TE3-FALC

Firmware Package, V1.2-1.3.x

PEF 3460 E, Version 1.2

Firmware Description

Wireline Communications

Never stop thinking.

Edition 2004-08-26

Published by Infineon Technologies AG,
St.-Martin-Strasse 53,
81669 München, Germany

Attention please!

The information herein is given to describe certain components and shall not be considered as a guarantee of characteristics.

Terms of delivery and rights to technical change reserved.

We hereby disclaim any and all warranties, including but not limited to warranties of non-infringement, regarding circuits, descriptions and charts stated herein.

Information

For further information on technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies Office (**www.infineon.com**).

Warnings

Due to technical requirements components may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies Office.

Infineon Technologies Components may only be used in life-support devices or systems with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system, or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body, or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.

Revision History: **2004-08-26** Rev. 1

Previous Version:

Page	Subjects (major changes since last revision)
12	**Differences to Firmware V1.2-1.2.x**
99	New Command **CMD_CTRL_BERT_ERROR** (page 99)
86	New Command **CMD_GET_FEAC_CODE** (page 86)
97	New Parameter BERT_GEN_TXER in **CMD_CTRL_BERT** (page 97)
66	New Parameter E3F_PL_FORMAT in **CMD_SET_BIT_E3G751_E3** (page 66)
66	New Parameter E3F_PL_FORMAT in **CMD_SET_BIT_E3G751** (page 66)
71	New Parameter E3F_PL_FORMAT in **CMD_CFG_E3G751_FRAMER** (page 71)
91	Handling of PTY information in E3 G.832 mode. See **Framer RDI Generation (Remote Defect Indication)** (page 91) for further information.

Table of Contents **Page**

Table of Contents **Page**

List of Figures **Page**

List of Tables **Page**

List of Messages **Page**

List of Messages **Page**

List of Messages **Page**

1 Introduction

This document provides a complete reference for the structure of the firmware image file and the Message Catalog of TE3-FALC firmware. Additionally a configuration setup example is described. More documentation and the latest firmware is available on the Infineon Technologies web site at **http://www.infineon.com/t3**.

1.1 Firmware Package

The firmware package of TE3-FALC includes the following files:

Table 1 Firmware Package

File	Description
t3f1213x.bin	Firmware image file in binary format. (see also **"Firmware Binary Format" on Page 21**). *Note: 'x' stands for the dedicated firmware development status*
msgid.h	Describes the firmware message IDs.
paramid.h	Describes the firmware parameter IDs.
bootmsg.h	Describes the boot message IDs using DCI interface
Release_Notes_1213x.txt	Provides comments to the supported firmware version

1.2 Firmware Version Numbering

The firmware version is according to the following numbering rules:

Table 2 Firmware Version Number: Va.b-c.d.e

a.b	Hardware version of the product	
c.d.e	Firmware version for hardware product version a.b	
	c	FW major feature package ID The major feature package ID represents major feature hubs.

Table 2 Firmware Version Number: Va.b-c.d.e (cont'd)

d	FW feature package ID The firmware feature package ID informs about the feature packages included in the firmware.
e	FW development status ID The firmware development status ID represents the timing of the product with the features coded in the FW feature package ID. That means that any minor change to the firmware will lead to the incrementation of the FW development status ID.

The described Firmware V1.2-1.3.x can be used with TE3-FALC (PEF 3460 E) Version 1.2 and provides mainly the following functionality:

– Setup of all data flows (DS3, E3 G.832, E3 G.751, ATM, PLCP, PPP, Bitstream Acces)
– Configuration of all modules (Line Interface Unit (LIU), Digital Jitter Attenuators (DJAT), DS3 Framer, E3 G.832 Framer, E3 G.751 Framer, HDLC Processor, ATM Processor, Data Buffers, System Interface, General Purpose I/O ports (GPIO))
– Support of Trail Trace Identifier (TTI), Payload Type (PTY), SSM nibble, Path Maintenance Data Link (MDL), FEAC Alarm and Status Codes
– Support of Alarm Messages, Defects and Failures
– Support of Performance Measurement (BERT, Performance Counters and Statistics)
– Support of Read Back command to read back the complete operating state of TE3-FALC.

1.3 Differences to Firmware V1.2-1.2.x

Firmware V1.2-1.3.x is based on firmware V1.2-1.2.10 with additional support for the following features and functional improvements:

New features:

– New Command **CMD_CTRL_BERT_ERROR** (page 99)
– New Command **CMD_GET_FEAC_CODE** (page 86)
– New Parameter BERT_GEN_TXER in **CMD_CTRL_BERT** (page 97)
– New Parameter E3F_PL_FORMAT in **CMD_SET_BIT_E3G751_E3** (page 66)
– New Parameter E3F_PL_FORMAT in **CMD_SET_BIT_E3G751** (page 66)
– New Parameter E3F_PL_FORMAT in **CMD_CFG_E3G751_FRAMER** (page 71)

Functional improvements:

– Handling of PTY information in E3 G.832 mode. See **Framer RDI Generation (Remote Defect Indication)** (page 91) for further information.

2 Initial Steps for TE3-FALC

This chapter shall give an overview which initial steps should be done for activating and configuring the TE3-FALC. After power-up a hardware reset must be provided. Then the firmware upload should be performed and the hierarchical initialization and configuration can be done.

2.1 Reset

Global Hardware Reset

A Global Reset of the device can be issued on hardware level by asserting pin $\overline{\text{POR}}$, where the provided reset signal is not required to be synchronous to any clock. After this type of reset the on-chip boot strap loader starts loading a new firmware image from the origin defined with the boot configuration pins.

Global Software Reset

Also on software level a Global Device Reset can be issued via message CMD_RESET_DEVICE. All internal modules are set back to reset state, just the configuration of the Device Control Interface (DCI) remains unchanged.

A parameter selects whether the firmware image shall be reloaded or the current image shall be re-used.

2.2 Boot Process

Prior to the boot process the host software must set the WM bit (bit0) of the MPI Control Register (MCR) to '1'.

The firmware upload can be performed in two ways. The firmware image data is read autonomously by TE3-FALC (acting as bus master) out of an attached EPROM or image data can be written to the device by issuing a set of boot messages via the Device Control Interface DCI. The boot mode configuration should be done via pin setting CFG[2:0].

Note: At the beginning of the boot process, right after reset, the input levels on pins P2, P1 and P0 are used to roughly program the on-chip central PLL. Wrong PLL frequency might lead to internal overclocking and can thus influence the reliability of the device's operation.

2.2.1 Boot via Device Control Interface (DCI)

Figure 1 lists the individual actions user software has to take for booting via DCI.

After internal initialization the device is ready to boot. If internal initialization was successful the **NFC_DCI_ACCESSIBLE** (page 16) notification is written into the low

priority egress queue (LPQOUT). At this point in time the interrupt system is not yet enabled, thus the LPQOUT queue needs to be polled.

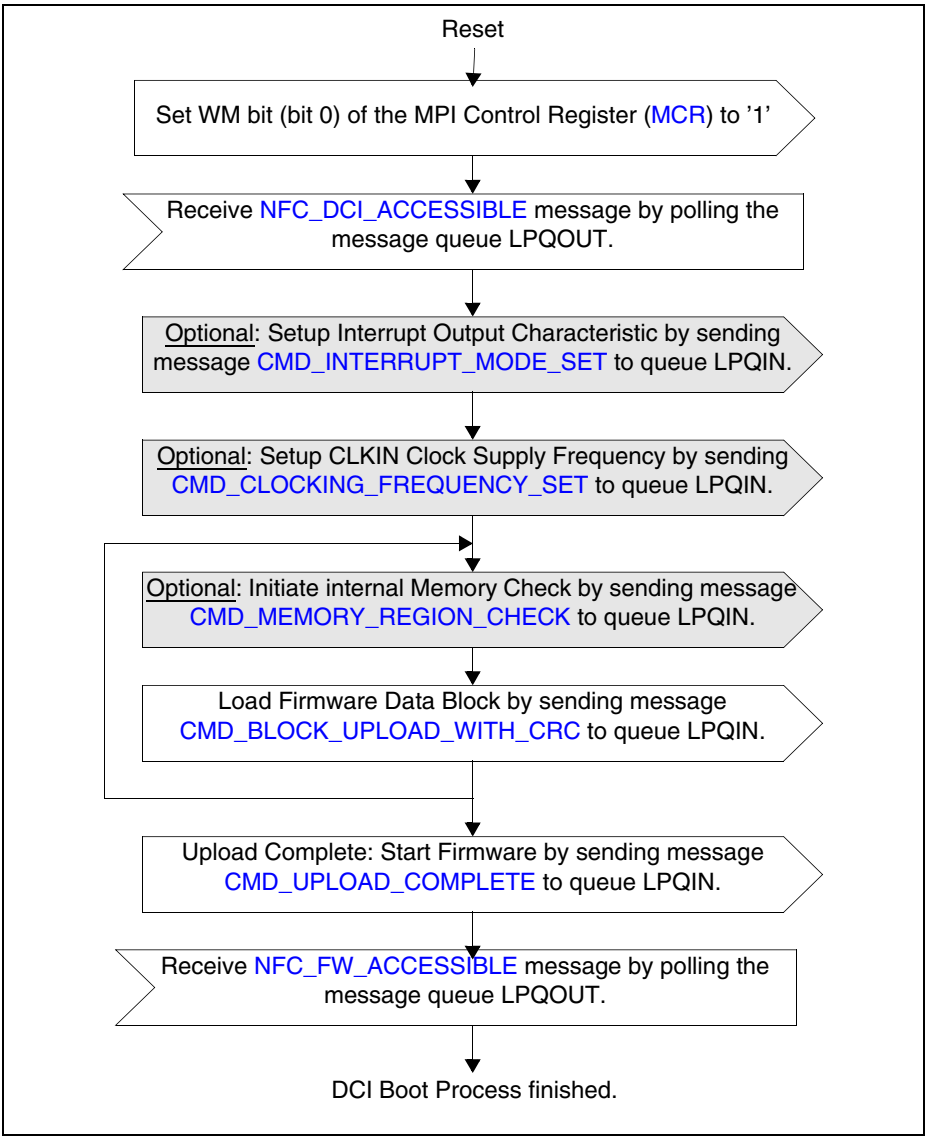
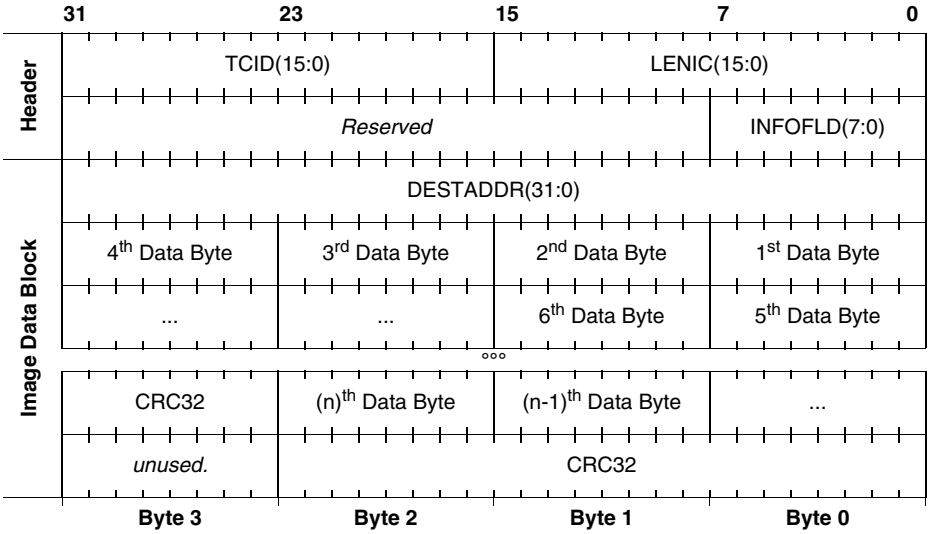


Figure 1 DCI Boot Process

Upload of firmware is performed with cyclic redundancy check (CRC32). The generator polynomial used for CRC32 check is:

$1 + x^1 + x^2 + x^4 + x^5 + x^7 + x^8 + x^{10} + x^{11} + x^{12} + x^{16} + x^{22} + x^{23} + x^{26} + x^{32}$

Table 3 Upload Image Data



A block of image data as shown in **Table 3** is to be written into the DCI memory area. The usable address range is 8000_H to $FF80_H$.

The length indicator (LENIC) includes the whole block size, including header, data field and CRC32 field. Its value should be (n + 16) with n > 0 being the number of bytes to be transferred.

The block format includes the internal destination address DESTADDR(31:0) for this block.

Note that the parameter data format during boot deviates from the parameter array format described in **Chapter 4.2.1.1**.

A block transfer starts after writing the **CMD_BLOCK_UPLOAD_WITH_CRC** (page 16) command into the low priority ingress queue (LPQIN).

To indicate that the upload of firmware is completed, the **CMD_UPLOAD_COMPLETE** (page 16) command must be written into the low priority ingress queue (LPQIN). After receiving this message the TE3-FALC starts running off the loaded firmware image. The fact that firmware started successfully is indicated with the **NFC_FW_ACCESSIBLE** (page 41) which can be read from queue LPQOUT.

*Note: The following messages are only applicable for the boot process, i.e. until message **NFC_FW_ACCESSIBLE** (page 41) is sent out by TE3-FALC.*

Notification ID: **NFC_DCI_ACCESSIBLE**

This message reports the successful initialization of the Device Control Interface (DCI) after reset. When receiving this message the interface is ready to interchange information. As long as the firmware is not uploaded, only boot messages can be interchanged via the Low Priority Queue In (LPQIN) and the Low Priority Queue OUT (LPQOUT). All other messages are acknowledged with with the Return Code "Message Denied".

Output Parameters	*none*

Command ID: **CMD_BLOCK_UPLOAD_WITH_CRC**

Command ID: **CMD_BLOCK_UPLOAD**

This message causes the firmware to load new image data via DCI. The image data is loaded sequentially in blocks assigned to each CMD_BLOCK_UPLOAD_WITH_CRC message. The data is stored in the code memory. In parallel a checksum is calculated. After storing the block of image data, the checksum is verified and the result is indicated to the host in the Return Code (RC) of the **ACK_BLOCK_UPLOAD_WITH_CRC** message.
As an alternative the upload can also take place without CRC, by using the command CMD_BLOCK_UPLOAD instead.

Input Parameters
Block of Upload Image Data as shown in **Table 3**.

Command ID: **CMD_UPLOAD_COMPLETE**

This message is used to identify the end of firmware upload process. After receiving this message the TE3-FALC starts with the firmware specific configuration. If the configuration is finished the **NFC_FW_ACCESSIBLE** (page 41) message is sent to the host.

Input Parameters
32-bit Program Start Pointer (PSP)

Optional Boot Messages

The following messages are optional and can be used as needed to improve the standard (minimum) boot process described above. Note that after the boot process messages **CMD_SPECIFY_CLOCKS** (page 41) and **CMD_SET_INTERRUPT_MODE**

(page 45) are available for analog functionality as the following boot messages **CMD_CLOCKING_FREQUENCY_SET** and **CMD_INTERRUPT_MODE_SET**.

Command ID: **CMD_MEMORY_REGION_CHECK**

This message is used to initiate a memory check. The memory check is non-destructive, i.e. after the check is complete, the original values are restored.
The following memory address regions can be checked with this message:

Memory	Start Address	End Address
Code/Data Section	C000 1000$_H$	C001 FFFC$_H$
Message Queues	DF00 8000$_H$	DF00 9FFC$_H$

Note: This Message is not applicable if the EPROM boot option has been selected!

Input Parameters *(each parameter is a 32-bit value)*		
Start Address	*32-bit address*	Word aligned start address of the memory region.
End Address	*32-bit address*	Word aligned end address of the memory region.
0000 0001$_H$	*32-bit value*	This fixed 32-bit value must be appended for proper operation.

Command ID: **CMD_CLOCKING_FREQUENCY_SET**

This optional message is used to set the TE3-FALC internal system clock to a fixed frequency in the range 58 to 60 MHz.
PLL multiplicator and divisor parameters are calculated from the clock frequency value passed as input parameter to this message (2nd stage PLL programming). The settings configured with port pins P2, P1 and P0 (1st stage PLL programming) are overwritten.

Input Parameters
32-bit value carrying the CLKIN frequency in Hertz (Hz). The allowed range is 4,000,000 Hz (003D 0900$_H$) to 52,000,000 Hz (0319 7500$_H$).

Command ID: **CMD_INTERRUPT_MODE_SET**

The external interrupt pin characteristic is configurable with this meassage. By default the interrupt output is disabled.
For interrupt driven download, the interrupt mask register INTMSK (see **Page 37**) must be programmed.

Note: Interrupts INTREG:HPQINSA and INTREG:LPQINSA are not available during download!

Input Parameters *(each parameter is a 32-bit value)*		
Interrupt Port Config	INT_DISABLE INT_OPEN_DRAIN INT_HIGH INT_LOW	Interrupt output is disabled (tri-state). Interrupt output is open-drain. Interrupt output is push/pull, active high. Interrupt output is push/pull, active low.
$0000\ 0000_H$	*32-bit value*	This fixed 32-bit value must be appended for proper operation.
$0000\ 001F_H$	*32-bit value*	This fixed 32-bit value must be appended for proper operation.

2.2.2 Boot from EPROM

No host action is required for booting from an external EPROM (expect setting bit WM in register MCR to '1'). After reset, the TE3-FALC autonomously starts loading the provided firmware image. After load is complete, firmware starts running and indicates to the host that it is ready to receive messages via the DCI interface by sending the notification **NFC_FW_ACCESSIBLE** (page 41) to the queue LPQOUT. The boot process is visualized in **Figure 2**.

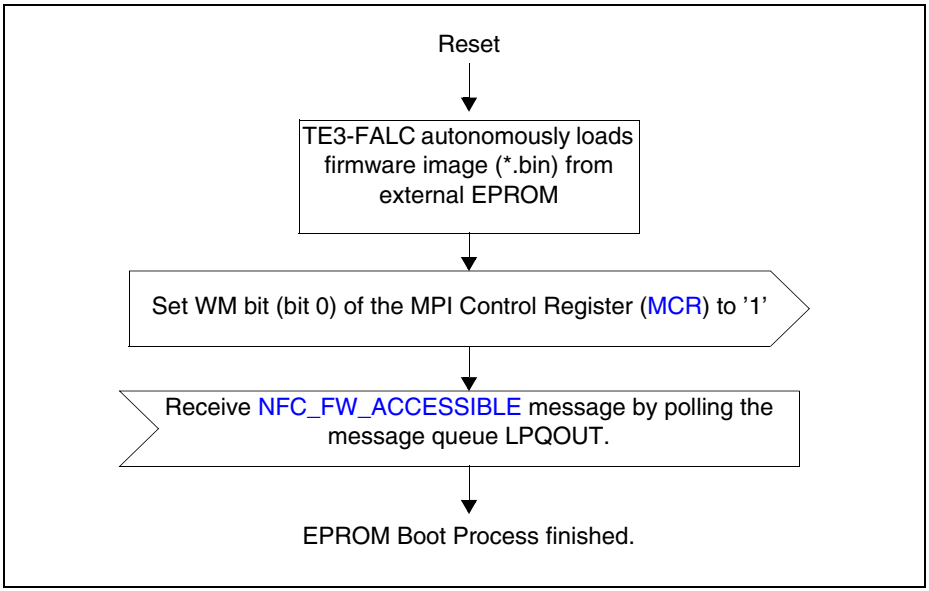


Figure 2 EPROM Boot Process

2.3 Initialization

After the boot process is completed, as indicated with the notification message **NFC_FW_ACCESSIBLE** (page 41), a few initialization messages need to be sent to the TE3-FALC's ingress queue LPQIN.

The first steps of the initialization procedure shown in **Figure 3** are configuration of the TE3-FALC interrupt output pin characteristic via the **CMD_SET_INTERRUPT_MODE** (page 45) message (unless device polling is the preferred method) and specification of input and output frequencies of the clocks used in the system surrounding TE3-FALC with message **CMD_SPECIFY_CLOCKS** (page 41).

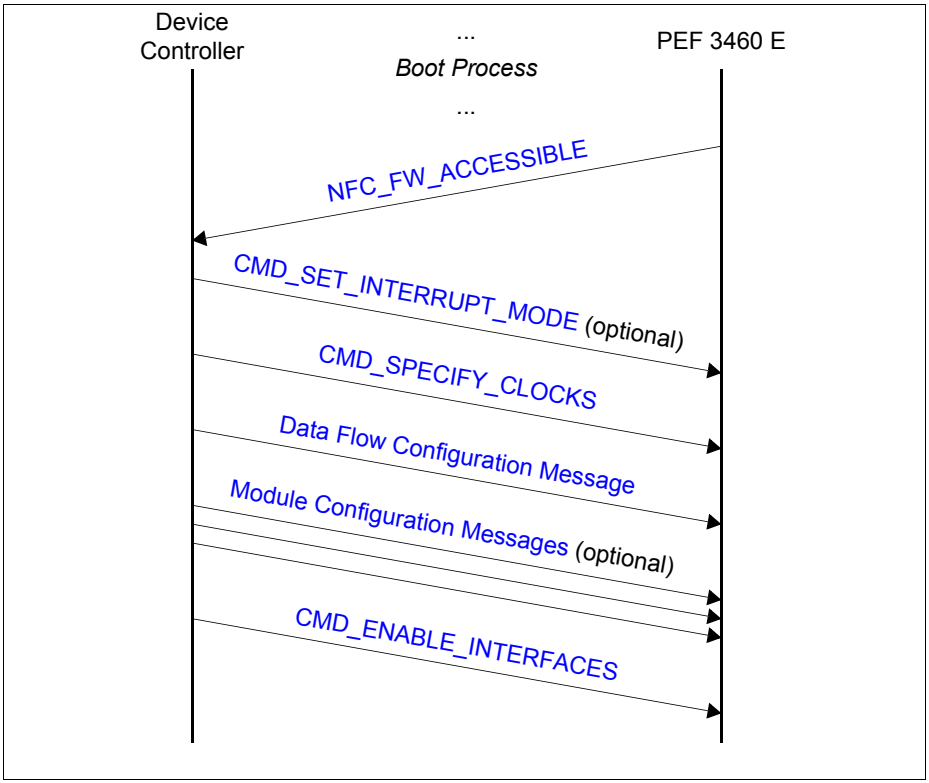


Figure 3 Minimum Initialization Message Flow

As an option, the RX PLL and TX PLL transitioning to their LOCK state can be monitored through notifications **NFC_PLL_LOCKED** (page 44) and **NFC_PLL_UNLOCKED** (page 44). These notifications must first be enabled by using the message **CMD_SET_NFC_ALM_MODE** (page 45).

The next initialization step is to select a **Data Flow Configuration Message** (page 51) from the ones listed in **Table 15**. All internal modules that are part of the selected data flow contain default settings which apply in most cases. If the desired module settings deviate from the default setting, **Module Configuration Messages** (page 67) can be sent to individual TE3-FALC modules.

Now configuration of the data flow is complete and the line and system side interfaces shall be enabled with the **CMD_ENABLE_INTERFACES** (page 67) message.

3 Firmware Binary Format

The firmware image file[1)] is delivered as a binary file and has the extension *.bin. Its special structure is optimized for the EPROM boot option. The boot loader detects the right message format and loads the image blocks automatically into the internal memory.

The parameter block of the binary file is structured as shown in **Table 4**. It contains a 32-bit LENGTH value, a 16-bit MSGID and an optional data array. The size of the parameter block is not necessarily a multiple of 4 bytes (no 32-bit alignment).

Table 4 Structure of Firmware Binary Parameter Block

Address	Value	Comment
addr+0	LENGTH [7:0]	Length of parameter block in bytes including MSGID, LENGTH and DATAARRAY. LENGTH=6+datasize
addr+1	LENGTH [15:8]	
addr+2	LENGTH [23:16]	
addr+3	LENGTH [31:24]	
addr+4	MSGID [7:0]	
addr+5	MSGID [15:8]	
addr+6	DATAARRAY byte[0]	Parameter array; Datasize may be zero for parameterless commands. Datasize is not required to be 32-bit-aligned.
addr+7		
addr+6+ datasize-1	DATAARRAY byte[datasize-1]	

3.1 Supported Parameter Blocks

The following parameter blocks are supported by TE3-FALC:

Table 5 Supported Parameter Blocks

Parameter Block	MSGID
CMD_BLOCK_UPLOAD_WITH_CRC	0001_H
CMD_BLOCK_UPLOAD	0002_H
CMD_UPLOAD_COMPLETE	0003_H
CMD_INTERRUPT_MODE_SET	0006_H
CMD_CLOCKING_FREQUENCY_SET	0021_H

1) There might be more files with different formats shipped with your development kit. Here, it's only refered to the *.bin file format.

