shadow |

### Assignment of OBCR bits

| Pos. | Access | Bit | Function |
|---|---|---|---|
| 22…16 | r | OBRH[6:0] | OBF Request Host,<br>number of message buffer available for Host access |
| 15 | r | OBSYS | OBF Busy Shadow,<br>signals ongoing transfer from Message RAM to OBF Shadow |
| 9 | r/w | REQ | Request Transfer from Message RAM to OBF Shadow |
| 8 | r/w | VIEW | View OBF Shadow, swap OBF Shadow and OBF Host |
| 6…0 | r/w | OBRS[6:0] | OBF Request Shadow,<br>number of message buffer for next request |

*19.11.3. FlexRay Protocol Controller access to Message RAM*

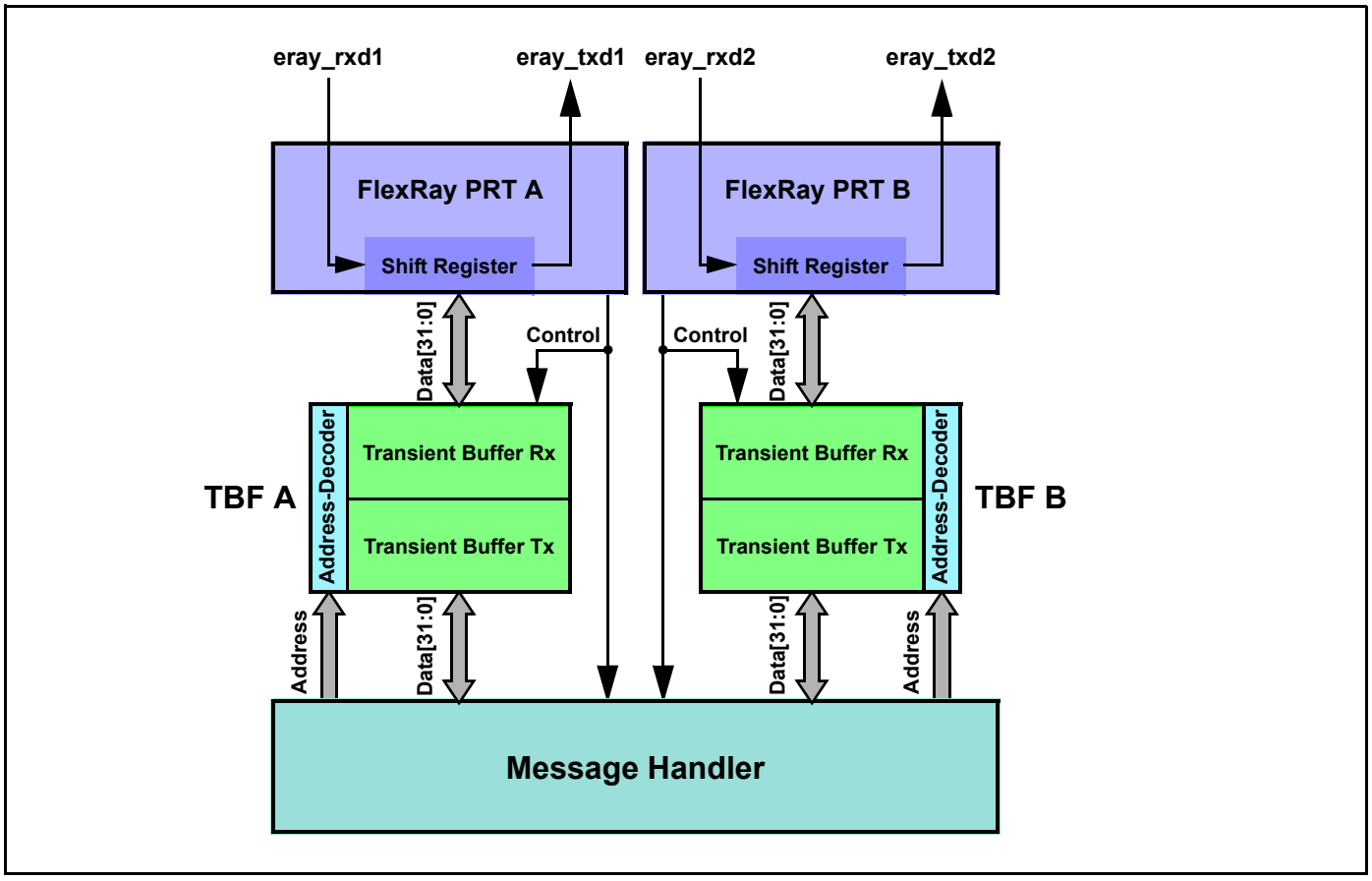
The two Transient Buffer RAMs (TBF A,B) are used to buffer the data for transfer between the two FlexRay Protocol Controllers and the Message RAM.

Each Transient Buffer RAM is build up as a double buffer, able to store two complete FlexRay messages. There is always one buffer assigned to the corresponding Protocol Controller while the other one is accessible by the Message Handler.

If e.g. the Message Handler writes the next message to be send to Transient Buffer Tx, the FlexRay Channel Protocol Controller can access Transient Buffer Rx to store the message it is actually receiving. During transmission of the message stored in Transient Buffer Tx, the Message Handler transfers the last received message stored in Transient Buffer Rx to the Message RAM (if it passes acceptance filtering) and updates the respective message buffer.

Data transfers between the Transient Buffer RAMs and the shift registers of the FlexRay Channel Protocol Controllers are done in words of 32 bit. This enables the use of a 32 bit shift register independent of the length of the FlexRay messages.

### Access to Transient Buffer RAMs



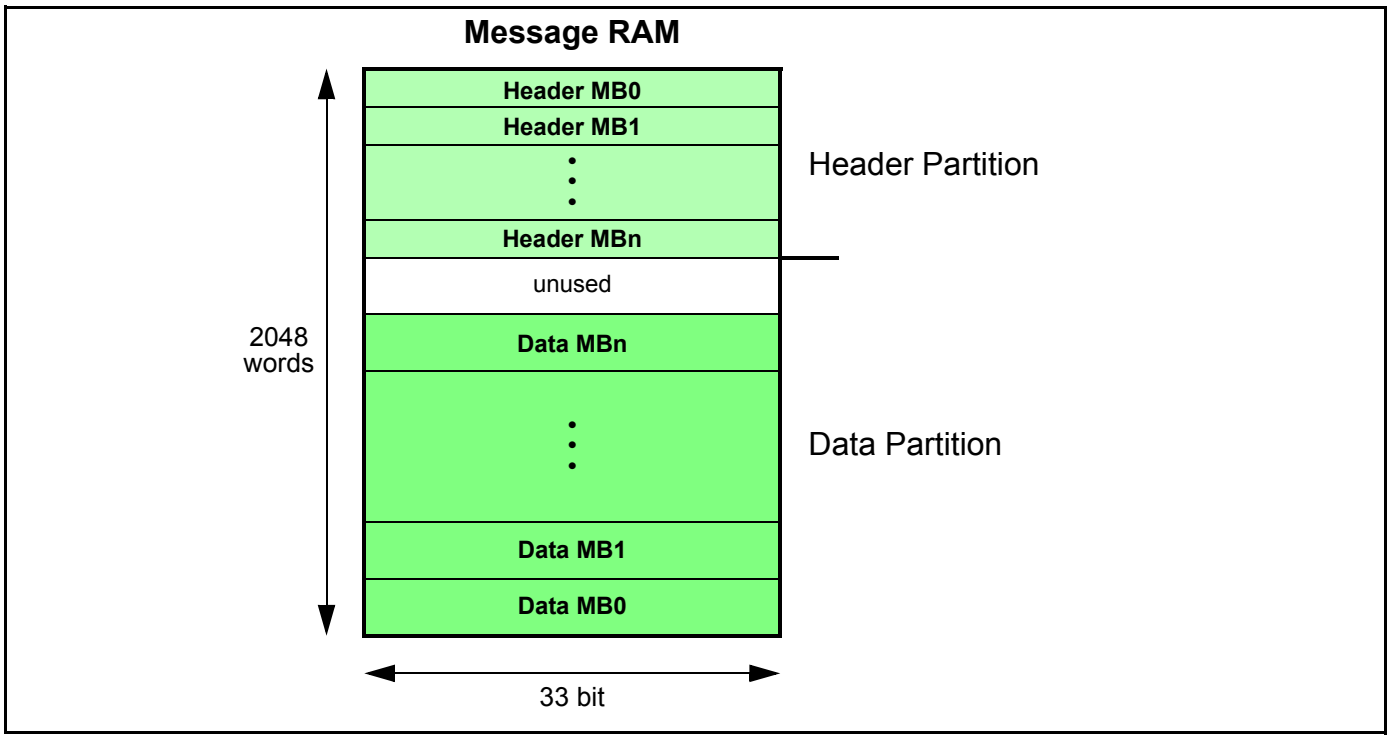
## 19.12. Message RAM

To avoid conflicts between Host access to the Message RAM and FlexRay message reception / transmission, the Host cannot directly access the message buffers in the Message RAM. These accesses are handled via the Input and Output Buffers. The Message RAM is able to store up to 128 message buffers depending on the configured payload length.

The Message RAM is organized 2048 x 33 = 67,584 bit. Each 32-bit word is protected by a parity bit. To achieve the required flexibility with respect to different numbers of data bytes per FlexRay frame (0...254), the Message RAM has a structure as shown in the figure “Configuration example of message buffers in the Message RAM”.

The data partition is allowed to start at Message RAM word number: (**MRC.LCB** + 1) • 4

### Configuration example of message buffers in the Message RAM




**Header Partition**

- Stores header sections of the configured message buffers:
- Supports a maximum of 128 message buffers
- Each message buffer has a header section of four 32+1 bit words
- Header 3 of each message buffer holds the 11-bit data pointer to the respective data section in the data partition

**Data Partition**

Flexible storage of data sections with different length. Some maximum values are:

- 30 message buffers with 254 byte data section each
- Or 56 message buffers with 128 byte data section each
- Or 128 message buffers with 48 byte data section each

**Restriction**: header partition + data partition may not occupy more than 2048 33-bit words.

### 19.12.1. Header Partition

The elements used for configuration of a message buffer as well as the actual message buffer status are stored in the header partition of the Message RAM as listed in the table “Header section of a message buffer in the Message RAM”. Configuration of the header sections of the message buffers is done via IBF (WRHS1...3). Read access to the header sections is done via OBF (RDHS1...3 + MBS). The data pointer has to be calculated by the programmer to define the starting point of the data section for the respective message buffer in the data partition of the Message RAM. The data pointer should not be modified during runtime. For message buffers belonging to the receive FIFO (re)configuration is possible in DEFAULT_CONFIG or CONFIG state only.

