E-Ray

FlexRay IP-Module

User's Manual

Revision 1.2.7

06.02.2009

Robert Bosch GmbH
Automotive Electronics

Copyright Notice and Proprietary Information

Disclaimer

Contents

1. About this Document

1.1 Change Control

1.1.1 Current Status

Revision 1.2.7

1.1.2 Change History

Issue	Date	By	Change
Revision 1.0	29.10.04	C. Horst	First complete revision
Revision 1.0.1	16.11.04	C. Horst	Message Buffer Status bits PLE, MLST, ES replaced by bits ESA, ESB, MLST
Revision 1.0.2	28.01.05	C. Horst	IBCR, IBCM, OBCR, OBCM: addresses changed MHDC2: register removed MHDC1: renamed to MHDC Message buffer 0 dedicated to hold key slot ID SFS: description updated ESIDn, OSIDn: description updated EIR: bit SCE removed EILS: bit SCEL removed EIES, EIER: bit SCEE removed
Revision 1.1 working	29.04.05	C. Horst	State DEFAULT_CONFIG added to POC CCSV: assignment of states to POCS[5:0] changed: POCS[5:0] = 00 0000 = DEFAULT_CONFIG POCS[5:0] = 00 1111 = CONFIG CCSV: bit DCREQ removed SIR: bit MBSI added SILS: bit MBSIL added SIES, SIER: bit MBSIE added Register BGSC removed EIR: bits SMEB, SMEA removed EILS: bits SMEBL, SMEAL removed EIES, EIER: bits SMEBE, SMEAE removed Registers TXRQ3, TXRQ4, NDAT3, NDAT4, MBSC3, MBSC4 added Bus guardian related pins **eray_arm**, **eray_bgt**, **eray_mt**, **eray_bge1**, and **eray_bge2** have no function PRTC1: Configuration parameter CASM[6:0] added WRHS1: Bit NME changed to PPIT RDHS1: Bit NME changed to PPIT Pin **eray_scanmode** for scan mode control added
Revision 1.1	04.08.05	C. Horst	EIR: Flags CCL, EFA, IIBA, IOBA, TABA, TABB added SIR: Flag SDS added EILS: Control bits CCLL, EFAL, IIBAL, IOBAL, TABAL, TABBL added SILS: Control bit SDSL added

			EIES: Control bits CCLE, EFAE, IIBAE, IOBAE, TABAE, TABBE added EIER: Control bits CCLE, EFAE, IIBAE, IOBAE, TABAE, TABBE added SIES: Control bit SDSE added SIER: Control bit SDSE added SUCC2: LT[20:0] range modified PRTC1: TSST[3:0] range modified, SPP[1:0] added, configuration of BRP[1:0] for 1.25 MBit/s removed PRTC2: RXL[5:0] range modified MHDC: SLT[12:0] range modified GTUC1: UT[19:0] range modified GTUC2: MPC[13:0] range modified GTUC3: Configuration parameter MTIO[5:0] replaced by MIOA[6:0] and MIOB[6:0] GTUC4: NIT[13:0] and OCS[13:0] range modified GTUC5: DEC[7:0] range modified GTUC7: SSL[9:0] range modified GTUC8: NMS[12:0] range modified GTUC9: APO[5:0] and DSI[1:0] range modified GTUC10: MOC[13:0] range modified GTUC11: Configuration parameter ECC[1:0] replaced by EOCC[1:0] and ERCC[1:0] OCV: OCV[18:0] range modified SCV: SCCA[10:0], SCCB[10:0] range modified ACS: Flags can only be reset MRC: Configuration bits SEC[1:0] added Register LDTS added Bus guardian related pins **eray_arm**, **eray_bgt**, **eray_mt**, **eray_bge1**, and **eray_bge2** removed from physical layer interface Chapter 6. Restrictions removed. Description of timing requirements for data transfers between Message RAM and IBF / OBF moved to "Addendum to E-Ray FlexRay IP-Module Specification Revision 1.1"
Revision 1.2	09.12.05	C. Horst	Section 3.2 renamed from "Interrupt Flag Interface" to "Internal Signal and Flag Interface" With this revision it is possible to use message buffer 1 for sync frame transmission in addition to message buffer 0 if sync frames should have different payloads on channel A and B EIR: Handling of bits PERR and RFO same as for other bits, bit MHF added SIR: Bit RFF renamed to RFCL, handling of bits RFNE, RFCL same as for other bits EILS: Bit MHFL added SILS: Bit RFFL renamed to RFCLL EIES, EIER: Bit MHFE added SIES, SIER: Bit RFFE renamed to RFCLE

			Register STPW renamed to STPW1 STPW2: Register added CCSV: Bits PSL[5:0] added SWNIT: Bits MTSA, MTSB added MRC: Bit SPLM added FCL: Register added FSR: Register added MHDF: Register added MBSC1/2/3/4: Naming of bits changed from MBS to MBC to distinguish between message buffer status flag (MBC) and message buffer status register (MBS) CREL: Register added ENDN: Register added Message buffers in Message RAM: Header 2 and 3 updated from received data frames only MBS: Bits FTA, FTB, CCS[5:0], RCIS, SFIS, SYNS, NFIS, PPIS, RESS added
Revision 1.2.1	17.03.06	C. Horst	All changes to previous release are described in detail in [5].
Revision 1.2.2	19.05.06	C. Horst	All changes to previous release are described in detail in [6].
Revision 1.2.3	15.08.06	C. Horst	All changes to previous release are described in detail in [7].
Revision 1.2.4	24.11.06	C. Horst	Not published.
Revision 1.2.5	15.12.06	C. Horst	All changes to previous release are described in detail in [8], [9].
Revision 1.2.6	09.11.07	C. Horst	All changes to previous release are described in detail in [10].
Revision 1.2.7	06.02.09	C. Horst	All changes to previous release are described in detail in [11].

1.2 Conventions

The following conventions are used within this document:

Times bold	Names of bits and signals
CAPITALS	POC states and CHI commands

1.3 Definitions

FlexRay Frame:	Header Segment + Payload Segment
Message Buffer:	Header Section + Data Section
Message RAM:	Header Partition + Data Partition
Data Frame:	FlexRay frame that is not a null frame

1.4 Scope

This document describes the E-Ray FlexRay IP-module and its features from the application programmer's point of view. All information necessary to integrate the E-Ray module into an user-defined ASIC is located in the Module Integration Guide. Information about a specific Customer CPU Interface can be found in the respective Customer CPU Interface Specification document.

1.5 References

This document refers to the following E-Ray release:

Revision 1.0.3

This document refers to the following documents:

Ref	Author(s)	Title
[1]	FlexRay Group	FlexRay Communication System Protocol Specification v2.1 (05/05/12)
[2]	FlexRay Group	FlexRay Communication System Protocol Specification v2.1 Revision A Errata Sheet Version 1 (06/03/29)
[3]	FlexRay Group	FlexRay Data Link Layer Conformance Test Specification v2.1 (06/03/27)
[4]	AE/EIP	Addendum to E-Ray FlexRay IP-Module Specification Revision 1.2.2
[5]	AE/EIP	Changes E-Ray FlexRay IP-Module Specification v1.2 to v1.2.1
[6]	AE/EIP	Changes E-Ray FlexRay IP-Module Specification v1.2.1 to v1.2.2
[7]	AE/EIP	Changes E-Ray FlexRay IP-Module Specification v1.2.2 to v1.2.3
[8]	AE/EIP	Changes E-Ray FlexRay IP-Module Specification v1.2.3 to v1.2.4
[9]	AE/EIP	Changes E-Ray FlexRay IP-Module Specification v1.2.4 to v1.2.5
[10]	AE/EIP	Changes E-Ray FlexRay IP-Module Specification v1.2.5 to v1.2.6
[11]	AE/EIY	Changes E-Ray FlexRay IP-Module Specification v1.2.6 to v1.2.7

1.6 Terms and Abbreviations

This document uses the following terms and abbreviations:

Term	Meaning
AP	Action Point
BD	Bus Driver
BSS	Byte Start Sequence
CAS	Collision Avoidance Symbol
CC	Communication Controller
CHI	Controller Host Interface
CIF	Customer Interface Block
CRC	Cyclic Redundancy Check
FES	Frame End Sequence
FSS	Frame Start Sequence
FIFO	First In First Out (message buffer structure)
FSM	Finite State Machine
FSP	Frame and Symbol Processing Block
FTM	Fault Tolerant Midpoint
GIF	Generic Interface Block
GTU	Global Time Unit Block
IBF	Input Buffer
INT	Interrupt Control Block
MHD	Message Handler Block
MT	Macrotick
MTS	Media Access Test Symbol
NCT	Network Communication Time
NEM	Network Management Block
NIT	Network Idle Time
NM	Network Management
OBF	Output Buffer
POC	Protocol Operation Control
PRT	Protocol Controller Block
SDL	Specification and Description Language
SUC	System Universal Control Block
TBF	Transient Buffer
TDMA	Time Division Multiple Access (media access method)

TSS	Transmission Start Sequence
TT-D	Time Triggered Distributed Synchronization
µT	Microtick
WUP	Wakeup Pattern
WUS	Wakeup Symbol

2. 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 MBit/s. 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

2.1 Block Diagram

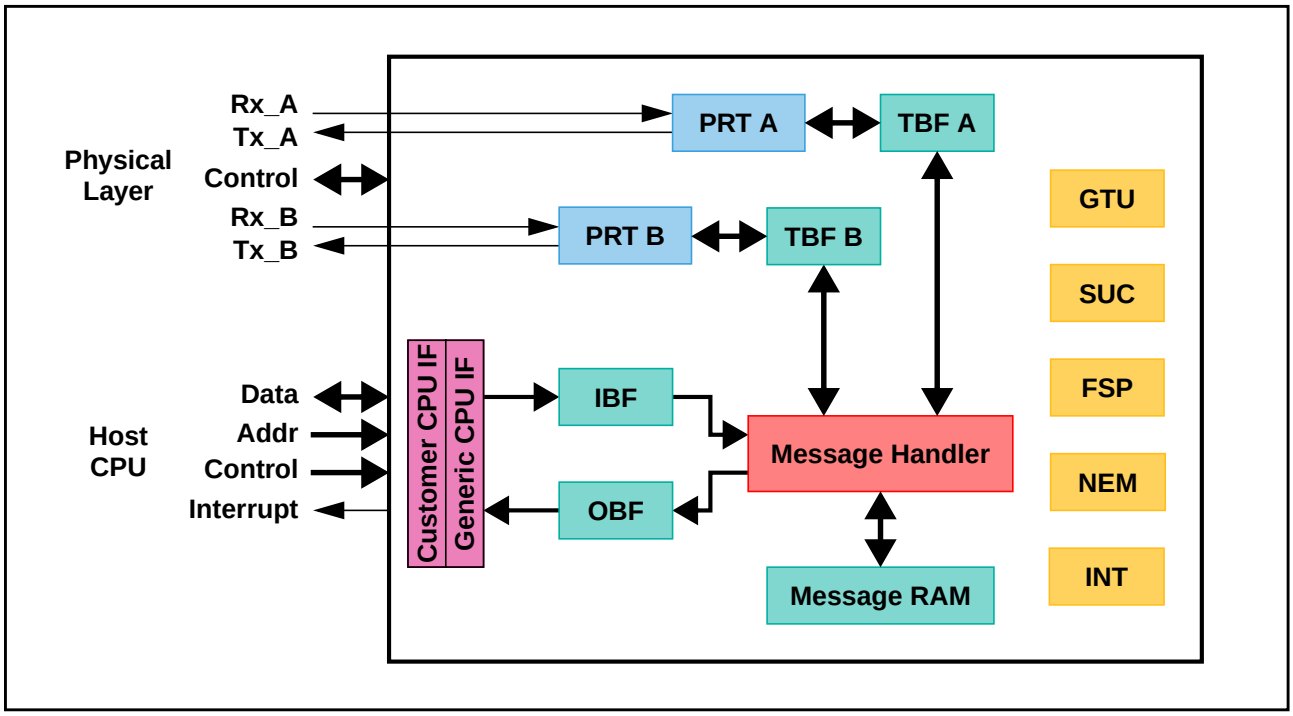


Figure 1: E-Ray block diagram

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 accesssed 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

3. 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 Generic CPU Interface connects the E-Ray module to a customer specific Host CPU via the Customer CPU Interface. It supports 8/16/32-bit access modes.