The described parameter blocks are related to the DCI boot load messages described in the data sheet.

Note: The binary file supports CMD_BLOCK_UPLOAD_WITH_CRC.

3.1.1 CMD_BLOCK_UPLOAD_WITH_CRC

Table 6 CMD_BLOCK_UPLOAD_WITH_CRC

Address	Size	Value	Comment
addr+0	4	LENGTH=14_D+n[1)]	Length of parameter block in bytes
addr+4	2	MSGID=0001_H	CMD_BLOCK_UPLOAD_WITH_CRC
addr+6	4	DESTADDR	Destination address
addr+10	n	n [1)]	n bytes of the user data
addr+10+n	4	CRC32	Checksum of data block excluding LENGTH, MSGID and DESTADDR

1) With the binary file the block size of n=244_D is provided.

The generator polynomial used for CRC32 check is:

$1 + x^1 + x^2 + x^4 + x^5 + x^7 + x^8 + x^{10} + x^{11} + x^{12} + x^{16} + x^{22} + x^{23} + x^{26} + x^{32}$

3.1.2 CMD_BLOCK_UPLOAD

Table 7 CMD_BLOCK_UPLOAD

Address	Size	Value	Comment
addr+0	4	LENGTH=10_D+n[1)]	Length of parameter block in bytes
addr+4	2	MSGID=0002_H	CMD_BLOCK_UPLOAD
addr+6	4	DESTADDR	Destination address
addr+10	n	n[1)]	n bytes of the user data

1) With the binary file the block size of n=244_D is provided.

3.1.3 CMD_UPLOAD_COMPLETE

Table 8 CMD_UPLOAD_COMPLETE

Address	Size	Value	Comment
addr+0	4	LENGTH=10_D	Length of parameter block in bytes
addr+4	2	MSGID=0003_H	CMD_UPLOAD_COMPLETE
addr+6	4	PSP	Process start pointer (entry point)

3.1.4 CMD_INTERRUPT_MODE_SET

Table 9 CMD_INTERRUPT_MODE_SET

Address	Size	Value	Comment
addr+0	4	LENGTH=10_D	Length of parameter block in bytes
addr+4	2	MSGID=0006_H	CMD_INTERRUPT_MODE_SET
addr+6	4	INT_SETTING	Interrupt setting

3.1.5 CMD_CLOCKING_FREQUENCY_SET

Table 10 CMD_CLOCKING_FREQUENCY_SET

Address	Size	Value	Comment
addr+0	4	LENGTH=10_D	Length of parameter block in bytes
addr+4	2	MSGID=0021_H	CMD_CLOCKING_FREQUENCY_SET
addr+6	4	CLOCKING_ FREQUENCY	Input clock frequency

3.2 Example: Structure of a typical image file

Table 11 Typical Structure of an image File

Offset	Size	Contents
0..3	4_D	LENGTH=102_H
4..5	2_D	MSGID=CMD_BLOCK_UPLOAD_WITH_CRC
6..9	4_D	DESTADDR
10..253	244_D	244_D bytes of user data
254..257	4_D	CRC32
...		... more block uploads

Table 11 Typical Structure of an image File (cont'd)

Offset	Size	Contents
x..x+3	4_D	LENGTH=$0A_H$
x+4..x+5	2_D	MSGID=CMD_UPLOAD_COMPLETE
x+6..x+9	4_D	PSP (start address)
x+10..x+13	4_D	LENGTH=0_D (identifies end of parameter block list)
x+14..		unused

4 Introduction to Message Catalog

Communication with the TE3-FALC is achieved by writing high level commands, and reading high level status, to and from the **Device Control Interface (DCI)**. DCI is a dual-ported memory and register area which is accessible via the microprocessor interface.

Table 12 TE3-FALC Version 1.2 Memory Map

	Address [1]		Comment
DCI Memory Area (32 kB)	$FFFF_H$ 9000_H	*reserved*	
	$8FFF_H$ 8800_H	Egress Buffer for Parameter Arrays	TE3-FALC is responsible for memory management.
	$87FF_H$ 8000_H	Ingress Buffer for Parameter Arrays	User is responsible for memory management.
	$7FFC_H$ 0294_H	*reserved*	
DCI Register Area	0290_H	HPQIN_CNT	
	0280_H	HPQIN message queue	
	0270_H	LPQIN_CNT	Access to High/Low Priority Ingress/ Egress Message Queues. 32-bit wide Message Actuators are exchanged here.
	0260_H	LPQIN message queue	
	0250_H	HPQOUT_CNT	
	0240_H	HPQOUT message queue	
	0230_H	LPQOUT_CNT	
	0220_H	LPQOUT message queue	
	0134_H	DCIERR	DCI Error Indicator Register
	0014_H	INTMSK	Interrupt Mask Register
	0010_H	INTREG	Interrupt Status Register
	0008_H	HW_VID	Hardware Version ID Register
	0000_H	MCR	MPI Control Register

[1] Address bit 15 is treated as Memory Select bit (MSEL): '0' = DCI Register Area, '1' = DCI Memory Area

This interface decouples the internal functional blocks from the external microprocessor bus. The RAM data width is 32 bits, the depth is 8192 words, resulting in an overall size of 32 kbytes. **Table 12** shows how memory and register areas are mapped to the 16-bit addressable range of the TE3-FALC.

With the exception of the initialization registers of the DCI, messages are the only way to communicate with the device.

4.1 Message Types

Each of the following message categories have a set of messages which are identified by the message identifier (MSGID) which is a unique for each message.

There are five general message types:

- Command (CMD)
- Progress Indication (PRG)
- Acknowledgement (ACK)
- Notification (NFC)
- Alarm (ALM)

The properties of the different messages are:

Command Messages (CMD)

Command Messages are used to configuring and control the TE3-FALC. Command messages are always written by the host processor to the DCI (ingress).

Commands can either be sent in format MAT 1-1 or MAT 2-1 (refer to **Chapter 4.2.1**).

Progress-Indication Message (PRG)

The Progress-Indication message is the device's response to having received a Command message. It is sent by the internal microcontroller to the DCI after the device has read the entire parameter array, indicating that the parameter array may be used again. An unmodified Transaction Correlation ID (TCID) is written to the DCI in order to maintain a direct relationship with the corresponding Command message.

Progress-Indication messages could either be sent in format MAT 1-2 or MAT 2-2.

Acknowledge Message (ACK)

The Acknowledgement message is returned by the device after processing of the Command message is complete. The Acknowledgement message also contains a Transaction Correlation ID and a Message ID in order to maintain a direct relationship with the corresponding Command message.

Acknowledgement messages could either be sent of format MAT 1-2 or MAT 2-2.

Note: It is recommended to always request an acknowledgement.

Notification Message (NFC)

Notification messages are written by the internal microcontroller to the DCI if an internal event occurs. Each Notification message must be enabled by the user software prior to the event.

Within a notification message the transaction correlation identifier TCID is always zero. Determination is only done with the message identifier MSGID.

Notification messages could either be sent of format MAT 1-1 or MAT 2-1. In both cases the P-Flag and A-Flag are not used and should be ignored by the receiver of this message.

Alarm Message (ALM)

Alarm messages are written by the internal microcontroller to the DCI if a user definable alarm occurs. Each Alarm message must be enabled by the user software prior to the event.

Within an alarm message the transaction correlation identifier TCID is always set to zero.

Alarm messages could either be sent of format MAT 1-1 or MAT 2-1. In both cases the P-Flag and A-Flag are not used and should be ignored by the receiver of this message.

The principal interaction of messages for Ingress (towards the device) and Egress (towards the user) is shown in **Figure 4**. Grey arrows mark optional messages.

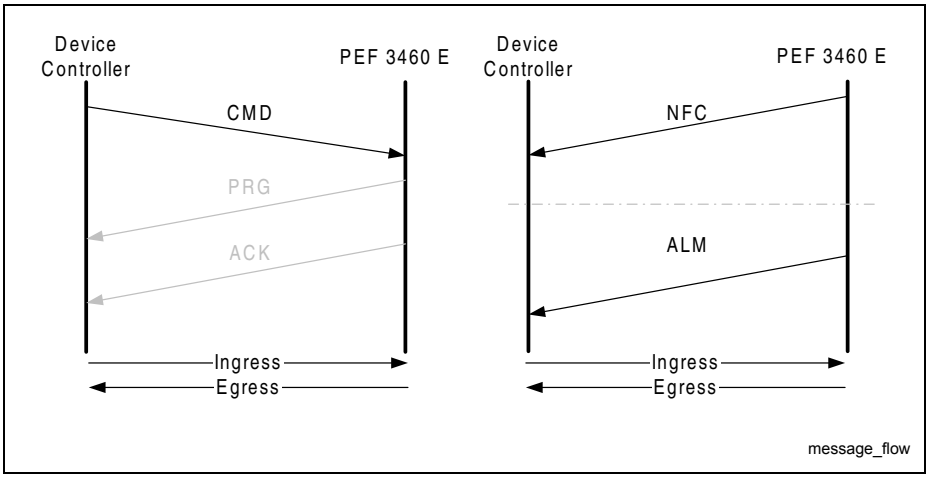


Figure 4 Dependency between Different Message Types

4.2 General Message Structures

The user can send several messages at once and then wait for the replies. Because the order of the messages is very important, this order must be maintained in the DCI which is achieved by means of a queue FIFO. A message comprises 32-bit message actuator (MAT) and optionally a parameter block. Message actuators of type MAT 1-1 and MAT 1-2 do carry a pointer MSGPT to an associated parameter array, message actuators of type MAT 2-1 and MAT 2-2 do not.

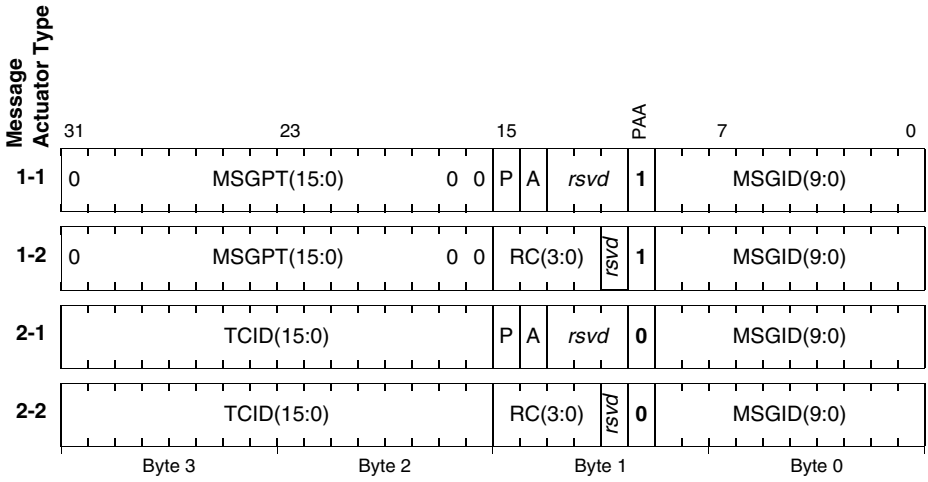
First the parameter array has to be set up in the related memory buffer before the message actuator is written into the message queue.

4.2.1 Message Actuators

An MAT 1-1 or MAT 2-1 actuator contains a progress-indication flag (P-Flag) and an acknowledgement flag (A-Flag). If the progress-indication flag is set, a progress-indication message is generated by the device after the appropriate parameter array has ben read. If the acknowledgement-flag is set, an acknowledgement message is generated after the received message is finally processed.

The MAT 1-2 or MAT 2-2 actuator contains a return code (RC) and is used for progress-indication and acknowledgement messages. The return code indicates whether the corresponding command message has been processed successfully or not.

To distinguish between the different messages, the actuator contains a 10-bit message identifier field MSGID.



Message Actuator Fields

MSGID (9:0) Message Identifier
This unique value is to distinguish between the different messages used within the system

PAA Parameter Array Attached
This is to distinguish between messages with and without attached parameter array. It defines how the bits (31:16) of the message actuator are to be interpreted.

- 0 No parameter array attached. Bits (31:16) are interpreted as Transaction Correlation ID TCID(15:0).
- 1 Parameter array attached. Bits (31:16) are interpreted as pointer MSGPT(15:0) to this parameter array.

A-Flag Acknowledgement Flag
This flag is only valid within command messages. It decides whether an acknowledgement shall be generated when the command has been processed.

- 0 No acknowledgement message is desired for the command message.
- 1 Send acknowledgement message after the processing of this command message is finished.

P-Flag Progress-Indication Flag
This flag is only valid within command messages. It decides whether a progress-indication message shall be generated when the command including its optional parameter array has been read. This indication can be used to free the memory space acquired by this command (actuator and parameter set).

- 0 No progress-indication message is desired for this command message.
- 1 Send progress-indication message after the command including its optional parameter array has been read.

Message Actuator Fields (cont'd)

RC(3:0) Return Code
This field is only valid within acknowledgement messages. It indicates the status of the acknowledged command message.

- 0000_2 positive acknowledgement
- 0001_2 negative acknowledgement
- 0010_2 message denied (MSGID of command not valid)
- 0011_2 parameter mismatch, parameter out of range
- 0100_2 parameter mismatch, configuration invalid
- 0101_2 parameter array mismatch
- 0110_2 resource occupied
- 0111_2 mandatory parameter(s) missing
- others *Reserved.*

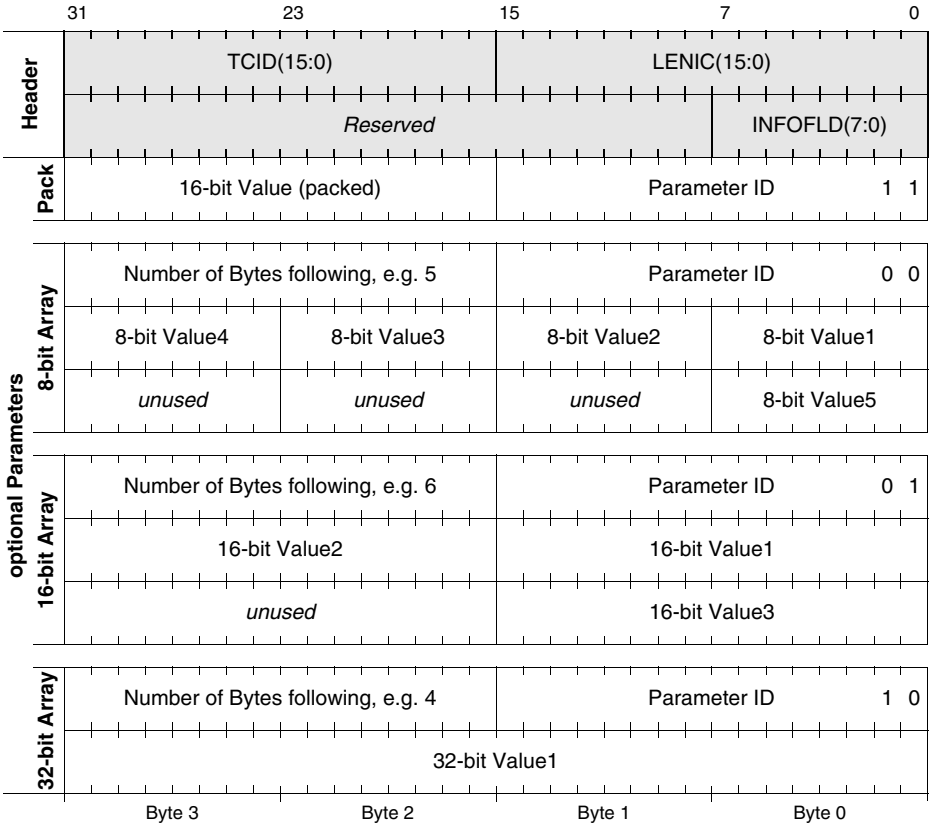
MSGPT (15:0) Message Pointer
16-bit pointer to the appropriate parameter array (located in in-/egress buffer, depends on message direction). This pointer must be 32-bit aligned, thus bits MSGPT(1:0) must be zero.

TCID (15:0) Transaction Correlation Identifier
This user definable value is used to get the match between the sent Command message and the received Progress-Indication or Acknowledgement message. The Transaction Correlation ID is returned unmodified from the TE3-FALC.

4.2.1.1 Optional Parameter Array

The parameter array (see **Table 13**) has variable length. Therefore every parameter array has its own header (shaded fields).

Table 13 Parameter Array Structure



*Note: This parameter array format is not suited for DCI boot messages as described in **Chapter 2.2.1**.*

Parameter Array Fields

LENIC (15:0) Length Indicator
The length indicator (LENIC) represents the size (in number of bytes) of the complete parameter array, including its 8-byte header.

TCID (15:0) Transaction Correlation Identifier
This field has the same meaning as the TCID(15:0) field in the message actuator (refer to **Page 30**).

INFOFLD (7:0) Information Field
The INFOFLD is used for memory management.
It marks the parameter array “valid” or “invalid”. If the parameter array has been marked “invalid” the associated memory area can be re-used. The user always has to set the INFOFLD to “valid” when writing a parameter array to the ingress buffer in DCI memory. The device sets the INFOFLD “invalid” after the parameter array has been read from the internal processor and the user can use the memory region that was allocated by this parameter array again.
The same applies to the opposite direction, i.e. for egress messages with associated output parameters the user has to set the INFOFLD to “invalid” as soon as the parameters have been read.

Param. ID (15:0) Parameter ID
Parameters consist of a 16-bit Parameter ID and one or more associated Values. The list of optional parameters is processed as long as the defined length (LENIC) of the parameter array is reached. The lower two bits of the parameter ID define the type of this parameter:

- ...11_2 Packed Parameter
  A single 32-bit unit is consumed by this parameter type. The lower 16 bits contain the parameter ID, thus a field of 16-bit width remains which is used for the parameter value.
- ...00_2 Parameter Array of 8-bit Values
  The 16-bit field following the parameter ID contains the count of 8-bit values (bytes) attached to this parameter.
- ...01_2 Parameter Array of 16-bit Values
  The 16-bit field following the parameter ID contains the length information of this parameter in number of bytes. If *n* 16-bit values are attached, the byte count is $2 \times n$ bytes.
- ...10_2 Parameter Array of 32-bit Values
  The 16-bit field following the parameter ID contains the length information of this parameter in number of bytes. If *n* 32-bit values are attached, the byte count is $4 \times n$ bytes.

If an optional parameter is not supplied, the value of this parameter stays unchanged. In case a parameter was never set by user software, a reset value is applied which is highlighted (underlined) in the message descriptions.

4.3 The Hardware Interface to the Messages

The DCI manages four message queues (LPQOUT, HPQOUT, LPQIN, HPQIN) which are accessible from the internal and from the external bus for read and write, with the external access having the higher priority.

The width of the message words is 32 bits. TE3-FALC only allows 16- or 8-bit accesses to these 32-bit structures. Thus, one byte of the 32-bit word is configured as an "indicator" byte. Writing to the indicator byte of a 32-bit word increments the write pointer, reading the indicator byte increments the read pointer (destructive read/write), while reading or writing other bytes of a word does not influence the pointers. Thus the message queue indicator byte should be read/written last (see **Table 14**).

A level counter (LPQOUT_CNT, HPQOUT_CNT, LPQIN_CNT, HPQIN_CNT) indicates how many words are stored inside the message queue. The counter is incremented with every indicator byte write and is decremented with every indicator byte read. This counter is stored in a read only register and this register is accessible from both sides. In case of reading an empty queue, all zeros get returned. In case of writing to a full queue the written word is discarded.

Table 14 8-/16-Bit Accesses to an Internal 32-Bit Word

		8-Bit Access		16-Bit Access				
A[1:0]		Motorola	Intel		Motorola		Intel	
00	1st	Byte 3	Byte 0	1st	Byte 3	Byte 2	Byte 1	Byte 0
01	2nd	Byte 2	Byte 1					
10	3rd	Byte 1	Byte 2	2nd	Byte 1	Byte 0	Byte 3	Byte 2
11	4th	Byte 0	Byte 3					

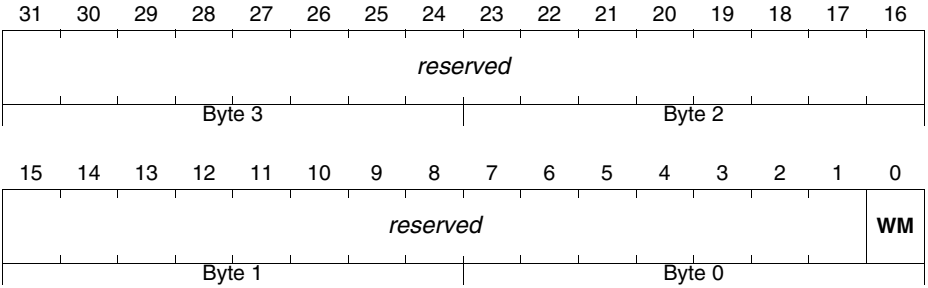
Internal 32-Bit Word	Byte 3	Byte 2	Byte 1	Byte 0

Indicator Byte for INTREG Indicator Byte for Message Queues

4.4 MPI Control Register

Prior to the boot process the host software must set the WM bit (bit 0) of the MPI Control Register (MCR) to ’1’.

MCR
MPI Control Register **rd/wr** **Reset value: 0000 0xxx$_H$**
Address: 0000$_H$



MCR Register Bits:

WM **Word Mode**
This bit must be set to ’1’ prior to any other access to the Device Control Interface DCI.

Note: This bit enables the collection of 2 16-bit (or 4 8-bit) accesses to consecutive (increasing) addresses into an internal 32-bit word access.

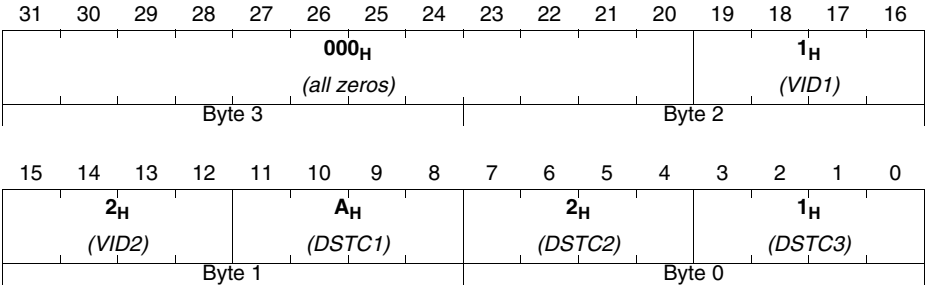
reserved **Reserved MPI Mode Bits**
The reset values of these bits depend on MPI mode pins MPIM(4:0).

Note: The reserved bits [31:1] must remain unchanged, hence the host must first read this register, logically OR its value with 0000 0001$_H$ and then write back the result!

4.5 Hardware Version ID Register

The 32-bit register HW_VID provides the Hardware Version ID of the used TE3-FALC as a 20-bit string.

HW_VID
Hardware Version ID Register **rd only** **Reset value: 0001 2A21$_H$**
Address: 0008$_H$



HW_VID Register Bits:

VID1(3:0)	**Version ID, 1st Digit**	= 1_H
VID2(3:0)	**Version ID, 2nd Digit**	= 2_H
DSTC1(3:0)	**Design Step Code, 1st Digit**	= A_H
DSTC2(3:0)	**Design Step Code, 2nd Digit**	= 2_H
DSTC3(3:0)	**Design Step Code, 3rd Digit**	= 1_H

The TE3-FALC Version 1.2 has the design step code A21, thus the 20-bit HW version string is 12A21$_H$

The upper 3 nibbles (12 bits) of this register are all zeros, the lower 5 nibbles all contain a value larger than zero, independent from the TE3-FALC hardware revision. This fact can be utilized to determine the host processor's addressing scheme (big endian or little endian). In a little endian environment for example, a byte access to address 0008$_H$ should result in a value different from zero, a byte access to address 000B$_H$ should result in the value 00$_H$.

Note: TE3-FALC Version 1.1 did not yet follow this version ID coding scheme; it returns the fixed value 0000 3802$_H$ on a read access to this register address!

4.6 Interrupt Registers

The DCI can assert the external interrupt pin $\overline{\text{MPINT}}$ to notify the host processor that messages in outgoing queues are ready to be read. The interrupt pin $\overline{\text{MPINT}}$ can be programmed to be open drain or push/pull active high or active low.