The header section of each message buffer occupies four 33-bit words in the header partition of the Message RAM. The header of message buffer 0 starts with the first word in the Message RAM.

For transmit buffers the Header CRC has to be calculated by the Host.

Payload Length Received **PLR[6:0]**, Receive Cycle Count **RCC[5:0]**, Received on Channel Indicator RCI, Startup Frame Indicator **SFI**, Sync Frame Indicator **SYN**, Null Frame Indicator **NFI**, Payload Preamble Indicator **PPI**, and Reserved Bit **RES** are updated from received valid data frames only.

Header word 3 of each configured message buffer holds the respective Message Buffer Status MBS.

**Header section of a message buffer in the Message RAM**

| Bit Word | 32 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | P | | | MBI | TXM | PPIT | CFG | CHB | CHA | | Cycle Code | | | | | | | | | | | | Frame ID | | | | | | | | | | |
| 1 | P | | Payload Length Received | | | | | | | | Payload Length Configured | | | | | | | | | | | | Tx Buffer: Header CRC Configured<br>Rx Buffer: Header CRC Received | | | | | | | | | | |
| 2 | P | | | RES | PPI | NFI | SYN | SFI | RCI | | | Receive Cycle Count | | | | | | | | | | | Data Pointer | | | | | | | | | | |
| 3 | P | | | RESS | PPIS | NFIS | SYNS | SFIS | RCIS | | | Cycle Count Status | | | | | | FTB | FTA | | MLST | ESB | ESA | TCIB | TCIA | SVOB | SVOA | CEOB | CEOA | SEOB | SEOA | VFRB | VFRA |
| … | P | … | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| … | P | … | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

- Frame Configuration
- Filter Configuration
- Message Buffer Control
- Message RAM Configuration
- Updated from received Data Frame
- Message Buffer Status MBS
- Parity Bit
- unused

**Header 1** (word 0)
Write access via WRHS1, read access via RDHS1:
- Frame ID - Slot counter filtering configuration
- Cycle Code - Cycle counter filtering configuration
- CHA, CHB - Channel filtering configuration
- CFG - Message buffer direction configuration: receive / transmit
- PPIT - Payload Preamble Indicator Transmit
- TXM - Transmit mode configuration: single-shot / continuous
- MBI - Message buffer receive / transmit interrupt enable

**Header 2** (word 1)
Write access via WRHS2, read access via RDHS2:
- Header CRC - Transmit Buffer: Configured by the Host (calculated from frame header)
  - Receive Buffer: Updated from received frame
- Payload Length Configured- Length of data section (2-byte words) as configured by the Host
- Payload Length Received - Length of payload segment (2-byte words) stored from received frame

**Header 3** (word 2)
Write access via WRHS3, read access via RDHS3:
- Data Pointer - Pointer to the beginning of the corresponding data section in the data partition

Read access via RDHS3, valid for receive buffers only, updated from received frames:
- Receive Cycle Count - Cycle count from received frame
- RCI - Received on Channel Indicator
- SFI - Startup Frame Indicator
- SYN - Sync Frame Indicator
- NFI - Null Frame Indicator
- PPI - Payload Preamble Indicator
- RES - Reserved bit

### Message Buffer Status MBS (word 3)
Read access via MBS, updated by the CC at the end of the configured slot.
- VFRA - Valid Frame Received on channel A
- VFRB - Valid Frame Received on channel B
- SEOA - Syntax Error Observed on channel A
- SEOB - Syntax Error Observed on channel B
- CEOA - Content Error Observed on channel A
- CEOB - Content Error Observed on channel B
- SVOA - Slot boundary Violation Observed on channel A
- SVOB - Slot boundary Violation Observed on channel B
- TCIA - Transmission Conflict Indication channel A
- TCIB - Transmission Conflict Indication channel B
- ESA - Empty Slot Channel A
- ESB - Empty Slot Channel B
- MLST - Message LoST
- FTA - Frame Transmitted on Channel A
- FTB - Frame Transmitted on Channel B
- Cycle Count Status - Actual cycle count when status was updated
- RCIS - Received on Channel Indicator Status
- SFIS - Startup Frame Indicator Status
- SYNS - Sync Frame Indicator Status
- NFIS - Null Frame Indicator Status
- PPIS - Payload Preamble Indicator Status
- RESS - Reserved bit Status

### 19.12.2. Data Partition

The data partition of the Message RAM stores the data sections of the message buffers configured for reception / transmission as defined in the header partition. The number of data bytes for each message buffer can vary from 0 to 254. To optimize the data transfer between the shift registers of the two FlexRay Protocol Controllers and the Message RAM as well as between the Host interface and the Message RAM, the physical width of the Message RAM is set to 4 bytes plus one parity bit.

The data partition starts after the last word of the header partition. When configuring the message buffers in the Message RAM the programmer has to assure that the data pointers point to addresses within the data partition. The table “Example for structure of the data partition in the Message RAM” below shows an example how the data sections of the configured message buffers can be stored in the data partition of the Message RAM.

The beginning and the end of a message buffer’s data section is determined by the data pointer and the payload length configured in the message buffer’s header section, respectively. This enables a flexible usage of the available RAM space for storage of message buffers with different data length.

If the size of the data section is an odd number of 2-byte words, the remaining 16 bits in the last 32-bit word are unused (see the table “Example for structure of the data partition in the Message RAM” below).

**Example for structure of the data partition in the Message RAM**

| Bit Word | 32 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| … | P | unused | | | | | | | | unused | | | | | | | | unused | | | | | | | | unused | | | | | | | |
| … | P | unused | | | | | | | | unused | | | | | | | | unused | | | | | | | | unused | | | | | | | |
| … | P | **MBn** Data3 | | | | | | | | **MBn** Data2 | | | | | | | | **MBn** Data1 | | | | | | | | **MBn** Data0 | | | | | | | |
| … | P | … | | | | | | | | … | | | | | | | | … | | | | | | | | … | | | | | | | |
| … | P | … | | | | | | | | … | | | | | | | | … | | | | | | | | … | | | | | | | |
| … | P | **MBn** Data(m) | | | | | | | | **MBn** Data(m-1) | | | | | | | | **MBn** Data(m-2) | | | | | | | | **MBn** Data(m-3) | | | | | | | |
| … | P | … | | | | | | | | … | | | | | | | | … | | | | | | | | … | | | | | | | |
| … | P | … | | | | | | | | … | | | | | | | | … | | | | | | | | … | | | | | | | |
| … | P | … | | | | | | | | … | | | | | | | | … | | | | | | | | … | | | | | | | |
| … | P | **MB1** Data3 | | | | | | | | **MB1** Data2 | | | | | | | | **MB1** Data1 | | | | | | | | **MB1** Data0 | | | | | | | |
| … | P | … | | | | | | | | … | | | | | | | | … | | | | | | | | … | | | | | | | |
| … | P | **MB1** Data(k) | | | | | | | | **MB1** Data(k-1) | | | | | | | | **MB1** Data(k-2) | | | | | | | | **MB1** Data(k-3) | | | | | | | |
| 2046 | P | **MB0** Data3 | | | | | | | | **MB0** Data2 | | | | | | | | **MB0** Data1 | | | | | | | | **MB0** Data0 | | | | | | | |
| 2047 | P | unused | | | | | | | | unused | | | | | | | | **MB0** Data5 | | | | | | | | **MB0** Data4 | | | | | | | |