4. Programmer's Model

4.1 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 assignment of the message buffers is done according to the scheme shown in Table 1 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 4.7.1 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.

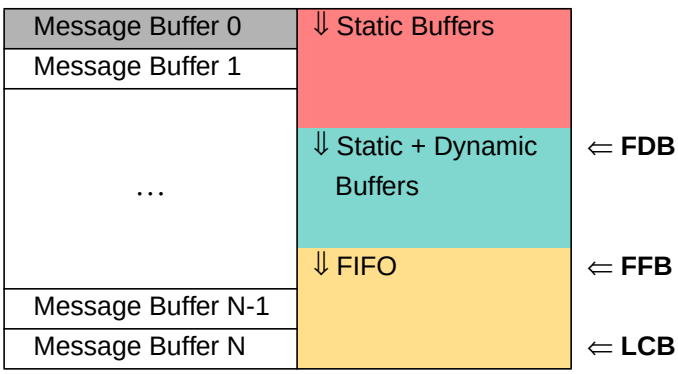


Table 1: Assignment of message buffers

Address	Symbol	Name	Page	Reset	Acc	Block
Customer Registers						
0x0000 0x0004 0x0008 0x000C	see Customer CPU Interface Specification					CIF
Special Registers						
0x0010 - 0x0018		*reserved (3)*		0000 0000	r	
0x001C	LCK	Lock Register	24	0000 0000	r/w	GIF
Interrupt Registers						
0x0020	EIR	Error Interrupt Register	25	0000 0000	r/w	INT
0x0024	SIR	Status Interrupt Register	28	0000 0000	r/w	
0x0028	EILS	Error Interrupt Line Select	31	0000 0000	r/w	
0x002C	SILS	Status Interrupt Line Select	32	0303 FFFF	r/w	
0x0030	EIES	Error Interrupt Enable Set	33	0000 0000	r/w	
0x0034	EIER	Error Interrupt Enable Reset	33	0000 0000	r/w	
0x0038	SIES	Status Interrupt Enable Set	34	0000 0000	r/w	
0x003C	SIER	Status Interrupt Enable Reset	34	0000 0000	r/w	
0x0040	ILE	Interrupt Line Enable	35	0000 0000	r/w	
0x0044	T0C	Timer 0 Configuration	36	0000 0000	r/w	
0x0048	T1C	Timer 1 Configuration	37	0002 0000	r/w	
0x004C	STPW1	Stop Watch Register 1	38	0000 0000	r/w	
0x0050	STPW2	Stop Watch Register 2	39	0000 0000	r/w	
0x0054 - 0x007C		*reserved (11)*		0000 0000	r	
CC Control Registers						
0x0080	SUCC1	SUC Configuration Register 1	40	0C40 1080	r/w	SUC
0x0084	SUCC2	SUC Configuration Register 2	45	0100 0504	r/w	
0x0088	SUCC3	SUC Configuration Register 3	45	0000 0011	r/w	
0x008C	NEMC	NEM Configuration Register	46	0000 0000	r/w	NEM
0x0090	PRTC1	PRT Configuration Register 1	47	084C 0633	r/w	PRT
0x0094	PRTC2	PRT Configuration Register 2	48	0F2D 0A0E	r/w	
0x0098	MHDC	MHD Configuration Register	49	0000 0000	r/w	MHD
0x009C		*reserved (1)*		0000 0000	r	

Address	Symbol	Name	Page	Reset	Acc	Block
0x00A0	GTUC1	GTU Configuration Register 1	50	0000 0280	r/w	GTU
0x00A4	GTUC2	GTU Configuration Register 2	50	0002 000A	r/w	
0x00A8	GTUC3	GTU Configuration Register 3	51	0202 0000	r/w	
0x00AC	GTUC4	GTU Configuration Register 4	52	0008 0007	r/w	
0x00B0	GTUC5	GTU Configuration Register 5	53	0E00 0000	r/w	
0x00B4	GTUC6	GTU Configuration Register 6	53	0002 0000	r/w	
0x00B8	GTUC7	GTU Configuration Register 7	54	0002 0004	r/w	
0x00BC	GTUC8	GTU Configuration Register 8	54	0000 0002	r/w	
0x00C0	GTUC9	GTU Configuration Register 9	55	0000 0101	r/w	
0x00C4	GTUC10	GTU Configuration Register 10	55	0002 0005	r/w	
0x00C8	GTUC11	GTU Configuration Register 11	56	0000 0000	r/w	
0x00CC - 0x00FC		*reserved (13)*		0000 0000	r	
CC Status Registers						
0x0100	CCSV	CC Status Vector	57	0010 4000	r	SUC
0x0104	CCEV	CC Error Vector	60	0000 0000	r	
0x0108 - 0x010C		*reserved (2)*		0000 0000	r	
0x0110	SCV	Slot Counter Value	61	0000 0000	r	GTU
0x0114	MTCCV	Macrotick and Cycle Counter Value	61	0000 0000	r	
0x0118	RCV	Rate Correction Value	62	0000 0000	r	
0x011C	OCV	Offset Correction Value	62	0000 0000	r	
0x0120	SFS	Sync Frame Status	63	0000 0000	r	
0x0124	SWNIT	Symbol Window and NIT Status	64	0000 0000	r	
0x0128	ACS	Aggregated Channel Status	66	0000 0000	r/w	
0x012C		*reserved (1)*		0000 0000	r	
0x0130 - 0x0168	ESIDn	Even Sync ID [1…15]	68	0000 0000	r	
0x016C		*reserved (1)*		0000 0000	r	
0x0170 - 0x01A8	OSIDn	Odd Sync ID [1…15]	69	0000 0000	r	
0x01AC		*reserved (1)*		0000 0000	r	
0x01B0 - 0x01B8	NMVn	Network Management Vector [1…3]	70	0000 0000	r	NEM
0x01BC - 0x02FC		*reserved (81)*		0000 0000	r	
Message Buffer Control Registers						
0x0300	MRC	Message RAM Configuration	71	0180 0000	r/w	MHD
0x0304	FRF	FIFO Rejection Filter	73	0180 0000	r/w	
0x0308	FRFM	FIFO Rejection Filter Mask	74	0000 0000	r/w	
0x030C	FCL	FIFO Critical Level	74	0000 0080	r/w	

Address	Symbol	Name	Page	Reset	Acc	Block
Message Buffer Status Registers						
0x0310	MHDS	Message Handler Status	75	0000 0080	r/w	MHD
0x0314	LDTS	Last Dynamic Transmit Slot	76	0000 0000	r	
0x0318	FSR	FIFO Status Register	77	0000 0000	r	
0x031C	MHDF	Message Handler Constraints Flags	78	0000 0000	r/w	
0x0320	TXRQ1	Transmission Request 1	80	0000 0000	r	
0x0324	TXRQ2	Transmission Request 2	80	0000 0000	r	
0x0328	TXRQ3	Transmission Request 3	80	0000 0000	r	
0x032C	TXRQ4	Transmission Request 4	80	0000 0000	r	
0x0330	NDAT1	New Data 1	81	0000 0000	r	
0x0334	NDAT2	New Data 2	81	0000 0000	r	
0x0338	NDAT3	New Data 3	81	0000 0000	r	
0x033C	NDAT4	New Data 4	81	0000 0000	r	
0x0340	MBSC1	Message Buffer Status Changed 1	82	0000 0000	r	
0x0344	MBSC2	Message Buffer Status Changed 2	82	0000 0000	r	
0x0348	MBSC3	Message Buffer Status Changed 3	82	0000 0000	r	
0x034C	MBSC4	Message Buffer Status Changed 4	82	0000 0000	r	
0x0350 - 0x03EC		*reserved (40)*		0000 0000	r	
Identification Registers						
0x03F0	CREL	Core Release Register	83	[release info]	r	GIF
0x03F4	ENDN	Endian Register	83	8765 4321	r	
0x03F8 - 0x03FC		*reserved (2)*		0000 0000	r	
Input Buffer						
0x0400 - 0x04FC	WRDSn	Write Data Section [1…64]	84	0000 0000	r/w	IBF
0x0500	WRHS1	Write Header Section 1	85	0000 0000	r/w	
0x0504	WRHS2	Write Header Section 2	86	0000 0000	r/w	
0x0508	WRHS3	Write Header Section 3	86	0000 0000	r/w	
0x050C		*reserved (1)*		0000 0000	r/w	
0x0510	IBCM	Input Buffer Command Mask	87	0000 0000	r/w	
0x0514	IBCR	Input Buffer Command Request	88	0000 0000	r/w	
0x0518 - 0x05FC		*reserved (58)*		0000 0000	r	
Output Buffer						
0x0600 - 0x06FC	RDDSn	Read Data Section [1…64]	89	0000 0000	r	OBF
0x0700	RDHS1	Read Header Section 1	90	0000 0000	r	
0x0704	RDHS2	Read Header Section 2	91	0000 0000	r	
0x0708	RDHS3	Read Header Section 3	92	0000 0000	r	
0x070C	MBS	Message Buffer Status	93	0000 0000	r	
0x0710	OBCM	Output Buffer Command Mask	96	0000 0000	r/w	
0x0714	OBCR	Output Buffer Command Request	97	0000 0000	r/w	
0x0718 - 0x07FC		*reserved (58)*		0000 0000	r	

Table 2: E-Ray register map

4.2 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.

4.3 Special Registers

4.3.1 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									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 write accesses to the Configuration Lock Key (unlock sequence). If the write sequence below is interrupted by other write accesses between the second write to the Configuration Lock Key and the write access to the SUCC1 register, 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]**

Note: In case that the Host uses 8/16-bit accesses to write **CLK[7:0]**, the programmer has to ensure that no "dummy accesses" e.g. to the remaining register bytes / words are inserted by the compiler.

4.4 Interrupt Registers

4.4.1 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 either the number of sync frames received during the last communication cycle or the total number of different sync frame IDs received during the last double 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 the execution of the previous CHI command has not yet completed. 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 and one of the following conditions applies:

1) The CC is not in CONFIG or DEFAULT_CONFIG state and 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

4.4.2 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 when **CCSV.WSV[2:0]** changes to a value other than UNDEFINED.

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 Table 17).

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 4.8.6 New Data 1/2/3/4 (NDAT1/2/3/4)), and if bit **MBI** of that message buffer is set to '1' (see Table 17).

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

This flag is set after a stop watch activation when the actual cycle counter and macrotick value are stored in the Stop Watch register (see section 4.4.10 Stop Watch Register 1 (STPW1)).

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 and if bit **MBI** of that message buffer is set (see Table 17).

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 preceding 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 preceding 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

4.4.3 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

4.4.4 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

4.4.5 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				0	0	0	0	0			
S:0x0030 R:0x0034	W						TABBE	LTVBE	EDBE						TABAE	LTVAE	EDAE
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												
	W					MHFE	IOBAE	IIBAE	EFAE	RFOE	PERRE	CCLE	CCFE	SFOE	SFBME	CNAE	PEMCE
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

4.4.6 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

4.4.7 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

4.4.8 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	T0MO 13	T0MO 12	T0MO 11	T0MO 10	T0MO 9	T0MO 8	T0MO 7	T0MO 6	T0MO 5	T0MO 4	T0MO 3	T0MO 2	T0MO 1	T0MO 0
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 5.7.2 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.

4.4.9 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.

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.

4.4.10 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 actual cycle counter and macrotick value are stored in the Stop Watch register.

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 15999.

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

4.4.11 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.

4.5 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 4.3.1 Lock Register (LCK).

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

4.5.1 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 case the previous CHI command has not yet completed, **EIR.CCL** is set to '1' together with **EIR.CNA**; the CHI command needs to be repeated. Except for HALT state, a POC state change command applied while the CC is already in the requested POC state neither causes a state change nor will **EIR.CNA** be set.

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.
In general the Host must check **SUCC1.PBSY** before writing a new CHI command.