32-bit wide interrupt status (INTREG) and interrupt mask (INTMSK) registers are provided. Since TE3-FALC only allows 16- or 8-bit accesses, a shadow register is implemented. One byte of the INTREG register bytes is the indicator byte (destructive read). This byte must be accessed first (see **Table 14**). At that moment the interrupt status register is copied to the shadow register and then cleared. All other read accesses to the interrupt status register not accessing the indicator byte are served by the shadow register.

In case bit INTREG:DCIERR was read as '1', the DCIERR register must be read too, in order to clear its status bits.

INTREG
Interrupt Status Register **rd only** **Reset value: 0000 0000$_H$**
Address: 0010$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved															
Byte 3								Byte 2							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved								SEC	HPQ INSA	LPQ INSA	HPQ INFL	LPQ INFL	HPQ OUT	LPQ OUT	DCI ERR
Byte 1								Byte 0							

INTREG Register Bits:

DCIERR **DCI Error**
This bit indicates that a Queue-Overflow happened for at least one queue. The error details must be read from register DCIERR.

LPQOUT **Egress Low/High Priority Queue Interrupt**
HPQOUT This bit indicates that at least one new message is waiting to be read by the host from the corresponding egress queue.

LPQINFL **Ingress Low/High Priority Queue Full Interrupt**
HPQINFL This bit indicates that the corresponding ingress queue is full, i.e. the next write access might lead to an overflow.

INTREG Register Bits: (cont'd)

LPQINSA **Ingress Low/High Priority Queue Space Available Interrupt**
HPQINSA This bit indicates that the corresponding ingress queue has space available again, i.e. the host can continue writing to this queue.
This interrupt occurs only if the full-condition was reached before (see LPQINFL/HPQINFL) and the amount of queued messages fell back to four.

SEC **One-Second Interrupt**
This interrupt status bit is set in 1-second intervals.

INTMSK
Interrupt Mask Register **rd/wr** **Reset value: FFFF FFFF$_H$**
Address: 0014$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved															
Byte 3								Byte 2							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved								IM_SEC	IM_HPQ INSA	IM_LPQ INSA	IM_HPQ INFL	IM_LPQ INFL	IM_HPQ OUT	IM_LPQ OUT	IM_DCI ERR
Byte 1								Byte 0							

INTMSK Register Bits:

Each bit in this register masks the $\overline{\text{MPINT}}$ interrupt generation for the corresponding interrupt status bit of register INTREG. This does not prevent the interrupt status bit from being set to '1' if the corresponding interrupt event occurs.

DCIERR
DCI Error Indicator Register **rd only** **Reset value: 0000 0000$_H$**
Address: 0134$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved															
Byte 3								**Byte 2**							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved											ANE	HPQ INOV	LPQ INOV	HPQ OUT OV	LPQ OUT OV
Byte 1								**Byte 0**							

Note: In case of an INTREG:DCIERR interrupt, this register must be read to clear the interrupt!

DCIERR Register Bits:

LPQOUT OV HPQOUT OV **Egress Low/High Priority Queue Overflow Error**
This bit indicates that the corresponding egress queue experienced an overflow error and at least one message got lost. The host processor should read the corresponding egress queue at a higher rate, or the amount of egress messages should be reduced.

LPQINOV HPQINOV **Ingress Low/High Priority Queue Overflow Error**
This bit indicates that the corresponding ingress queue experienced an overflow error and at least one mesaage got lost. The host processor should limit the ingress message flow by evaluating the interrupt status bits INTREG:LPQINFL and INTREG:HPQINFL.

ANE **Access to Non-Existent Address**
This bit indicates that the host attempted to access an address which is out of the allowed range.

4.7 A Message Example

Consider the message **CMD_GET_VERSION_ID** with MSGID=0D$_H$. The acknowledgement (ACK) and its progress indication (PRG) is granted. The TCID is set to ABCD$_H$. Since this message is parameterless, one has to use the MAT21 and to write to LPQIN (see also **Table 13** and **Chapter 4.3**).

$ABCDC00D_H$ => @LPQIN : MAT21 ($TCID=ABCD_H$, A/P-Flag=1, PAA=0, $MSGID=0D_H$)

Then read from LPQOUT_CNT which was increased[1] by 2 accounting for the PRG and ACK[2].

@LPQOUT => $ABCD0200_H$: MAT22 ($TCID=ABCD_H$, $MSGID=200_H$)
@LPQOUT => $1180060D_H$: MAT12 ($MSGPT=1180_H$, RC=0, PAA=1, $MSGID=20D_H$)

The message **CMD_GET_VERSION_ID** has an acknowledgement (**ACK_GET_VERSION_ID**) of it's own with parameters. Hence it must be interpreted as MAT12 and then one has to read from the egress buffer:

@1180_H => $ABCD0018_H$: MSGHDR0 ($TCID=ABCD_H$, $LENIC=18_H$)
@1184_H => 00000001_H : MSGHDR1 (INFOFLD=1)
@1188_H => 00040006_H : 32-bit Array (Number of bytes=4_H, $ParameterID=4_H$)
@$118C_H$ => $00012A21_H$: 32-bit Value (Version ID; Hardware Version depending)
@1190_H => $0004000A_H$: 32-bit Array (Number of bytes=4_H, $ParameterID=8_H$)
@1194_H => 00122083_H : 32-bit Value (Version ID; Firmware Version depending)

After reading the parameter array the buffer is released:
(e.g. Hardware Version: V1.2 Build A21; Firmware Version: V1.2-1.2.3)

0000000_H => @1184 : MSGHDR1 (INFOFLD=0)

*Note: The device checks the LENIC against the expected LENIC and if there is a mismatch it acknowledges with RC=5 (see **"RC(3:0)" on Page 30**).*

1) Assume that the queue was empty before. Otherwise the pending messages should be read first.
2) Change can be recognized either by polling or by interrupt.

5 Message Catalog of Firmware V1.2-1.3.x

The below example of a command and corresponding acknowledge message show the structure of message descriptions used in the following chapters.

Command ID: **CMD_EXAMPLE_MSGID**

A brief description of the message is posted here.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
PARAMETER_ID$_P$	_VALUE1 _VALUE2	Value 1 of the optional parameter. Value 2 of the optional parameter (the underline marks it as default value).

Acknowledgement ID: **ACK_EXAMPLE_MSGID**

This acknowledgement is sent out when the command **CMD_EXAMPLE_MSGID** has been processed, provided that the A-flag was set within the message actuator for CMD_EXAMPLE_MSGID.

Output Parameters

Parameter ID	*Value*	*Description*
PARAMETER_ID$_{32}$	*32-bit field*	This is a 32-bit Value

All parameters consist of a 16-bit Parameter ID in capital letters (example: "PARAMETER_ID") and a 16-bit Value (example: "PARAMETER_ID_VALUE1" or "PARAMETER_ID_VALUE2"). The default value for each optional input parameter is underlined.

The type of each parameter is indicated with the appended index: ’P’ (packed), ’8’ (8-bit array), ’16’ (16-bit array) or ’32’ (32-bit array). Refer to **Chapter 4.2.1.1** for details.

When developing software for the TE3-FALC it is recommended to use the predefined value names (in capital letters), e.g. in a C-header file.

5.1 Device Maintenance Messages

Notification ID: **NFC_FW_ACCESSIBLE**

This notification is sent to the host after receipt firmware boot and initialization was completed successfully. The attached output parameters identify the hardware and firmware version ID of the device (see **CMD_GET_VERSION_ID** (page 47) for encoding).
This notification is enabled by default and can not be disabled.

Output Parameters

Parameter ID	*Value*	*Description*
$HW\_VERSION\_ID_{32}$	32-bit value	Hardware version ID
$FW\_VERSION\_ID_{32}$	32-bit value	Firmware version ID

Command ID: **CMD_SPECIFY_CLOCKS**

This message causes the firmware to set the internal clocks of the device. The frequency of an input clock is used to calculate the Clock Multiplication Factor for raising input clock to high frequency master clock, and the Clock Division Factor for generating the desired clock (e.g., Line Speed clock) from high frequency master clock.
The parameter for CLKIN frequency is mandatory.
This message should be the first message sent to TE3-FALC.

Note: Wrong PLL frequency programming might lead to internal overclocking and can thus influence the reliablitiy of the device's operation!

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$CLK\_FRQ\_CLKIN_{32}$	*32-bit field*	Value providing the CLKIN input frequency in Hz (4,000,000 to 52,000,000 Hz).
$CLK\_FRQ\_CLKOUT_{32}$	*32-bit field*	Optional value providing the CLKOUT output frequency in Hz (15,000,000 to 52,000,000 Hz). Default value zero means this clock output is disabled.

Command ID: **CMD_SPECIFY_CLOCKS** (cont'd)

$CLK\_SRC\_TCLKIN_P$	_DIRECT	TCLKIN is used directly as transmit clock.
	_VIA_PLL	The transmit PLL can derive the target transmit clock frequency from the clock provided on pin TCLKIN. In this case the TCLKIN frequency must be specified with parameter CLK_FRQ_TCLKIN.
	If the transmit master clock shall be derived from CLKIN, the clock input TCLKIN is not used and this parameter is not relevant.	
$CLK\_FRQ\_TCLKIN_{32}$	*32-bit field*	Optional value providing the TCLKIN input frequency in Hz (8,000 to 52,000,000 Hz). Default value: 0
$CLK\_FRQ\_RSYNC_{32}$	*32-bit field*	Optional value providing the RSYNC input frequency in Hz (8,000 to 52,000,000 Hz). Default value: 0

Command ID: **CMD_SPECIFY_CLOCKS** (cont'd)

$CLK\_SRC\_RCLKOUT_P$	_DIRECT	The recovered receive clock is directly output on pin RCLKOUT. If the receive DJAT is enabled, the jitter attenuated receive clock is output.
	_VIA_PLL	The receive PLL can derive the frequency programmed with CLK_FRQ_RCLKOUT from the clock provided on RSYNC and output it on pin RCLKOUT. If the parameter of CLK_FRQ_RSYNC is 0 automatically the recovered receive route clock is used instead of the clock provided on pin RSYNC.
	*The option CLK_SRC_RCLKOUT_VIA_PLL is only available if the receive DJAT is disabled (**CMD_CFG_DJAT** (page 68)), otherwise the receive PLL is already occupied for the DJAT function.*	
$CLK\_FRQ\_RCLKOUT_{32}$	*32-bit field*	Optional value providing the RCLKOUT output frequency in Hz (8,000 to 52,000,000 Hz). Default value zero means this clock output is disabled. If CLK_SRC_RCLKOUT_DIRECT is selected only frequencies devided by an integer value of the recovered route clock can be output.

Command ID: **CMD_CFG_PLL**

This message controls the characteristic of the receive and transmit PLLs.
If the PLL characteristic shall be changed this message must be issued prior to sending command **CMD_CFG_DJAT** (page 68)!

Command ID: **CMD_CFG_PLL** (cont'd)

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$PLL\_RX\_CHARAC_P$	_P_ELEMENT	The receive PLL has a P characteristic. In case of loss of reference clock the PLL enters the "Center Mode"
	_PI_ELEMENT	The receive PLL has a PI characteristic. In case of loss of reference clock the PLL enters the "Holdover Mode"
$PLL\_TX\_CHARAC_P$	_P_ELEMENT	The transmit PLL has a P characteristic. In case of loss of reference clock the PLL enters the "Center Mode"
	_PI_ELEMENT	The transmit PLL has a PI characteristic. In case of loss of reference clock the PLL enters the "Holdover Mode"

Notification ID: **NFC_PLL_LOCKED**

After power up the internal PLLs are in the unlocked state. If all clocking setting are applied, the internal PLLs shall reach the locked state. This is reported by this message.
To enable this notification use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters

Parameter ID	*Value*	*Description*
$INTERNAL\_PLL_P$	_RX	The reference clock of RX PLL is running
	_TX	The reference clock of TX PLL is running
	_ANALOG	On-chip main PLL (CLKIN) entered locked state

Notification ID: **NFC_PLL_UNLOCKED**

If an internal PLL switched from the locked state into the unlocked state this message will be generated.
To enable this notification use message **CMD_SET_NFC_ALM_MODE** (page 45).

Notification ID: **NFC_PLL_UNLOCKED** (cont'd)

Output Parameters

Parameter ID	*Value*	*Description*
INTERNAL_PLL$_P$	_RX	The reference clock of RX PLL is missing. The PLL is in hold/center mode (depends on CMD_CFG_PLL)
	_TX	The reference clock of TX PLL is missing. The PLL is in hold/center mode (depends on CMD_CFG_PLL)
	_ANALOG	On-chip main PLL (CLKIN) entered unlocked state

Command ID: **CMD_SET_INTERRUPT_MODE**

With this command the output characteristics of the interrupt output pin $\overline{\text{MPINT}}$ can be programmed. Interrupt sources are determined by reading register INTREG (see data sheet). Interrupt events can be masked via register INTMSK (see data sheet).

Input Parameters (optional)

Parameter ID	*Value*	*Description*
INT_PORT_CFG$_P$	_DISABLE	Interrupt output is disabled (tri-state).
	_OPEN_DRAIN	Interrupt output is open-drain.
	_ACT_HIGH	Interrupt output is push/pull, active high.
	_ACT_LOW	Interrupt output is push/pull, active low.

Command ID: **CMD_SET_NFC_ALM_MODE**

This command assigns the selected notification or alarm message to one of the outgoing message queues (LPQOUT, HPQOUT) at the user interface. Assigning NOQOUT inhibits sending out a message to the user interface.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
MSG_TO_HPQOUT$_P$	*16-bit field*	10-bit message identifier MSGID(9:0) of the Alarm or Notification that shall be assigned to high priority egress queue.

Command ID: **CMD_SET_NFC_ALM_MODE** (cont'd)

$MSG\_TO\_LPQOUT_P$	*16-bit field*	10-bit message identifier MSGID(9:0) of the Alarm or Notification that shall be assigned to low priority egress queue.
$MSG\_TO\_NOQOUT_P$	*16-bit field*	10-bit message identifier MSGID(9:0) of the Alarm or Notification that shall not be assigned to any egress queue, i.e. Alarm/ Notification generation is suppressed.

Note: The above input parameters can be issued multiple times. That way it is possible to e.g. enable a group of alarm messages by issuing this message CMD_SET_NFC_ALM_MODE only once.

Command ID: **CMD_SET_LOOP**

This message causes the firmware to activate or deactivate a specific data path loop. For details on provided loop options refer to data sheet.

*Note: It's expected to enable a loop after the setup of the data flow configuration is finished (e.g. after **CMD_ENABLE_INTERFACES** (page 67).*

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$LOOP\_LIU_P$	_REMOTE	Remote Line Loop (LIU)
	_LOCAL	Local Line Loop (LIU)
	_OFF	No analog Line Loop
$LOOP\_FRAMER_P$	_REMOTE	✧Remote Line Loop (Framer)[1)]
	_LOCAL	Local Line Loop (Framer)
	_OFF	No Line Loop (Framer)
$LOOP\_SYSIF_P$	_LOCAL	Local System Interface Loop
	_OFF	No System Interface Loop

1) This is an option which is not available with the current firmware release.

Command ID: **CMD_GET_VERSION_ID**

This message initiates the firmware to send the device hardware version ID and firmware version ID.

The hardware version ID is decoded from the 32-bit hexadecimal HW_VERSION_ID as follows:

<as_digit[19:16]> **.** *<as_digit[15:12]>* – *<as_alpha[11:8]><as_digit[7:4]><as_digit[3:0]>*

E.g., HW_VERSION_ID = $00012A21_H$ reads as "V1.2-A21" and means Hardware Version 1.2, Build A21.

The firmware version ID is maintained by the firmware in a constant data type and is decoded from the 32-bit hexadecimal FW_VERSION_ID as follows:

Firmware Build: *<as_digit[23:20]>***.***<as_digit[19:16]>*
Feature Set: *<as_digit[15:13]>***.***<as_digit[12:6]>*
Development Step: *<as_digit[5:0]>*

E.g., FW_VERSION_ID = 00122083_H means Firmware Build 1.2, Feature Set 1.2, Development Step 3.

Input Parameters	*none*

Acknowledgement ID: **ACK_GET_VERSION_ID**

This acknowledgement is sent to the host after receipt of the **CMD_GET_VERSION_ID** message to identify the hardware and firmware version of the device.

Output Parameters

Parameter ID	*Value*	*Description*
$HW\_VERSION\_ID_{32}$	32-bit value	Hardware version ID
$FW\_VERSION\_ID_{32}$	32-bit value	Firmware version ID

Command ID: **CMD_SELF_DIAGNOSTICS**

This commands triggers the execution of the device-selftest. The result of the device selftest is returned to the user via the standard acknowledge message.
Because the execution of this command is service disruptive a subsequent device reset with initialization is mandatory.

Command ID: **CMD_SELF_DIAGNOSTICS** (cont'd)

Input Parameters *none*

Acknowledgement ID: **ACK_SELF_DIAGNOSTICS**

This message is sent to the user to provide the results of the self-diagnostics run after requested so by the host via **CMD_SELF_DIAGNOSTICS** message.

Output Parameters

Parameter ID	*Value*	*Description*
$TEST\_RESULT_P$	_OK	The self-test has been executed successfully.
	_FAIL	One or more tests failed.
$ERROR\_CODE_P$	*16-bit field*	Manufacturer specific error code.

Command ID: **CMD_RESET_DEVICE**

This message causes the firmware to perform a reset of the TE3-FALC. All hardware modules are reset. All interrupts except for DCI interrupts are disabled.
For details on the boot process refer to data sheet.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$REBOOT\_SOURCE_P$	_CURR_IMAGE	Reboot from current image.
	_PRESET	Load new image according to preset Hardware Boot Selection and reboot.

Command ID: **CMD_READBACK**

This command allows to read back the complete operating state of the TE3-FALC. Note that the **CMD_GET_STATUS** (page 100) message delivers additional information of the chip state.

Command ID: **CMD_READBACK** (cont'd)

Input Parameters (optional)

Parameter ID	*Value*	*Description*
READ_BLOCK$_P$	_MODULE_CFG	Read back the current configuration of all chip internal modules. This includes the parameter set of the following messages (partly depending on the set data flow): CMD_SPECIFY_CLOCKS, CMD_CFG_PLL CMD_SET_INTERRUPT_MODE, CMD_SET_LOOP, CMD_CFG_LIU, CMD_CFG_DJAT, CMD_CFG_DS3_FRAMER, CMD_CFG_E3G832_FRAMER, CMD_CFG_E3G751_FRAMER, CMD_CFG_HDLC_PROCESSOR, CMD_ENABLE_INTERFACES, CMD_CFG_ATM_PROCESSOR, CMD_CFG_SYSIF, CMD_CFG_BUFFERS,
	_LINK_MAINT	Read back the state and configuration of the link maintenance channels. This includes the parameter set of the following messages (partly depending on the set data flow): CMD_CFG_TTI_CHANNEL (E3 G.832) CMD_CFG_PTY_CHANNEL (E3 G.832) CMD_CFG_SSM_CHANNEL (E3 G.832) CMD_CFG_MDL_CHANNEL (E3 G.832) CMD_SET_FEAC_CODE (DS3) CMD_CFG_RDI_DECLARATION CMD_CFG_FEAC_GENERATION (DS3) CMD_CFG_GPIO

Acknowledgement ID: **ACK_READBACK**

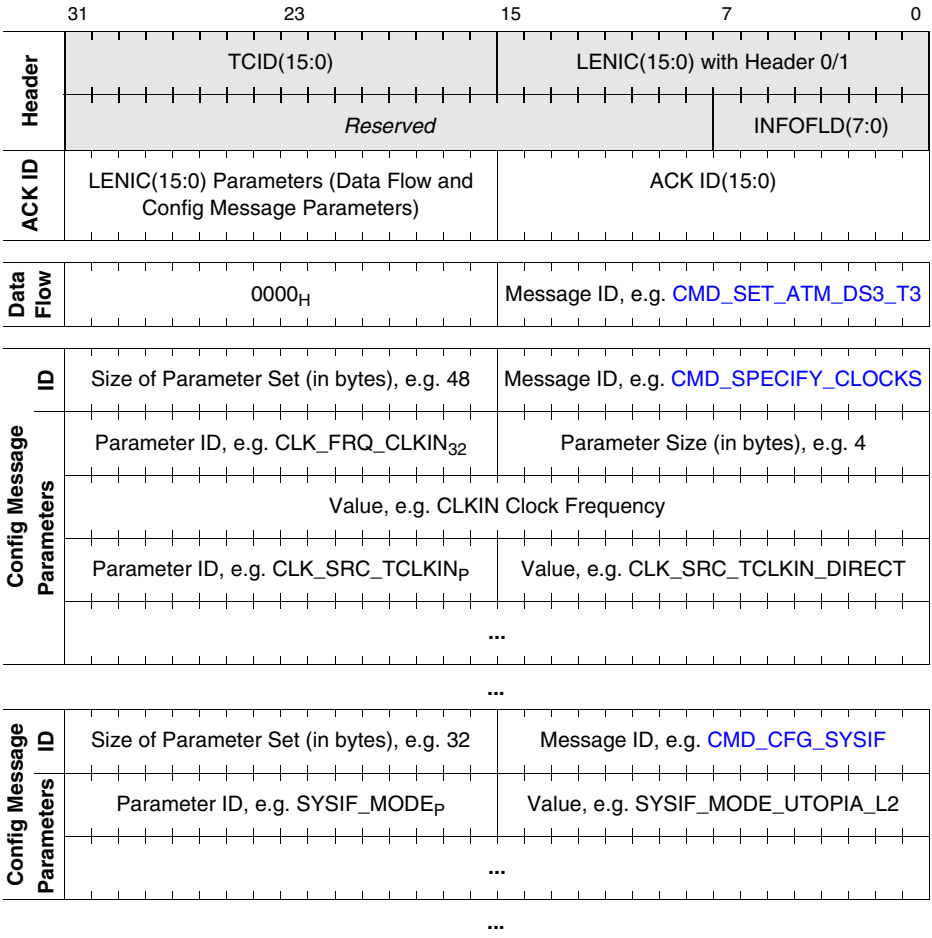
This message is sent to the user to provide TE3-FALC's state and configuration data requested by the host via **CMD_READBACK** message.

Note: Readback data can be split into 1..4 ***ACK_READBACK*** *messages!*

Note: Parameters, not applicable to the selected mode, have to be ignored!

The read-back data block contains all configuration parameters, grouped in the way the individual configuration messages are defined. The first 32-bit word of read-back data always contains the configured data flow in form of the corresponding message ID. The following data sections each consist of the message ID of the corresponding configuration message followed by the parameter set of that message. The set of messages that are part of the read-back data structure depends on the selected data flow, for example if the DS3 framer is active in the current flow, the E3-specific configuration messages/parameters will not be delivered.

The structure of the acknowledgement read-back data is shown below:



The above generic structure allows the host to pick only the relevant parameters out of the prepared read-back data structure as needed. The host's reading sequence would be as follows:

- Read Message ID
  – If relevant, read the parameter set
  – If not relevant, skip to the next Message ID by evaluating the byte count of the current message's parameter set.
- Read next Message ID

...

Notification ID: **NFC_DEVICE_NOT_CONFIGURED**

This message informs the user that CMD_READBACK was done before the first flow (see **Table 15**) was setup.

Output Parameters	*none*

5.2 Data Flow Configuration Message

The configuration process should be hierarchical. A *Data Flow Configuration Function* selects a specific data flow between the system side and the line side of the TE3-FALC device. According to the selected data flow all involved modules are configured by module specific configuration functions. A number of default settings are applied which can individually be changed by **Module Configuration Messages** (page 67).

After configuration is completed, the interfaces can be enabled with the **CMD_ENABLE_INTERFACES** (page 67) command.

All applicable types of data flow are listed in **Table 15**. The left column (Type of Data Flow) identifies the selected type of data flow, whereas the other columns identify the module specific operating modes.