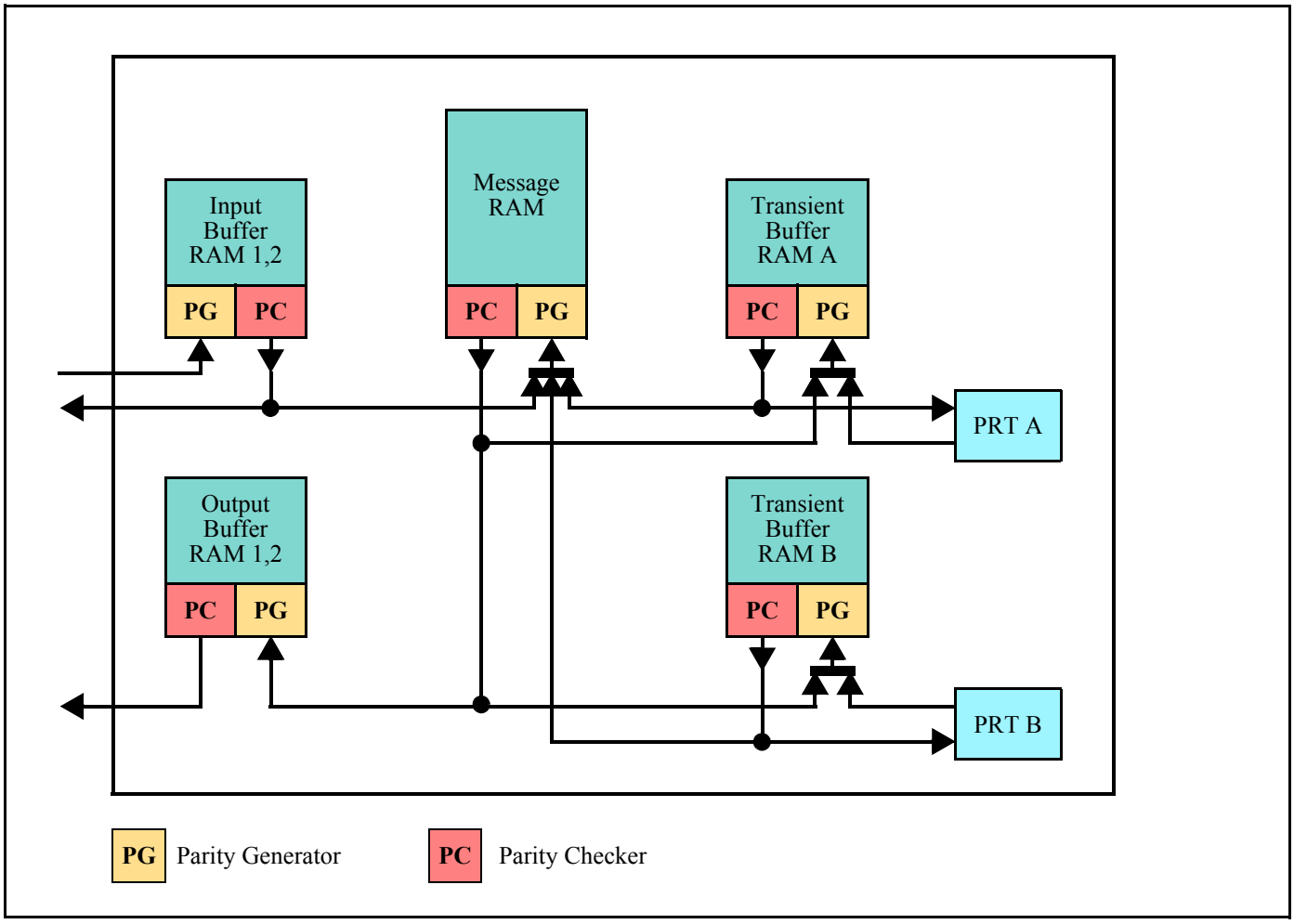
### 19.12.3. Parity Check

There is a parity checking mechanism implemented in the E-Ray core to assure the integrity of the data stored in the seven RAM blocks. The RAM blocks have a parity generator / checker attached as shown in the figure “Parity generation and check”. When data is written to a RAM block, the local parity generator generates the parity bit. The E-Ray core uses an even parity (with an even number of ones in the 32-bit data word a zero parity bit is generated). The parity bit is stored together with the respective data word. The parity is checked each time a data word is read from any of the RAM blocks. The E-Ray core’s internal data buses have a width of 32 bits.

If a parity error is detected, the respective error flag is set. The parity error flags **MHDS.PIBF**, **MHDS.POBF**, **MHDS.PMR**, **MHDS.PTBF1**, **MHDS.PTBF2**, and the faulty message buffer indicators MHDS.FMBD, MHDS.MFMB, MHDS.FMB[6:0] are located in the Message Handler Status register. These single error flags control the error interrupt flag **EIR.PERR**.

The figure “Parity generation and check” shows the data paths between the RAM blocks and the parity generators/checkers.

### Parity generation and check




Note: Parity generator & checker are not part of the RAM blocks, but of the RAM access logic which is part of the E-Ray core.
When a parity error has been detected the following actions will be performed:

**In all cases**

- The respective parity error flag in register MHDS is set
- The parity error flag EIR.PERR is set and, if enabled, a module interrupt to the Host will be generated.

**Additionally in specific cases**

1) Parity error during data transfer from Input Buffer RAM 1,2 ⇒ Message RAM

a) Transfer of header and data section:

- **MHDS.PIBF** bit is set
- **MHDS.FMBD** bit is set to indicate that **MHDS.FMB[6:0]** points to a faulty message buffer
- **MHDS.FMB[6:0]** indicates the number of the faulty message buffer
- Transmit buffer: Transmission request for the respective message buffer is not set

b) Transfer of data section only:

Parity error when reading header section of respective message buffer from Message RAM.

- **MHDS.PMR** bit is set
- **MHDS.FMBD** bit is set to indicate that **MHDS.FMB[6:0]** points to a faulty message buffer
- **MHDS.FMB[6:0]** indicates the number of the faulty message buffer
- The data section of the respective message buffer is not updated
- Transmit buffer: Transmission request for the respective message buffer is not set

2) Parity error during Host reading Input Buffer RAM 1,2
- **MHDS.PIBF** bit is set

3) Parity error during scan of header sections in Message RAM
- **MHDS.PMR** bit is set
- **MHDS.FMBD** bit is set to indicate that **MHDS.FMB[6:0]** points to a faulty message buffer
- **MHDS.FMB[6:0]** indicates the number of the faulty message buffer
- Ignore message buffer (message buffer is skipped)

4) Parity error during data transfer from Message RAM ⇒Transient Buffer RAM 1, 2
- **MHDS.PMR** bit is set
- **MHDS.FMBD** bit is set to indicate that **MHDS.FMB[6:0]** points to a faulty message buffer
- **MHDS.FMB[6:0]** indicates the number of the faulty message buffer
- Frame not transmitted, frames already in transmission are invalidated by setting the frame CRC to zero

5) Parity error during data transfer from Transient Buffer RAM 1, 2 ⇒ Protocol Controller 1, 2
- **MHDS.PTBF1,2** bit is set
- Frames already in transmission are invalidated by setting the frame CRC to zero

6) Parity error during data transfer from Transient Buffer RAM 1, 2 ⇒ Message RAM
a) Parity error when reading header section of respective message buffer from Message RAM:
- **MHDS.PMR** bit is set
- **MHDS.FMBD** bit is set to indicate that **MHDS.FMB[6:0]** points to a faulty message buffer
- **MHDS.FMB[6:0]** indicates the number of the faulty message buffer
- The data section of the respective message buffer is not updated

b) Parity error when reading Transient Buffer RAM 1, 2:
- **MHDS.PTBF1,2** bit is set
- **MHDS.FMBD** bit is set to indicate that **MHDS.FMB[6:0]** points to a faulty message buffer
- **MHDS.FMB[6:0]** indicates the number of the faulty message buffer

7) Parity error during data transfer from Message RAM ⇒Output Buffer RAM
- **MHDS.PMR** bit is set
- **MHDS.FMBD** bit is set to indicate that **MHDS.FMB[6:0]** points to a faulty message buffer
- **MHDS.FMB[6:0]** indicates the number of the faulty message buffer

8) Parity error during Host reading Output Buffer RAM 1,2
- **MHDS.POBF** bit is set

9) Parity error during data read of Transient Buffer RAM 1, 2
When a parity error occurs when the Message Handler reads a frame with network management information (PPI = ’1’) from the Transient Buffer RAM 1, 2 the corresponding network management vector register NMV1...3 is not updated from that frame.

## 19.13. Module Interrupt

In general, interrupts provide a close link to the protocol timing as they are triggered almost immediately when an error or status change is detected by the CC, a frame is received or transmitted, a configured timer interrupt is activated, or a stop watch event occurred. This enables the Host to react very quickly on specific error conditions, status changes, or timer events. On the other hand too many interrupts can cause the Host to miss deadlines required for the application. Therefore the CC supports enable / disable controls for each individual interrupt source separately.

- An interrupt may be triggered when
- An error was detected
- A status flag is set
- A timer reaches a preconfigured value
- A message transfer from Input Buffer to Message RAM or from Message RAM to Output Buffer has completed
- A stop watch event occurred

Tracking status and generating interrupts when a status change or an error occurs are two independent tasks. Regardless of whether an interrupt is enabled or not, the corresponding status is tracked and indicated by the CC. The Host has access to the actual status and error information by reading registers EIR and SIR.

### Module interrupt flags and interrupt line enable

| Register | Bit | Function |
|---|---|---|
| EIR | PEMC | Protocol Error Mode Changed |
| | CNA | Command Not Valid |
| | SFBM | Sync Frames Below Minimum |
| | SFO | Sync Frame Overflow |
| | CCF | Clock Correction Failure |
| | CCL | CHI Command Locked |
| | PERR | Parity Error |
| | RFO | Receive FIFO Overrun |
| | EFA | Empty FIFO Access |
| | IIBA | Illegal Input Buffer Access |
| | IOBA | Illegal Output Buffer Access |
| | MHF | Message Handler Constraints Flag |
| | EDA | Error Detected on Channel A |
| | LTVA | Latest Transmit Violation Channel A |
| | TABA | Transmission Across Boundary Channel A |
| | EDB | Error Detected on Channel B |
| | LTVB | Latest Transmit Violation Channel B |
| | TABB | Transmission Across Boundary Channel B |

| Register | Bit | Function |
|---|---|---|
| SIR | WST | Wakeup Status |
| | CAS | Collision Avoidance Symbol |
| | CYCS | Cycle Start Interrupt |
| | TXI | Transmit Interrupt |
| | RXI | Receive Interrupt |
| | RFNE | Receive FIFO not Empty |
| | RFCL | Receive FIFO Critical Level |
| | NMVC | Network Management Vector Changed |
| | TI0 | Timer Interrupt 0 |
| | TI1 | Timer Interrupt 1 |
| | TIBC | Transfer Input Buffer Completed |
| | TOBC | Transfer Output Buffer Completed |
| | SWE | Stop Watch Event |
| | SUCS | Startup Completed Successfully |
| | MBSI | Message Buffer Status Interrupt |
| | SDS | Start of Dynamic Segment |
| | WUPA | Wakeup Pattern Channel A |
| | MTSA | MTS Received on Channel A |
| | WUPB | Wakeup Pattern Channel B |
| | MTSB | MTS Received on Channel B |
| ILE | EINT0 | Enable Interrupt Line 0 |
| | EINT1 | Enable Interrupt Line 1 |

The interrupt lines to the Host, eray_int0 and eray_int1, are controlled by the enabled interrupts. In addition each of the two interrupt lines can be enabled/disabled separately by programming bit **ILE.EINT0** and **ILE.EINT1**.

The two timer interrupts generated by interrupt timer 0 and 1 are available on pins eray_tint0 and eray_tint1. They can be configured via registers T0C and T1C.

A stop watch event may be triggered via input pin **eray_stpwt**.

The status of the data transfer between IBF/OBF and the Message RAM is signalled on pins **eray_ibusy** and **eray_obusy**. When a transfer has completed bit **SIR.TIBC** or **SIR.TOBC** is set.