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, HALT, or MONITOR_MODE, **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
Resets status flags **CCSV.CSNI**, **CCSV.CSAI**, and **CCSV.WSV[2:0]** to their default values. May be called in POC states READY and STARTUP. When called in any other state, **CMD[3:0]** will be reset to "0000" = command_not_accepted.

MONITOR_MODE
Enter MONITOR_MODE when called in POC state CONFIG. In this mode the CC is able to receive FlexRay frames and wakeup pattern. 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 5.5.4 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 the POC state in the **eray_sclk** domain 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 additionally delayed by synchronization from **eray_sclk** to **eray_bclk** domain and by the Host-specific CPU interface. The maximum additional delay is 12 cycles of the slower of the two clocks **eray_bclk** and **eray_sclk**.

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

Note: 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

Note: **MTSA,B** may also be changed outside DEFAULT_CONFIG or CONFIG state when the write to SUCC1 register is directly preceded by the unlock sequence for the Configuration Lock Key as described in 4.3.1 Lock Register (LCK). This may be combined with CHI command SEND_MTS. 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

Table 3 below references the CHI commands from the FlexRay Protocol Specification v2.1 (section 2.2.1.1, Table 2-2) to the E-Ray CHI command vector CMD[3:0].

CHI command	Where processed (POC States)	CHI Command Vector CMD[3:0]
ALL_SLOTS	POC:normal active, POC:normal passive	ALL_SLOTS
ALLOW_COLDSTART	All except POC:default config, POC:config, POC:halt	ALLOW_COLDSTART
CONFIG	POC:default config, POC:ready	CONFIG
CONFIG_COMPLETE	POC:config	Unlock sequence & READY
DEFAULT_CONFIG	POC:halt	CONFIG
FREEZE	All	FREEZE
HALT	POC:normal active, POC:normal passive	HALT
READY	All except POC:default config, POC:config, POC:ready, POC:halt	READY
RUN	POC:ready	RUN
WAKEUP	POC:ready	WAKEUP

Table 3: Reference to CHI Host command summary from FlexRay protocol specification

4.5.2 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)

4.5.3 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.

Note: The transition to HALT state is prevented if **SUCC1.HCSE** is not set.

4.5.4 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.

4.5.5 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

Note: 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.

4.5.6 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.

4.5.7 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.

4.5.8 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.

4.5.9 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.

4.5.10 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.

4.5.11 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 5.1.5 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.

4.5.12 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.

4.5.13 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.

4.5.14 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.

4.5.15 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.

4.5.16 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	MAPO 4*	MAPO 3*	MAPO 2*	MAPO 1*	MAPO 0*	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.

4.5.17 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.

4.5.18 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.

4.6 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). The status vector may change faster than the Host can poll the status vector, depending on **eray_bclk** frequency.

4.6.1 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 = STARTUP_SUCCESS state
10 1100…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 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 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 in states READY, WAKEUP, STARTUP, NORMAL_ACTIVE, and NORMAL_PASSIVE. 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. Set to SINGLE in all other states.

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 due to CHI command READY. 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 when entering Wakeup state, by CHI command RESET_STATUS_INDICATORS, or by transition from DEFAULT_CONFIG to CONFIG state.

000 = UNDEFINED. Wakeup not yet executed by the CC.

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 RUN command resets this counter to the maximum number of coldstart attempts as configured by **SUCC1.CSA[4:0]**. The initial value of **RCA[4:0]** during CONFIG and DEFAULT_CONFIG state is also **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 and **FSI** is not already set i.e. the HALT state was not reached by FREEZE command. Reset to "00 0000" when leaving HALT state.

4.6.2 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 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**.

4.6.3 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

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

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.

4.6.4 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

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

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 15999.

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.

4.6.5 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

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

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'.

4.6.6 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

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

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'.

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

4.6.7 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

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

VSAE[3:0] Valid Sync Frames Channel A, **even** communication cycle

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

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

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

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.

Note: 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

4.6.8 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. The register is reset when the CC leaves CONFIG state or enters STARTUP state.

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 preceding 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 preceding 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

4.6.9 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. The register is 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
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

Note: 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'.

4.6.10 Even Sync ID [1…15] (ESIDn)

Registers ESID1 to ESID15 hold the frame IDs of the sync frames received in **even** communication cycles used for clock synchronisation up to the limit of gSyncNodeMax. The values are sorted in ascending order, with register ESID1 holding the lowest received sync frame ID. If the node itself transmits a sync frame in an even communication cycle, register ESID1 holds the respective sync frame ID as configured in message buffer 0 and flags **RXEA**, **RXEB** are set. The value is updated during the NIT of each even communication cycle. The register is 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
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 / Configured Even Sync ID on Channel A

Signals that a sync frame corresponding to the stored even sync ID was received on channel A or that the node is configured to be a sync node with key slot = **EID[9:0]** (ESID1 only).

1 = Sync frame received on channel A / node configured to transmit sync frames

0 = No sync frame received on channel A / node not configured to transmit sync frames

RXEB Received / Configured Even Sync ID on Channel B

Signals that a sync frame corresponding to the stored even sync ID was received on channel B or that the node is configured to be a sync node with key slot = **EID[9:0]** (ESID1 only).

1 = Sync frame received on channel B / node configured to transmit sync frames

0 = No sync frame received on channel B / node not configured to transmit sync frames

4.6.11 Odd Sync ID [1…15] (OSIDn)

Registers OSID1 to OSID15 hold the frame IDs of the sync frames received in **odd** and **even** communication cycles used for clock synchronisation up to the limit of gSyncNodeMax. The values are sorted in ascending order, with register OSID1 holding the lowest received sync frame ID. If the node itself transmits a sync frame in an odd communication cycle, register OSID1 holds the respective sync frame ID as configured in message buffer 0 and flags **RXOA**, **RXOB** are set. The value is updated during the NIT of each odd communication cycle. The register is 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
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 / Configured Odd Sync ID on Channel A

Signals that a sync frame corresponding to the stored odd sync ID was received on channel A or that the node is configured to be a sync node with key slot = **OID[9:0]** (OSID1 only).

1 = Sync frame received on channel A / node configured to transmit sync frames

0 = No sync frame received on channel A / node not configured to transmit sync frames

RXOB Received / Configured Odd Sync ID on Channel B

Signals that a sync frame corresponding to the stored odd sync ID was received on channel B or that the node is configured to be a sync node with key slot = **OID[9:0]** (OSID1 only).

1 = Sync frame received on channel B / node configured to transmit sync frames

0 = No sync frame received on channel B / node not configured to transmit sync frames

4.6.12 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 5.6 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. The NM vector is reset when the CC leaves CONFIG state or enters STARTUP 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 4 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							

Table 4: Assignment of data bytes to network management vector

4.7 Message Buffer Control Registers

4.7.1 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. For temporary unlocking see 5.12.4 Host Handling of Parity Errors.

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

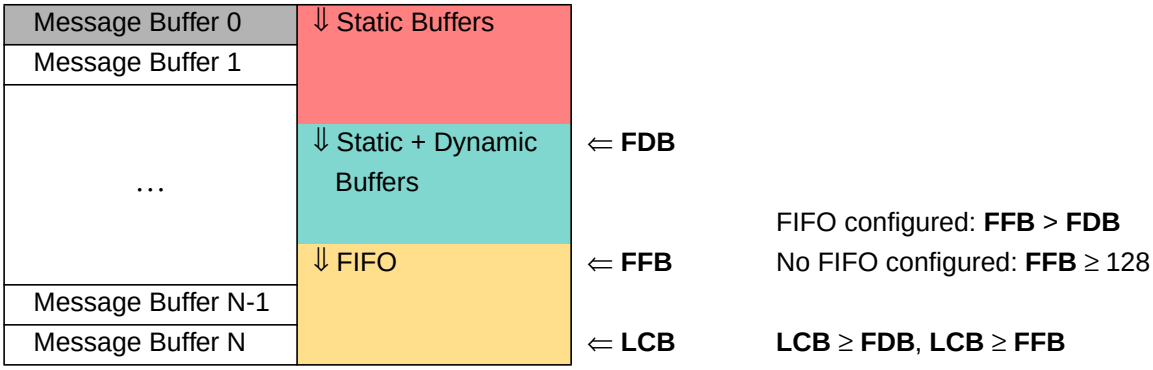
SPLM Sync Frame Payload Multiplex

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!**

Note: 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 5.12 Message RAM.

In case two or more message buffers are assigned to slot 1 by use of cycle filtering, all of them must be located either in the "Static Buffers" or at the beginning of the "Static + Dynamic Buffers" section.

The FlexRay protocol specification requires that each node has to send a frame in its key slot. Therefore at least message buffer 0 is reserved for transmission in the key slot. Due to this requirement a maximum number of 127 message buffers can be assigned to the FIFO. Nevertheless, a non protocol conform configuration without a transmission slot in the static segment would still be operational.

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.

4.7.2 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

Note: 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

Determines the frame ID to be rejected by the FIFO. With the additional configuration of register FRFM, the corresponding frame ID filter bits are ignored, which results in further rejected frame IDs. When **FRFM.MFID[10:0]** is zero, 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 5.7.2 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

4.7.3 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

4.7.4 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.

4.8 Message Buffer Status Registers

4.8.1 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

Note: 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. Is not updated while flag FMBD is set.

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.

Note: **MBT[6:0]** and **MBU[6:0]** are reset when the CC leaves CONFIG state or enters STARTUP state.

4.8.2 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, enters STARTUP state, or by CHI command CLEAR_RAMS.

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.

4.8.3 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, enters STARTUP state, or by CHI command CLEAR_RAMS.

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.

4.8.4 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	TNSB	TNSA	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, enters STARTUP state, or by CHI command CLEAR_RAMS.

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

TNSA Transmission Not Started Channel A

This flag is set by the CC when the Message Handler was not ready to start a scheduled transmission on channel A at the action point of the configured slot.

1 = Transmission not started on channel A

0 = No transmission not started on channel A

TNSB Transmission Not Started Channel B

This flag is set by the CC when the Message Handler was not ready to start a scheduled transmission on channel B at the action point of the configured slot.

1 = Transmission not started on channel B

0 = No transmission not started on channel B

WAHP Write Attempt to Header Partition

Outside DEFAULT_CONFIG and CONFIG state 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

Note: When one of the flags **SNUA**, **SNUB**, **FNFA**, **FNFB**, **TBFA**, **TBFB**, **TNSA**, **TNSB**, **WAHP** changes from '0' to '1', interrupt flag **EIR.MHF** is set to '1'.

4.8.5 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.

4.8.6 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.

4.8.7 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 4.11.5 Message Buffer Status (MBS) and 5.12.1 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.

4.9 Identification Registers

4.9.1 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.

Release	Step	Year	Month	Day	Name
0	70	6	02	03	Beta2, Date 2006/02/03
0	71	6	03	24	Beta2ct, Date 2006/03/24
0	72	6	04	12	Revision 1.0RC1, Date 2006/04/12
1	00	6	05	19	Revision 1.0.0, Date 2006/05/19
1	01	6	12	11	Revision 1.0.1, Date 2006/12/11
1	02	7	10	31	Revision 1.0.2, Date 2007/10/31
1	03	9	02	06	Revision 1.0.3, Date 2009/02/06

Table 5: Coding for releases

4.9.2 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.

4.10 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 4.11.5 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 5.11.2.1 Data Transfer from Input Buffer to Message RAM.

4.10.1 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																
0x0400 - 0x04FC	W	MD31	MD30	MD29	MD28	MD27	MD26	MD25	MD24	MD23	MD22	MD21	MD20	MD19	MD18	MD17	MD16
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																
	W	MD15	MD14	MD13	MD12	MD11	MD10	MD9	MD8	MD7	MD6	MD5	MD4	MD3	MD2	MD1	MD0
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_{2n-1}, $byte_{4n-4}$

MD[15:8] = DW_{2n-1}, $byte_{4n-3}$

MD[23:16] = DW_{2n}, $byte_{4n-2}$

MD[31:24] = DW_{2n}, $byte_{4n-1}$

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.

Transmission order on the FlexRay bus is WRDSn[7:0], WRDSn[15:8], WRDSn[23:16], WRDSn[31:24] with the most significant bit transmitted first. To check how the E-Ray's endianness matches with the Host CPU's endianness, read register ENDN.