*Note: It's strongly recommended to perform a software reset (**CMD_RESET_DEVICE** (page 48)) before re-configuration of data flows.*

Table 15 Supported Operational Modes

Type of Data Flow (Command Message)	LIU	Framer [1]	FDL/ TTI	PLCP	G.804	PPP	System side Interface [2]
ATM - G.832 - E3 (CMD_SET_ATM_E3G832_E3)	E3	E3 G.832	TTI	*n.a.*	G.804	*n.a.*	UTOPIA/ POS-PHY/ UTOPIA-X
ATM - G.751 - E3 (CMD_SET_ATM_E3G751_E3)	E3	E3 G.751	*n.a.*	*n.a.*	G.804	*n.a.*	UTOPIA/ POS-PHY/ UTOPIA-X
ATM - PLCP - G.751 - E3 (CMD_SET_ATM_PLCP_E3G751_E3)	E3	E3 G.751	*n.a.*	PLCP	G.804	*n.a.*	UTOPIA/ POS-PHY/ UTOPIA-X
ATM DS3 T3 (CMD_SET_ATM_DS3_T3)	T3	DS3	FEAC MDL	*n.a.*	G.804	*n.a.*	UTOPIA/ POS-PHY/ UTOPIA-X
ATM - PLCP - DS3 - T3 (CMD_SET_ATM_PLCP_DS3_T3)	T3	DS3	FEAC MDL	PLCP	G.804	*n.a.*	UTOPIA/ POS-PHY/ UTOPIA-X
PPP - G.832 - E3 (CMD_SET_PPP_E3G832_E3)	E3	E3 G.832	TTI	*n.a.*	*n.a.*	PPP	POS-PHY/ UTOPIA-X
PPP - G.751 - E3 (CMD_SET_PPP_E3G751_E3)	E3	E3 G.751	*n.a.*	*n.a.*	*n.a.*	PPP	POS-PHY/ UTOPIA-X
PPP - DS3 - T3 (CMD_SET_PPP_DS3_T3)	T3	DS3	FEAC MDL	*n.a.*	*n.a.*	PPP	POS-PHY/ UTOPIA-X
ATM - E3 (CMD_SET_ATM_E3)	E3	bypass	*n.a.*	*n.a.*	G.804	*n.a.*	UTOPIA/ POS-PHY/ UTOPIA-X
ATM - T3 (CMD_SET_ATM_T3)	T3	bypass	*n.a.*	*n.a.*	G.804	*n.a.*	UTOPIA/ POS-PHY/ UTOPIA-X
PPP - E3 (CMD_SET_PPP_E3)	E3	bypass	*n.a.*	*n.a.*	*n.a.*	PPP	POS-PHY/ UTOPIA-X
PPP - T3 (CMD_SET_PPP_T3)	T3	bypass	*n.a.*	*n.a.*	*n.a.*	PPP	POS-PHY/ UTOPIA-X
ATM - G.832 (CMD_SET_ATM_E3G832)	bypass	E3 G.832	TTI	*n.a.*	G.804	*n.a.*	UTOPIA/ POS-PHY/ UTOPIA-X
ATM - G.751 (CMD_SET_ATM_E3G751)	bypass	E3 G.751	*n.a.*	*n.a.*	G.804	*n.a.*	UTOPIA/ POS-PHY/ UTOPIA-X

Table 15 Supported Operational Modes (cont'd)

Type of Data Flow (Command Message)	LIU	Framer [1]	FDL/ TTI	PLCP	G.804	PPP	System side Interface[2]
ATM - PLCP - G.751 (CMD_SET_ATM_PLCP_E3G751)	bypass	E3 G.751	*n.a.*	PLCP	G.804	*n.a.*	UTOPIA/ POS-PHY/ UTOPIA-X
ATM - DS3 (CMD_SET_ATM_DS3)	bypass	DS3	FEAC MDL	*n.a.*	G.804	*n.a.*	UTOPIA/ POS-PHY/ UTOPIA-X
ATM - PLCP - DS3 (CMD_SET_ATM_PLCP_DS3)	bypass	DS3	FEAC MDL	PLCP	G.804	*n.a.*	UTOPIA/ POS-PHY/ UTOPIA-X
PPP - G.832 (CMD_SET_PPP_E3G832)	bypass	E3 G.832	TTI	*n.a.*	*n.a.*	PPP	POS-PHY/ UTOPIA-X
PPP - G.751 (CMD_SET_PPP_E3G751)	bypass	E3 G.751	*n.a.*	*n.a.*	*n.a.*	PPP	POS-PHY/ UTOPIA-X
PPP - DS3 (CMD_SET_PPP_DS3)	bypass	DS3	FEAC MDL	*n.a.*	*n.a.*	PPP	POS-PHY/ UTOPIA-X
Bitstream G.832 E3 (CMD_SET_BIT_E3G832_E3)	E3	E3 G.832	TTI	not used			Bitstream Access
Bitstream G.751 E3 (CMD_SET_BIT_E3G751_E3)	E3	E3 G.751	*n.a.*	not used			Bitstream Access
Bitstream - DS3 - T3 (CMD_SET_BIT_DS3_T3)	T3	DS3	FEAC MDL	not used			Bitstream Access
Bitstream - G.832 (CMD_SET_BIT_E3G832)	bypass	E3 G.832	TTI	not used			Bitstream Access
Bitstream - G.751 (CMD_SET_BIT_E3G751)	bypass	E3 G.751	*n.a.*	not used			Bitstream Access
Bitstream - DS3 (CMD_SET_BIT_DS3)	bypass	DS3	FEAC MDL	not used			Bitstream Access

1) For DS3 modes, the framer evaluates the AIC bit to dynamically select C-bit parity or M23 unchannelized mode

2) One of the alternative system side interface configurations can be selected at configuration time

Command ID: **CMD_SET_ATM_E3G832_E3**
Command ID: **CMD_SET_ATM_E3G832**

Message CMD_SET_ATM_E3G832_E3 sets up the path
System Interface <=> G.804 <=> E3:G.832 <=> E3 LIU.
Message CMD_SET_ATM_E3G832 sets the same data path with digital line interface:
System Interface <=> G.804 <=> E3:G.832.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
LINE_TIMING$_P$	_MASTER_INT	E3 Line internal Master timing (CLKIN)
	_MASTER_EXT	E3 Line external Master timing (TCLKIN)
	_LOOP	E3 Line Loop timing
SYSIF_MODE$_P$	_UTOPIA_L1	UTOPIA Level 1 mode
	_UTOPIA_L2	UTOPIA Level 2 mode
	_POSPHY_L2	POS-PHY Level 2 mode
	_UTOPIA_L2X	UTOPIA Level 2 eXtended mode
SYSIF_WIDTH$_P$	_8BIT	8-bit data bus width
	_16BIT	16-bit data bus width
SYSIF_ADDR$_P$	*16-bit field*	PHY address (0 to 30). Default is 0.
SYSIF_STATUS$_P$	_POLLING	MPHY Polling
	_DIRECT_IND	Direct Status Indication

Default Settings *(These settings can be modified by Module Configuration Messages)*

E3 G.832 Framer	• SSM pattern is disabled (set to 0000_2). • Payload Type field set to 'ATM'
ATM Processor	• Payload Scrambling enabled • Header Correction enabled • ALPHA = 7, DELTA = 6
Serial Data Access	• E3 Bitstream Breakout interface disabled • E3 Overhead Access interface disabled
Data Buffers	• Burst-/Chunksize set to 64 bytes (only for POS-PHY / UTOPIA-L2X) • Forward Threshold set to 64 bytes (only for POS-PHY / UTOPIA-L2X)
System Interface	• Cell-Level handshake (UTOPIA only) / Packet-Level handshake (POS-PHY only) • Tri-statable bus, allows configuration with multiple PHYs • Parity check enabled

Command ID: **CMD_SET_ATM_E3G751_E3**
Command ID: **CMD_SET_ATM_E3G751**

Message CMD_SET_ATM_E3G751_E3 sets up the path
System Interface <=> G.804 <=> E3:G.751 <=> E3 LIU.
Message CMD_SET_ATM_E3G751 sets the same data path with digital line interface:
System Interface <=> G.804 <=> E3:G.751.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
LINE_TIMING$_P$	_MASTER_INT	E3 Line internal Master timing (CLKIN)
	_MASTER_EXT	E3 Line external Master timing (TCLKIN)
	_LOOP	E3 Line Loop timing
SYSIF_MODE$_P$	_UTOPIA_L1	UTOPIA Level 1 mode
	_UTOPIA_L2	UTOPIA Level 2 mode
	_POSPHY_L2	POS-PHY Level 2 mode
	_UTOPIA_L2X	UTOPIA Level 2 eXtended mode
SYSIF_WIDTH$_P$	_8BIT	8-bit data bus width
	_16BIT	16-bit data bus width
SYSIF_ADDR$_P$	*16-bit field*	PHY address (0 to 30). Default is 0.
SYSIF_STATUS$_P$	_POLLING	MPHY Polling
	_DIRECT_IND	Direct Status Indication

Default Settings *(These settings can be modified by Module Configuration Messages)*

E3 G.751 Framer	• ATM Frame Format
ATM Processor	• Payload Scrambling enabled • Header Correction enabled • ALPHA = 7, DELTA = 6
Serial Data Access	• E3 Bitstream Breakout interface disabled • E3 Overhead Access interface disabled
Data Buffers	• Burst-/Chunksize set to 64 bytes (only for POS-PHY / UTOPIA-L2X) • Forward Threshold set to 64 bytes (only for POS-PHY / UTOPIA-L2X)
System Interface	• Cell-Level handshake (UTOPIA only) / Packet-Level handshake (POS-PHY only) • Tri-statable bus, allows configuration with multiple PHYs • Parity check enabled

Command ID: **CMD_SET_ATM_PLCP_E3G751_E3**
Command ID: **CMD_SET_ATM_PLCP_E3G751**

Message CMD_SET_ATM_PLCP_E3G751_E3 sets up the data path
System Interface <=> G.804 <=> PLCP <=> E3:G.751 <=> E3 LIU.
Message CMD_SET_ATM_PLCP_E3G751 sets the same data path with digital line i/f:
System Interface <=> G.804 <=> PLCP <=> E3:G.751.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$LINE\_TIMING_P$	_MASTER_INT	E3 Line internal Master timing (CLKIN)
	_MASTER_EXT	E3 Line external Master timing (TCLKIN)
	_LOOP	E3 Line Loop timing
$SYSIF\_MODE_P$	_UTOPIA_L1	UTOPIA Level 1 mode
	_UTOPIA_L2	UTOPIA Level 2 mode
	_POSPHY_L2	POS-PHY Level 2 mode
	_UTOPIA_L2X	UTOPIA Level 2 eXtended mode
$SYSIF\_WIDTH_P$	_8BIT	8-bit data bus width
	_16BIT	16-bit data bus width
$SYSIF\_ADDR_P$	*16-bit field*	PHY address (0 to 30). Default is 0.
$SYSIF\_STATUS_P$	_POLLING	MPHY Polling
	_DIRECT_IND	Direct Status Indication

Default Settings *(These settings can be modified by Module Configuration Messages)*

E3 G.751 Framer	• ATM Frame Format.
ATM Processor	• PLCP mapping • Payload Scrambling enabled • Header Correction enabled
Serial Data Access	• DS3 Bitstream Breakout interface disabled • DS3 Overhead Access interface disabled
Data Buffers	• Burst-/Chunksize set to 64 bytes (only for POS-PHY / UTOPIA-L2X) • Forward Threshold set to 64 bytes (only for POS-PHY / UTOPIA-L2X)
System Interface	• Cell-Level handshake (UTOPIA only) / Packet-Level handshake (POS-PHY only) • Tri-statable bus, allows configuration with multiple PHYs • Parity check enabled

Command ID: **CMD_SET_ATM_DS3_T3**
Command ID: **CMD_SET_ATM_DS3**

Message CMD_SET_ATM_DS3_T3 sets up the data path
System Interface <=> G.804 <=> DS3 <=> T3 LIU.
Message CMD_SET_ATM_DS3 sets the same data path with digital line interface:
System Interface <=> G.804 <=> DS3.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$LINE\_TIMING_P$	_MASTER_INT	T3 Line internal Master timing (CLKIN)
	_MASTER_EXT	T3 Line external Master timing (TCLKIN)
	_LOOP	T3 Line Loop timing
$SYSIF\_MODE_P$	_UTOPIA_L1	UTOPIA Level 1 mode
	_UTOPIA_L2	UTOPIA Level 2 mode
	_POSPHY_L2	POS-PHY Level 2 mode
	_UTOPIA_L2X	UTOPIA Level 2 eXtended mode
$SYSIF\_WIDTH_P$	_8BIT	8-bit data bus width
	_16BIT	16-bit data bus width
$SYSIF\_ADDR_P$	*16-bit field*	PHY address (0 to 30). Default is 0.
$SYSIF\_STATUS_P$	_POLLING	MPHY Polling
	_DIRECT_IND	Direct Status Indication

Default Settings *(These settings can be modified by Module Configuration Messages)*

DS3 Framer	• Auto sensing of application via AIC channel (C-Bit Parity / M23 unchannelized mode) • AIS signal defined as 101010_2... sequence.
ATM Processor	• Payload Scrambling enabled • Header Correction enabled • Direct Mapping (ALPHA = 7, DELTA = 6)
Serial Data Access	• DS3 Bitstream Breakout interface disabled • DS3 Overhead Access interface disabled
Data Buffers	• Burst-/Chunksize set to 64 bytes (only for POS-PHY / UTOPIA-L2X) • Forward Threshold set to 64 bytes (only for POS-PHY / UTOPIA-L2X)
System Interface	• Cell-Level handshake (UTOPIA only) / Packet-Level handshake (POS-PHY only) • Tri-statable bus, allows configuration with multiple PHYs • Parity check enabled

Command ID: **CMD_SET_ATM_PLCP_DS3_T3**
Command ID: **CMD_SET_ATM_PLCP_DS3**

Message CMD_SET_ATM_PLCP_DS3_T3 sets up the data path
System Interface <=> G.804 <=> PLCP <=> DS3 <=> T3 LIU.
Message CMD_SET_ATM_PLCP_DS3 sets the same data path with digital line i/f:
System Interface <=> G.804 <=> PLCP <=> DS3.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$LINE\_TIMING_P$	_MASTER_INT	T3 Line internal Master timing (CLKIN)
	_MASTER_EXT	T3 Line external Master timing (TCLKIN)
	_LOOP	T3 Line Loop timing
$SYSIF\_MODE_P$	_UTOPIA_L1	UTOPIA Level 1 mode
	_UTOPIA_L2	UTOPIA Level 2 mode
	_POSPHY_L2	POS-PHY Level 2 mode
	_UTOPIA_L2X	UTOPIA Level 2 eXtended mode
$SYSIF\_WIDTH_P$	_8BIT	8-bit data bus width
	_16BIT	16-bit data bus width
$SYSIF\_ADDR_P$	*16-bit field*	PHY address (0 to 30). Default is 0.
$SYSIF\_STATUS_P$	_POLLING	MPHY Polling
	_DIRECT_IND	Direct Status Indication

Default Settings *(These settings can be modified by Module Configuration Messages)*

DS3 Framer	• Auto sensing of application via AIC channel (C-Bit Parity / M23 unchannelized mode) • AIS signal defined as 101010_2... sequence.
ATM Processor	• PLCP mapping • Payload Scrambling enabled • Header Correction enabled
Serial Data Access	• DS3 Bitstream Breakout interface disabled • DS3 Overhead Access interface disabled
Data Buffers	• Burst-/Chunksize set to 64 bytes (only for POS-PHY / UTOPIA-L2X) • Forward Threshold set to 64 bytes (only for POS-PHY / UTOPIA-L2X)
System Interface	• Cell-Level handshake (UTOPIA only) / Packet-Level handshake (POS-PHY only) • Tri-statable bus, allows configuration with multiple PHYs • Parity check enabled

Command ID: **CMD_SET_PPP_E3G832_E3**
Command ID: **CMD_SET_PPP_E3G832**

Message CMD_SET_PPP_E3G832_E3 sets up the data path
System Interface <=> PPP <=> E3:G.832 <=> E3 LIU.
Message CMD_SET_PPP_E3G832 sets the same data path with digital line interface:
System Interface <=> PPP <=> E3:G.832.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$LINE\_TIMING_P$	_MASTER_INT	E3 Line internal Master timing (CLKIN)
	_MASTER_EXT	E3 Line external Master timing (TCLKIN)
	_LOOP	E3 Line Loop timing
$SYSIF\_MODE_P$	_POSPHY_L2	POS-PHY Level 2 mode
	_UTOPIA_L2X	UTOPIA Level 2 eXtended mode
$SYSIF\_WIDTH_P$	_8BIT	8 bit data bus width
	_16BIT	16 bit data bus width
$SYSIF\_ADDR_P$	*16-bit field*	PHY address (0 to 30). Default is 0.
$SYSIF\_STATUS_P$	_POLLING	MPHY Polling
	_DIRECT_IND	Direct Status Indication

Default Settings *(These settings can be modified by Module Configuration Messages)*

E3 G.832 Framer	• SSM pattern is disabled (set to 0000_2). • Payload Type field set to 'non-specific'
PPP Processor	• Bit-synchronous PPP mode • 16-bit CRC • Shared Flag between two frames • Address/Control-Field Compression disabled
Serial Data Access	• E3 Bitstream Breakout interface disabled • E3 Overhead Access interface disabled
Data Buffers	• Burst-/Chunksize set to 64 bytes • Forward Threshold set to 64 bytes
System Interface	• Packet-Level handshake (POS-PHY only) • Tri-statable bus, allows configuration with multiple PHYs

Command ID: **CMD_SET_PPP_E3G751_E3**
Command ID: **CMD_SET_PPP_E3G751**

Message CMD_SET_PPP_E3G751_E3 sets up the data path
System Interface <=> PPP <=> E3:G.751 <=> E3 LIU.
Message CMD_SET_PPP_E3G751 sets the same data path with digital line interface:
System Interface <=> PPP <=> E3:G.751.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$LINE\_TIMING_P$	_MASTER_INT	E3 Line internal Master timing (CLKIN)
	_MASTER_EXT	E3 Line external Master timing (TCLKIN)
	_LOOP	E3 Line Loop timing
$SYSIF\_MODE_P$	_POSPHY_L2	POS-PHY Level 2 mode
	_UTOPIA_L2X	UTOPIA Level 2 eXtended mode
$SYSIF\_WIDTH_P$	_8BIT	8-bit data bus width
	_16BIT	16-bit data bus width
$SYSIF\_ADDR_P$	*16-bit field*	PHY address (0 to 30). Default is 0.
$SYSIF\_STATUS_P$	_POLLING	MPHY Polling
	_DIRECT_IND	Direct Status Indication

Default Settings *(These settings can be modified by Module Configuration Messages)*

E3 G.751 Framer	• Non-ATM Frame Format with C-bits set to '1'.
PPP Processor	• Bit-synchronous PPP mode • 16-bit CRC • Shared Flag between two frames • Address/Control-Field Compression disabled
Serial Data Access	• E3 Bitstream Breakout interface disabled • E3 Overhead Access interface disabled
Data Buffers	• Burst-/Chunksize set to 64 bytes • Forward Threshold set to 64 bytes
System Interface	• Packet-Level handshake (POS-PHY only) • Tri-statable bus, allows configuration with multiple PHYs

Command ID: **CMD_SET_PPP_DS3_T3**
Command ID: **CMD_SET_PPP_DS3**

Message CMD_SET_PPP_DS3_T3 sets up the data path
System Interface <=> PPP <=> DS3 <=> T3 LIU.
Message CMD_SET_PPP_DS3 sets the same data path with digital line interface:
System Interface <=> PPP <=> DS3.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$LINE\_TIMING_P$	_MASTER_INT _MASTER_EXT _LOOP	T3 Line internal Master timing (CLKIN) T3 Line external Master timing (TCLKIN) T3 Line Loop timing
$SYSIF\_MODE_P$	_POSPHY_L2 _UTOPIA_L2X	POS-PHY Level 2 mode UTOPIA Level 2 eXtended mode
$SYSIF\_WIDTH_P$	_8BIT _16BIT	8-bit data bus width 16-bit data bus width
$SYSIF\_ADDR_P$	*16-bit field*	PHY address (0 to 30). Default is 0.
$SYSIF\_STATUS_P$	_POLLING _DIRECT_IND	MPHY Polling Direct Status Indication

Default Settings *(These settings can be modified by Module Configuration Messages)*

DS3 Framer	• Auto sensing of application via AIC channel (C-Bit Parity / M23 unchannelized mode) • AIS signal defined as 101010_2... sequence.
PPP Processor	• 16-bit CRC • Shared Flag between two frames • Address/Control-Field Compression disabled
Serial Data Access	• DS3 Bitstream Breakout interface disabled • DS3 Overhead Access interface disabled
Data Buffers	• Burst-/Chunksize set to 64 bytes • Forward Threshold set to 64 bytes
System Interface	• Packet-Level handshake (POS-PHY only) • Tri-statable bus, allows configuration with multiple PHYs

Command ID: **CMD_SET_ATM_E3**

This message sets up the path

System Interface <=> G.804 <=> E3 LIU ,

with the E3 framer being bypassed.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$LINE\_TIMING_P$	_MASTER_INT	E3 Line internal Master timing (CLKIN)
	_MASTER_EXT	E3 Line external Master timing (TCLKIN)
	_LOOP	E3 Line Loop timing
$SYSIF\_MODE_P$	_UTOPIA_L1	UTOPIA Level 1 mode
	_UTOPIA_L2	UTOPIA Level 2 mode
	_POSPHY_L2	POS-PHY Level 2 mode
	_UTOPIA_L2X	UTOPIA Level 2 eXtended mode
$SYSIF\_WIDTH_P$	_8BIT	8-bit data bus width
	_16BIT	16-bit data bus width
$SYSIF\_ADDR_P$	*16-bit field*	Physical address (0 to 30). Default is 0.
$SYSIF\_STATUS_P$	_POLLING	MPHY Polling
	_DIRECT_IND	Direct Status Indication

Default Settings *(These settings can be modified by Module Configuration Messages)*

ATM Processor	• Payload Scrambling enabled • Header Correction enabled • ALPHA = 7, DELTA = 6
Serial Data Access	• E3 Bitstream Breakout interface disabled
Data Buffers	• Burst-/Chunksize set to 64 bytes (only for POS-PHY / UTOPIA-L2X) • Forward Threshold set to 64 bytes (only for POS-PHY / UTOPIA-L2X)
System Interface	• Cell-Level handshake (UTOPIA only) / Packet-Level handshake (POS-PHY only) • Tri-statable bus, allows configuration with multiple PHYs • Parity check enabled

Command ID: **CMD_SET_ATM_T3**

This message sets up the path

System Interface <=> G.804 <=> T3 LIU ,

with the DS3 framer being bypassed.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$LINE\_TIMING_P$	_MASTER_INT	T3 Line internal Master timing (CLKIN)
	_MASTER_EXT	T3 Line external Master timing (TCLKIN)
	_LOOP	T3 Line Loop timing
$SYSIF\_MODE_P$	_UTOPIA_L1	UTOPIA Level 1 mode
	_UTOPIA_L2	UTOPIA Level 2 mode
	_POSPHY_L2	POS-PHY Level 2 mode
	_UTOPIA_L2X	UTOPIA Level 2 eXtended mode
$SYSIF\_WIDTH_P$	_8BIT	8-bit data bus width
	_16BIT	16-bit data bus width
$SYSIF\_ADDR_P$	*16-bit field*	PHY address (0 to 30). Default is 0.
$SYSIF\_STATUS_P$	_POLLING	MPHY Polling
	_DIRECT_IND	Direct Status Indication

Default Settings *(These settings can be modified by Module Configuration Messages)*

ATM Processor	• Payload Scrambling enabled • Header Correction enabled • Direct Mapping (ALPHA = 7, DELTA = 6)
Serial Data Access	• DS3 Bitstream Breakout interface disabled
Data Buffers	• Burst-/Chunksize set to 64 bytes (only for POS-PHY / UTOPIA-L2X) • Forward Threshold set to 64 bytes (only for POS-PHY / UTOPIA-L2X)
System Interface	• Cell-Level handshake (UTOPIA only) / Packet-Level handshake (POS-PHY only) • Tri-statable bus, allows configuration with multiple PHYs • Parity check enabled

Command ID: **CMD_SET_PPP_E3**

This message sets up the data path

System Interface <=> PPP <=> E3 LIU ,

with the E3 framer being bypassed.