# 20. Appendix

## 20.1. Register Bit Overview

| TEST1 | Test Register 1 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0010 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | CERB3 | CERB2 | CERB1 | CERB0 | CERA3 | CERA2 | CERA1 | CERA0 | 0 | 0 | TXENB | TXENA | TXB | TXA | RXB | RXA |
| W | | | | | | | | | | | | | | | | |
| p33 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | AOB | AOA | 0 | 0 | TMC1 | TMC0 | 0 | 0 | ELBE | WRTEN |
| W | | | | | | | | | | | | | | | | |
| **TEST2** | **Test Register 2** | | | | | | | | | | | | | | | |
| 0x0014 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p37 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | RDPB | WRPB | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSEL2 | SSEL1 | SSEL0 | 0 | RS2 | RS1 | RS0 |
| W | | | | | | | | | | | | | | | | |
| **LCK** | **Lock Register** | | | | | | | | | | | | | | | |
| 0x001C | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p39 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | TMK7 | TMK6 | TMK5 | TMK4 | TMK3 | TMK2 | TMK1 | TMK0 | CLK7 | CLK6 | CLK5 | CLK4 | CLK3 | CLK2 | CLK1 | CLK0 |
| **EIR** | **Error Interrupt Register** | | | | | | | | | | | | | | | |
| 0x0020 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | TABB | LTVB | EDB | 0 | 0 | 0 | 0 | 0 | TABA | LTVA | EDA |
| W | | | | | | | | | | | | | | | | |
| p40 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | MHF | IOBA | IIBA | EFA | RFO | PERR | CCL | CCF | SFO | SFBM | CNA | PEMC |
| W | | | | | | | | | | | | | | | | |
| **SIR** | **Status Interrupt Register** | | | | | | | | | | | | | | | |
| 0x0024 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | MTSB | WUPB | 0 | 0 | 0 | 0 | 0 | 0 | MTSA | WUPA |
| W | | | | | | | | | | | | | | | | |
| p43 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | SDS | MBSI | SUCS | SWE | TOBC | TIBC | TI1 | TI0 | NMVC | RFCL | RFNE | RXI | TXI | CYCS | CAS | WST |
| W | | | | | | | | | | | | | | | | |
| **EILS** | **Error Interrupt Line Select** | | | | | | | | | | | | | | | |
| 0x0028 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | TABBL | LTVBL | EDBL | 0 | 0 | 0 | 0 | 0 | TABAL | LTVAL | EDAL |
| W | | | | | | | | | | | | | | | | |
| p46 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | MHFL | IOBAL | IIBAL | EFAL | RFOL | PERRL | CCLL | CCFL | SFOL | SFBML | CNAL | PEMCL |
| W | | | | | | | | | | | | | | | | |

| SILS | Status Interrupt Line Select | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x002C | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | MTSBL | WUPBL | 0 | 0 | 0 | 0 | 0 | 0 | MTSAL | WUPAL |
| W | | | | | | | | | | | | | | | | |
| p47 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | SDSL | MBSIL | SUCSL | SWEL | TOBCL | TIBCL | TI1L | TI0L | NMVCL | RFCLL | RFNEL | RXIL | TXIL | CYCSL | CASL | WSTL |
| W | | | | | | | | | | | | | | | | |

| EIES<br>EIER | Error Interrupt Enable Set<br>Error Interrupt Enable Reset | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0030<br>0x0034 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | TABBE | LTVBE | EDBE | 0 | 0 | 0 | 0 | 0 | TABAE | LTVAE | EDAE |
| W | | | | | | | | | | | | | | | | |
| p48 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | MHFE | IOBAE | IIBAE | EFAE | RFOE | PERRE | CCLE | CCFE | SFOE | SFBME | CNAE | PEMCE |
| W | | | | | | | | | | | | | | | | |

| SIES<br>SIER | Status Interrupt Enable Set<br>Status Interrupt Enable Reset | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0038<br>0x003C | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | MTSBE | WUPBE | 0 | 0 | 0 | 0 | 0 | 0 | MTSAE | WUPAE |
| W | | | | | | | | | | | | | | | | |
| p49 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | SDSE | MBSIE | SUCSE | SWEE | TOBCE | TIBCE | TI1E | TI0E | NMVCE | RFCLE | RFNEE | RXIE | TXIE | CYCSE | CASE | WSTE |
| W | | | | | | | | | | | | | | | | |

| ILE | Interrupt Line Enable | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0040 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p50 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | EINT1 | EINT0 |
| W | | | | | | | | | | | | | | | | |

| T0C | Timer 0 Configuration | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0044 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | T0MO 13 | T0MO 12 | T0MO 11 | T0MO 10 | T0MO9 | T0MO8 | T0MO7 | T0MO6 | T0MO5 | T0MO4 | T0MO3 | T0MO2 | T0MO1 | T0MO0 |
| W | | | | | | | | | | | | | | | | |
| p51 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | T0CC6 | T0CC5 | T0CC4 | T0CC3 | T0CC2 | T0CC1 | T0CC0 | 0 | 0 | 0 | 0 | 0 | 0 | T0MS | T0RC |
| W | | | | | | | | | | | | | | | | |

| T1C | Timer 1 Configuration | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0048 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | T1MC 13 | T1MC 12 | T1MC 11 | T1MC 10 | T1MC9 | T1MC8 | T1MC7 | T1MC6 | T1MC5 | T1MC4 | T1MC3 | T1MC2 | T1MC1 | T1MC0 |
| W | | | | | | | | | | | | | | | | |
| p52 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | T1MS | T1RC |
| W | | | | | | | | | | | | | | | | |

| STPW1 | Stop Watch Register 1 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x004C | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | SMTV 13 | SMTV 12 | SMTV 11 | SMTV 10 | SMTV9 | SMTV8 | SMTV7 | SMTV6 | SMTV5 | SMTV4 | SMTV3 | SMTV2 | SMTV1 | SMTV0 |
| W | | | | | | | | | | | | | | | | |
| p53 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | SCCV5 | SCCV4 | SCCV3 | SCCV2 | SCCV1 | SCCV0 | 0 | EINT1 | EINT0 | EETP | SSWT | EDGE | SWMS | ESWT |
| W | | | | | | | | | | | | | | | | |

| STPW2 | Stop Watch Register 2 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0050 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | SSCVB10 | SSCVB9 | SSCVB8 | SSCVB7 | SSCVB6 | SSCVB5 | SSCVB4 | SSCVB3 | SSCVB2 | SSCVB1 | SSCVB0 |
| W | | | | | | | | | | | | | | | | |
| p54 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | SSCVA10 | SSCVA9 | SSCVA8 | SSCVA7 | SSCVA6 | SSCVA5 | SSCVA4 | SSCVA3 | SSCVA2 | SSCVA1 | SSCVA0 |
| W | | | | | | | | | | | | | | | | |

| SUCC1 | SUC Configuration Register 1 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0080 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | CCHB* | CCHA* | MTSB* | MTSA* | HCSE* | TSM* | WUCS* | PTA4* | PTA3* | PTA2* | PTA1* | PTA0* |
| W | | | | | | | | | | | | | | | | |
| p55 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | CSA4* | CSA3* | CSA2* | CSA1* | CSA0* | 0 | TXSY* | TXST* | PBSY | 0 | 0 | 0 | CMD3 | CMD2 | CMD1 | CMD0 |
| W | | | | | | | | | | | | | | | | |

| SUCC2 | SUC Configuration Register 2 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0084 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | LTN3* | LTN2* | LTN1* | LTN0* | 0 | 0 | 0 | LT20* | LT19* | LT18* | LT17* | LT16* |
| W | | | | | | | | | | | | | | | | |
| p59 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | LT15* | LT14* | LT13* | LT12* | LT11* | LT10* | LT9* | LT8* | LT7* | LT6* | LT5* | LT4* | LT3* | LT2* | LT1* | LT0* |
| W | | | | | | | | | | | | | | | | |

| SUCC3 | SUC Configuration Register 3 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0088 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p60 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | WCF3* | WCF2* | WCF1* | WCF0* | WCP3* | WCP2* | WCP1* | WCP0* |
| W | | | | | | | | | | | | | | | | |

| NEMC | NEM Configuration Register | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x008C | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p61 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | NML3* | NML2* | NML1* | NML0* |
| W | | | | | | | | | | | | | | | | |

| PRTC1 | PRT Configuration Register 1 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0090 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | RWP5* | RWP4* | RWP3* | RWP2* | RWP1* | RWP0* | 0 | RXW8* | RXW7* | RXW6* | RXW5* | RXW4* | RXW3* | RXW2* | RXW1* | RXW0* |
| W | | | | | | | | | | | | | | | | |
| p62 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | BRP1* | BRP0* | SPP1* | SPP0* | 0 | CASM6 | CASM5* | CASM4* | CASM3* | CASM2* | CASM1* | CASM0* | TSST3* | TSST2* | TSST1* | TSST0* |
| W | | | | | | | | | | | | | | | | |

| PRTC2 | PRT Configuration Register 2 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0094 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | TXL5* | TXL4* | TXL3* | TXL2* | TXL1* | TXL0* | TXI7* | TXI6* | TXI5* | TXI4* | TXI3* | TXI2* | TXI1* | TXI0* |
| W | | | | | | | | | | | | | | | | |
| p63 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | RXL5* | RXL4* | RXL3* | RXL2* | RXL1* | RXL0* | 0 | 0 | RXI5* | RXI4* | RXI3* | RXI2* | RXI1* | RXI0* |
| W | | | | | | | | | | | | | | | | |

| MHDC | MHD Configuration Register | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0098 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | SLT12* | SLT11* | SLT10* | SLT9* | SLT8* | SLT7* | SLT6* | SLT5* | SLT4* | SLT3* | SLT2* | SLT1* | SLT0* |
| W | | | | | | | | | | | | | | | | |
| p64 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SFDL6* | SFDL5* | SFDL4* | SFDL3* | SFDL2* | SFDL1* | SFDL0* |
| W | | | | | | | | | | | | | | | | |

| GTUC1 | GTU Configuration Register 1 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x00A0 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | UT19* | UT18* | UT17* | UT16* |
| W | | | | | | | | | | | | | | | | |
| p65 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | UT15* | UT14* | UT13* | UT12* | UT11* | UT10* | UT9* | UT8* | UT7* | UT6* | UT5* | UT4* | UT3* | UT2* | UT1* | UT0* |
| W | | | | | | | | | | | | | | | | |

| GTUC2 | GTU Configuration Register 2 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x00A4 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SNM3* | SNM2* | SNM1* | SNM0* |
| W | | | | | | | | | | | | | | | | |
| p65 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | MPC13* | MPC12* | MPC11* | MPC10* | MPC9* | MPC8* | MPC7* | MPC6* | MPC5* | MPC4* | MPC3* | MPC2* | MPC1* | MPC0* |
| W | | | | | | | | | | | | | | | | |