4.10.2 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 5.7.2 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** transmit frame on	**Receive Buffer** store frame received from
1	1	both channels (static segment only)	channel A or B (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

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')

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 5.8.3 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

4.10.3 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 (see also Section 5.8.3 Transmit Buffers).

4.10.4 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.

4.10.5 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

4.10.6 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 5.11.2.1 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

4.11 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 5.11.2.2 Data Transfer from Message RAM to Output Buffer.

4.11.1 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_{2n-1}, $byte_{4n-4}$

MD[15:8] = DW_{2n-1}, $byte_{4n-3}$

MD[23:16] = DW_{2n}, $byte_{4n-2}$

MD[31:24] = DW_{2n}, $byte_{4n-1}$

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.

4.11.2 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'.

4.11.3 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.

4.11.4 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'.

Note: Header 3 is updated from data frames only.

4.11.5 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 5.7 Filtering and Masking, 5.8 Transmit Process, and 5.9 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

Note: 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.

For receive buffers (**CFG** = '0') the following status bits are updated from both valid data and null frames. If no valid frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.

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'.

4.11.6 Output Buffer Command Mask (OBCM)

Configures how the Output Buffer is updated from the message buffer in the Message RAM selected by OBCR.OBRS[6:0]. Mask bits **RDSS** and **RHSS** are copied to the register internal storage when a Message RAM transfer is requested by **OBCR.REQ**. When OBF Host and OBF Shadow are swapped, mask bits **RDSH** and **RHSH** are swapped with the register internal storage to keep them attached to the respective Output Buffer transfer. The data transfer between Output Buffer and Message RAM is described in detail in Section 5.11.2.2 Data Transfer from Message RAM to Output Buffer.

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.

4.11.7 Output Buffer Command Request (OBCR)

After setting bit **REQ** to '1' while **OBSYS** is '0', **OBSYS** is automatically set to '1', **OBRS[6:0]** is copied to the register internal storage, mask bits **OBCM.RDSS** and **OBCM.RHSS** are copied to register OBCM internal storage, 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 bit **VIEW** to '1' while **OBSYS** is '0', OBF Host and OBF Shadow are swapped. Additionally mask bits **OBCM.RDSH** and **OBCM.RHSH** are swapped with the register OBCM internal storage to keep them attached to the respective Output Buffer transfer. **OBRH[6:0]** signals the number of the message buffer currently accessible by the Host.

If bits **REQ** and **VIEW** are set to '1' with the same write access while **OBSYS** is '0', **OBSYS** is automatically set to '1' and OBF Shadow and OBF Host are swapped. Additionally mask bits **OBCM.RDSH** and **OBCM.RHSH** are swapped with the registers internal storage to keep them attached to the respective Output Buffer transfer. Afterwards **OBRS[6:0]** is copied to the register internal storage, and the transfer of the selected message buffer from the Message RAM to OBF Shadow is started. While the transfer is ongoing the Host can read the message buffer transferred by the previous transfer from OBF Host. When the current transfer between Message RAM and OBF Shadow has completed, this is signalled by setting **OBSYS** back to '0'.

Any write access to **OBCR[15:8]** while **OBSYS** is set will cause the error flag **EIR.IOBA** to be set. In this case the Output Buffer will not be changed.

The data transfer between Output Buffer and Message RAM is described in detail in Section 5.11.2.2 Data Transfer from Message RAM to Output Buffer.

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 5.10 FIFO Function) to OBF Shadow.

VIEW View Shadow Buffer

Toggles between OBF Shadow and OBF Host. Only writeable 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 writeable 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).

5. 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.

5.1 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.

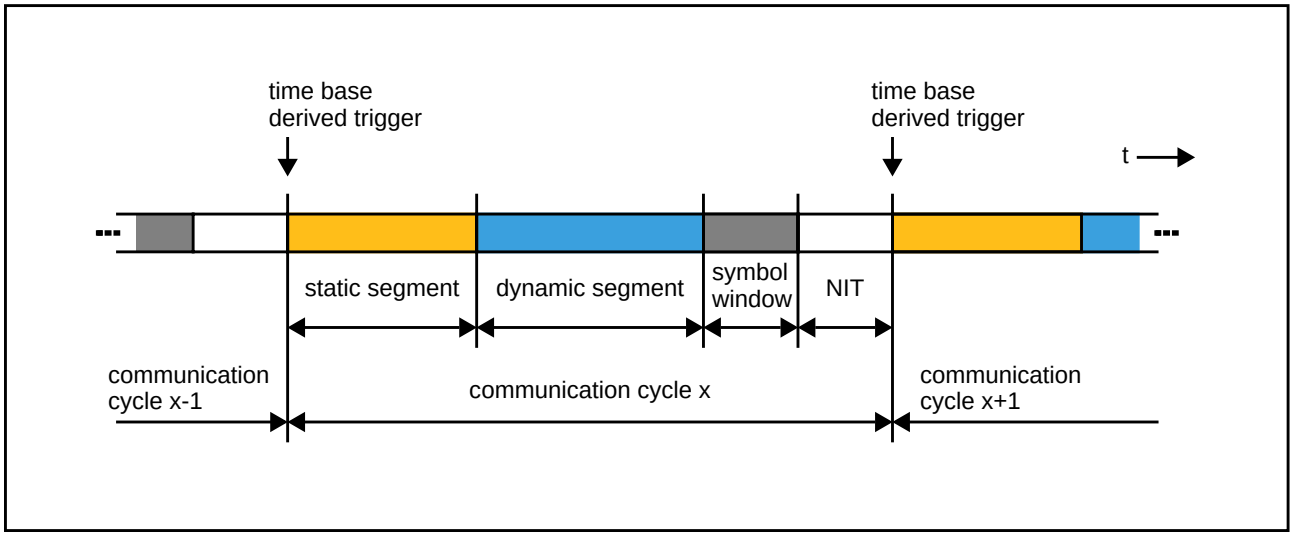


Figure 2: Structure of communication cycle

5.1.1 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]**

5.1.2 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]**

5.1.3 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]**

5.1.4 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]**

5.1.5 Configuration of NIT Start and Offset Correction Start

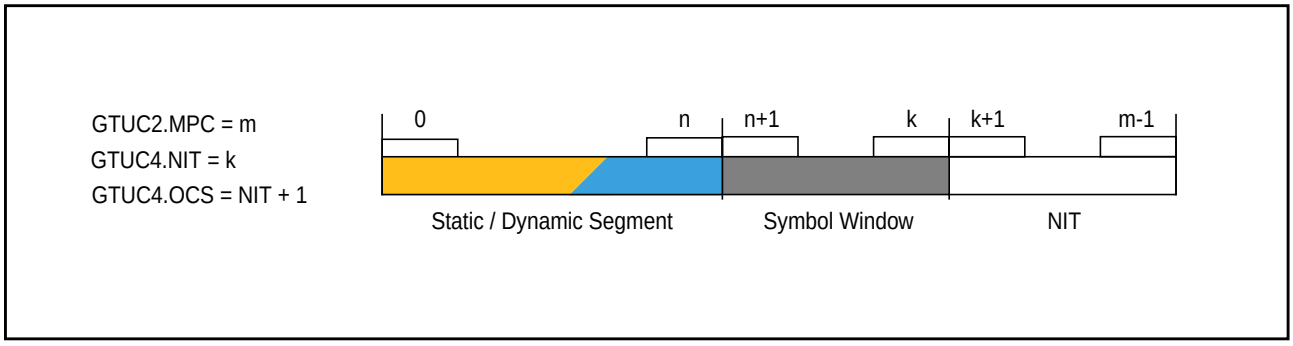


Figure 3: Configuration of NIT start and offset correction start

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.

5.2 Communication Modes

The FlexRay Protocol Specification v2.1 defines the Time-Triggered Distributed (TT-D) mode.

5.2.1 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.

5.3 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.

5.3.1 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 (µT)
- Cycle length = duration of a communication cycle in units of macroticks (MT)

5.3.2 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 (µT).

Node specific:

- Oscillator clock -> prescaler -> microtick (µT)
- µT = basic unit of time measurement in a CC, clock correction is done in units of µTs
- Cycle counter + macrotick counter = nodes local view of the global time

5.3.3 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:

- Max. one sync frame per node in one communication cycle
- Max. 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.

5.3.3.1 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 µTs
- 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)

5.3.3.2 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 µTs
- Distributed over macroticks comprising the next **even / odd** cycle pair (MTs lengthened / shortened)

5.3.3.3 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'.

5.3.4 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

5.4 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 Mode	Activity
ACTIVE (green)	**Full operation**, State: NORMAL_ACTIVE 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 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 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.

Table 6: Error modes of the POC (degradation model)

5.4.1 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.

Note: The transition to HALT state is prevented if **SUCC1.HCSE** is not set.

5.4.2 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.

5.4.3 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.

5.4.4 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]**.

5.5 Communication Controller States

5.5.1 Communication Controller State Diagram

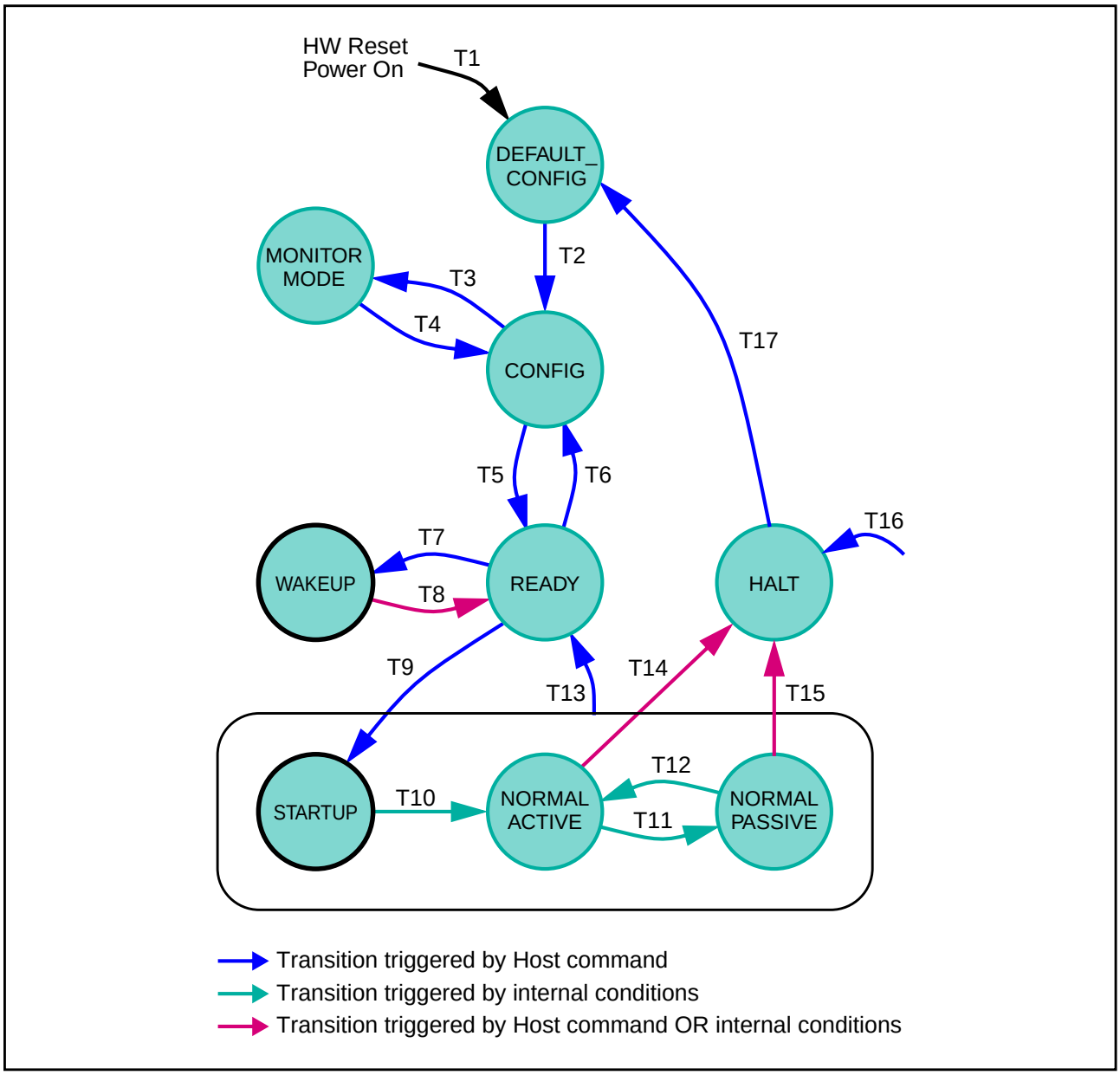


Figure 4: Overall state diagram of E-Ray communication controller

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").