Input Parameters (optional)		
Parameter ID	*Value*	*Description*
LINE_TIMING$_P$	_MASTER_INT	E3 Line internal Master timing (CLKIN)
	_MASTER_EXT	E3 Line external Master timing (TCLKIN)
	_LOOP	E3 Line Loop timing
SYSIF_MODE$_P$	_POSPHY_L2	POS-PHY Level 2 mode
	_UTOPIA_L2X	UTOPIA Level 2 eXtended mode
SYSIF_WIDTH$_P$	_8BIT	8-bit data bus width
	_16BIT	16-bit data bus width
SYSIF_ADDR$_P$	*16-bit field*	PHY address (0 to 30). Default is 0.
SYSIF_STATUS$_P$	_POLLING	MPHY Polling
	_DIRECT_IND	Direct Status Indication

Default Settings *(These settings can be modified by Module Configuration Messages)*	
PPP Processor	• 16-bit CRC • Shared Flag between two frames • Address/Control-Field Compression disabled
Data Buffers	• Burst-/Chunksize set to 64 bytes • Forward Threshold set to 64 bytes
System Interface	• Packet-Level handshake (POS-PHY only) • Tri-statable bus, allows configuration with multiple PHYs

Command ID: **CMD_SET_PPP_T3**

This message sets up the data path

System Interface <=> PPP <=> T3 LIU ,

with the DS3 framer being bypassed.

Input Parameters (optional)		
Parameter ID	*Value*	*Description*
LINE_TIMING$_P$	_MASTER_INT	T3 Line internal Master timing (CLKIN)
	_MASTER_EXT	T3 Line external Master timing (TCLKIN)
	_LOOP	T3 Line Loop timing
SYSIF_MODE$_P$	_POSPHY_L2	POS-PHY Level 2 mode
	_UTOPIA_L2X	UTOPIA Level 2 eXtended mode

Command ID:	**CMD_SET_PPP_T3** (cont'd)	
$SYSIF\_WIDTH_P$	_8BIT	8-bit data bus width
	_16BIT	16-bit data bus width
$SYSIF\_ADDR_P$	*16-bit field*	PHY address (0 to 30). Default is 0.
$SYSIF\_STATUS_P$	_POLLING	MPHY Polling
	_DIRECT_IND	Direct Status Indication

Default Settings *(These settings can be modified by Module Configuration Messages)*	
PPP Processor	• 16-bit CRC • Shared Flag between two frames • Address/Control-Field Compression disabled
Data Buffers	• Burst-/Chunksize set to 64 bytes • Forward Threshold set to 64 bytes
System Interface	• Packet-Level handshake (POS-PHY only) • Tri-statable bus, allows configuration with multiple PHYs

Command ID: **CMD_SET_BIT_E3G832_E3**
Command ID: **CMD_SET_BIT_E3G832**

Message CMD_SET_BIT_E3G832_E3 sets up a serial E3 framer + E3 LIU data path *Bitstream Interface <=> E3:G.832 <=> E3 LIU*.
Message CMD_SET_BIT_E3G832 sets this data path without E3 LIU (pure E3 framer): *Bitstream Interface <=> E3:G.832.*

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$LINE\_TIMING_P$	_MASTER_INT	E3 Line internal Master timing (CLKIN)
	_MASTER_EXT	E3 Line external Master timing (TCLKIN)
	_LOOP	E3 Line Loop timing

Default Settings *(These settings can be modified by Module Configuration Messages)*	
E3 G.832 Framer	• SSM pattern is disabled (set to 0000_2). • Payload Type field set to 'non-specific'

Command ID: **CMD_SET_BIT_E3G751_E3**
Command ID: **CMD_SET_BIT_E3G751**

Message CMD_SET_BIT_E3G751_E3 sets up a serial E3 framer + E3 LIU data path *Bitstream Interface <=> E3:G.751 <=> E3 LIU*.
Message CMD_SET_BIT_E3G751 sets this data path without E3 LIU (pure E3 framer): *Bitstream Interface <=> E3:G.751.*

Input Parameters (optional)		
Parameter ID	*Value*	*Description*
$LINE\_TIMING_P$	_MASTER_INT _MASTER_EXT _LOOP	E3 Line internal Master timing (CLKIN) E3 Line external Master timing (TCLKIN) E3 Line Loop timing
$E3F\_PL\_FORMAT_P$	_PPP _ATM	Selects either normal (PPP) framing mode or ATM framing mode.
Default Settings *(These settings can be modified by Module Configuration Messages)*		
E3 G.751 Framer	• Non-ATM Frame Format with C-bits set to '1'.	

Command ID: **CMD_SET_BIT_DS3_T3**
Command ID: **CMD_SET_BIT_DS3**

Message CMD_SET_BIT_DS3_T3 sets up a serial DS3 framer + T3 LIU data path *Bitstream Interface <=> DS3 <=> T3 LIU*.
Message CMD_SET_BIT_DS3 sets this data path without T3 LIU (pure DS3 framer): *Bitstream Interface <=> DS3.*

Input Parameters (optional)		
Parameter ID	*Value*	*Description*
$LINE\_TIMING_P$	_MASTER_INT _MASTER_EXT _LOOP	T3 Line internal Master timing (CLKIN) T3 Line external Master timing (TCLKIN) T3 Line Loop timing
Default Settings *(These settings can be modified by Module Configuration Messages)*		
DS3 Framer	• Auto sensing of application via AIC channel (C-Bit Parity / M23 unchannelized mode) • AIS signal defined as 101010_2... sequence.	

5.3 Interfaces Control Messages

Command ID: **CMD_ENABLE_INTERFACES**

This message controls enabling and disabling of both line interface (digital or analog) and system side interface (UTOPIA, POS-PHY or Bitstream).
Also the bitstream break-out and DS3/E3 framer overhead access interfaces are controlled by this message.
Prior to this command, the data flow and the interfaces must be configured properly.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
IF_SYSTEM_P	_DISABLE	Disable system side interface.
	_ENABLE	Enable system side interface.
IF_LINE_P	_DISABLE	Disable line side interface.
	_ENABLE	Enable line side interface.
$IF\_BREAKOUT_P$	_DISABLE	Disable serial bitstream break-out.
	_ENABLE	Enable serial bitstream break-out between DS3/E3 framer and ATM/HDLC protocol processor.
	This option is not available in data flow configurations with system side interface set to "Bitstream Access".	
$IF\_OVERHEAD_P$	_DISABLE	Disable DS3/E3 framer overhead access interface.
	_ENABLE	Enable DS3/E3 framer overhead access interface.

5.4 Module Configuration Messages

Command ID: **CMD_CFG_LIU**

This message causes the firmware to configure the Line Interface Unit (LIU).

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$LIU\_AMPLIFIER_P$	_NORMAL	Normal amplifier setting.
	_MONITOR	Monitor mode: additional 20 dB gain stage at RL1/RL2 activated.
$LIU\_TRANSMIT_P$	_DISABLE	Transmitter disabled.
	_ENABLE	Transmitter enabled.

Command ID: **CMD_CFG_DJAT**

This message causes the firmware to configure the Digital Jitter Attenuators.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
DJAT_RECEIVE$_P$	_DISABLE	Receive DJAT disabled. Receive PLL can be utilized to generate arbitrary frequency on pin RCLKOUT.
	_ENABLE	Receive DJAT enabled.
DJAT_RX_REF$_P$	_RCLK_PLL	Receive DJAT PLL reference is recovered route clock[1].
	_RSYNC_PLL	Receive DJAT PLL reference is RSYNC.
DJAT_RX_LOSMODE$_P$	_AUTO_REF1	If setting DJAT_RX_REF_RCLK_PLL is selected and LOS is detected, TE3-FALC automatically switches the receive PLL reference clock to pin RSYNC[2].
	_AUTO_REF2	If setting DJAT_RX_REF_RCLK_PLL is selected and LOS is detected, the receive PLL enters the center/holdover mode[3].
	_NON_AUTO	In case LOS is detected, TE3-FALC does not automatically switch the receive PLL reference clock.
DJAT_TRANSMIT$_P$	_DISABLE	Transmit DJAT disabled.
	_ENABLE	Transmit DJAT enabled.
DJAT_TX_LOSMODE$_P$	_AUTO_REF1	If setting LINE_TIMING_LOOP (see Data Flow Configuration Messages) is selected and receiver LOS is detected, TE3-FALC automatically switches the transmit PLL reference clock to pin TCLKIN[4].
	_AUTO_REF2	If setting LINE_TIMING_LOOP is selected and receiver LOS is detected, the transmit PLL enters the center/holdover mode[3].
	_NON_AUTO	In case receiver LOS is detected, TE3-FALC does not automatically switch the transmit PLL reference clock.

1) In digital line interface mode, the clock on pin RCLKI is used as reference instead.

2) If there is a loss of RSYNC clock, the receive PLL automatically enters center/holdover mode[3] in any case.

3) Whether center or holdover mode is entered depends on PLL characteristic selected with CMD_CFG_PLL.

4) If there is a loss of TCLKIN clock, the transmit PLL automatically enters center/holdover mode[3] in any case.

Command ID: **CMD_CFG_DS3_FRAMER**

This message causes the firmware to configure the DS3 framer. The DS3 framer is configured in C-Bit Parity mode by default. If the M23 mode is sensed from the far end, a M23 indication message **NFC_DS3_FRAMER_MODE** (page 70) is sent to the host.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$DS3F\_MODE_P$	_AUTOSENSE	Automatically sense C-Bit Parity mode or M23 mode via the AIC channel
	_CBIT_PARITY	Select C-Bit Parity mode
	_M23	Select M23 unchannelized mode
	_BYPASS	Bypass DS3 framer
	Setting the DS3F_MODE parameter forces the DS3 framer to start a new search for the framing pattern (reframe).	
$DS3F\_AIS\_PATTERN_P$	_TOGGLE	AIS pattern is 101010_2... between overhead bits, C-bits all zeros, X-bits all ones (standard)
	_ALL_ONES	AIS pattern is unframed all ones
$DS3F\_F\_REFRAME_P$	_3_OF_8	New F-frame search when 3 out of 8 contiguous F bits are in error.
	_3_OF_16	New F-frame search when 3 out of 16 contiguous F bits are in error.
$DS3F\_M\_REFRAME_P$	_DISABLE	No new F-frame search due to M-bit errors, only for F-bit errors
	_2_OF_4	New F-frame search if M-bit errors are detected in 2 out of 4 consecutive M-frames.
	_3_OF_4	New F-frame search if M-bit errors are detected in 3 out of 4 consecutive M-frames.

Notification ID: **NFC_DS3_FRAMER_MODE**

This message notificates to the host, that the C-Bit Parity mode (AIC signal is set to 1) or the M23 mode has been detected (AIC signal is random 1s and 0s) on the DS3 interface. It is only generated if the DS3 framer auto sensing mode is selected. Auto sensing is inhibited by TE3-FALC during times where AIS is received.
To enable this notification use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters

Parameter ID	*Value*	*Description*
$DS3F\_MODE_P$	_CBIT_PARITY	Sensed C-Bit Parity mode via AIC signal.
	_M23	Sensed M23 unchannelized mode via AIC signal.

Command ID: **CMD_CFG_E3G832_FRAMER**
Command ID: **CMD_CFG_E3_FRAMER (outdated)**

This message causes the firmware to configure the E3 G.832 framer.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$E3F\_MODE_P$	_G832	Select E3 framing according to ITU-T G.832.
	_BYPASS	Bypass E3 framer.
	Setting the E3F_MODE parameter forces the E3 framer to start a new search for the framing pattern (reframe).	
$E3F\_REFRAME_P$	_3_FAE	The framer gets asynchronous after detecting three consecutive frame alignment errors.
	_4_FAE	The framer gets asynchronous after detecting four consecutive frame alignment errors.
$E3F\_REFRAME\_BIPE_P$	_DISABLE	The only condition for getting asynchronous is as described in parameter E3F_REFRAME.
	_ENABLE	The framer additionally gets asynchronous when 986 out of 1000 frames had BIP errors (ETS300 689).

Command ID: **CMD_CFG_E3G751_FRAMER**

This message causes the firmware to configure the E3 G.751 framer.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$E3F\_MODE_P$	_G751	Select E3 framing according to ITU-T G.751.
	_BYPASS	Bypass E3 framer.
	Setting the E3F_MODE parameter forces the E3 framer to start a new search for the framing pattern (reframe).	
$E3F\_REFRAME_P$	_3_FAE	The framer gets asynchronous after detecting three consecutive frame alignment errors.
	_4_FAE	The framer gets asynchronous after detecting four consecutive frame alignment errors.
$E3F\_OVHD\_FORMAT_P$	_C_ONLY	C-bits are set to 1 in transmit direction (positive justification). J-bits account to payload.
	_C_AND_J	C-bits are set to 0 in transmit direction (no justification). J-bits do not account to payload.
	_NONE	Justification mechanism is disabled. C-bits and J-bits account to payload.
	Parameter E3F_OVHD_FORMAT only applies to non-ATM frame format. The setting applies to both transmit and receive direction. In receive direction the C-bits are not evaluated.	
$E3F\_PL\_FORMAT_P$	_PPP _ATM	Selects either normal (PPP) framing mode or ATM framing mode. *Note: The default value is the previous selected one from the prior issued command flow setup command.* *Note: Only applicable in a dataflow setup with enabled bitstream interface.*

Command ID: **CMD_CFG_HDLC_PROCESSOR**

This message causes the firmware to configure the HDLC/PPP Module.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$HDLC\_MODE_P$	_BIT_HDLC	Bit-synchronous HDLC operation.
	_BYTE_HDLC	Byte-synchronous HDLC operation. Setting only valid for E3 G.832 flows.
	_BIT_PPP	Bit-synchronous PPP operation.
	_BYTE_PPP	Byte-synchronous PPP operation. Setting only valid for E3 G.832 flows.
	_TRANSP	Transparent mode: No flags, no CRC, no bit/byte stuffing.
$HDLC\_CRC\_MODE_P$	_CRC16	Select CRC-16 generation/check.
	_CRC32	Select CRC-32 generation/check.
	_NOCRC	Disable CRC generation/check.
$HDLC\_SHARED\_FLG_P$	_DISABLE	Disable shared flag transmission.
	_ENABLE	Enable shared flag transmission.
$HDLC\_IFTF_P$	_FLAGS	Inter-frame time fill pattern is HDLC flags.
	_ALL_ONES	Inter-frame time fill pattern is continuous ones.
$HDLC\_MAX\_FRMLEN_P$	*16-bit field*	Maximum Frame Length in number of bytes (1 to 8187 bytes). Calculation accounts all bytes between start and end flag excluding the CRC field. When programming to zero, the Maximum Frame Length check is disabled.
$HDLC\_MIN\_FRMLEN_P$	*16-bit field*	Minimum Frame Length in number of bytes (1 to 64 bytes). A frame is considered invalid when the number of bytes received is less than this specified length. Calculation accounts all bytes between start and end flag, excluding CRC. When programming to zero, minimum frame length check is disabled.

Command ID: **CMD_CFG_HDLC_PROCESSOR** (cont'd)

PPP_ACFC$_P$	_DISABLE[1)] _ENABLE	No compression of Address/Control-field. Enable Address- and Control-Field-Compression.
	Only applicable for PPP modes.	
PPP_PFC$_P$	_DISABLE _ENABLE	No compression of Protocol-field. Enable Protocol-Field-Compression.
	Only applicable for PPP modes.	

1) In TE3-FALC V1.1 this option was called "PPP_COMPRESS_NONE" and "PPP_COMPRESS_ADR_CTRL".

Command ID: **CMD_CFG_ATM_PROCESSOR**

This message causes the firmware to configure the G.804 Module.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
ATM_MAPPING$_P$	_DIRECT _PLCP	Direct mapped ATM. PLCP mapped ATM. Setting only available with DS3 flow.
ATM_SCRAMBLING$_P$	_DISABLE _ENABLE	Disable ATM payload scrambling. Enable ATM payload scrambling acc. to I.432.1.
ATM_HEAD_CORR$_P$	_DISABLE _ENABLE	Disable header error correction. Enable single-bit header error correction.
ATM_ALPHA$_P$	*16-bit field*	Determines the transition conditions (number of consecutive incorrect HECs) from SYNCH to HUNT state in the cell delineation state diagram (1 to 31). Default is 7.
ATM_DELTA$_P$	*16-bit field*	Determines the transition conditions (number of consecutive correct HECs) from PRESYNC to SYNCH state in the cell delineation state diagram (1 to 31). Default is 6.

Command ID: **CMD_CFG_SYSIF**

This message causes the firmware to configure the System Interface.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$SYSIF\_MODE_P$	_UTOPIA_L1	UTOPIA Level 1 mode
	_UTOPIA_L2	UTOPIA Level 2 mode
	_POSPHY_L2	POS-PHY Level 2 mode
	_UTOPIA_L2X	UTOPIA Level 2 eXtended mode
$SYSIF\_WIDTH_P$	_8BIT	8 bit data bus width
	_16BIT	16 bit data bus width
$SYSIF\_ADDR_P$	*16-bit field*	PHY address (0 to 30). Default is 0.
	Not applicable for UTOPIA Level 1 mode.	
$SYSIF\_STATUS_P$	_POLLING	MPHY Polling
	_DIRECT_IND	Direct Status Indication
	Not applicable for UTOPIA Level 1 mode.	
$SYSIF\_CHUNK\_SIZE_P$	*16-bit field*	UTOPIA-L2X Chunk Size: The UTOPIA-L2X Chunk Size is programmable between 16 and 128 bytes, in multiples of 4. Default: 64 bytes. POS-PHY-L2 Burst Size: The POS-PHY burst size is programmable between 4 and 256 bytes, in multiples of 4. Default: 64 bytes.
	Not applicable for UTOPIA Level 1 / Level 2 mode.	
$SYSIF\_PARITY\_CHK_P$	_DISABLE	Parity checking across the transmit input data bus is disabled.
	_ENABLE	Parity checking across the transmit input data bus is enabled. Parity errored transmit cells/packets are discarded.

Command ID: **CMD_CFG_SYSIF** (cont'd)

$SYSIF\_PARITY_P$	_ODD	Calculated parity over data bus is Odd.
	_EVEN	Calculated parity over data bus is Even.
$SYSIF\_OUTPUTS_P$	_TRISTATE	Outputs tri-state during idle times. This is the recommended setting.
	_DRIVING	Outputs do not tri-state. This is only applicable for a point-to-point connection.
$SYSIF\_HANDSHAKE_P$	_PKT_CELL	Packet (POS-PHY) / Cell (UTOPIA) Level handshake
	_BYTE	Byte (POS-PHY) / Octet (UTOPIA) Level handshake

Not applicable for UTOPIA Level 2 eXtended mode.

Command ID: **CMD_CFG_BUFFERS**

This message causes the firmware to configure the Data Buffers.
The forward threshold parameter has no relevance for ATM operation.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
BUF_SIZE_P	*16-bit field*	Buffer Size. The buffer size can be programmed in multiples of 64-byte blocks. The lower limit is 2 blocks (128 bytes). The upper limit is 14 blocks (896 bytes).
$BUF\_FW\_THRESH_P$	*16-bit field*	Forward Threshold (not for ATM cells). In order to prevent transmit underrun situations, data is no earlier forwarded to the protocol processor than the forward threshold has been reached, i.e. the amount of data bytes reaches this programmed value (4 to 308 bytes, in multiples of 4). Short packets that do not reach the threshold are forwarded as soon as the packet end is stored in the buffer. Default setting is 64 bytes.

Command ID: **CMD_CFG_GPIO**

This message causes the firmware to configure the General Purpose I/O port pins.

Input Parameters (optional)		
Parameter ID	*Value*	*Description*
$GPIO\_0\_CTRL_P$	_IN _IN_PU _IN_PD _OUT _OUT_OD _LED_STATUS	Input Pin Input Pin with internal Pull-Up Input Pin with internal Pull-Down Output Pin (Push/Pull) Output Pin (Open Drain) Configurable Status Output (Push/Pull). The condition that leads to activation (high level) of this pin is programmable with parameter GPIO_0_LED_STAT.
$GPIO\_1\_CTRL_P$	*as above*	*analog to GPIO Pin 0 above*
...	...	...
$GPIO\_7\_CTRL_P$	*as above*	*analog to GPIO Pin 0 above*
$GPIO\_ALL\_CTRL_P$	*as above*	This option allows to configure all 8 port pins at the same time.
$GPIO\_0\_LED\_STAT_{32}$	*32-bit field*	This 32-bit field is only evaluated if option GPIO_0_CTRL_LED_STATUS from above is selected. For the 32-bit value refer to **ACK_GET_STATUS** (page 100). If a combination of several indications/ failures is entered, the LED status output is the OR function of the respective bits. Default: $0000\ 0800_H$ (Out Of Service)
$GPIO\_1\_LED\_STAT_{32}$	*32-bit field*	Default: $0040\ 0000_H$ (Local Loop)
$GPIO\_2\_LED\_STAT_{32}$	*32-bit field*	Default: $0000\ 0080_H$ (Remote Loop)
$GPIO\_3\_LED\_STAT_{32}$	*32-bit field*	Default: $0002\ 0000_H$ (OOF Defect)
$GPIO\_4\_LED\_STAT_{32}$	*32-bit field*	Default: $0000\ 0001_H$ (No Alarm)
$GPIO\_5\_LED\_STAT_{32}$	*32-bit field*	Default: $0004\ 0000_H$ (RDI Defect)
$GPIO\_6\_LED\_STAT_{32}$	*32-bit field*	Default: $0008\ 0000_H$ (LOS Defect)
$GPIO\_7\_LED\_STAT_{32}$	*32-bit field*	Default: $0001\ 0000_H$ (AIS Defect)

Command ID: **CMD_READ_GPIO**

This message causes the firmware to perform a read access to the General Purpose I/O port pins.

Input Parameters	*none*

Acknowledgement ID: **ACK_READ_GPIO**
Returned input level of the GPIO port after requested by the host via message **CMD_READ_GPIO**.

Output Parameters

Parameter ID	*Value*	*Description*
$GPIO\_VALUE_P$	*16-bit field*	Current input level of the 8 GPIO port pins (including those configured as outputs).

Command ID: **CMD_WRITE_GPIO**

This message causes the firmware to perform a write access to the General Purpose I/O port pins.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$GPIO\_VALUE_P$	*16-bit field*	The lower 8-bit value gets mapped to the 8 port pins. Individual bits can get masked with parameter GPIO_MASK. Default is 00_H.
$GPIO\_MASK_P$	*16-bit field*	This bit mask defines which of the GPIO output ports get overwritten and which ones do not. A '1' at the corresponding bit postion enables overwriting. Default is FF_H.

5.5 Link Maintenance Messages

5.5.1 TTI Support (E3 G.832)

The firmware synchronizes to the Trail Trace Identifier (TTI) as specified in ITU-T G.832 Annex A and checks and generates the CRC-7 checksums. TTI data are not generated or evaluated by the firmware but passed through the user interface. The received TTI is compared against a user-supplied TTI, and deviations are reported. The 16-byte frame format is shown in **Table 16**.

Table 16 TTI Message - 16-Byte Frame Format

	MSB	LSB
Byte 1	1[1)]	CRC-7 (over previous frame) [2)]
Byte 2	0	7-bit Character
Byte 3	0	7-bit Character
	...	...
Byte 16	0	7-bit Character

1) Frame Start Marker; must be set to '1' in transmit direction

2) If in transmit direction automatic CRC-7 generation is enabled, set these 7 bits to '0'.

Command ID: CMD_CFG_TTI_CHANNEL

This message causes the firmware to start receiving TTI data. The TTI transmitter is always active.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$TTI\_RX\_CHNG\_CNT_P$	*16-bit field*	Number of repetitions (3 to 15) of new valid TTI before a change is accepted/ reported. Default is 3.
$TTI\_TX\_DATA_8$	*16 × 8-bit array*	16 bytes of new Transmit TTI Data (includes CRC)
$TTI\_CRC\_GENERATE_P$	_DISABLE _ENABLE	Disable CRC-7 checksum generation. Enable CRC-7 checksum generation. This is valid for both receive and transmit TTI.
$TTI\_RX\_CMP\_DATA_8$	*16 × 8-bit array*	16 bytes of new Receive TTI Data to compare with (includes CRC)

Notification ID: **NFC_TTI_RX_DATA_CHANGE**

This message indicates to the host that TTI data different from the data set by message **CMD_CFG_TTI_CHANNEL** was received.
To enable this notification use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters

Parameter ID	*Value*	*Description*
$TTI\_RX\_STATUS_P$	_MATCH	New received TTI data matches preset receive data, including correct CRC.
	_MISMATCH	New received TTI data does not match preset receive data but has correct CRC.
	_INVALID	Received TTI data is invalid (CRC errors and/or invalid data format).
$TTI\_RX\_DATA_8$	*16 × 8-bit array*	16 bytes of new Receive TTI Data (includes CRC)

5.5.2 Payload Type (PTY) Support (E3 G.832)

Command ID: **CMD_CFG_PTY_CHANNEL**

This message causes the firmware to start evaluating the received PTY field and sets the transmit and receive Payload Type field. The PTY transmitter is always active. Note that the Payload Type field is also set with the data path configuration messages.