| GTUC3 | GTU Configuration Register 3 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x00A8 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | MIOB6* | MIOB5* | MIOB4* | MIOB3* | MIOB2* | MIOB1* | MIOB0* | 0 | MIOA6* | MIOA5* | MIOA4* | MIOA3* | MIOA2* | MIOA1* | MIOA0* |
| W | | | | | | | | | | | | | | | | |
| p66 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | UIOB7* | UIOB6* | UIOB5* | UIOB4* | UIOB3* | UIOB2* | UIOB1* | UIOB0* | UIOA7* | UIOA6* | UIOA5* | UIOA4* | UIOA3* | UIOA2* | UIOA1* | UIOA0* |
| W | | | | | | | | | | | | | | | | |

| GTUC4 | GTU Configuration Register 4 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x00AC | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | OCS13* | OCS12* | OCS11* | OCS10* | OCS9* | OCS8* | OCS7* | OCS6* | OCS5* | OCS4* | OCS3* | OCS2* | OCS1* | OCS0* |
| W | | | | | | | | | | | | | | | | |
| p67 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | NIT13* | NIT12* | NIT11* | NIT10* | NIT9* | NIT8* | NIT7* | NIT6* | NIT5* | NIT4* | NIT3* | NIT2* | NIT1* | NIT0* |
| W | | | | | | | | | | | | | | | | |
| **GTUC5** | **GTU Configuration Register 5** | | | | | | | | | | | | | | | |
| 0x00B0 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | DEC7* | DEC6* | DEC5* | DEC4* | DEC3* | DEC2* | DEC1* | DEC0* | 0 | 0 | 0 | CDD4* | CDD3* | CDD2* | CDD1* | CDD0* |
| W | | | | | | | | | | | | | | | | |
| p68 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | DCB7* | DCB6* | DCB5* | DCB4* | DCB3* | DCB2* | DCB1* | DCB0* | DCA7* | DCA6* | DCA5* | DCA4* | DCA3* | DCA2* | DCA1* | DCA0* |
| W | | | | | | | | | | | | | | | | |
| **GTUC6** | **GTU Configuration Register 6** | | | | | | | | | | | | | | | |
| 0x00B4 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | MOD 10* | MOD9* | MOD8* | MOD7* | MOD6* | MOD5* | MOD4* | MOD3* | MOD2* | MOD1* | MOD0* |
| W | | | | | | | | | | | | | | | | |
| p68 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | ASR10* | ASR9* | ASR8* | ASR7* | ASR6* | ASR5* | ASR4* | ASR3* | ASR2* | ASR1* | ASR0* |
| W | | | | | | | | | | | | | | | | |
| **GTUC7** | **GTU Configuration Register 7** | | | | | | | | | | | | | | | |
| 0x00B8 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | NSS9* | NSS8* | NSS7* | NSS6* | NSS5* | NSS4* | NSS3* | NSS2* | NSS1* | NSS0* |
| W | | | | | | | | | | | | | | | | |
| p69 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | SSL9* | SSL8* | SSL7* | SSL6* | SSL5* | SSL4* | SSL3* | SSL2* | SSL1* | SSL0* |
| W | | | | | | | | | | | | | | | | |
| **GTUC8** | **GTU Configuration Register 8** | | | | | | | | | | | | | | | |
| 0x00BC | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | NMS 12* | NMS 11* | NMS 10* | NMS9* | NMS8* | NMS7* | NMS6* | NMS5* | NMS4* | NMS3* | NMS2* | NMS1* | NMS0* |
| W | | | | | | | | | | | | | | | | |
| p69 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | MSL5* | MSL4* | MSL3* | MSL2* | MSL1* | MSL0* |
| W | | | | | | | | | | | | | | | | |
| **GTUC9** | **GTU Configuration Register 9** | | | | | | | | | | | | | | | |
| 0x00C0 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DSI1* | DSI0* |
| W | | | | | | | | | | | | | | | | |
| p70 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | MAPO4* | MAPO3* | MAPO2* | MAPO1* | MAPO0* | 0 | 0 | APO5* | APO4* | APO3* | APO2* | APO1* | APO0* |
| W | | | | | | | | | | | | | | | | |

| GTUC10 | GTU Configuration Register 10 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x00C4 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | MRC 10* | MRC9* | MRC8* | MRC7* | MRC6* | MRC5* | MRC4* | MRC3* | MRC2* | MRC1* | MRC0* |
| W | | | | | | | | | | | | | | | | |
| p70 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | MOC 13* | MOC 12* | MOC 11* | MOC 10* | MOC9* | MOC8* | MOC7* | MOC6* | MOC5* | MOC4* | MOC3* | MOC2* | MOC1* | MOC0* |
| W | | | | | | | | | | | | | | | | |

| GTUC11 | GTU Configuration Register 11 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x00C8 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | ERC2* | ERC1* | ERC0* | 0 | 0 | 0 | 0 | 0 | EOC2* | EOC1* | EOC0* |
| W | | | | | | | | | | | | | | | | |
| p71 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | ERCC1 | ERCC0 | 0 | 0 | 0 | 0 | 0 | 0 | EOCC1 | EOCC0 |
| W | | | | | | | | | | | | | | | | |

| CCSV | CC Status Vector | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0100 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | PSL5 | PSL4 | PSL3 | PSL2 | PSL1 | PSL0 | RCA4 | RCA3 | RCA2 | RCA1 | RCA0 | WSV2 | WSV1 | WSV0 |
| W | | | | | | | | | | | | | | | | |
| p72 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | CSI | CSAI | CSNI | 0 | 0 | SLM1 | SLM0 | HRQ | FSI | POCS5 | POCS4 | POCS3 | POCS2 | POCS1 | POCS0 |
| W | | | | | | | | | | | | | | | | |

| CCEV | CC Error Vector | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0104 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p75 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | PTAC4 | PTAC3 | PTAC2 | PTAC1 | PTAC0 | ERRM1 | ERRM0 | 0 | 0 | CCFC3 | CCFC2 | CCFC1 | CCFC0 |
| W | | | | | | | | | | | | | | | | |

| SCV | Slot Counter Value | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0110 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | SCCB10 | SCCB9 | SCCB8 | SCCB7 | SCCB6 | SCCB5 | SCCB4 | SCCB3 | SCCB2 | SCCB1 | SCCB0 |
| W | | | | | | | | | | | | | | | | |
| p76 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | SCCA10 | SCCA9 | SCCA8 | SCCA7 | SCCA6 | SCCA5 | SCCA4 | SCCA3 | SCCA2 | SCCA1 | SCCA0 |
| W | | | | | | | | | | | | | | | | |

| MTCCV | Macrotick and Cycle Counter Value | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0114 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | CCV5 | CCV4 | CCV3 | CCV2 | CCV1 | CCV0 |
| W | | | | | | | | | | | | | | | | |
| p76 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | MTV13 | MTV12 | MTV11 | MTV10 | MTV9 | MTV8 | MTV7 | MTV6 | MTV5 | MTV4 | MTV3 | MTV2 | MTV1 | MTV0 |
| W | | | | | | | | | | | | | | | | |

| RCV | Rate Correction Value | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0118 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p77 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | RCV11 | RCV10 | RCV9 | RCV8 | RCV7 | RCV6 | RCV5 | RCV4 | RCV3 | RCV2 | RCV1 | RCV0 |
| W | | | | | | | | | | | | | | | | |

| OCV | Offset Correction Value | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x011C | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | OCV18 | OCV17 | OCV16 |
| W | | | | | | | | | | | | | | | | |
| p77 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | OCV15 | OCV14 | OCV13 | OCV12 | OCV11 | OCV10 | OCV9 | OCV8 | OCV7 | OCV6 | OCV5 | OCV4 | OCV3 | OCV2 | OCV1 | OCV0 |
| W | | | | | | | | | | | | | | | | |

| SFS | Sync Frame Status | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0120 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RCLR | MRCS | OCLR | MOCS |
| W | | | | | | | | | | | | | | | | |
| p78 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | VSBO3 | VSBO2 | VSBO1 | VSBO0 | VSBE3 | VSBE2 | VSBE1 | VSBE0 | VSAO3 | VSAO2 | VSAO1 | VSAO0 | VSAE3 | VSAE2 | VSAE1 | VSAE0 |
| W | | | | | | | | | | | | | | | | |

| SWNIT | Symbol Window and NIT Status | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0124 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p79 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | SBNB | SENB | SBNA | SENA | MTSB | MTSA | TCSB | SBSB | SESB | TCSA | SBSA | SESA |
| W | | | | | | | | | | | | | | | | |

| ACS | Aggregated Channel Status | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0128 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p81 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | SBVB | CIB | CEDB | SEDB | VFRB | 0 | 0 | 0 | SBVA | CIA | CEDA | SEDA | VFRA |
| W | | | | | | | | | | | | | | | | |

| ESIDn | Even Sync ID [1...15] | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0130 to 0x0168 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p83 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | RXEB | RXEA | 0 | 0 | 0 | 0 | EID9 | EID8 | EID7 | EID6 | EID5 | EID4 | EID3 | EID2 | EID1 | EID0 |
| W | | | | | | | | | | | | | | | | |