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

Table 7: State transitions of E-Ray overall state machine

5.5.2 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.

5.5.3 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 4.3.1 Lock Register (LCK). Directly after unlocking the CONFIG state the Host has to write **SUCC1.CMD[3:0]** to enter the next 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.

5.5.4 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 and to detect wakeup pattern. 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 CC is not able to distinguish between CAS and MTS symbols. In case one of these symbols is received on one or both of the two channels, the flags **SIR.MTSA** resp. **SIR.MTSB** are set. **SIR.CAS** has no function in MONITOR_MODE.

5.5.5 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.

5.5.6 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.

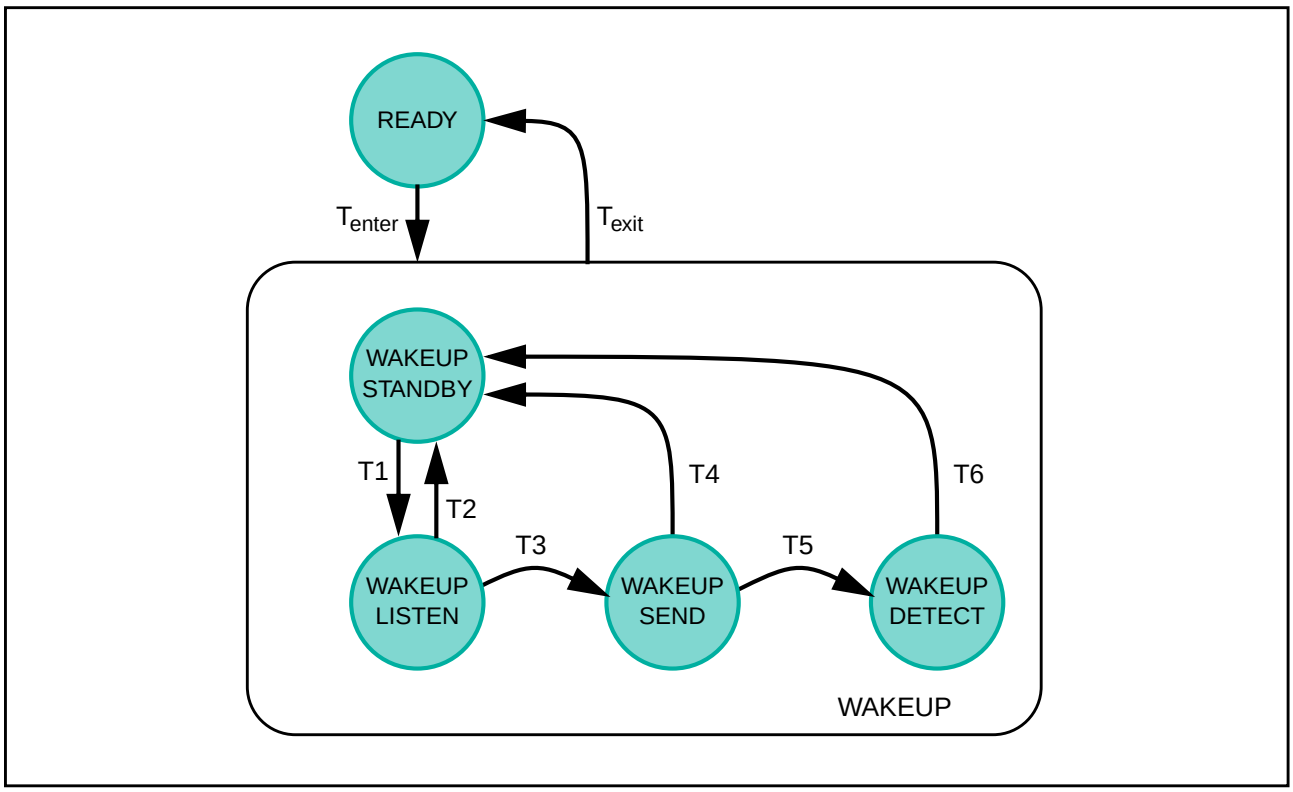


Figure 5: Structure of POC state WAKEUP

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

Table 8: State transitions WAKEUP

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.

5.5.6.1 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)

5.5.6.2 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]**

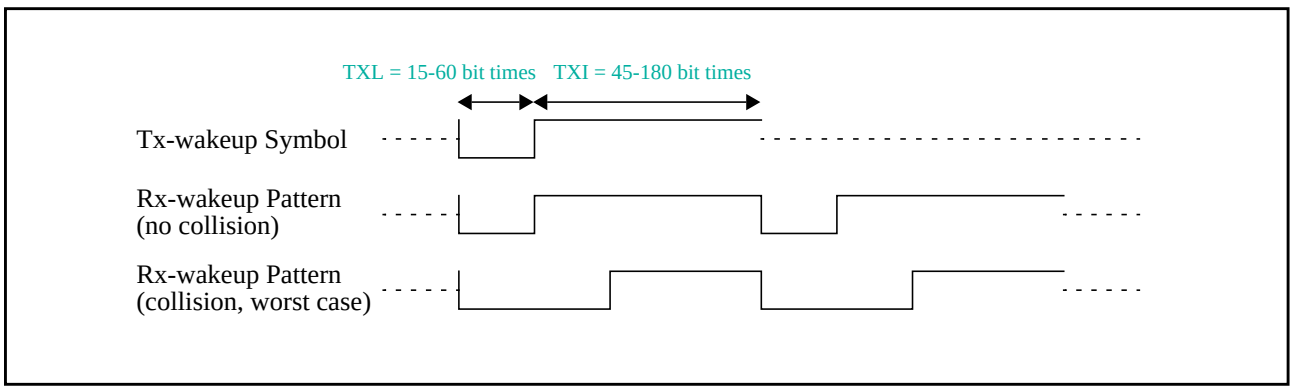


Figure 6: Timing of wakeup pattern

5.5.7 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 Figure 7):

- 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.

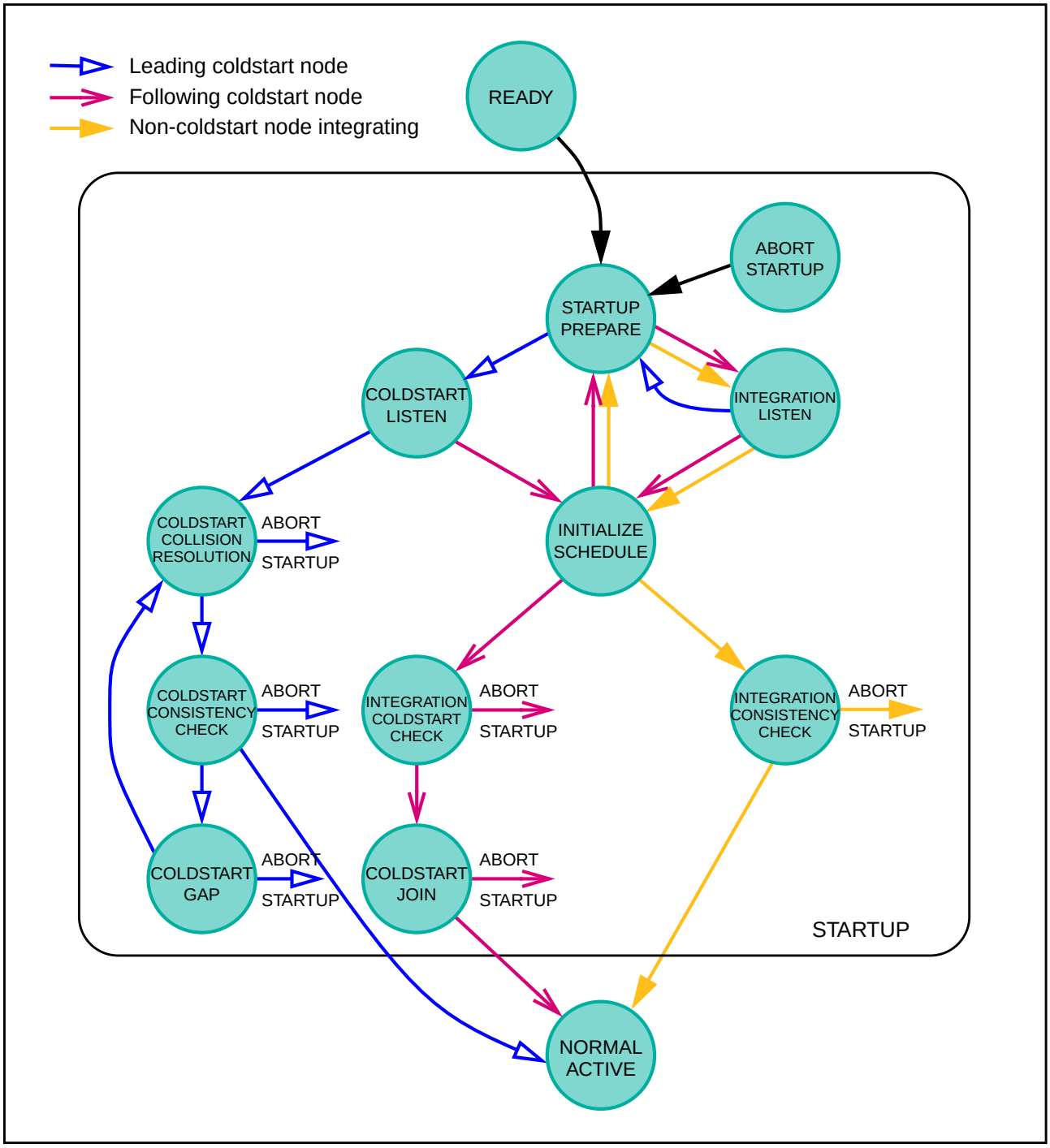


Figure 7: State diagram time-triggered startup

5.5.7.1 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")

5.5.7.2 Startup Timeouts

The CC supplies two different µT 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 4.5.2 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 4.5.2 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.

5.5.7.3 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.

5.5.7.4 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.

5.5.7.5 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.

5.5.8 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)

5.5.9 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)

5.5.10 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 and **SUCC1.HCSE** is set
- When exiting from NORMAL_PASSIVE state because the clock correction failed counter reached the "maximum without clock correction fatal" limit and **SUCC1.HCSE** is set

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 analysing purposes.

When the Host writes **SUCC1.CMD[3:0]** = "0110" (HALT command), the CC sets bit **CCSV.HRQ** and enters HALT state at the next end of cycle.

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]**.

5.6 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]**.

When the CC transits to HALT state, the cycle count is not incremented and therefore the NM vector is not updated. In this case NMV1…3 holds the value from the cycle before.

5.7 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.

5.7.1 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 channel 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.

5.7.2 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 shall 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 Table 9.

Cycle Code	Matching Cycle Counter Values		
0b000000x	all Cycles		
0b000001c	every second Cycle	at (Cycle Count)mod2	= c
0b00001cc	every fourth Cycle	at (Cycle Count)mod4	= cc
0b0001ccc	every eighth Cycle	at (Cycle Count)mod8	= ccc
0b001cccc	every sixteenth Cycle	at (Cycle Count)mod16	= cccc
0b01ccccc	every thirty-second Cycle	at (Cycle Count)mod32	= ccccc
0b1cccccc	every sixty-fourth Cycle	at (Cycle Count)mod64	= cccccc

Table 9: Definition of cycle set

Table 10 below gives some examples for valid cycle sets to be used for cycle counter filtering:

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 ↵

Table 10: Examples for valid cycle sets

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.

5.7.3 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 Table 11).