Command ID: **CMD_CFG_PTY_CHANNEL** (cont'd)

Input Parameters (optional)

Parameter ID	*Value*	*Description*
PTY_RX_CHNG_CNT$_P$	*16-bit field*	Number of repetitions (3 to 15) of new valid PTY before a change is accepted/ reported. Default is 3.
PTY_TX_FIELD$_P$	_UNEQUIPPED	Set transmit Payload Type field to 'unequipped'.
	_NONSPECIFIC	Set transmit Payload Type field to 'equipped, non-specific'.
	_ATM	Set transmit Payload Type field to 'ATM'.
	_SDH_TU12	Set transmit Payload Type field to 'SDH TU 12s'.
PTY_RX_CMP_FIELD$_P$	_UNEQUIPPED	Set expected receive Payload Type field to 'unequipped'.
	_NONSPECIFIC	Set expected receive Payload Type field to 'equipped, non-specific'.
	_ATM	Set expected receive Payload Type field to 'ATM'.
	_SDH_TU12	Set expected receive Payload Type field to 'SDH TU 12s'.

Notification ID: **NFC_PTY_RX_DATA_CHANGE**

This message indicates to the host that a PTY field different from the field set by message **CMD_CFG_PTY_CHANNEL** was received.

Output Parameters

Parameter ID	*Value*	*Description*
PTY_RX_STATUS$_P$	_MATCH	New received PTY data matches preset receive data.
	_MISMATCH	New received PTY data does not match preset receive data.
	_UNSTABLE	Received PTY data is unstable.
PTY_RX_FIELD$_P$	_UNEQUIPPED	Received Payload Type field is 'unequipped'.
	_NONSPECIFIC	Received Payload Type field is 'equipped, non-specific'.
	_ATM	Received Payload Type field is 'ATM'.
	_SDH_TU12	Received Payload Type field is 'SDH TU 12s'.

5.5.3 SSM Nibble Support (E3 G.832)

Command ID: **CMD_CFG_SSM_CHANNEL**

This message causes the firmware to start receiving SSM nibble data. The SSM transmitter is always active.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$SSM\_RX\_CHNG\_CNT_P$	*16-bit field*	Sets the number of repetitions of new valid SSM before a change is accepted/ reported.
$SSM\_TX\_DATA_P$	*16-bit field*	Transmit SSM nibble data (4-bits)
$SSM\_RX\_CMP\_DATA_P$	*16-bit field*	Expected receive SSM nibble data (4-bits)

Notification ID: **NFC_SSM_RX_DATA_CHANGE**

This message indicates to the host that SSM data different from the data set by message **CMD_CFG_SSM_CHANNEL** was received.
To enable this notification use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters

Parameter ID	*Value*	*Description*
$SSM\_RX\_DATA_P$	*16-bit field*	Receive SSM nibble data (4-bits)
$SSM\_RX\_STATUS_P$	_MATCH	New received SSM data matches preset receive data.
	_MISMATCH	New received SSM data does not match preset receive data.
	_UNSTABLE	Received SSM data is unstable.

5.5.4 MDL Support (DS3)

Three bits C-Bits per multiframe (DL_t bits) are used for the MDL channel (Path Maintenance Data Link). An HDLC controller processes these bits, checks and generates CRC16 checksums, adds and detects HDLC frame delimiting flags etc.

The HDLC idle pattern when no data are transmitted is the HDLC flag (01111110).

Received and transmitted frames are not generated or evaluated by the firmware but passed through the user interface.

Command ID: **CMD_CFG_MDL_CHANNEL**

This message causes the firmware to configure the MDL controller.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$MDL\_RECEIVE_P$	_DISABLE _ENABLE	Disable MDL Receive Channel. Enable MDL Receive Channel. Make sure that the **NFC_MDL_RX_DATA** (page 82) notification gets enabled too.
$MDL\_TRANSMIT_P$	_DISABLE _ENABLE	Disable MDL Transmit Channel. Enable MDL Transmit Channel.
$MDL\_MAX\_FRM\_LEN_P$	*16-bit field*	Maximum Frame Length (1 to 256 octets)
$MDL\_RX\_ERR\_RPT_P$	_DISABLE	Errored MDL data gets discarded without notification to the user.
	_ENABLE	The notification **NFC_MDL_RX_DATA** (page 82) gets sent for both valid and errored MDL data.

Command ID: **CMD_SET_MDL_TX_DATA**

This message causes the firmware to transmit MDL data.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$MDL\_TX\_DATA_8$	*$n \times$ 8-bit array ($1 \leq n \leq 256$)*	Transmit MDL data (maximum length as specified via message **CMD_CFG_MDL_CHANNEL**)

Notification ID: **NFC_MDL_RX_DATA**

This message indicates the host about new receive MDL data. Along with the MDL data, the parameter MDL_RX_STATUS indicates whether the received HDLC frame is valid or not.
To enable this notification use message **CMD_SET_NFC_ALM_MODE** (page 45).

Notification ID: **NFC_MDL_RX_DATA** (cont'd)

Output Parameters

Parameter ID	*Value*	*Description*
$MDL\_RX\_DATA_8$	*n × 8-bit array (1 ≤ n ≤ 256)*	Receive MDL data (maximum length as specified via message **CMD_CFG_MDL_CHANNEL**)
$MDL\_RX\_STATUS_P$	_VALID_FRAME	Valid Frame
	_CRC_ERR	CRC error
	_ABORT_ERR	Aborted frame
	_MAXFL_ERR	Frame exceeded maximum length
	_MINFL_ERR	Frame too short
	_ILEN_ERR	Frame length not multiple of 8 bits

5.5.5 FEAC Support (DS3)

The firmware synchronizes to the 8-ones bit pattern in the FEAC codes delivered by the hardware. **Table 17** and **Table 18** list the FEAC codes with predefined parameter values in the TE3-FALC firmware, according to ANSI T1.107-1995 (Figures 23 and 24). Any other FEAC codeword (like unassigned codes listed in ANSI T1.107-1995 Table 25) can be generated by assembling the user defined parameter value as described in message **CMD_SET_FEAC_CODE** (page 84).

The rightmost bit of each codeword is transmitted first. Specific codewords are transmitted continuously for the duration of the alarm or status condition, or for a minimum of n codeword repetitions (n is programmable), whichever is longer. When no alarm or status condition is being transmitted, the FEAC code contains all ones.

Table 17 FEAC Alarm and Status Codes

Alarm / Status Codes	ID	Code Word	(Hex)	Prio Code
DS3 equiment failure (SA)	_DS3_SA	0 011001 0 11111111	32_H	38_H
DS3 Loss of Signal (LOS)	_DS3_LOS	0 001110 0 11111111	$1C_H$	30_H
DS3 Out-of-Frame	_DS3_OOF	0 000000 0 11111111	00_H	28_H
DS3 AIS received	_DS3_AIS	0 010110 0 11111111	$2C_H$	20_H
DS3 IDLE received	_DS3_IDLE	0 011010 0 11111111	34_H	18_H
DS3 equiment failure (NSA)	_DS3_NSA	0 001111 0 11111111	$1E_H$	10_H
Common equipment failure (NSA)	_COM_NSA	0 011101 0 11111111	$3A_H$	08_H
No FEAC Code	_NONE	1 111111 1 11111111	FF_H	00_H

The DS3 line loopback codes consist of a sequence of two FEAC codewords (see **Table 18**). These codewords are sent sequentially, each with the programmed repetition count (inifinite repetition count is not allowed). This ensures that no gap occurs between the 2 codewords.

Table 18 FEAC Loopback Control Codes

Loopback Control Codes	ID	Code Word	(Hex)	Prio Code
DS3 Line loopback activate	_DS3_LP_ON[1)]	0 011011 0 11111111	36_H	40_H
		0 000111 0 11111111	$0E_H$	40_H
DS3 Line loopback deactivate	_DS3_LP_OFF[2)]	0 011011 0 11111111	36_H	40_H
		0 011100 0 11111111	38_H	40_H

1) The corresponding 16-bit parameter value of this code sequence is $8F40_H$

2) The corresponding 16-bit parameter value of this code sequence is $B940_H$

Command ID: CMD_SET_FEAC_CODE

This message causes the firmware to transmit the FEAC code word followed by the DS3 line code word acc. to ANSI T1.107-1995.

Command ID: **CMD_SET_FEAC_CODE** (cont'd)

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$FEAC\_CODEWORD_P$	_DS3_SA	DS3 equiment failure (Service Affecting)
	_DS3_LOS	DS3 Loss of Signal LOS
	_DS3_OOF	DS3 Out-of-Frame
	_DS3_AIS	DS3 AIS received
	_DS3_IDLE	DS3 IDLE Signal received
	_DS3_NSA	DS3 equiment failure (Non-Service Affecting)
	_COM_NSA	Common equipment failure (Non-Service Affecting)
	_DS3_LP_ON	DS3 Line Loopback activate
	_DS3_LP_OFF	DS3 Line Loopback deactivate
	_NONE	No valid FEAC code to be transmitted
	other 16-bit value	High byte is the actual FEAC code (excluding FF_H), low byte defines priority of this code. Refer to **Table 17** and **Table 18** for the predefined code words with their priority code.
$FEAC\_CODE\_CNT_P$	*16-bit field*	Repetition Counter for FEAC code word (10 to 65535). Progamming to zero leads to infinite repetition count. Default is 10.

Notification ID: **NFC_FEAC_LOOP_STATUS**

This message indicates to the host, that the loop status changed, indicated via FEAC channel.
To enable this notification use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters

Parameter ID	*Value*	*Description*
$FEAC\_DS3\_LOOP_P$	_INACTIVE	DS3 Line loopback is inactive.
	_ACTIVE	DS3 Line loopback is active.

Command ID: **CMD_GET_FEAC_CODE**

This command returns the currently received FEAC code..

Input Parameters	*none*

Acknowledgement ID: **ACK_GET_FEAC_CODE**

This message is sent to the user to provide the currently received FEAC code requested by the **CMD_GET_FEAC_CODE** message.

Output Parameters

Parameter ID	*Value*	*Description*
$FEAC\_CODEWORD_P$	*16-bit field*	FEAC-Code (8-bit) Priority (8-bit, always FF_H). Example: $xxFF_H$ with FEAC code value xx_H

5.6 Alarm Messages

5.6.1 Framer LOS Detection (Loss of Signal)

Alarm ID: **ALM_FRM_LOS_DEFECT**

An LOS defect occurs when the TE3-FALC does not detect the expected signal type on the line.
In DS3 mode, LOS defects are integrated to generate an LOS failure after 2.5 sec.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters

Parameter ID	*Value*	*Description*
$FRM\_LOS\_DEFECT_P$	_DECLARED _CLEARED	LOS defect has been declared LOS defect has been cleared

Firmware Actions	
Actions when declared	– Declare RDI to far end (if enabled) – Increment LOS counter – Activate LOS LED (if enabled) – Start integration for LOS Failure (DS3 only)
Actions when cleared	– Clear RDI to far end – Deactivate LOS LED (if enabled) – Stop integration for LOS Failure (DS3 only)

Alarm ID: **ALM_FRM_LOS_FAILURE**

This alarm indicates both declaring and clearing a Loss of Signal (LOS) failure. It is only supported in DS3 mode.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters		
Parameter ID	*Value*	*Description*
$FRM\_LOS\_FAILURE_P$	_DECLARED	LOS failure has been declared since LOS defect persisted for 2.5 sec.
	_CLEARED	LOS failure has been cleared since the last 10.0 sec had no LOS defects.
Firmware Actions		
Actions when declared	– Start transmission of the FEAC code 'DS3 LOS' to the far end (if enabled), unless a higher priority FEAC code needs to be sent	
Actions when cleared	– Stop transmission of the FEAC code 'DS3 LOS'	

5.6.2 Framer OOF/LOF Detection (Out of Frame, Loss of Frame)

Alarm ID: **ALM_FRM_OOF_DEFECT**

This alarm indicates both declaring and clearing a Out-of-Frame (OOF) defect. An OOF defect is indicated by the framer if several F- and M-bit errors are detected. In DS3 mode, OOF defects are integrated to generate an LOF failure after 2.5 sec.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters		
Parameter ID	*Value*	*Description*
$FRM\_OOF\_DEFECT_P$	_DECLARED	OOF defect has been declared.
	_CLEARED	OOF defect has been cleared.
Firmware Actions		
Actions when declared	– Declare RDI to far end (if enabled) – Incremet OOF counter – Activate OOF LED (if enabled) – Start integration for LOF Failure (DS3 only)	
Actions when cleared	– Clear RDI to far end – Deactivate OOF LED (if enabled) – Stop integration for LOF Failure (DS3 only)	

Alarm ID: **ALM_FRM_LOF_FAILURE**

This alarm indicates both declaring and clearing a Loss of Frame (LOF) failure. It is only supported in DS3 mode.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters		
Parameter ID	*Value*	*Description*
$FRM\_LOF\_FAILURE_P$	_DECLARED	LOF failure has been declared since OOF defect persisted for 2.5 sec.
	_CLEARED	LOF failure has been cleared since the last 10.0 sec had no OOF defects.
Firmware Actions		
Actions when declared	– Start transmission of the FEAC code 'DS3 out-of-frame' to the far end (if enabled), unless a higher priority FEAC code needs to be sent	
Actions when cleared	– Stop transmission of the FEAC code 'DS3 out-of-frame'	

5.6.3 Framer AIS/IDLE Detection (Alarm Indication, DS3 Idle Signal)

Alarm ID: **ALM_FRM_AIS_DEFECT**

An AIS defect is declared when the AIS pattern is received from the far end. In DS3 mode, AIS defects are monitored to generate an AIS failure after 2.5 sec.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters		
Parameter ID	*Value*	*Description*
$FRM\_AIS\_DEFECT_P$	_DECLARED	AIS defect has been declared.
	_CLEARED	AIS defect has been cleared.
Firmware Actions		
Actions when declared	– Declare RDI to far end (if enabled) – Activate AIS LED (if enabled) – Start integration for AIS Failure (DS3 only)	
Actions when cleared	– Clear RDI to far end – Deactivate AIS LED (if enabled) – Stop integration for AIS Failure (DS3 only)	

Alarm ID: **ALM_FRM_AIS_FAILURE**

This alarm indicates both declaring and clearing an AIS failure. It is only supported in DS3 mode.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters		
Parameter ID	*Value*	*Description*
$FRM\_AIS\_FAILURE_P$	_DECLARED	AIS failure has been declared since AIS defect persisted for 2.5 sec.
	_CLEARED	AIS failure has been cleared since the last 10.0 sec had no AIS defects.

Firmware Actions	
Actions when declared	– Start transmission of the FEAC code 'DS3 AIS Received' to the far end (if enabled), unless a higher priority FEAC code needs to be sent
Actions when cleared	– Stop transmission of the FEAC code 'DS3 AIS Received'

Alarm ID: **ALM_FRM_IDLE_RECEIVED**

This alarm is generated if the DS3 Idle Signal is detected on the receive line. The Idle Signal is defined as a 1100_2... inside the DS3 payload. This message is supported in DS3 mode only.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters		
Parameter ID	*Value*	*Description*
$FRM\_IDLE\_RCVD_P$	_DECLARED	DS3 Idle is being received.
	_CLEARED	No DS3 Idle is received.

Firmware Actions	
Actions when declared	– Start transmission of the FEAC code 'DS3 IDLE Received' to the far end (if enabled), unless a higher priority FEAC code needs to be sent
Actions when cleared	– Stop transmission of the FEAC code 'DS3 IDLE Received'

5.6.4 Framer AIS/IDLE Generation (Alarm Indication, DS3 Idle Signal)

Command ID: **CMD_FRM_TX_AIS_IDLE**

With this message the transmit signal can be set to AIS or Idle (DS3 only). In DS3 mode, the AIS signal definition can be programmed with message **CMD_CFG_DS3_FRAMER** (page 69).

Input Parameters (optional)		
Parameter ID	*Value*	*Description*
$FRM\_TX\_SIGNAL_P$	_NORMAL	Transmit Normal Data
	_AIS	Transmit AIS Signal
	_IDLE	Transmit DS3 Idle Signal (DS3 only)

5.6.5 Framer RDI Detection (Remote Defect Indication)

Older DS3 standards (esp. ANSI) referred to this signal element as a “Yellow Alarm” or RAI signal. In newer standards, the usage does not correspond to a typical Yellow Alarm or RAI signal, which is only sent after a multi-second soaking period. The more accurate RDI terminology has been adopted in ANSI standards. Note that the defect that is detected upon receipt of this signal is sometimes referred to as a “far-end SEF/AIS defect".

Alarm ID: **ALM_FRM_RDI_DEFECT**

In DS3 mode, an RDI defect (also known as “far-end SEF/AIS defect") is declared when both X-bits in an M-frame are received to zero. It is cleared when both X-bits are received to one.
In E3 mode a dedicated RDI bit exists in the MA byte.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters		
Parameter ID	*Value*	*Description*
$FRM\_RDI\_DEFECT_P$	_DECLARED	RDI defect has been declared.
	_CLEARED	RDI defect has been cleared.
Firmware Actions		
Actions when declared	– Increment RDI counter – Activate RDI LED (if enabled) – Start integration for RDI Failure (DS3 only)	
Actions when cleared	– Deactivate RDI LED (if enabled) – Stop integration for RDI Failure (DS3 only)	

Alarm ID: **ALM_FRM_RDI_FAILURE**

This alarm indicates both declaring and clearing an Remote Defect Indication (RDI) failure. It is only supported in DS3 mode.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters		
Parameter ID	*Value*	*Description*
$FRM\_RDI\_FAILURE_P$	_DECLARED	RDI failure has been declared since RDI defects persisted for 2.5 sec.
	_CLEARED	RDI failure has been cleared since the last 10.0 sec had no RDI defects.
Firmware Actions		
Actions when declared	–	
Actions when cleared	–	

5.6.6 Framer RDI Generation (Remote Defect Indication)

Command ID: **CMD_CFG_RDI_DECLARATION**

This message configures the criteria to declare "Remote Defect Indication" (RDI) towards the far end. By default, RDI is only declared on LOS, OOF and AIS defects. In an E3 configration, also the G.804 "Loss of Cell Delineation" (LCD) event is taken into account.
To be compliant to the 'old' ANSI interpretation of the DS3 "Yellow Alarm" (RAI) set the parameter RDI_ON_LOF_FAIL_ENABLE.

Input Parameters (optional)		
Parameter ID	*Value*	*Description*
$RDI\_ON\_LOS\_DEFCT_P$	_DISABLE	Do not declare RDI on LOS Defect
	_ENABLE	Declare RDI on LOS Defect
$RDI\_ON\_OOF\_DEFCT_P$	_DISABLE	Do not declare RDI on OOF Defect
	_ENABLE	Declare RDI on OOF Defect
$RDI\_ON\_LOF\_FAIL_P$	_DISABLE	Do not declare RDI on LOF Failure
	_ENABLE	Declare RDI on LOF Failure (DS3 only)
$RDI\_ON\_AIS\_DEFCT_P$	_DISABLE	Do not declare RDI on AIS Defect
	_ENABLE	Declare RDI on AIS Defect
$RDI\_ON\_LCD_P$	_DISABLE	Do not declare RDI on G.804 LCD event
	_ENABLE	Declare RDI on G.804 LCD event (E3 only)

Command ID: **CMD_CFG_RDI_DECLARATION** (cont'd)

$RDI\_ON\_TIM_P$	_DISABLE	Do not declare RDI on G.832 Trail Trace Identifier Mismatch
	_ENABLE	Declare RDI on G.832 Trail Trace Identifier Mismatch (E3 only)
$RDI\_ON\_UNEQ_P$	_DISABLE	Do not declare RDI on receiving G.832 PTY field 'unequipped'
	_ENABLE	Declare RDI on receiving G.832 PTY field 'unequipped' (E3 only). In case of LOF no RDI is transmitted
$RDI\_ON\_PLM_P$	_DISABLE	Do not declare RDI on G.832 PTY field mismatch
	_ENABLE	Declare RDI on G.832 PTY field mismatch (E3 only). In case of LOF a RDI is transmitted sins no PTY match can be achieved

Command ID: **CMD_FRM_TX_RDI**

With this message the transmit RDI signal can be set manually, in addition to the automatic RDI declaration. If the RDI signal shall be fully controlled manually, the automatic RDI declaration should be disabled via the **CMD_CFG_RDI_DECLARATION** (page 91) message.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$FRM\_TX\_RDI_P$	_INACTIVE	RDI is only transmitted if one or more of the conditions listed in message **CMD_CFG_RDI_DECLARATION** (page 91) is active and enabled (automatic RDI declaration only).
	_ACTIVE	RDI is declared. This setting overrides the automatic RDI declaration.

5.6.7 Framer FEAC Detection (Far End Alarm Control)

Alarm ID: **ALM_FEAC_ALARM**

This message indicates to the host, that a FEAC alarm has occured.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters

Parameter ID	*Value*	*Description*
$FEAC\_ALARM_P$	_DS3_SA	DS3 equiment failure (Service Affecting)
	_DS3_LOS	DS3 Loss of Signal LOS
	_DS3_OOF	DS3 Out-of-Frame
	_DS3_AIS	DS3 AIS received
	_DS3_IDLE	DS3 IDLE Signal received
	_DS3_NSA	DS3 equiment failure (Non-Service Affecting)
	_COM_NSA	Common equipment failure (Non-Service Affecting)
	_NONE	No valid FEAC code is being received

5.6.8 Framer FEAC Generation (Far End Alarm Control)

Command ID: **CMD_CFG_FEAC_GENERATION**

This message controls the automatic generation of FEAC codes towards the far end. By default all FEAC codes are disabled, i.e. the FEAC channel transmits all ones.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$FEAC\_ON\_LOS\_FAIL_P$	_DISABLE	Do not generate FEAC code on LOS Failure
	_ENABLE	Generate FEAC code on LOS Failure
$FEAC\_ON\_LOF\_FAIL_P$	_DISABLE	Do not generate FEAC code on LOF Failure
	_ENABLE	Generate FEAC code on LOF Failure
$FEAC\_ON\_AIS\_FAIL_P$	_DISABLE	Do not generate FEAC code on AIS Failure
	_ENABLE	Generate FEAC code on AIS Failure
$FEAC\_ON\_IDLE\_RCV_P$	_DISABLE	Do not generate FEAC code on reception of IDLE Pattern
	_ENABLE	Generate FEAC code on reception of IDLE Pattern

5.6.9 PLCP OOF Detection (Out-of-Frame)

Alarm ID: **ALM_PLCP_OOF_DEFECT**

This alarm indicates both declaring and clearing a PLCP Out-of-Frame (OOF) defect.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters		
Parameter ID	*Value*	*Description*
$PLCP\_OOF\_DEFECT_P$	_DECLARED _CLEARED	PLCP-OOF defect has been declared. PLCP-OOF defect has been cleared.
Firmware Actions		
Actions when declared	– Declare PLCP-RAI to far end – Increment PLCP-OOF counter	
Actions when cleared	– Clear PLCP-RAI to far end	

5.6.10 PLCP RAI Detection (Remote Alarm Indication)

Alarm ID: **ALM_PLCP_RAI_DEFECT**

This alarm indicates both declaring and clearing a receive PLCP Remote Alarm Indication (RAI) defect.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters		
Parameter ID	*Value*	*Description*
$PLCP\_RAI\_DEFECT_P$	_DECLARED _CLEARED	RDI defect has been declared. RDI defect has been cleared.
Firmware Actions		
Actions when declared	– Increment PLCP-RAI counter	
Actions when cleared	–	

5.6.11 G.804 LCD Detection (Loss of Cell Delineation)

Alarm ID: **ALM_G804_LCD**

This message indicates to the host, that the G.804 receiver has lost or gained cell delineation synchronization.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters		
Parameter ID	*Value*	*Description*
$G804\_LCD_P$	_DECLARED	Loss of cell delineation has been declared.
	_CLEARED	Loss of cell delineation has been cleared.
Firmware Actions		
Actions when declared	– Declare RDI to far end (if enabled; E3 only) – Increment LCD counter	
Actions when cleared	– Clear RDI to far end (E3 only)	

5.6.12 Buffer XDU/RDO Detection (Tx Underun, Rx Overflow)

Alarm ID: **ALM_TX_UNDERRUN**

This alarm indicates a transmit data underrun (XDU) event in the PPP/HDLC packet processor. It is not applicable in ATM operation.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters	*none*

Alarm ID: **ALM_RX_OVERFLOW**

This alarm indicates a receive data overflow (RDO) event in the receive buffer.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

*Note: The overflow alarm indication is output max. about 20 times per second in order to reduce the load of the message queue. Every occurance of overflow condition is properly counted in the **ATM TC-Layer Performance Counters** (page 104).*

Output Parameters	*none*

5.6.13 System Interface Parity Defect

Alarm ID: **ALM_SYSIF_PARITY_DEFECT**

This alarm indicates a parity defect in transmit direction at Utopia/POS-PHY system interface.
To enable this alarm use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters	*none*

5.7 Performance Measurement

5.7.1 Bit Error Ratio Testing (BERT)

Command ID: **CMD_CTRL_BERT**

This message configures the bit error ratio test unit with a programmable bit pattern. Note that the framer gets reset with each start of a BER measurement.