| OSIDn | Odd Sync ID [1…15] | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0170 to 0x01A8 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p84 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | RXOB | RXOA | 0 | 0 | 0 | 0 | OID9 | OID8 | OID7 | OID6 | OID5 | OID4 | OID3 | OID2 | OID1 | OID0 |
| W | | | | | | | | | | | | | | | | |
| **NMVn** | **Network Management Vector [1…3]** | | | | | | | | | | | | | | | |
| 0x01B0 to 0x01B8 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | NM31 | NM30 | NM29 | NM28 | NM27 | NM26 | NM25 | NM24 | NM23 | NM22 | NM21 | NM20 | NM19 | NM18 | NM17 | NM16 |
| W | | | | | | | | | | | | | | | | |
| p85 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | NM15 | NM14 | NM13 | NM12 | NM11 | NM10 | NM9 | NM8 | NM7 | NM6 | NM5 | NM4 | NM3 | NM2 | NM1 | NM0 |
| W | | | | | | | | | | | | | | | | |
| **MRC** | **Message RAM Configuration** | | | | | | | | | | | | | | | |
| 0x0300 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | SPLM* | SEC1* | SEC0* | LCB7* | LCB6* | LCB5* | LCB4* | LCB3* | LCB2* | LCB1* | LCB0* |
| W | | | | | | | | | | | | | | | | |
| p86 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | FFB7* | FFB6* | FFB5* | FFB4* | FFB3* | FFB2* | FFB1* | FFB0* | FDB7* | FDB6* | FDB5* | FDB4* | FDB3* | FDB2* | FDB1* | FDB0* |
| W | | | | | | | | | | | | | | | | |
| **FRF** | **FIFO Rejection Filter** | | | | | | | | | | | | | | | |
| 0x0304 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RNF* | RSS* | CYF6* | CYF5* | CYF4* | CYF3* | CYF2* | CYF1* | CYF0* |
| W | | | | | | | | | | | | | | | | |
| p88 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | FID10* | FID9* | FID8* | FID7* | FID6* | FID5* | FID4* | FID3* | FID2* | FID1* | FID0* | CH1* | CH0* |
| W | | | | | | | | | | | | | | | | |
| **FRFM** | **FIFO Rejection Filter Mask** | | | | | | | | | | | | | | | |
| 0x0308 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p89 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | MFID 10* | MFID 9* | MFID 8* | MFID 7* | MFID 6* | MFID 5* | MFID 4* | MFID 3* | MFID 2* | MFID 1* | MFID 0* | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| **FCL** | **FIFO Critical Level** | | | | | | | | | | | | | | | |
| 0x030C | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p89 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | CL7* | CL6* | CL5* | CL4* | CL3* | CL2* | CL1* | CL0* |
| W | | | | | | | | | | | | | | | | |

| MHDS | Message Handler Status | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0310 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | MBU6 | MBU5 | MBU4 | MBU3 | MBU2 | MBU1 | MBU0 | 0 | MBT6 | MBT5 | MBT4 | MBT3 | MBT2 | MBT1 | MBT0 |
| W | | | | | | | | | | | | | | | | |
| p90 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | FMB6 | FMB5 | FMB4 | FMB3 | FMB2 | FMB1 | FMB0 | CRAM | MFMB | FMBD | PTBF2 | PTBF1 | PMR | POBF | PIBF |
| W | | | | | | | | | | | | | | | | |

| LDTS | Last Dynamic Transmit Slot | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0314 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | LDTB10 | LDTB9 | LDTB8 | LDTB7 | LDTB6 | LDTB5 | LDTB4 | LDTB3 | LDTB2 | LDTB1 | LDTB0 |
| W | | | | | | | | | | | | | | | | |
| p91 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | LDTA10 | LDTA9 | LDTA8 | LDTA7 | LDTA6 | LDTA5 | LDTA4 | LDTA3 | LDTA2 | LDTA1 | LDTA0 |
| W | | | | | | | | | | | | | | | | |

| FSR | FIFO Status Register | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0318 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p92 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | RFFL7 | RFFL6 | RFFL5 | RFFL4 | RFFL3 | RFFL2 | RFFL1 | RFFL0 | 0 | 0 | 0 | 0 | 0 | RFO | RFCL | RFNE |
| W | | | | | | | | | | | | | | | | |

| MHDF | Message Handler Constraints Flags | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x031C | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p93 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | WAHP | 0 | 0 | TBFB | TBFA | FNFB | FNFA | SNUB | SNUA |
| W | | | | | | | | | | | | | | | | |

| TXRQ1 | Transmission Request 1 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0320 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | TXR31 | TXR30 | TXR29 | TXR28 | TXR27 | TXR26 | TXR25 | TXR24 | TXR23 | TXR22 | TXR21 | TXR20 | TXR19 | TXR18 | TXR17 | TXR16 |
| W | | | | | | | | | | | | | | | | |
| p95 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | TXR15 | TXR14 | TXR13 | TXR12 | TXR11 | TXR10 | TXR9 | TXR8 | TXR7 | TXR6 | TXR5 | TXR4 | TXR3 | TXR2 | TXR1 | TXR0 |
| W | | | | | | | | | | | | | | | | |

| TXRQ2 | Transmission Request 2 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0324 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | TXR63 | TXR62 | TXR61 | TXR60 | TXR59 | TXR58 | TXR57 | TXR56 | TXR55 | TXR54 | TXR53 | TXR52 | TXR51 | TXR50 | TXR49 | TXR48 |
| W | | | | | | | | | | | | | | | | |
| p95 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | TXR47 | TXR46 | TXR45 | TXR44 | TXR43 | TXR42 | TXR41 | TXR40 | TXR39 | TXR38 | TXR37 | TXR36 | TXR35 | TXR34 | TXR33 | TXR32 |
| W | | | | | | | | | | | | | | | | |

| TXRQ3 | Transmission Request 3 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0328 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | TXR95 | TXR94 | TXR93 | TXR92 | TXR91 | TXR90 | TXR89 | TXR88 | TXR87 | TXR86 | TXR85 | TXR84 | TXR83 | TXR82 | TXR81 | TXR80 |
| W | | | | | | | | | | | | | | | | |
| p95 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | TXR79 | TXR78 | TXR77 | TXR76 | TXR75 | TXR74 | TXR73 | TXR72 | TXR71 | TXR70 | TXR69 | TXR68 | TXR67 | TXR66 | TXR65 | TXR64 |
| W | | | | | | | | | | | | | | | | |
| **TXRQ4** | **Transmission Request 4** | | | | | | | | | | | | | | | |
| 0x032C | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | TXR127 | TXR126 | TXR125 | TXR124 | TXR123 | TXR122 | TXR121 | TXR120 | TXR119 | TXR118 | TXR117 | TXR116 | TXR115 | TXR114 | TXR113 | TXR112 |
| W | | | | | | | | | | | | | | | | |
| p95 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | TXR111 | TXR110 | TXR109 | TXR108 | TXR107 | TXR106 | TXR105 | TXR104 | TXR103 | TXR102 | TXR101 | TXR100 | TXR99 | TXR98 | TXR97 | TXR96 |
| W | | | | | | | | | | | | | | | | |
| **NDAT1** | **New Data 1** | | | | | | | | | | | | | | | |
| 0x0330 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | ND31 | ND30 | ND29 | ND28 | ND27 | ND26 | ND25 | ND24 | ND23 | ND22 | ND21 | ND20 | ND19 | ND18 | ND17 | ND16 |
| W | | | | | | | | | | | | | | | | |
| p96 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | ND15 | ND14 | ND13 | ND12 | ND11 | ND10 | ND9 | ND8 | ND7 | ND6 | ND5 | ND4 | ND3 | ND2 | ND1 | ND0 |
| W | | | | | | | | | | | | | | | | |
| **NDAT2** | **New Data 2** | | | | | | | | | | | | | | | |
| 0x0334 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | ND63 | ND62 | ND61 | ND60 | ND59 | ND58 | ND57 | ND56 | ND55 | ND54 | ND53 | ND52 | ND51 | ND50 | ND49 | ND48 |
| W | | | | | | | | | | | | | | | | |
| p96 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | ND47 | ND46 | ND45 | ND44 | ND43 | ND42 | ND41 | ND40 | ND39 | ND38 | ND37 | ND36 | ND35 | ND34 | ND33 | ND32 |
| W | | | | | | | | | | | | | | | | |
| **NDAT3** | **New Data 3** | | | | | | | | | | | | | | | |
| 0x0338 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | ND95 | ND94 | ND93 | ND92 | ND91 | ND90 | ND89 | ND88 | ND87 | ND86 | ND85 | ND84 | ND83 | ND82 | ND81 | ND80 |
| W | | | | | | | | | | | | | | | | |
| p96 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | ND79 | ND78 | ND77 | ND76 | ND75 | ND74 | ND73 | ND72 | ND71 | ND70 | ND69 | ND68 | ND67 | ND66 | ND65 | ND64 |
| W | | | | | | | | | | | | | | | | |
| **NDAT4** | **New Data 4** | | | | | | | | | | | | | | | |
| 0x033C | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | ND127 | ND126 | ND125 | ND124 | ND123 | ND122 | ND121 | ND120 | ND119 | ND118 | ND117 | ND116 | ND115 | ND114 | ND113 | ND112 |
| W | | | | | | | | | | | | | | | | |
| p96 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | ND111 | ND110 | ND109 | ND108 | ND107 | ND106 | ND105 | ND104 | ND103 | ND102 | ND101 | ND100 | ND99 | ND98 | ND97 | ND96 |
| W | | | | | | | | | | | | | | | | |