CHA	CHB	Transmit Buffer transmit frame	Receive Buffer store valid receive frame
1	1	on both channels (static segment only)	received on channel A or B (store first semantically valid frame, 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

Table 11: Channel filtering configuration

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').

5.7.4 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.

5.8 Transmit Process

5.8.1 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.

5.8.2 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.

5.8.3 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 5.11.1 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 5.6 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.

Note: In case of an odd payload length (PLC = 1,3,5,...) the application has to write zero to the last 16 bit of the message buffers data section to ensure that the padding pattern is all 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.

5.8.4 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.

5.8.5 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.

5.9 Receive Process

5.9.1 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 5.11.1 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.

5.9.2 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 5.11.2.2 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.

5.9.3 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.

5.10 FIFO Function

5.10.1 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 127 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 Figure 8 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.

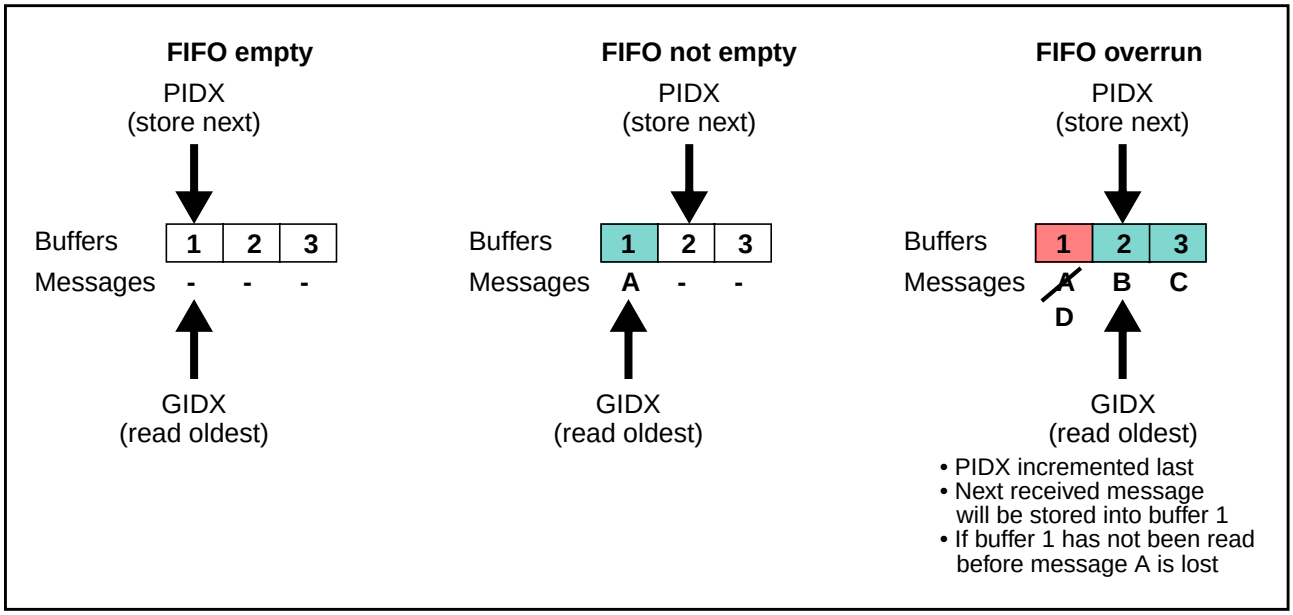


Figure 8: FIFO status: empty, not empty, overrun

5.10.2 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: It is recommended to program the MBI bits of the message buffers belonging to the FIFO to '0' via **WRHS1.MBI** to avoid generation of RX interrupts.

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.

5.10.3 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.

5.11 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.

5.11.1 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 Table 6 below:

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)
…	…

Table 12: Scan of Message RAM

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.

5.11.2 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 **IBCM.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.

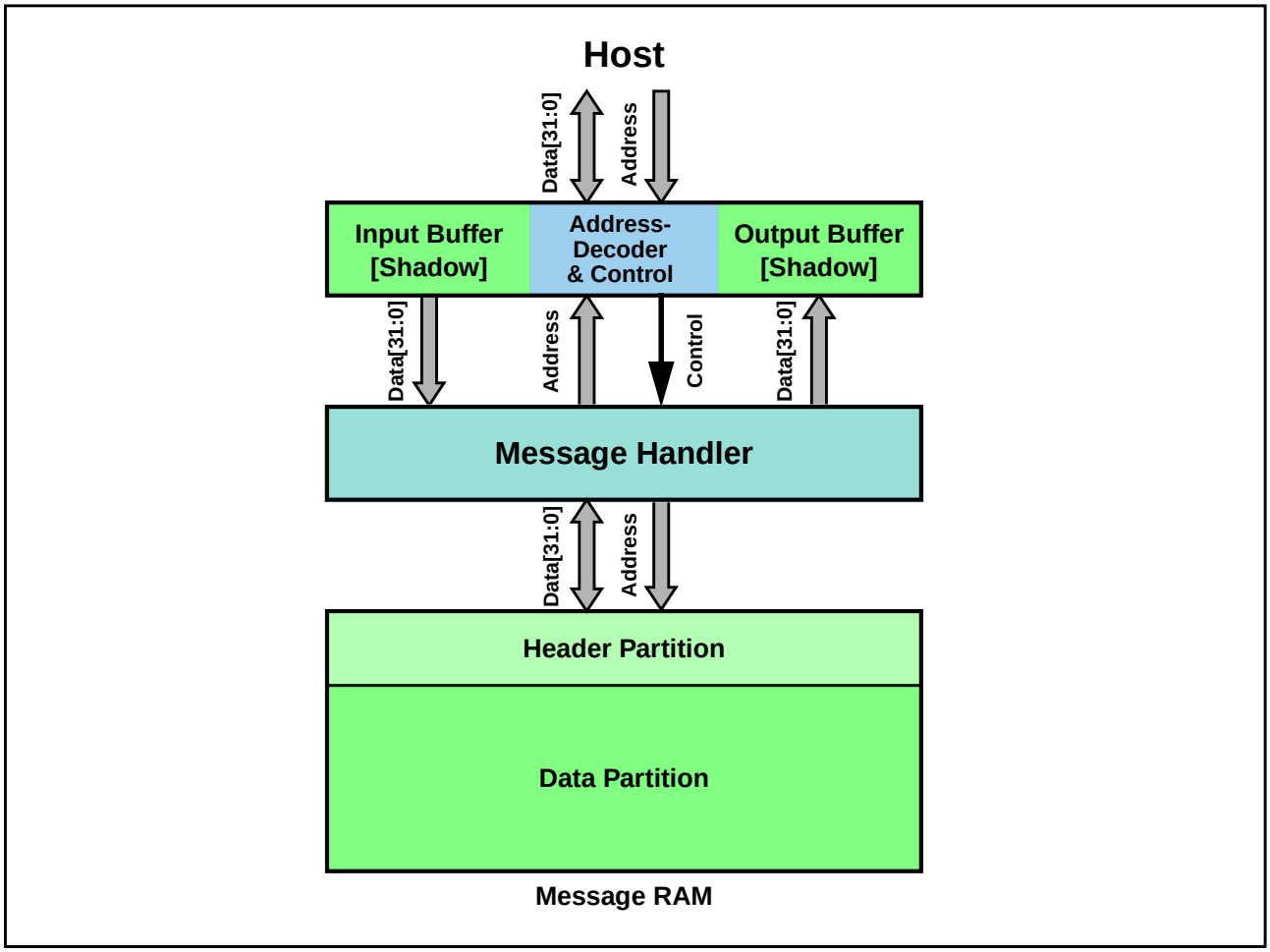


Figure 9: Host access to Message RAM

5.11.2.1 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 Figure 10).

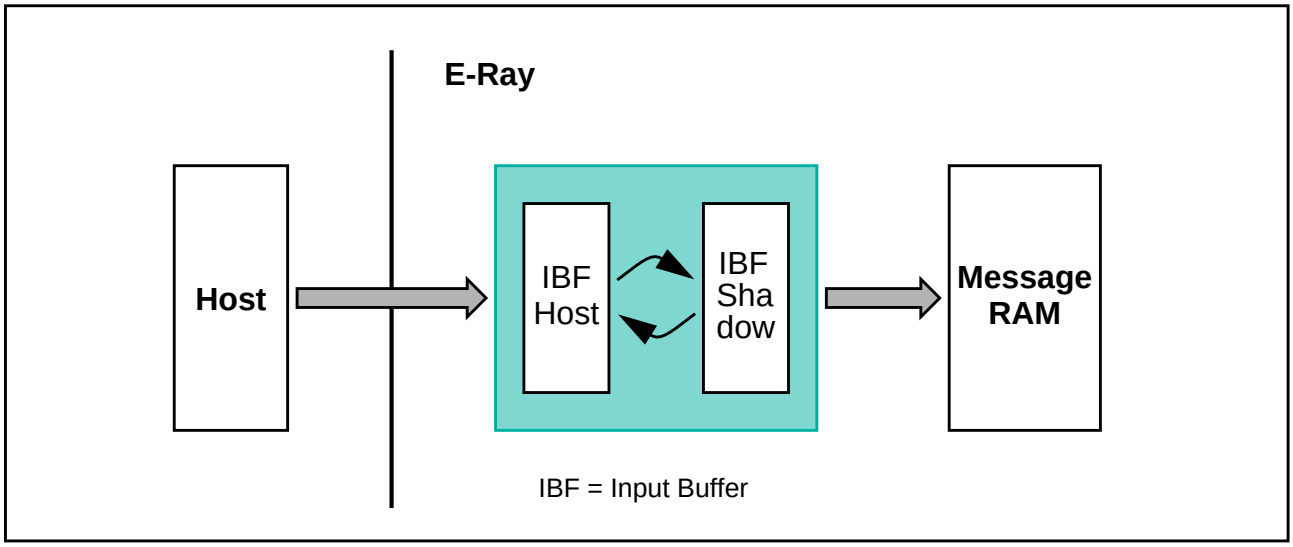


Figure 10: Double buffer structure Input Buffer

In addition the bits in the IBCM and IBCR registers are also swapped to keep them attached to the respective IBF section (see Figure 11).

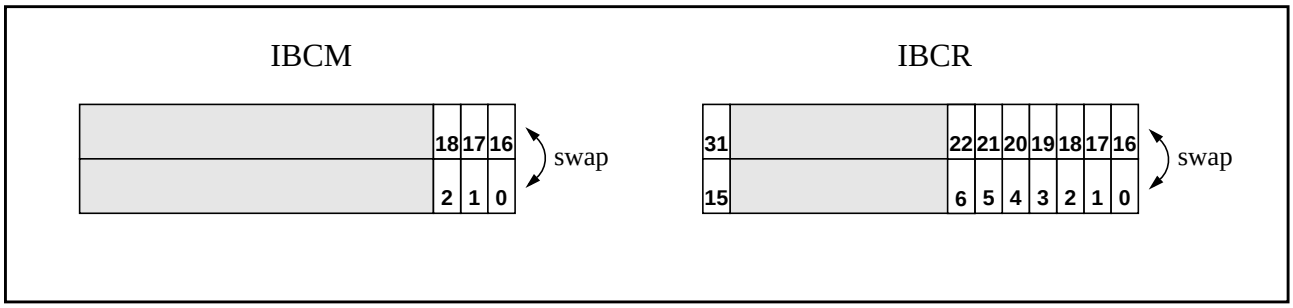


Figure 11: Swapping of IBCM and IBCR bits

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.

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

Table 13: Assignment of IBCM bits

Pos.	Access	Bit	Function
31	r	IBSYS	IBF Busy Shadow, signals ongoing transfer from IBF Shadow to Message RAM
22…16	r	IBRS[6:0]	IBF Request Shadow, number of message buffer currently / lately updated
15	r	IBSYH	IBF Busy Host, transfer request pending for message buffer referenced by IBRH[6:0]
6…0	r/w	IBRH[6:0]	IBF Request Host, number of message buffer to be updated next

Table 14: Assignment of IBCR bits

5.11.2.2 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.