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$BERT\_CONTROL_P$	_START	Start bit error ratio measurement with programmed measurement interval. If an interval > 0 is programmed, the BERT result is reported via message **NFC_BERT_RESULT** (page 98).
	_STOP	Stop current bit error ratio measurement. The BERT results can be determined with message **CMD_GET_STATISTICS** (page 105).
$BERT\_PATTERN_P$	_2P15	2^{15} -1 pseudo-random pattern (O.151)
	_2P20	2^{20} -1 pseudo-random pattern (O.153)
	_2P20_QRSS	2^{20} -1 pseudo-random pattern with zero suppression (O.151)
	_2P23	2^{23} -1 pseudo-random pattern (O.151)
	_ALL_ONES	All Ones fixed pattern
	_ALL_ZEROS	All Zeros fixed pattern
	_TOGGLE	Toggling 101010_2... fixed pattern
	_GEN_RAND	Generic pseudo-random test sequence built based on BERT_GEN_LENGTH and BERT_GEN_FBTAP.
	_GEN_FIXED	Generic fixed pattern test sequence built based on BERT_GEN_LENGTH and BERT_GEN_FIXPAT.
$BERT\_INTERVAL_P$	*16-bit field*	Defines a bit error ratio measurement interval from 1 to 1440 minutes (24h). Programming to zero leads to infinite measurement interval (to be stopped manually with BERT_CONTROL_STOP).

Note: The parameters listed below are only relevant if a generic pattern shall be built.

Command ID: **CMD_CTRL_BERT** (cont'd)

$BERT\_GEN\_LENGTH_P$	*16-bit field*	For a *pseudo-random* pattern, the test sequence length is $2^{BERT\_GEN\_LENGTH} - 1$. For a *fixed* pattern, this field defines the length (1..32 bit) of the test sequence. Default is 32.
$BERT\_GEN\_FBTAP_P$	*16-bit field*	Defines the position of the feedback tap (1..32) for a *pseudo-random* pattern. Default is 16.
$BERT\_GEN\_FIXPAT_{32}$	*32-bit field*	For a generic *fixed* pattern, this field, together with BERT_GEN_LENGTH defines the pattern to be generated. Default is 0xAAAAAAAA.
$BERT\_GEN\_INVERT_P$	_DISABLE _ENABLE	No pattern inversion. Pattern generator output gets inverted.
$BERT\_GEN\_TXER_P$	_NONE _3 _4 _5 _6 _7	Selects the bit error rate inserted into the datastream generated by the BERT. Either no bit errors or the rate of the selected power of the tenth of bit errors is inserted. The parameters _x configure a bit error rate of 10^{-x}.

Notification ID: **NFC_BERT_RESULT**

To enable this notification use message **CMD_SET_NFC_ALM_MODE** (page 45).

Output Parameters

Parameter ID	*Value*	*Description*
$RPT\_BERT\_TOTAL_{32}$	*2 × 32-bit array*	The first 32-bit word contains the Receive Bit Count. The second 32-bit word contains the Receive Error Count (only incremented in synchronous state).

Command ID: **CMD_CTRL_BERT_ERROR**

This message causes the firmware to perform the selected bit error insertion in the BERT block of the device

Input Parameters (optional)

Parameter ID	*Value*	*Description*
$BERT\_GEN\_TXER_P$	_NONE _3 _4 _5 _6 _7 _SINGLE	Selects the bit error rate inserted into the datastream generated by the BERT. Either no bit errors, single bit errors or the rate of the selected power of the tenth of bit errors is inserted. The parameters _x configure a bit error rate of 10^{-x}. *Note: The selected bit error rate is not changed upon transmission of single bit error.*

5.7.2 Status Messages

Command ID: **CMD_GET_STATUS**

This message initiates the firmware to send the device status in form of a 32-bit status word.

Input Parameters	*none*

Acknowledgement ID: **ACK_GET_STATUS**

The device status is coded as a 32-bit word with the bit assignment shown below. If multiple failures/indications are active simultaneously, the resulting status word is represented as the sum of the individual status words.

Output Parameters

Parameter ID	*Value*	*Description*
$DEVICE\_STATUS_{32}$	$0000\ 0001_H$	No Alarm
	$0000\ 0002_H$	Receiving DS3/E3 RDI Failure
	$0000\ 0004_H$	Transmitting DS3/E3 RDI
	$0000\ 0008_H$	Receiving DS3/E3 AIS Failure
	$0000\ 0010_H$	Transmitting DS3/E3 AIS
	$0000\ 0020_H$	Receiving DS3/E3 LOF Failure
	$0000\ 0040_H$	Receiving DS3/E3 LOS Failure
	$0000\ 0080_H$	Remote Loop active
	$0000\ 0100_H$	Recognizing Test Pattern (BERT)
	$0000\ 0200_H$	Other Failure
	$0000\ 0400_H$	Unavailable State (UAS)
	$0000\ 0800_H$	Out Of Service (OOS)
	$0000\ 4000_H$	Unavailable State Far-End (UAS-FE)
	$0000\ 8000_H$	DS3 Framer in M23 Mode (0: C-bit parity)

Command ID: **CMD_GET_STATUS** (cont'd)

0001 0000$_H$	Receiving DS3/E3 AIS Defect
0002 0000$_H$	Receiving DS3/E3 OOF Defect
0004 0000$_H$	Receiving DS3/E3 RDI Defect
0008 0000$_H$	Receiving DS3/E3 LOS Defect
0010 0000$_H$	Receiving IDLE Pattern
0020 0000$_H$	Transmitting IDLE Pattern
0040 0000$_H$	Local Loop active
0080 0000$_H$	Transmitting LOS (No Output Signal)
0100 0000$_H$	Receiving E3 G.832 Trail Trace Mismatch
0200 0000$_H$	Receiving E3 G.832 PTY "Unequipped"
0400 0000$_H$	Receiving E3 G.832 PTY Mismatch
0800 0000$_H$	Receiving E3 G.832 SSM Mismatch
1000 0000$_H$	Receive G.804 lost cell delineation (LCD)
2000 0000$_H$	Receiving PLCP OOF Defect
4000 0000$_H$	Receiving PLCP Remote Alarm (RAI)
8000 0000$_H$	Transmitting PLCP Remote Alarm (RAI)

Note: The Out of Service (OOS) status reflects the status of the device system interface (enable/disable).

5.7.3 Statistics Messages

To support performance measurement statistics, several performance counters are realized in TE3-FALC. These counters can be requested anytime with a single message **CMD_GET_STATISTICS** (page 105). It is possible to provide the total count since the last clearing of counters ("TOTAL") and/or the count of the last completed 1-second ("1SEC") interval. **Figure 5** illustrates the accumulation of counts over several seconds and the Read & Clear operation on the total count. The value of the 1-second count is updated periodically every second.

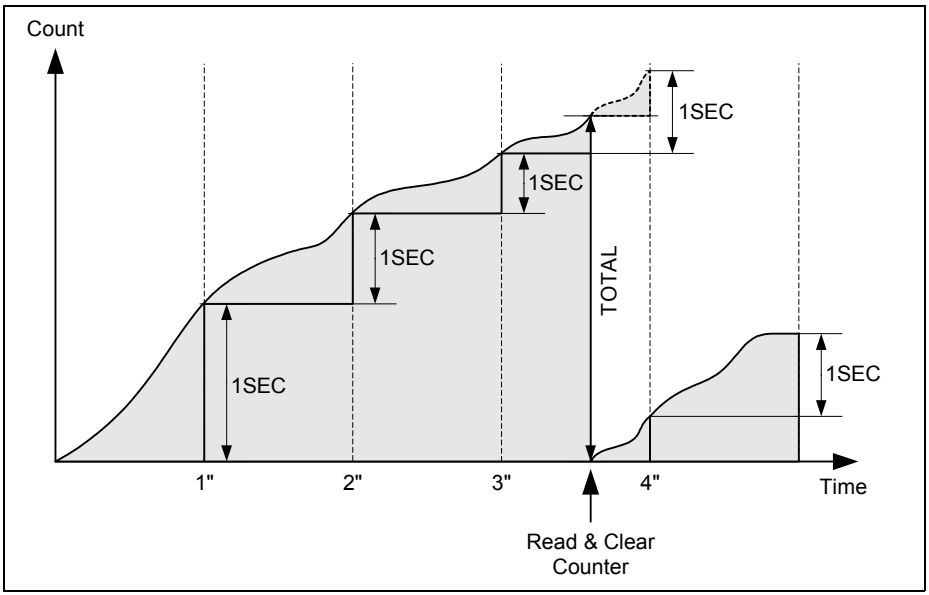


Figure 5 Statistics Counter Operation

The implemented performance counters are listed in **Table 19**, **Table 20**, **Table 21**, **Table 22** and **Table 23**. Reporting of the counter values is performed in the order they are listed.

Table 19 DS3/E3 Framer Performance Counters

Word	Name	Near-/ Far-end	Counter Value	Counter Type
0	PES	- NE	DS3/E3: P-bit Errored Seconds	1-sec interval
1	PSES	- NE	DS3/E3: P-bit Severely Errored Seconds	1-sec interval
2	SEFS	- NE	DS3/E3: Severely Errored Frame Seconds	1-sec interval
3	UAS	- NE	DS3/E3: Unavailable Seconds	1-sec interval

Table 19 DS3/E3 Framer Performance Counters (cont'd)

Word	Name	Near-/ Far-end	Counter Value	Counter Type
4	LCV	- NE	DS3/E3: Line Coding Violations	Hardware
5	PCV	- NE	DS3: P-bit Coding Violations E3 G832: BIP-8 errors	Hardware
6	LES	- NE	DS3/E3: Line Errored Seconds	1-sec interval
7	CCV	- NE	DS3: C-bit Coding Violations	Hardware
8	CES	- NE	DS3: C-bit Errored Seconds	1-sec interval
9	CSES	- NE	DS3: C-bit Severely Errored Seconds	1-sec interval
10	FAE	- NE	DS3/E3: Frame Alignment errors	Hardware
11	OOF	- NE	DS3/E3: OOF defects	Software
12	LOS	- NE	DS3/E3: LOS defects	Software
13	BCV	- NE	DS3/E3: Bipolar Coding Violations	Hardware
14	EXZ	- NE	DS3/E3: Excessive Zeros	Hardware
15	CCV	- FE	DS3: Far end block errors E3 G832: Remote error indications	Hardware
16	CES	- FE	DS3: C-bit Errored Seconds	1-sec interval
17	CSES	- FE	DS3: C-bit Severely Errored Seconds	1-sec interval
18	UAS	- FE	DS3/ Unavailable Seconds E3 G832:	1-sec interval
19	RDI	- FE	DS3/E3: Received RDI errors	Software

Table 20 PLCP Performance Counters

Word	Name	Near-/ Far-end	Counter Value	Counter Type
0	PLCP_SEFS	- NE	PLCP Severely Errored Frame Seconds	1-sec interval
1	PLCP_UAS	- NE	✧PLCP Unavailable Seconds[1)]	1-sec interval
2	PLCP_FAE	- NE	PLCP Frame Alignment errors	Hardware
3	PLCP_BIPE	- NE	PCLP BIP-8 errors	Hardware
4	PLCP_OOF	- NE	PLCP OOF defects	Software
5	PLCP_FEBE	- FE	PLCP Far end block errors (FEBE)	Hardware
6	PLCP_RAI	- FE	PLCP RAI errors	Software

1) This is an option which is not available with the current firmware release.

Table 21 ATM TC-Layer Performance Counters

Word	Counter Value	Counter Type
0	Loss of Cell Delineation (LCD) events	Software
1	Uncorrectable HEC errors	Hardware
2	HEC corrections	Hardware
3	Received user cells	Hardware
4	Received idle cells	Hardware
5	Received bytes	Software
6	Transmitted user cells	Hardware
7	Transmitted idle cells	Hardware
8	Transmitted bytes	Software
9	Buffer Overflow Events	Software

Table 22 HDLC Performance Counters

Word	Counter Value	Counter Type
0	Received errored packets	Hardware
1	CRC errored packets	Hardware
2	Misaligned packets (bitcount of packet not multiple of 8 bits)	Hardware
3	Too long packets	Hardware
4	Too short packets	Hardware
5	Bad PPP address fields	Hardware
6	Bad PPP control fields	Hardware
7	Overflown packets (rx packets aborted due to buffer overflow)	Software
8	Received valid packets	Hardware
9	Received bytes	Hardware
10	Underflown packets (tx packets aborted due to buffer underrun)	Software
11	Transmitted valid packets	Hardware
12	Transmitted bytes	Hardware
13	Received packets ending with HDLC-abort	Hardware

Table 23 BERT Performance Counters

Word	Counter Value	Counter Type
0	Receive Bit Counter	Hardware
1	Receive Error Counter	Hardware

Command ID: CMD_GET_STATISTICS

This message causes the firmware to control the reporting of the provided Performance Counters.

Input Parameters (optional)		
Parameter ID	*Value*	*Description*
STAT_FRAMR_TOTAL$_P$	_RPT	Report Total-Counts for Framer statistics, do NOT clear counters (refer to **Table 19**)
	_RPT_CLEAR	Report Total-Counts for Framer statistics and clear counters
	_NO_RPT	Do not report Total-Counts of Framer statistics
STAT_FRAMR_1SEC$_P$	_RPT	Get counts of Framer statistics for the last completed 1-second interval
	_NO_RPT	Do not get counts of Framer statistics for the last 1-second interval
STAT_FRAMR_T1231$_P$	_RPT	Report 10-second delayed counts for ANSI T1.231 compliant Framer statistics, do NOT clear counters (refer to **Table 19**)
	_RPT_CLEAR	Report 10-second delayed counts for Framer statistics and clear counters
	_NO_RPT	Do not report 10-second delayed counts of Framer statistics
STAT_PLCP_TOTAL$_P$	*as above*	*as above, for PLCP Mapper (refer to **Table 20**)*
STAT_PLCP_1SEC$_P$	*as above*	
STAT_ATM_TOTAL$_P$	*as above*	*as above, for ATM TC-Sublayer (refer to **Table 21**)*
STAT_ATM_1SEC$_P$	*as above*	
STAT_HDLC_TOTAL$_P$	*as above*	*as above, for HDLC Protocol (refer to **Table 22**)*
STAT_HDLC_1SEC$_P$	*as above*	
STAT_BERT_TOTAL$_P$	*as above*	*as above, for Bit Error Rate Tester BERT (refer to **Table 23**)*
STAT_BERT_1SEC$_P$	*as above*	

Command ID: **CMD_GET_STATISTICS** (cont'd)

Acknowledgement ID: **ACK_GET_STATISTICS**

If activated, this acknowledgement is sent out when the command has been processed. This message provides Performance Counters, when requested so by the host via **CMD_GET_STATISTICS** message.

Output Parameters		
$STAT\_TIMESTAMP_P$	*16-bit field*	Continuous running time stamp that increments every second (wrapping from $FFFF_H$ to 0000_H). This allows to align the TE3-FALC's internal 1-second tick with the host's 1-second timer.
$RPT\_FRAMR\_TOTAL_{32}$	*20 × 32-bit array*	The statistics report consists of several 32-bit parameter arrays which are reported if requested so by command message **CMD_GET_STATISTICS**. E.g., parameter array "RPT_FRAMR_TOTAL" contains a report of the total count of all seven 32-bit wide Framer performance counters in the order listed in **Table 19** (analogue for the PLCP, ATM, HDLC and BERT).
$RPT\_FRAMR\_1SEC_{32}$	*20 × 32-bit array*	
$RPT\_FRAMR\_T1231_{32}$	*20 × 32-bit array*	
$RPT\_PLCP\_TOTAL_{32}$	*7 × 32-bit array*	
$RPT\_PLCP\_1SEC_{32}$	*7 × 32-bit array*	
$RPT\_ATM\_TOTAL_{32}$	*10 × 32-bit array*	
$RPT\_ATM\_1SEC_{32}$	*10 × 32-bit array*	
$RPT\_HDLC\_TOTAL_{32}$	*14 × 32-bit array*	
$RPT\_HDLC\_1SEC_{32}$	*14 × 32-bit array*	
$RPT\_BERT\_TOTAL_{32}$	*2 × 32-bit array*	
$RPT\_BERT\_1SEC_{32}$	*2 × 32-bit array*	

6 Configuration Setup Example - Step by Step

Following an example of a general initialization sequence - step by step:

Power up, RESET

/ MCR Register - read out; logically OR its value with 0000 0001_H; write value back */*
read address 0000_H / e.g. read out value = 0000 0006_H */*
write 0000 0007_H to MCR register (address 0000_H)

/ Poll the LPQOUT for NFC_DCI_ACCESSIBLE */*
read address 0220_H / LPQOUT */*
/ Structure of the Notification see message type description of the data sheet*/*
check if MSGID is 0201_H / 0201_H is the right ID for NFC_DCI_ACCESSIBLE */*

Wait until NFC_DCI_ACCESSIBLE received in LPQOUT

Firmware download, either via DCI or via EPROM
/ see “DCI Boot process” description data sheet and Firmware description */*
/ “Firmware Binary Format” */*
.....
/ Poll the LPQOUT for NFC_FW_ACCESSIBLE */*
read address 0220_H / LPQOUT */*
/ Structure of the Notification see message type description of the data sheet*/*
check if MSGID is $021F_H$ / MSGID see file msgid.h of the firmware package*/*

Wait for NFC_FW_ACCESSIBLE

/ Setup interrupt pin characteristic (e.g. active low)*/*
/ Send in LPQIN CMD_SET_INTERRUPT_MODE_INT_PORT_CFG_ACT_LOW*/*
/ first send parameters with parameter array structure according to data sheet, */*
/ then message*
Parameters / ParameterID see file paramid.h */*
/ write to Ingress buffer, and keep in mind used offset address (here e.g. 0100_H) */*

0001000C$_H$ / TCID=0001$_H$, LENIC=000C$_H$ because of 12 byte message */*
00000001$_H$ / Infofield='valid' => memory area allocated */*
0003003B$_H$ / Pack (Value=0003$_H$, ID=38$_H$ => INT_PORT_CFG_ACT_LOW) */*

/ Write message to LPQIN (address 0260$_H$)*
/ MSGPT=0100$_H$, A-Flag=1, P-Flag=0, PAA=1, MSGID=016$_H$ => see file msgid.h */*
write 0100 4416$_H$ to address 0260$_H$

/ Poll for Acknowledgement Flag*
read address 0220$_H$ / LPQOUT */*
/ Structure of the Acknowledgement see message type description of the data sheet*/*
check if MSGID is 0216$_H$ / MSGID see file msgid.h of the firmware package*/*

Wait for ACK

/ De-mask Interrupts - e.g. de-mask all valid interrupts*/*
write FFFF FF00$_H$ to INTMSK (address 0014$_H$)

/ Please note: User is responsible for Memory Management of the Ingress buffer */*
/ Recommendation: store each message in a separate memory location of the */*
/ Ingress buffer as long as there is space; store address of the infofields to check */*
/ later on if the memory is free for new message allocation */*

/ Send in LPQIN CMD_SPECIFY_CLOCKS */*
/ as above, first parameter, then message*
Parameters / ParameterID see file paramid.h */*
/ write to Ingress buffer, and keep in mind used offset address (here e.g. 0140$_H$) */*
00020028$_H$ / TCID=0002$_H$, LENIC=0010$_H$ because of 16byte message */*
00000001$_H$ / Infofield='valid' => memory area allocated */*
00040012$_H$ / 32-bit Array (Number of bytes=4$_H$, ID=10$_H$ => CLK_FRQ_CLKIN */*
02AA9E00$_H$ / 44.736 MHz on pin CLKIN */*

/ Write message to LPQIN (address 0260$_H$)*
/ MSGPT=0140$_H$, A-Flag=1, P-Flag=0, PAA=1, MSGID=028$_H$ => see file msgid.h */*

write 0140 4428_H to address 0260_H

/ Wait for Interrupt */*
if interrupt active, read out INTREG (address 0010_H)
store all interrupts to handle all interrupt requests
/ here LPQOUT interrupt occured */*
read address 0220_H / LPQOUT */*
/ Structure of the Acknowledgement see message type description of the data sheet*/*
check if MSGID is 0228_H / MSGID see file msgid.h of the firmware package*/*
check return code (RC(3..0) and react accordingly (see data sheet/ firmware description)

/ Setup main flow (e.g. CMD_SET_ATM_DS3_T3 */*
/ Send in LPQIN CMD_SET_ATM_DS3_T3 */*
/ as above, first parameter, then message*
/ write to Ingress buffer, and keep in mind used offset address (here e.g. 0180_H) */*
Parameters / ParameterID see file paramid.h */*
00030010_H / TCID=0003_H, LENIC=$000C_H$ because of 12 byte message*
00000001_H / Infofield='valid' => memory area allocated */*
00010087_H / Pack (Value=0001_H, ID=84_H => SYSIF_MODE_UTOPIA_L2) */*

/ Write message to LPQIN (address 0260_H)*
/ MSGPT=0180_H, A-Flag=1, P-Flag=0, PAA=1, MSGID=$1E2_H$ => see file msgid.h */*
write 0180 $45E2_H$ to address 0260_H

/ Wait for Interrupt */*
if interrupt active, read out INTREG (address 0010_H)
store all interrupts to handle all interrupt requests
/ here LPQOUT interrupt occured */*
read address 0220_H / LPQOUT */*
/ Structure of the Acknowledgement see message type description of the data sheet*/*
check if MSGID is $03E2_H$ / MSGID see file msgid.h of the firmware package*/*
check return code (RC(3..0) and react accordingly (see data sheet/ firmware description)

/ Do changes of default configuration in some module blocks */*

/ Send in LPQIN CMD_CFG_DS3_FRAMER */*
/ as above, first parameter, then message*
/ write to Ingress buffer, and keep in mind used offset address (here e.g. $01C0_H$) */*
Parameters / ParameterID see file paramid.h */*
$0004000C_H$ / TCID=0004_H, LENIC=$000C_H$ because of 12 byte message*
00000001_H / Infofield='valid' => memory area allocated */*
$000200C3_H$ / Pack (Value=0002_H, ID=$C0_H$ => DS3F_MODE_M23) */*

/ Write message to LPQIN (address 0260_H)*
/ MSGPT=$01C0_H$, A-Flag=1, P-Flag=0, PAA=1, MSGID=041_H => see file msgid.h */*
write 01C0 4441_H to address 0260_H

/ Wait for Interrupt */*
if interrupt active, read out INTREG (address 0010_H)
store all interrupts to handle all interrupt requests
/ here LPQOUT interrupt occured */*
read address 0220_H / LPQOUT */*
/ Structure of the Acknowledgement see message type description of the data sheet*/*
check if MSGID is 0241_H / MSGID see file msgid.h of the firmware package*/*
check return code (RC(3..0) and react accordingly (see data sheet/ firmware description)

/ Send in LPQIN CMD_CFG_GPIO */*
/ as above, first parameter, then message*
/ write to Ingress buffer, and keep in mind used offset address (here e.g. 0200_H) */*
Parameters / ParameterID see file paramid.h */*
$0005000C_H$ / TCID=0005_H, LENIC=$000C_H$ because of 12 byte message*
00000001_H / Infofield='valid' => memory area allocated */*
$00FF0F23_H$ / Pack (Value=$00FF_H$, ID=$0F20_H$ => GPIO_ALL_CTRL_LED_STATUS) */*

/ Write message to LPQIN (address 0260_H)*
/ MSGPT=0200_H, A-Flag=1, P-Flag=0, PAA=1, MSGID=148_H => see file msgid.h */*
write 0200 4548_H to address 0260_H

/ Wait for Interrupt */*

if interrupt active, read out INTREG (address 0010_H)
store all interrupts to handle all interrupt requests
/ here LPQOUT interrupt occured */*
read address 0220_H / LPQOUT */*
/ Structure of the Acknowledgement see message type description of the data sheet*/*
check if MSGID is 0348_H / MSGID see file msgid.h of the firmware package*/*
check return code (RC(3..0) and react accordingly (see data sheet/ firmware description)

/ Send in LPQIN CMD_ENABLE_INTERFACES */*
/ as above, first parameter, then message*
/ write to Ingress buffer, and keep in mind used offset address (here e.g. 0240_H) */*
Parameters / ParameterID see file paramid.h */*
00060010_H / TCID=0006_H, LENIC=0010_H because of 16 byte message*
00000001_H / Infofield='valid' => memory area allocated */*
$000101F3_H$ / Pack (Value=0001_H, ID=$01F0_H$ => IF_SYSTEM_ENABLE) */*
$000101F7_H$ / Pack (Value=0001_H, ID=$01F4_H$ => IF_LINE_ENABLE) */*

/ Write message to LPQIN (address 0260_H)*
/ MSGPT=0240_H, A-Flag=1, P-Flag=0, PAA=1, MSGID=011_H => see file msgid.h */*
Message 02404411

/ Wait for Interrupt */*
if interrupt active, read out INTREG (address 0010_H)
store all interrupts to handle all interrupt requests
/ here LPQOUT interrupt occured */*
read address 0220_H / LPQOUT */*
/ Structure of the Acknowledgement see message type description of the data sheet*/*
check if MSGID is 0211_H / MSGID see file msgid.h of the firmware package*/*
check return code (RC(3..0) and react accordingly (see data sheet/ firmware description)

7 Message IDs

Please find following the Message IDs which are supported by the firmware and provided within the firmware package (file msgid.h and bootmsg.h).