| MBSC1 | Message Buffer Status Changed 1 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0340 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | MBC31 | MBC30 | MBC29 | MBC28 | MBC27 | MBC26 | MBC25 | MBC24 | MBC23 | MBC22 | MBC21 | MBC20 | MBC19 | MBC18 | MBC17 | MBC16 |
| W | | | | | | | | | | | | | | | | |
| p97 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | MBC15 | MBC14 | MBC13 | MBC12 | MBC11 | MBC10 | MBC9 | MBC8 | MBC7 | MBC6 | MBC5 | MBC4 | MBC3 | MBC2 | MBC1 | MBC0 |
| W | | | | | | | | | | | | | | | | |
| **MBSC2** | **Message Buffer Status Changed 2** | | | | | | | | | | | | | | | |
| 0x0344 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | MBC63 | MBC62 | MBC61 | MBC60 | MBC59 | MBC58 | MBC57 | MBC56 | MBC55 | MBC54 | MBC53 | MBC52 | MBC51 | MBC50 | MBC49 | MBC48 |
| W | | | | | | | | | | | | | | | | |
| p97 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | MBC47 | MBC46 | MBC45 | MBC44 | MBC43 | MBC42 | MBC41 | MBC40 | MBC39 | MBC38 | MBC37 | MBC36 | MBC35 | MBC34 | MBC33 | MBC32 |
| W | | | | | | | | | | | | | | | | |
| **MBSC3** | **Message Buffer Status Changed 3** | | | | | | | | | | | | | | | |
| 0x0348 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | MBC95 | MBC94 | MBC93 | MBC92 | MBC91 | MBC90 | MBC89 | MBC88 | MBC87 | MBC86 | MBC85 | MBC84 | MBC83 | MBC82 | MBC81 | MBC80 |
| W | | | | | | | | | | | | | | | | |
| p97 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | MBC79 | MBC78 | MBC77 | MBC76 | MBC75 | MBC74 | MBC73 | MBC72 | MBC71 | MBC70 | MBC69 | MBC68 | MBC67 | MBC66 | MBC65 | MBC64 |
| W | | | | | | | | | | | | | | | | |
| **MBSC4** | **Message Buffer Status Changed 4** | | | | | | | | | | | | | | | |
| 0x034C | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | MBC127 | MBC126 | MBC125 | MBC124 | MBC123 | MBC122 | MBC121 | MBC120 | MBC119 | MBC118 | MBC117 | MBC116 | MBC115 | MBC114 | MBC113 | MBC112 |
| W | | | | | | | | | | | | | | | | |
| p97 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | MBC111 | MBC110 | MBC109 | MBC108 | MBC107 | MBC106 | MBC105 | MBC104 | MBC103 | MBC102 | MBC101 | MBC100 | MBC99 | MBC98 | MBC97 | MBC96 |
| W | | | | | | | | | | | | | | | | |
| **CREL** | **Core Release Register** | | | | | | | | | | | | | | | |
| 0x03F0 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | REL3 | REL2 | REL1 | REL0 | STEP7 | STEP6 | STEP5 | STEP4 | STEP3 | STEP2 | STEP1 | STEP0 | YEAR3 | YEAR2 | YEAR1 | YEAR0 |
| W | | | | | | | | | | | | | | | | |
| p98 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | MON7 | MON6 | MON5 | MON4 | MON3 | MON2 | MON1 | MON0 | DAY7 | DAY6 | DAY5 | DAY4 | DAY3 | DAY2 | DAY1 | DAY0 |
| W | | | | | | | | | | | | | | | | |
| **ENDN** | **Endian Register** | | | | | | | | | | | | | | | |
| 0x03F4 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | ETV31 | ETV30 | ETV29 | ETV28 | ETV27 | ETV26 | ETV25 | ETV24 | ETV23 | ETV22 | ETV21 | ETV20 | ETV19 | ETV18 | ETV17 | ETV16 |
| W | | | | | | | | | | | | | | | | |
| p98 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | ETV15 | ETV14 | ETV13 | ETV12 | ETV11 | ETV10 | ETV9 | ETV8 | ETV7 | ETV6 | ETV5 | ETV4 | ETV3 | ETV2 | ETV1 | ETV0 |
| W | | | | | | | | | | | | | | | | |

| WRDSn | Write Data Section [1…64] | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0400 to 0x04FC | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | MD31 | MD30 | MD29 | MD28 | MD27 | MD26 | MD25 | MD24 | MD23 | MD22 | MD21 | MD20 | MD19 | MD18 | MD17 | MD16 |
| W | | | | | | | | | | | | | | | | |
| p99 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | MD15 | MD14 | MD13 | MD12 | MD11 | MD10 | MD9 | MD8 | MD7 | MD6 | MD5 | MD4 | MD3 | MD2 | MD1 | MD0 |
| W | | | | | | | | | | | | | | | | |

| WRHS1 | Write Header Section 1 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0500 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | MBI | TXM | PPIT | CFG | CHB | CHA | 0 | CYC6 | CYC5 | CYC4 | CYC3 | CYC2 | CYC1 | CYC0 |
| W | | | | | | | | | | | | | | | | |
| p100 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | FID10 | FID9 | FID8 | FID7 | FID6 | FID5 | FID4 | FID3 | FID2 | FID1 | FID0 |
| W | | | | | | | | | | | | | | | | |

| WRHS2 | Write Header Section 2 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0504 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PLC6 | PLC5 | PLC4 | PLC3 | PLC2 | PLC1 | PLC0 |
| W | | | | | | | | | | | | | | | | |
| p101 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | CRC10 | CRC9 | CRC8 | CRC7 | CRC6 | CRC5 | CRC4 | CRC3 | CRC2 | CRC1 | CRC0 |
| W | | | | | | | | | | | | | | | | |

| WRHS3 | Write Header Section 3 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0508 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| p101 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | DP10* | DP9* | DP8* | DP7* | DP6* | DP5* | DP4* | DP3* | DP2* | DP1* | DP0* |
| W | | | | | | | | | | | | | | | | |

| IBCM | Input Buffer Command Mask | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0510 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | STXRS | LDSS | LHSS |
| W | | | | | | | | | | | | | | | | |
| p102 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | STXRH | LDSH | LHSH |
| W | | | | | | | | | | | | | | | | |

| IBCR | Input Buffer Command Request | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0514 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | IBSYS | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | IBRS6 | IBRS5 | IBRS4 | IBRS3 | IBRS2 | IBRS1 | IBRS0 |
| W | | | | | | | | | | | | | | | | |
| p103 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | IBSYH | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | IBRH6 | IBRH5 | IBRH4 | IBRH3 | IBRH2 | IBRH1 | IBRH0 |
| W | | | | | | | | | | | | | | | | |

| RDDSn | Read Data Section [1…64] | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0600 to 0x06FC | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | MD31 | MD30 | MD29 | MD28 | MD27 | MD26 | MD25 | MD24 | MD23 | MD22 | MD21 | MD20 | MD19 | MD18 | MD17 | MD16 |
| W | | | | | | | | | | | | | | | | |
| p104 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | MD15 | MD14 | MD13 | MD12 | MD11 | MD10 | MD9 | MD8 | MD7 | MD6 | MD5 | MD4 | MD3 | MD2 | MD1 | MD0 |
| W | | | | | | | | | | | | | | | | |
| **RDHS1** | **Read Header Section 1** | | | | | | | | | | | | | | | |
| 0x0700 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | MBI | TXM | PPIT | CFG | CHB | CHA | 0 | CYC6 | CYC5 | CYC4 | CYC3 | CYC2 | CYC1 | CYC0 |
| W | | | | | | | | | | | | | | | | |
| p105 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | FID10 | FID9 | FID8 | FID7 | FID6 | FID5 | FID4 | FID3 | FID2 | FID1 | FID0 |
| W | | | | | | | | | | | | | | | | |
| **RDHS2** | **Read Header Section 2** | | | | | | | | | | | | | | | |
| 0x0704 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | PLR6 | PLR5 | PLR4 | PLR3 | PLR2 | PLR1 | PLR0 | 0 | PLC6 | PLC5 | PLC4 | PLC3 | PLC2 | PLC1 | PLC0 |
| W | | | | | | | | | | | | | | | | |
| p106 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | CRC10 | CRC9 | CRC8 | CRC7 | CRC6 | CRC5 | CRC4 | CRC3 | CRC2 | CRC1 | CRC0 |
| W | | | | | | | | | | | | | | | | |
| **RDHS3** | **Read Header Section 3** | | | | | | | | | | | | | | | |
| 0x0708 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | RES | PPI | NFI | SYN | SFI | RCI | 0 | 0 | RCC5 | RCC4 | RCC3 | RCC2 | RCC1 | RCC0 |
| W | | | | | | | | | | | | | | | | |
| p107 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | DP10 | DP9 | DP8 | DP7 | DP6 | DP5 | DP4 | DP3 | DP2 | DP1 | DP0 |
| W | | | | | | | | | | | | | | | | |
| **MBS** | **Message Buffer Status** | | | | | | | | | | | | | | | |
| 0x070C | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | RESS | PPIS | NFIS | SYNS | SFIS | RCIS | 0 | 0 | CCS5 | CCS4 | CCS3 | CCS2 | CCS1 | CCS0 |
| W | | | | | | | | | | | | | | | | |
| p108 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | FTB | FTA | 0 | MLST | ESB | ESA | TCIB | TCIA | SVOB | SVOA | CEOB | CEOA | SEOB | SEOA | VFRB | VFRA |
| W | | | | | | | | | | | | | | | | |
| **OBCM** | **Output Buffer Command Mask** | | | | | | | | | | | | | | | |
| 0x0710 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RDSH | RHSH |
| W | | | | | | | | | | | | | | | | |
| p111 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | | | | | | | | | | | | | | | RDSS | RHSS |
| W | | | | | | | | | | | | | | | | |

| OBCR | Output Buffer Command Request | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0714 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | OBRH6 | OBRH5 | OBRH4 | OBRH3 | OBRH2 | OBRH1 | OBRH0 |
| W | | | | | | | | | | | | | | | | |
| p112 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | OBSYS | 0 | 0 | 0 | 0 | 0 | REQ | VIEW | | OBRS6 | OBRS5 | OBRS4 | OBRS3 | OBRS2 | OBRS1 | OBRS0 |
| W | | | | | | | | | | | | | | | | |

## 20.2. Assignment of FlexRay Configuration Parameters

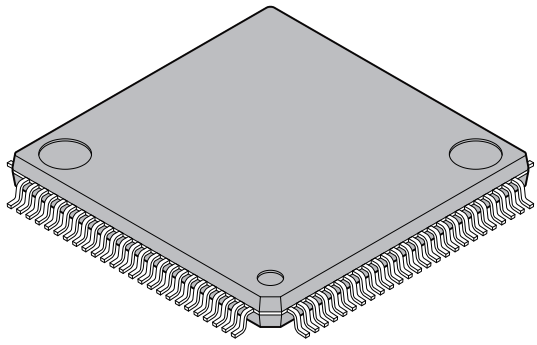
| Parameter | Bit(field) | Page |
|---|---|---|
| pKeySlotusedForStartup | SUCC1.TXST | 55 |
| pKeySlotUsedForSync | SUCC1.TXSY | 55 |
| gColdStartAttempts | SUCC1.CSA[4:0] | 55 |
| pAllowPassiveToActive | SUCC1.PTA[4:0] | 55 |
| pWakeupChannel | SUCC1.WUCS | 55 |
| pSingleSlotEnabled | SUCC1.TSM | 55 |
| pAllowHaltDueToClock | SUCC1.HCSE | 55 |
| pChannels | SUCC1.CCHA SUCC1.CCHB | 55 |
| pdListenTimeOut | SUCC2.LT[20:0] | 59 |
| gListenNoise | SUCC2.LTN[3:0] | 59 |
| gMaxWithoutClockCorrectionPassive | SUCC3.WCP[3:0] | 60 |
| gMaxWithoutClockCorrectionFatal | SUCC3.WCF[3:0] | 60 |
| gNetworkManagementVectorLength | NEMC.NML[3:0] | 61 |
| gdTSSTransmitter | PRTC1.TSST[3:0] | 62 |
| gdCASRxLowMax | PRTC1.CASM[6:0] | 62 |
| gdSampleClockPeriod | PRTC1.BRP[1:0] | 62 |
| pSamplesPerMicrotick | PRTC1.BRP[1:0] | 62 |
| gdWakeupSymbolRxWindow | PRTC1.RXW[8:0] | 62 |
| pWakeupPattern | PRTC1.RWP[5:0] | 62 |
| gdWakeupSymbolRxIdle | PRTC2.RXI[5:0] | 63 |
| gdWakeupSymbolRxLow | PRTC2.RXL[5:0] | 63 |
| gdWakeupSymbolTxIdle | PRTC2.TXI[7:0] | 63 |
| gdWakeupSymbolTxLow | PRTC2.TXL[5:0] | 63 |
| gPayloadLengthStatic | MHDC.SFDL[6:0] | 64 |
| pLatestTx | MHDC.SLT[12:0] | 64 |
| pMicroPerCycle | GTUC1.UT[19:0] | 65 |
| gMacroPerCycle | GTUC2.MPC[13:0] | 65 |
| gSyncNodeMax | GTUC2.SNM[3:0] | 65 |
| pMicroInitialOffset[A] | GTUC3.UIOA[7:0] | 66 |
| pMicroInitialOffset[B] | GTUC3.UIOB[7:0] | 66 |
| pMacroInitialOffset[A] | GTUC3.MIOA[6:0] | 66 |
| pMacroInitialOffset[B] | GTUC3.MIOB[6:0] | 66 |