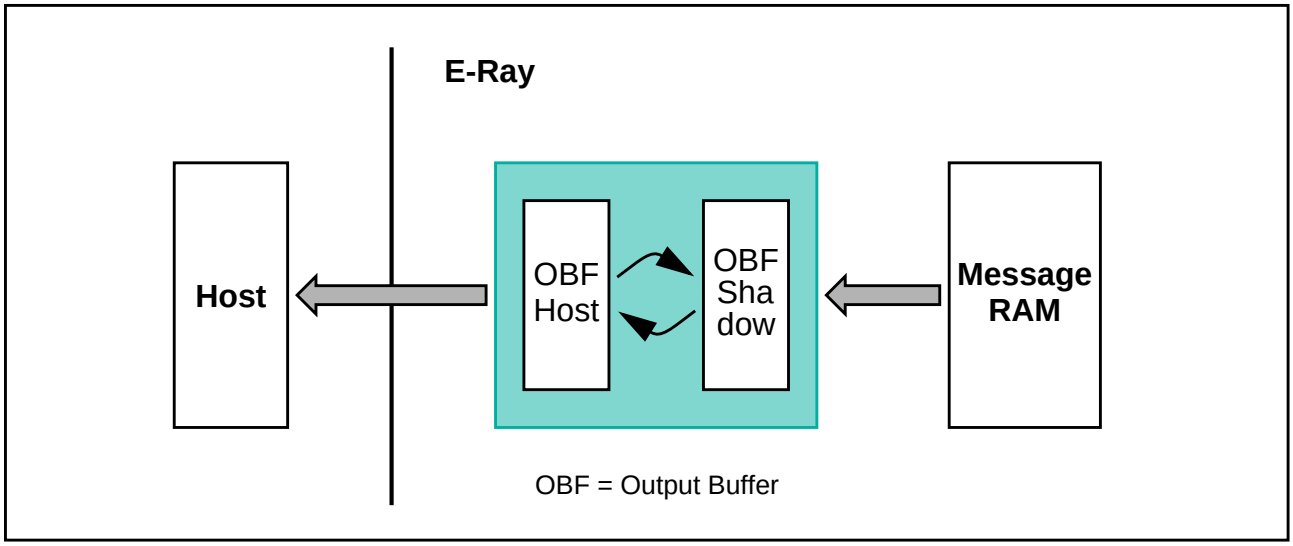


Figure 12: Double buffer structure Output Buffer

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 13).

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'.

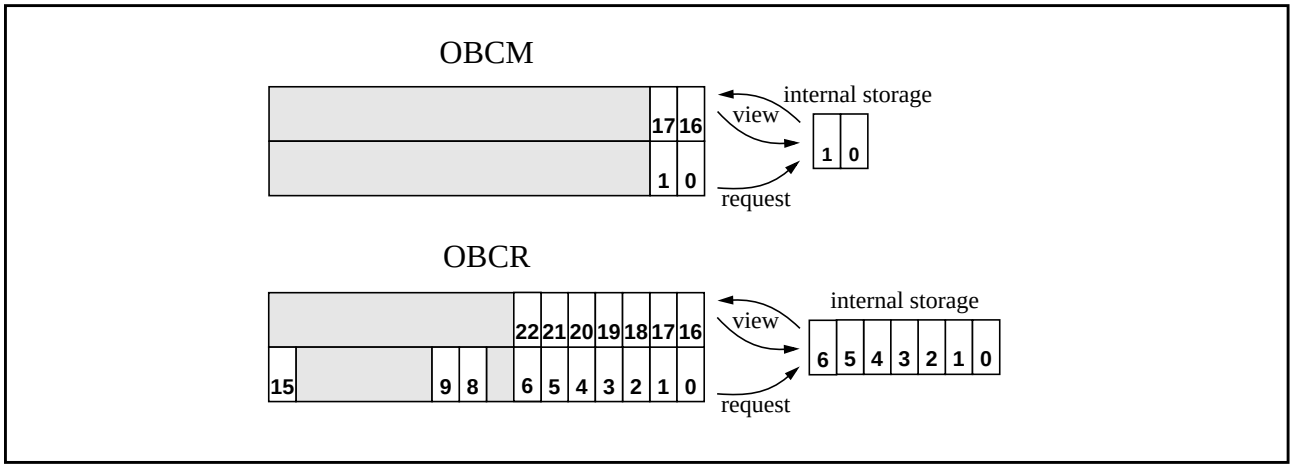


Figure 13: Swapping of OBCM and OBCR bits

OBF Host and OBF Shadow are swapped by setting bit **OBCR.VIEW** to '1' while bit **OBCR.OBSYS** is '0' (see Figure 12).

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 13).

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.

If bits **REQ** and **VIEW** are set to '1' with the same write access while **OBSYS** is '0', **OBSYS** is automatically set to '1' and OBF Shadow and OBF Host are swapped. Additionally mask bits **OBCM.RDSH** and **OBCM.RHSH** are swapped with the registers internal storage to keep them attached to the respective Output Buffer transfer. Afterwards **OBRS[6:0]** is copied to the register internal storage, mask bits **OBCM.RDSS** and **OBCM.RHSS** are copied to register OBCM internal storage, and the transfer of the selected message buffer from the Message RAM to OBF Shadow is started. While the transfer is ongoing the Host can read the message buffer transferred by the previous transfer from OBF Host. When the current transfer between Message RAM and OBF Shadow has completed, this is signalled by setting **OBSYS** back to '0'.

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

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

Table 15: Assignment of OBCM bits

Pos.	Access	Bit	Function
22…16	r	OBRH[6:0]	OBF Request Host, number of message buffer available for Host access
15	r	OBSYS	OBF Busy Shadow, 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, number of message buffer for next request

Table 16: Assignment of OBCR bits

5.11.3 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.

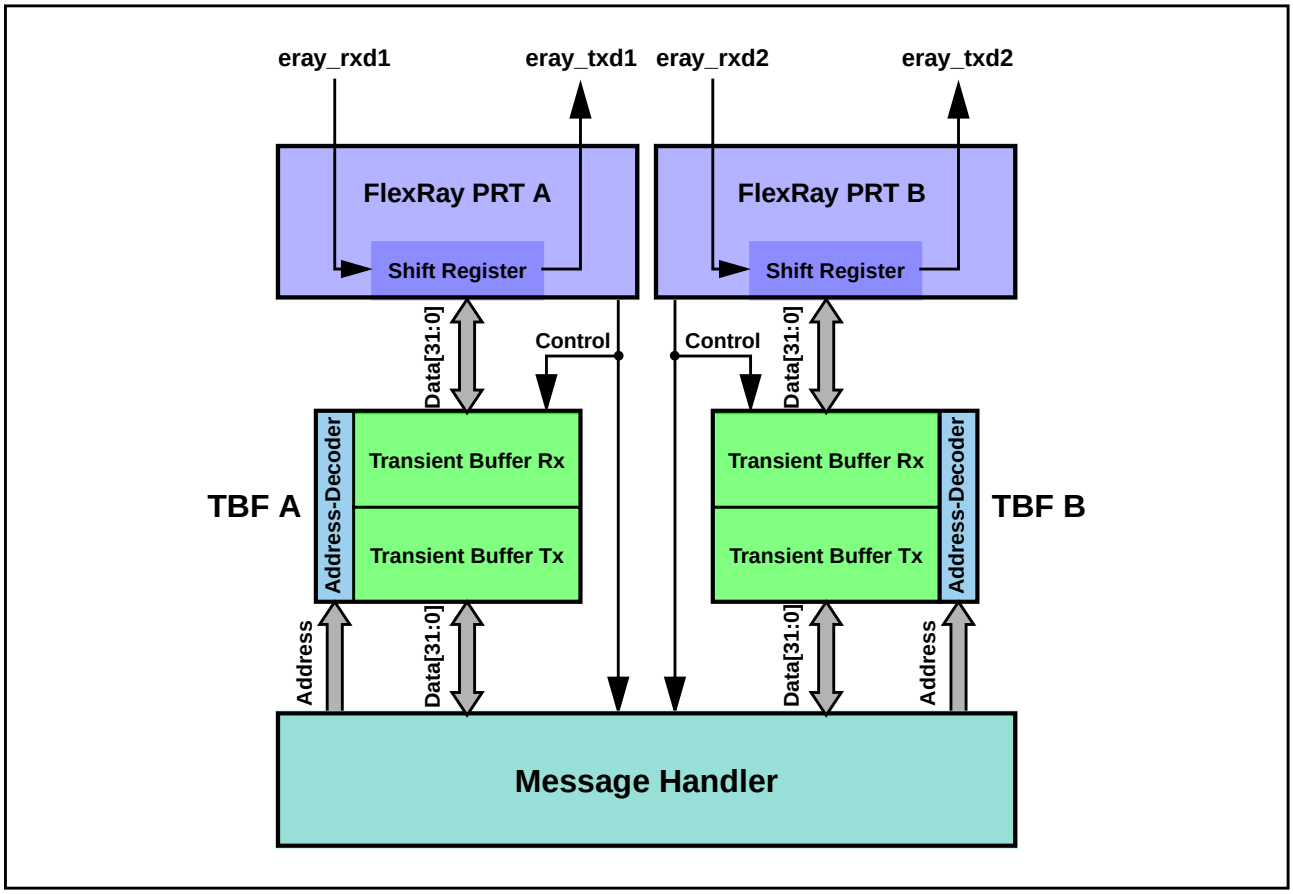


Figure 14: Access to Transient Buffer RAMs

5.12 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 Figure 15.

The data partition is allowed to start at Message RAM word number: (**MRC.LCB** + 1) • 4

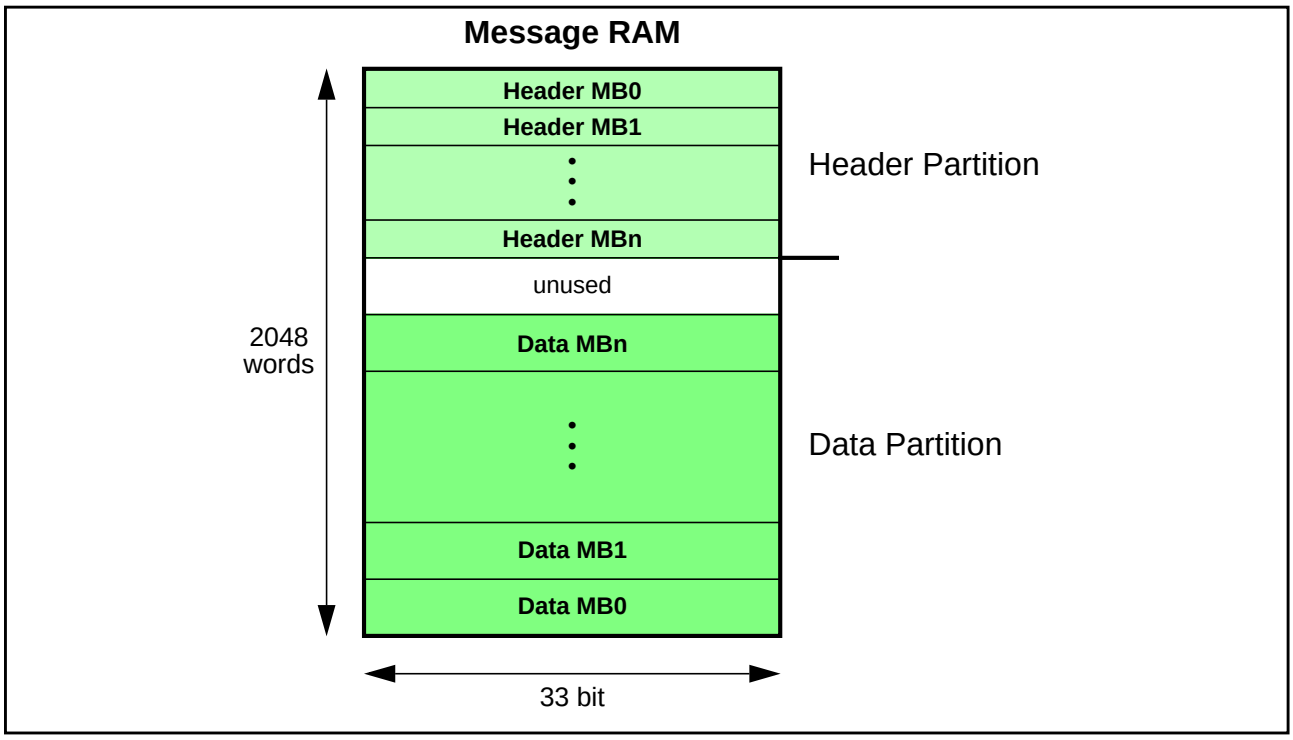


Figure 15: Configuration example of message buffers in the Message RAM

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.