| Parameter | Bit(field) | Page |
|---|---|---|
| gdNIT | GTUC4.NIT[13:0] | 67 |
| gOffsetCorrectionStart | GTUC4.OCS[13.0] | 67 |
| pDelayCompensation[A] | GTUC5.DCA[7:0] | 68 |
| pDelayCompensation[B] | GTUC5.DCB[7:0] | 68 |
| pClusterDriftDamping | GTUC5.CDD[4:0] | 68 |
| pDecodingCorrection | GTUC5.DEC[7:0] | 68 |
| pdAcceptedStartupRange | GTUC6.ASR[10:0] | 68 |
| pdMaxDrift | GTUC6.MOD[10:0] | 68 |
| gdStaticSlot | GTUC7.SSL[9:0] | 69 |
| gNumberOfStaticSlots | GTUC7.NSS[9:0] | 69 |
| gdMinislot | GTUC8.MSL[5:0] | 69 |
| gNumberOfMinislots | GTUC8.NMS[12:0] | 69 |
| gdActionPointOffset | GTUC9.APO[5:0] | 70 |
| gdMinislotActionPoint | GTUC9.MAPO[4:0] | 70 |
| gdDynamicSlotIdlePhase | GTUC9.DSI[1:0] | 70 |
| pOffsetCorrectionOut | GTUC10.MOC[13:0] | 70 |
| pRateCorrectionOut | GTUC10.MRC[10:0] | 70 |
| pExternOffsetCorrection | GTUC11.EOC[2:0] | 71 |
| pExternRateCorrection | GTUC11.ERC[2:0] | 71 |

## 21. Ordering Information

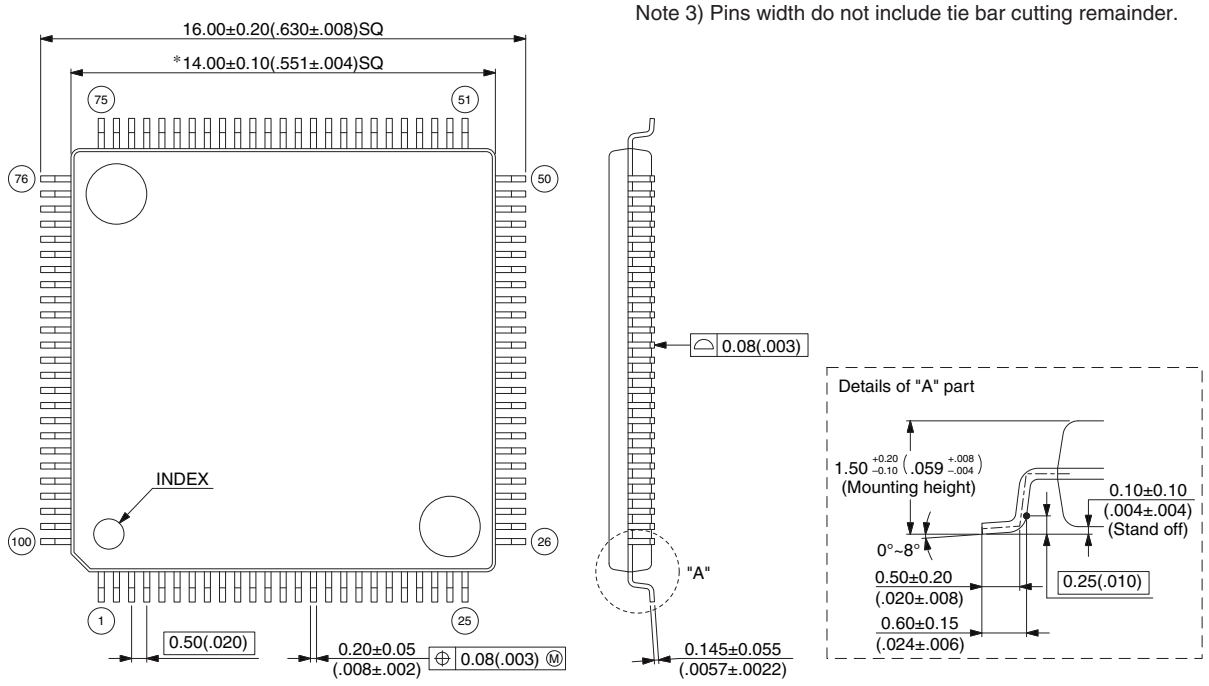
| Part number | Package | Remarks |
|---|---|---|
| MB91F465XAPMC-GE1 | 100-pin plastic QFP<br>(FPT-100P-M20) | Lead-free package |

## 22. Package Dimension

| 100-pin plastic LQFP | | |
|---|---|---|
|  | Lead pitch | 0.50 mm |
| | Package width × package length | 14.0 mm × 14.0 mm |
| | Lead shape | Gullwing |
| | Sealing method | Plastic mold |
| | Mounting height | 1.70 mm Max |
| | Weight | 0.65 g |
| (FPT-100P-M20) | Code (Reference) | P-LFQFP100-14×14-0.50 |

100-pin plastic LQFP
(FPT-100P-M20)

Note 1) * : These dimensions do not include resin protrusion.
Note 2) Pins width and pins thickness include plating thickness.
Note 3) Pins width do not include tie bar cutting remainder.




© 2005 -2008 FUJITSU MICROELECTRONICS LIMITED F100031S-c-3-3

Dimensions in mm (inches).
Note: The values in parentheses are reference values

## 23. Revision History

| Version | Date | Remark |
|---|---|---|
| 2.0 | 2008-10-15 | Initial version |
| 2.1 | ------- | Pinout: Increased the size for better readability<br>Notes on PS register: Updated for better readability<br>Embedded Program/Data Memory:<br>- Added note about flash memory operation mode switching<br>Flash Security:<br>- Corrected FSV2 table header<br>- added Notes About Flash Memory CRC Calculation<br>DC Characteristics:<br>- Corrected pin name of Analog input leakage current into ANn and added footnote<br>A/D converter characteristics<br>- added Offset between input channels<br>- corrected “linearity error” into “nonlinearity error” |

## 24. Main Changes

**Spansion Publication Number: DS07-16611**

| Page | Section | Change Results |
|---|---|---|
| Rev. 1.0 | | |
| 84 | Electrical Characteristics | Changed the table in “4. A/D converter characteristics”.<br>(LSB → V<br>AVRL + 2.5 → AVRL + 2.5 LSB<br>AVRL + 0.5 → AVRL + 0.5 LSB<br>AVRL - 1.5 → AVRL - 1.5 LSB )<br>(LSB → V<br>AVRH + 0.5 → AVRH + 0.5 LSB<br>AVRH - 1.5 → AVRH - 1.5 LSB<br>AVRH - 3.5 → AVRH - 3.5 LSB ) |
| 92 | | Changed the figure in “6.3. LIN-USART Timings at VDD5 = 3.0 to 5.5 V”.<br>($V_{OH}$ → $V_{IH}$<br>$V_{OL}$ → $V_{IL}$ ) |
| 115 | Programmer’s Model | Changed “2. Customer Registers”.<br>(CIF0 = 0xD000, CIF1 = 0xD004, CIF2 = 0xD008 und CIF3 = 0xD00C<br>→<br>CIF0 = 0xD000, CIF1 = 0xD004, CIF2 = 0xD008 and CIF3 = 0xD00C) |
| Rev. 2.0 | | |
| 239 | Ordering Information | Changed part number |
| - | - | Company name and layout design change |

**NOTE: Please see “Document History” about later revised information.**

## Document History

| Document Title: MB91460X Series FR60 32-bit Microcontrollers<br>Document Number: 002-04611 | | | | |
|---|---|---|---|---|
| **Revision** | **ECN** | **Orig. of Change** | **Submission Date** | **Description of Change** |
| ** | — | AKIH | 09/26/2014 | Migrated to Cypress and assigned document number 002-04611.<br>No change to document contents or format. |
| *A | 5206445 | AKIH | 04/07/2016 | Updated to Cypress format. |

## Sales, Solutions, and Legal Information

### Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

### Products

| | |
|---|---|
| ARM® Cortex® Microcontrollers | cypress.com/arm |
| Automotive | cypress.com/automotive |
| Clocks & Buffers | cypress.com/clocks |
| Interface | cypress.com/interface |
| Lighting & Power Control | cypress.com/powerpsoc |
| Memory | cypress.com/memory |
| PSoC | cypress.com/psoc |
| Touch Sensing | cypress.com/touch |
| USB Controllers | cypress.com/usb |
| Wireless/RF | cypress.com/wireless |

### PSoC®Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

### Cypress Developer Community

Forums | Projects | Video | Blogs | Training | Components

### Technical Support

cypress.com/support

© Cypress Semiconductor Corporation, 2009-2016. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.