5.12.1 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 Table 17 below. 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.

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 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

Table 17: Header section of a message buffer in the Message RAM

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

5.12.2 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. Table 18 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 Table 18 below).

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							

Table 18: Example for structure of the data partition in the Message RAM

5.12.3 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 Figure 16. 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**.

Figure 16 shows the data paths between the RAM blocks and the parity generators / checkers.

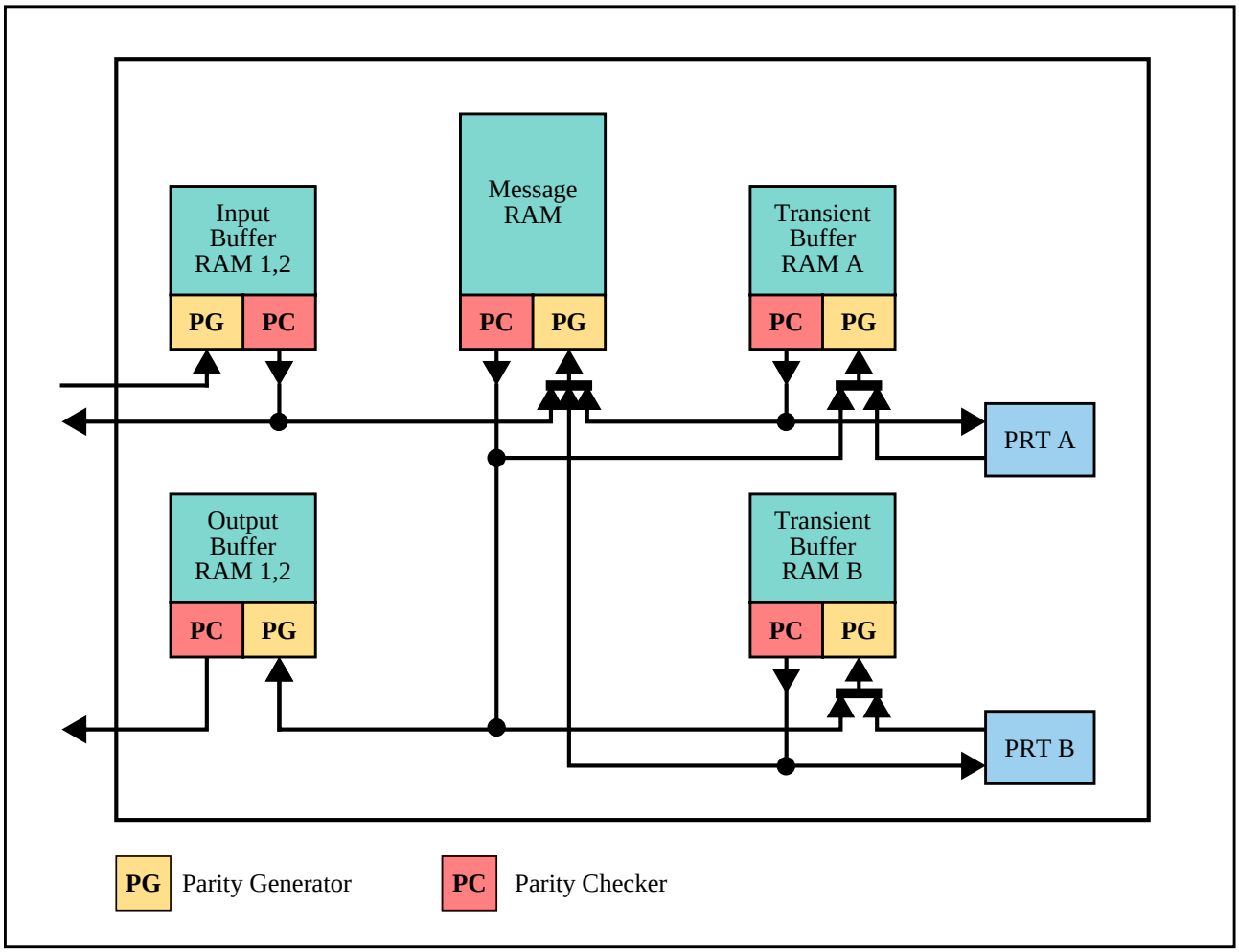


Figure 16: Parity generation and check

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/or 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.

5.12.4 Host Handling of Parity Errors

Parity error caused by transient bit flips can be fixed by:

5.12.4.1 Self-healing

Parity errors located in

• Input Buffer RAM 1,2

• Output Buffer RAM 1,2

• Data Section of Message RAM

• Transient Buffer RAM A

• Transient Buffer RAM B

are overwritten with the next write access to the disturbed bit(s) caused by Host access or by FlexRay communication.

5.12.4.2 CLEAR_RAMS Command

When called in DEFAULT_CONFIG or CONFIG state POC command CLEAR_RAMS initializes all module-internal RAMs to zero.

5.12.4.3 Temporary Unlocking of Header Section

A parity error in the header section of a locked message buffer can be fixed by a transfer from the Input Buffer to the locked buffer Header Section. For this transfer, the write-access to the IBCR (specifying the message buffer number) must be immediately preceded by the unlock sequence normally used to leave CONFIG state (see 4.3.3 Lock Register (LCK)).

For that single transfer the respective message buffer header is unlocked, regardless whether it belongs to the FIFO or whether its locking is controlled by MRC.SEC[1:0], and will be updated with new data.

5.13 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.

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

Table 19: Module interrupt flags and interrupt line enable

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.

6. Appendix

6.1 Register Bit Overview

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																
p24	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									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																
p25	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																
p28	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																
p31	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																
p32	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 EIER	Error Interrupt Enable Set Error Interrupt Enable Reset															
0x0030 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																
p33	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 SIER	Status Interrupt Enable Set Status Interrupt Enable Reset															
0x0038 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																
p34	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																
p35	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	T0MO 9	T0MO 8	T0MO 7	T0MO 6	T0MO 5	T0MO 4	T0MO 3	T0MO 2	T0MO 1	T0MO 0
W																
p36	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																
p37	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																
p38	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	SSCVB 10	SSCVB 9	SSCVB 8	SSCVB 7	SSCVB 6	SSCVB 5	SSCVB 4	SSCVB 3	SSCVB 2	SSCVB 1	SSCVB 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	SSCVA 10	SSCVA 9	SSCVA 8	SSCVA 7	SSCVA 6	SSCVA 5	SSCVA 4	SSCVA 3	SSCVA 2	SSCVA 1	SSCVA 0
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																
p40	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																
p45	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																
p45	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																
p46	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																
p47	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																
p48	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																
p49	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																
p50	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																
p50	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																
p51	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																
p52	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																
p53	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																
p53	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																
p54	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																
p54	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																
p55	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	MAPO 4*	MAPO 3*	MAPO 2*	MAPO 1*	MAPO 0*	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																
p55	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																
p56	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																
p57	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																
p60	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																
p61	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																
p61	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																
p62	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																
p62	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																
p63	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																
p64	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																
p66	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																
p68	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																
p69	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																
p70	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																
p71	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																
p73	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																
p74	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																
p74	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																
p75	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																
p76	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																
p77	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																
p78	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	TNSB	TNSA	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																
p80	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																
p80	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																
p80	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																
p80	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																
p81	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																
p81	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																
p81	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																
p81	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																
p82	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																
p82	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																
p82	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																
p82	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																
p83	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																
p83	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 W	MD31	MD30	MD29	MD28	MD27	MD26	MD25	MD24	MD23	MD22	MD21	MD20	MD19	MD18	MD17	MD16
p84	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MD15	MD14	MD13	MD12	MD11	MD10	MD9	MD8	MD7	MD6	MD5	MD4	MD3	MD2	MD1	MD0

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																
p85	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																
p86	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																
p86	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																
p87	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																
p88	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																
p89	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																
p90	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																
p91	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																
p92	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																
p93	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																
p96	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																
p97	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																

Table 20: Register bit overview

6.2 Assignment of FlexRay Configuration Parameters

Parameter	Bit(field)	Page
pKeySlotusedForStartup	SUCC1.TXST	40
pKeySlotUsedForSync	SUCC1.TXSY	40
gColdStartAttempts	SUCC1.CSA[4:0]	40
pAllowPassiveToActive	SUCC1.PTA[4:0]	40
pWakeupChannel	SUCC1.WUCS	40
pSingleSlotEnabled	SUCC1.TSM	40
pAllowHaltDueToClock	SUCC1.HCSE	40
pChannels	SUCC1.CCHA SUCC1.CCHB	40
pdListenTimeOut	SUCC2.LT[20:0]	45
gListenNoise	SUCC2.LTN[3:0]	45
gMaxWithoutClockCorrectionPassive	SUCC3.WCP[3:0]	45
gMaxWithoutClockCorrectionFatal	SUCC3.WCF[3:0]	45
gNetworkManagementVectorLength	NEMC.NML[3:0]	46
gdTSSTransmitter	PRTC1.TSST[3:0]	47
gdCASRxLowMax	PRTC1.CASM[6:0]	47
gdSampleClockPeriod	PRTC1.BRP[1:0]	47
pSamplesPerMicrotick	PRTC1.BRP[1:0]	47
gdWakeupSymbolRxWindow	PRTC1.RXW[8:0]	47
pWakeupPattern	PRTC1.RWP[5:0]	47
gdWakeupSymbolRxIdle	PRTC2.RXI[5:0]	48
gdWakeupSymbolRxLow	PRTC2.RXL[5:0]	48
gdWakeupSymbolTxIdle	PRTC2.TXI[7:0]	48
gdWakeupSymbolTxLow	PRTC2.TXL[5:0]	48
gPayloadLengthStatic	MHDC.SFDL[6:0]	49
pLatestTx	MHDC.SLT[12:0]	49
pMicroPerCycle	GTUC1.UT[19:0]	50
gMacroPerCycle	GTUC2.MPC[13:0]	50
gSyncNodeMax	GTUC2.SNM[3:0]	50
pMicroInitialOffset[A]	GTUC3.UIOA[7:0]	51
pMicroInitialOffset[B]	GTUC3.UIOB[7:0]	51
pMacroInitialOffset[A]	GTUC3.MIOA[6:0]	51
pMacroInitialOffset[B]	GTUC3.MIOB[6:0]	51
gdNIT	GTUC4.NIT[13:0]	52
gOffsetCorrectionStart	GTUC4.OCS[13.0]	52

Parameter	Bit(field)	Page
pDelayCompensation[A]	GTUC5.DCA[7:0]	53
pDelayCompensation[B]	GTUC5.DCB[7:0]	53
pClusterDriftDamping	GTUC5.CDD[4:0]	53
pDecodingCorrection	GTUC5.DEC[7:0]	53
pdAcceptedStartupRange	GTUC6.ASR[10:0]	53
pdMaxDrift	GTUC6.MOD[10:0]	53
gdStaticSlot	GTUC7.SSL[9:0]	54
gNumberOfStaticSlots	GTUC7.NSS[9:0]	54
gdMinislot	GTUC8.MSL[5:0]	54
gNumberOfMinislots	GTUC8.NMS[12:0]	54
gdActionPointOffset	GTUC9.APO[5:0]	55
gdMinislotActionPointOffset	GTUC9.MAPO[4:0]	55
gdDynamicSlotIdlePhase	GTUC9.DSI[1:0]	55
pOffsetCorrectionOut	GTUC10.MOC[13:0]	55
pRateCorrectionOut	GTUC10.MRC[10:0]	55
pExternOffsetCorrection	GTUC11.EOC[2:0]	56
pExternRateCorrection	GTUC11.ERC[2:0]	56

Table 21: FlexRay configuration parameters

List of Figures

List of Tables