Note: It can be possible that in the files 'msgid.h' and 'bootmsg.h' further message IDs are described but not mentioned in the tables below. Such messages are not supported by the current firmware.

7.1 File 'MSGID.H'

Table 7-1 Message IDs Maintenance

Messages	Message ID
CMD_READBACK (page 48)	183_H
NFC_DEVICE_NOT_CONFIGURED (page 51)	283_H
ACK_READBACK (page 49)	383_H
CMD_GET_VERSION_ID (page 47)	$00D_H$
ACK_GET_VERSION_ID (page 47)	$20D_H$
CMD_ENABLE_INTERFACES (page 67)	011_H
ACK_ENABLE_INTERFACES	211_H
CMD_SET_LOOP (page 46)	012_H
ACK_SET_LOOP	212_H
CMD_SET_INTERRUPT_MODE (page 45)	016_H
ACK_SET_INTERRUPT_MODE	216_H
CMD_SET_NFC_ALM_MODE (page 45)	$01D_H$
ACK_SET_NFC_ALM_MODE	$21D_H$
CMD_SELF_DIAGNOSTICS (page 47)	$01E_H$
ACK_SELF_DIAGNOSTICS (page 48)	$21E_H$
CMD_RESET_DEVICE (page 48)	$01F_H$
NFC_FW_ACCESSIBLE (page 41)	$21F_H$

Table 7-2 Message IDs Clocking Unit

Messages	Message ID
CMD_SPECIFY_CLOCKS (page 41)	028_H
ACK_SPECIFY_CLOCKS	228

Table 7-2 Message IDs Clocking Unit (cont'd)

NFC_PLL_LOCKED (page 44)	$22E_H$
NFC_PLL_UNLOCKED (page 44)	$22F_H$

Table 7-3 Message IDs Line Interface Unit (LIU)

Messages	Message ID
CMD_CFG_LIU (page 67)	030_H
ACK_CFG_LIU	230_H

Table 7-4 Message IDs Bit Error Rate Tester (BERT)

Messages	Message ID
NFC_BERT_RESULT (page 98)	237_H
CMD_CTRL_BERT (page 97)	038_H
ACK_CTRL_BERT	238_H
CMD_CTRL_BERT_ERROR (page 99)	039_H
ACK_CTRL_BERT_ERROR	239_H

Table 7-5 Message IDs DS3/E3 Framer

Messages	Message ID
CMD_CFG_E3G751_FRAMER (page 71)	050_H
ACK_CFG_E3G751_FRAMER	250_H
NFC_DS3_FRAMER_MODE (page 70)	240_H
CMD_CFG_DS3_FRAMER (page 69)	041_H
ACK_CFG_DS3_FRAMER	241_H
CMD_CFG_E3G832_FRAMER (page 70)	045_H
ACK_CFG_E3G832_FRAMER	245_H
CMD_FRM_TX_AIS_IDLE (page 90)	048_H
ACK_FRM_TX_AIS_IDLE	248_H
CMD_CFG_RDI_DECLARATION (page 91)	049_H
ACK_CFG_RDI_DECLARATION	249_H
CMD_FRM_TX_RDI (page 92)	051_H
ACK_FRM_TX_RDI	251_H

Table 7-6 Message IDs DJAT

Messages	Message ID
CMD_CFG_DJAT (page 68)	052_H
ACK_CFG_DJAT	252_H

Table 7-7 Message IDs PLL

Messages	Message ID
CMD_CFG_PLL (page 43)	053_H
ACK_CFG_PLL	253_H

Table 7-8 Message IDs Alarms

Messages	Message ID
ALM_FRM_LOS_DEFECT (page 86)	260_H
ALM_FRM_LOS_FAILURE (page 87)	261_H
ALM_FRM_OOF_DEFECT (page 87)	262_H
ALM_FRM_LOF_FAILURE (page 88)	263_H
ALM_FRM_AIS_DEFECT (page 88)	264_H
ALM_FRM_AIS_FAILURE (page 89)	265_H
ALM_FRM_IDLE_RECEIVED (page 89)	266_H
ALM_FRM_RDI_DEFECT (page 90)	267_H
ALM_FRM_RDI_FAILURE (page 91)	268_H
ALM_FEAC_ALARM (page 93)	269_H
ALM_PLCP_OOF_DEFECT (page 94)	$26A_H$
ALM_PLCP_RAI_DEFECT (page 94)	$26B_H$
ALM_G804_LCD (page 95)	$26C_H$
ALM_TX_UNDERRUN (page 96)	$26D_H$
ALM_RX_OVERFLOW (page 96)	$26E_H$
ALM_SYSIF_PARITY_DEFECT (page 96)	$26F_H$

Table 7-9 Message IDs HDLC/PPP

Messages	Message ID
CMD_CFG_HDLC_PROCESSOR (page 72)	121_H
ACK_CFG_HDLC_PROCESSOR	321_H

Table 7-10 Message IDs G804

Messages	Message ID
CMD_CFG_ATM_PROCESSOR (page 73)	111_H
ACK_CFG_ATM_PROCESSOR	311_H

Table 7-11 Message IDs Insertion/Extraction Buffer

Messages	Message ID
CMD_CFG_BUFFERS (page 75)	131_H
ACK_CFG_BUFFERS	331_H

Table 7-12 Message IDs System Interface

Messages	Message ID
CMD_CFG_SYSIF (page 74)	141_H
ACK_CFG_SYSIF	341_H

Table 7-13 Message IDs TTI

Messages	Message ID
NFC_TTI_RX_DATA_CHANGE (page 79)	$2C0_H$
CMD_CFG_TTI_CHANNEL (page 78)	$0C1_H$
ACK_CFG_TTI_CHANNEL	$2C1_H$

Table 7-14 Message IDs PTY

Messages	Message ID
NFC_PTY_RX_DATA_CHANGE (page 80)	$2C4_H$
CMD_CFG_PTY_CHANNEL (page 79)	$0C5_H$
ACK_CFG_PTY_CHANNEL	$2C5_H$

Table 7-15 Message IDs SSM

Messages	Message ID
NFC_SSM_RX_DATA_CHANGE (page 81)	$2C8_H$
CMD_CFG_SSM_CHANNEL (page 81)	$0C9_H$
ACK_CFG_SSM_CHANNEL	$2C9_H$

Table 7-16 Message IDs MDL

Messages	Message ID
NFC_MDL_RX_DATA (page 82)	$2A0_H$
CMD_CFG_MDL_CHANNEL (page 82)	$0A1_H$
ACK_CFG_MDL_CHANNEL	$2A1_H$
CMD_SET_MDL_TX_DATA (page 82)	$0A2_H$
ACK_SET_MDL_TX_DATA	$2A2_H$

Table 7-17 Message IDs FEAC

Messages	Message ID
NFC_FEAC_LOOP_STATUS (page 85)	$2B0_H$
CMD_SET_FEAC_CODE (page 84)	$0B1_H$
ACK_SET_FEAC_CODE	$2B1_H$
CMD_CFG_FEAC_GENERATION (page 93)	$0B2_H$
ACK_CFG_FEAC_GENERATION	$2B2_H$
CMD_GET_FEAC_CODE (page 86)	$0B3_H$
ACK_GET_FEAC_CODE	$2B3_H$

Table 7-18 Message IDs Datapath

Messages	Message ID
CMD_SET_ATM_E3G751_E3 (page 55)	$1D9_H$
ACK_SET_ATM_E3G751_E3	$3D9_H$
CMD_SET_ATM_E3G751 (page 55)	$1DA_H$
ACK_SET_ATM_E3G751	$3DA_H$

Table 7-18 Message IDs Datapath (cont'd)

CMD_SET_ATM_PLCP_E3G751_E3 (page 56)	$1DB_H$
ACK_SET_ATM_PLCP_E3G751_E3	$3DB_H$
CMD_SET_ATM_PLCP_E3G751 (page 56)	$1DC_H$
ACK_SET_ATM_PLCP_E3G751	$3DC_H$
CMD_SET_PPP_E3G751_E3 (page 60)	$1DD_H$
ACK_SET_PPP_E3G751_E3	$3DD_H$
CMD_SET_PPP_E3G751 (page 60)	$1DE_H$
ACK_SET_PPP_E3G751	$3DE_H$
CMD_SET_BIT_E3G751_E3 (page 66)	$1DF_H$
ACK_SET_BIT_E3G751_E3	$3DF_H$
CMD_SET_BIT_E3G751 (page 66)	$1E0_H$
ACK_SET_BIT_E3G751	$3E0_H$
CMD_SET_ATM_E3G832_E3 (page 54)	$1E1_H$
ACK_SET_ATM_E3G832_E3	$3E1_H$
CMD_SET_ATM_DS3_T3 (page 57)	$1E2_H$
ACK_SET_ATM_DS3_T3	$3E2_H$
CMD_SET_ATM_PLCP_DS3_T3 (page 58)	$1E3_H$
ACK_SET_ATM_PLCP_DS3_T3	$3E3_H$
CMD_SET_PPP_E3G832_E3 (page 59)	$1E4_H$
ACK_SET_PPP_E3G832_E3	$3E4_H$
CMD_SET_PPP_DS3_T3 (page 61)	$1E5_H$
ACK_SET_PPP_DS3_T3	$3E5_H$
CMD_SET_ATM_E3 (page 62)	$1E6_H$
ACK_SET_ATM_E3	$3E6_H$
CMD_SET_ATM_T3 (page 63)	$1E7_H$
ACK_SET_ATM_E3	$3E7_H$
CMD_SET_PPP_E3 (page 64)	$1E8_H$
ACK_SET_PPP_E3	$3E8_H$
CMD_SET_PPP_T3 (page 64)	$1E9_H$
CMD_SET_PPP_T3	$3E9_H$
CMD_SET_ATM_E3G832 (page 54)	$1EA_H$

Table 7-18 Message IDs Datapath (cont'd)

ACK_SET_ATM_E3G832	$3EA_H$
CMD_SET_ATM_DS3 (page 57)	$1EB_H$
ACK_SET_ATM_DS3	$3EB_H$
CMD_SET_ATM_PLCP_DS3 (page 58)	$1EC_H$
ACK_SET_ATM_PLCP_DS3	$3EC_H$
CMD_SET_PPP_E3G832 (page 59)	$1ED_H$
ACK_SET_PPP_E3G832	$3ED_H$
CMD_SET_PPP_DS3 (page 61)	$1EE_H$
ACK_SET_PPP_DS3	$3EE_H$
CMD_SET_BIT_E3G832_E3 (page 65)	$1F1_H$
ACK_SET_BIT_E3G832_E3	$3F1_H$
CMD_SET_BIT_DS3_T3 (page 66)	$1F2_H$
ACK_SET_BIT_DS3_T3	$3F2_H$
CMD_SET_BIT_E3G832 (page 65)	$1F5_H$
ACK_SET_BIT_E3G832	$3F5_H$
CMD_SET_BIT_DS3 (page 66)	$1F6_H$
ACK_SET_BIT_DS3	$3F6_H$

Table 7-19 Message IDs Statistics

Messages	**Message ID**
CMD_GET_STATISTICS (page 105)	081_H
ACK_GET_STATISTICS	281_H

Table 7-20 Message IDs Device Status

Messages	**Message ID**
CMD_GET_STATUS (page 100)	088_H
ACK_GET_STATUS	288_H
CMD_CFG_GPIO (page 76)	148_H
ACK_CFG_GPIO	348_H
CMD_READ_GPIO (page 77)	149_H
ACK_READ_GPIO	349_H

Table 7-20 Message IDs Device Status (cont'd)

CMD_WRITE_GPIO (page 77)	$14A_H$
ACK_WRITE_GPIO	$34A_H$

7.2 File 'BOOTMSG.H'

Table 7-21 Message IDs Boot Messages

Messages	Message ID
CMD_BLOCK_UPLOAD_WITH_CRC (page 16)	0001_H
CMD_BLOCK_UPLOAD (page 16)	0002_H
CMD_UPLOAD_COMPLETE (page 16)	0003_H
CMD_MEMORY_REGION_CHECK (page 17)	0004_H
CMD_INTERRUPT_MODE_SET (page 18)	0006_H
CMD_CLOCKING_FREQUENCY_SET (page 17)	0021_H
NFC_DCI_ACCESSIBLE (page 16)	0201_H

8 Parameter IDs

Please find following the Parameter IDs which are supported by the firmware and provided within the firmware package (file paramid.h).

Note: It can be possible that in the file 'paramid.h' further parameter IDs are described but not mentioned in the description below. Such parameters are not supported by the current firmware.

```
/* definition of the parameter types */

#define P_08 0 /* parameter array of 8-bit values */
#define P_16 1 /* parameter array of 16-bit values */
#define P_32 2 /* parameter array of 32-bit values */
#define P_P 3 /* one 16-bit value ('packed') */

#define HW_VERSION_ID_P (0x0004|P_32) /* 32-bit value Hardware version ID */
#define FW_VERSION_ID_P (0x0008|P_32) /* 32-bit value Firmware version ID */

#define CLK_FRQ_CLKIN_32 (0x0010|P_32)

#define CLK_SRC_TCLKIN_P (0x0014|P_P)
#define CLK_SRC_TCLKIN_DIRECT 0
#define CLK_SRC_TCLKIN_VIA_PLL 1
#define CLK_SRC_TCLKIN_DEFAULT CLK_SRC_TCLKIN_DIRECT

#define CLK_FRQ_TCLKIN_32 (0x0018|P_32)

#define CLK_FRQ_RSYNC_32 (0x001C|P_32)

#define CLK_FRQ_CLKOUT_32 (0x0020|P_32)

#define CLK_SRC_RCLKOUT_P (0x0024|P_P)
#define CLK_SRC_RCLKOUT_DIRECT 0
#define CLK_SRC_RCLKOUT_VIA_PLL 1
#define CLK_SRC_RCLKOUT_DEFAULT CLK_SRC_RCLKOUT_DIRECT
```

```
#define CLK_FRQ_RCLKOUT_32 (0x0028|P_32)

#define INTERNAL_PLL_P (0x0030|P_P)
#define INTERNAL_PLL_RX 0x5258
#define INTERNAL_PLL_TX 0x5458
#define INTERNAL_PLL_ANALOG 0x5658

#define INT_PORT_CFG_P (0x0038|P_P)
#define INT_PORT_CFG_DISABLE 0 /* Interrupt output is disabled (tri-state). */
#define INT_PORT_CFG_OPEN_DRAIN 1 /* Interrupt output is open-drain. */
#define INT_PORT_CFG_ACT_HIGH 2 /* Interrupt output is push/pull, active high. */
#define INT_PORT_CFG_ACT_LOW 3 /* Interrupt output is push/pull, active low. */

#define MSG_TO_HPQOUT_P (0x0040|P_P)
#define MSG_TO_LPQOUT_P (0x0044|P_P)
#define MSG_TO_NOQOUT_P (0x0048|P_P)

#define LOOP_LIU_P (0x0050|P_P)
#define LOOP_LIU_OFF 0 /* No analog line loop */
#define LOOP_LIU_LOCAL 1 /* Local line loop (analog) */
#define LOOP_LIU_REMOTE 2 /* remote line loop (analog) */
#define LOOP_LIU_DEFAULT LOOP_LIU_LOCAL

#define LOOP_FRAMER_P (0x0054|P_P)
#define LOOP_FRAMER_OFF 0 /* no digital line (framer) loop */
#define LOOP_FRAMER_LOCAL 1 /* local digital line (framer) loop */
#define LOOP_FRAMER_DEFAULT LOOP_FRAMER_OFF

#define LOOP_SYSIF_P (0x0058|P_P)
#define LOOP_SYSIF_OFF 0 /* no system interface loop */
#define LOOP_SYSIF_LOCAL 1 /* local system interface loop */
#define LOOP_SYSIF_DEFAULT LOOP_SYSIF_OFF
```

```
#define REGISTER_ADDR_32 (0x0060|P_32)
#define REGISTER_VALUE_32 (0x0064|P_32)
#define REGISTER_CNT_P (0x0068|P_P)
#define REGISTER_CNT_DEFAULT 1

#define REBOOT_SOURCE_P (0x0070|P_P)
#define REBOOT_SOURCE_CURR_IMAGE 0 /* Reboot from current image.*/
#define REBOOT_SOURCE_PRESET 1 /* Load new image according to preset
Hardware Boot Selection and reboot.*/
#define REBOOT_SOURCE_DEFAULT REBOOT_SOURCE_CURR_IMAGE

#define TEST_RESULT_P (0x0078|P_P)
#define TEST_RESULT_OK 0
#define TEST_RESULT_FAIL 1

#define ERROR_CODE_P (0x007C|P_P)

#define LINE_TIMING_P (0x0080|P_P)
#define LINE_TIMING_MASTER_INT 0 /* Master (internal), CLKIN */
#define LINE_TIMING_MASTER_EXT 1 /* Master (external), TCLKIN */
#define LINE_TIMING_LOOP 2 /* Loop timing  */
#define LINE_TIMING_DEFAULT LINE_TIMING_MASTER_INT

#define SYSIF_MODE_P (0x0084|P_P)
#define SYSIF_MODE_UTOPIA_L1 0 /* UTOPIA Level 1 mode  */
#define SYSIF_MODE_UTOPIA_L2 1 /* UTOPIA Level 2 mode  */
#define SYSIF_MODE_POSPHY_L2 2 /* POS-PHY Level 2 mode  */
#define SYSIF_MODE_UTOPIA_L2X 3 /* UTOPIA Level 2 Plus mode  */
#define SYSIF_MODE_DEFAULT SYSIF_MODE_UTOPIA_L2X

#define SYSIF_WIDTH_P (0x0088|P_P)
#define SYSIF_WIDTH_8BIT 8 /* 8-bit data bus width */
#define SYSIF_WIDTH_16BIT 16 /* 16-bit data bus width */
#define SYSIF_WIDTH_DEFAULT SYSIF_WIDTH_8BIT
```

```
#define SYSIF_STATUS_P (0x0090|P_P)
#define SYSIF_STATUS_POLLING 0 /* MPHY Polling */
#define SYSIF_STATUS_DIRECT_IND 1 /* Direct Status Indication */
#define SYSIF_STATUS_DEFAULT SYSIF_STATUS_POLLING

#define SYSIF_ADDR_P (0x0094|P_P)
#define SYSIF_ADDR_DEFAULT 0x0000

#define SYSIF_CHUNK_SIZE_P (0x0098|P_P)
#define SYSIF_CHUNK_SIZE_DEFAULT 64

#define SYSIF_OUTPUTS_P (0x009C|P_P)
#define SYSIF_OUTPUTS_TRISTATE 0
#define SYSIF_OUTPUTS_DRIVING 1
#define SYSIF_OUTPUTS_DEFAULT SYSIF_OUTPUTS_TRISTATE

#define SYSIF_HANDSHAKE_P (0x00A0|P_P)
#define SYSIF_HANDSHAKE_PKT_CELL 0
#define SYSIF_HANDSHAKE_BYTE 1
#define SYSIF_HANDSHAKE_DEFAULT SYSIF_HANDSHAKE_PKT_CELL

#define SYSIF_PARITY_CHK_P (0x00A4|P_P)
#define SYSIF_PARITY_CHK_DISABLE 0
#define SYSIF_PARITY_CHK_ENABLE 1
#define SYSIF_PARITY_CHK_DEFAULT SYSIF_PARITY_CHK_ENABLE

#define SYSIF_PARITY_P (0x00A8|P_P)
#define SYSIF_PARITY_ODD 0
#define SYSIF_PARITY_EVEN 1
#define SYSIF_PARITY_DEFAULT SYSIF_PARITY_ODD

#define LIU_AMPLIFIER_P (0x00B0|P_P)
#define LIU_AMPLIFIER_NORMAL 0 /* Normal amplifier setting */
```

#define LIU_AMPLIFIER_MONITOR 1 /* Monitor mode: additional 20 dB gain stage at RL1/RL2 activated */
#define LIU_AMPLIFIER_DEFAULT LIU_AMPLIFIER_NORMAL

#define LIU_TRANSMIT_P (0x00B4|P_P)
#define LIU_TRANSMIT_DISABLE 0 /* Transmitter disabled */
#define LIU_TRANSMIT_ENABLE 1 /* Transmitter enabled */
#define LIU_TRANSMIT_DEFAULT LIU_TRANSMIT_ENABLE

#define DS3F_MODE_P (0x00C0|P_P)
#define DS3F_MODE_CBIT_PARITY 1 /* Select C-Bit Parity mode */
#define DS3F_MODE_M23 2 /* Select M23 unchannelized mode */
#define DS3F_MODE_AUTOSENSE 3 /* Automatically sense C-Bit Parity mode or M23 mode via the AIC channel*/
#define DS3F_MODE_BYPASS 0 /* Bypass DS3 framer */
#define DS3F_MODE_DEFAULT DS3F_MODE_AUTOSENSE

#define DS3F_AIS_PATTERN_P (0x00C4|P_P)
#define DS3F_AIS_PATTERN_TOGGLE 0xAAAA /* AIS pattern is 101010... between overhead bits, C-bits all zeros, X-bits all ones (standard) */
#define DS3F_AIS_PATTERN_ALL_ONES 0xFFFF /* AIS pattern is unframed all ones */
#define DS3F_AIS_PATTERN_DEFAULT DS3F_AIS_PATTERN_TOGGLE

#define DS3F_F_REFRAME_P (0x00C8|P_P)
#define DS3F_F_REFRAME_3_OF_8 0x0803 /* New F-frame search when 3 out of 8 contiguous F bits are in error */
#define DS3F_F_REFRAME_3_OF_16 0x1003 /* New F-frame search when 3 out of 16 contiguous F bits are in error */
#define DS3F_F_REFRAME_DEFAULT DS3F_F_REFRAME_3_OF_8

#define DS3F_M_REFRAME_P (0x00CC|P_P)
#define DS3F_M_REFRAME_DISABLE 0 /* No new F-frame search due to M-bit errors, only for F-bit errors */

```
#define DS3F_M_REFRAME_2_OF_4 0x0402 /* New F-frame search if M-bit errors are
detected in 2 out of 4 consecutive M-frames */
#define DS3F_M_REFRAME_3_OF_4 0x0403 /* New F-frame search if M-bit errors are
detected in 3 out of 4 consecutive M-frames */
#define DS3F_M_REFRAME_DEFAULT DS3F_M_REFRAME_2_OF_4

#define E3F_MODE_P (0x00D0|P_P)
#define E3F_MODE_BYPASS 0 /* Bypass E3 framer */
#define E3F_MODE_G832 1 /* Select E3 framing according to ITU-T G.832 */
#define E3F_MODE_G751 2 /* Select E3 framing according to ITU-T G.751 */
#define E3F_MODE_DEFAULT E3F_MODE_G832

#define E3F_REFRAME_P (0x00D4|P_P)
#define E3F_REFRAME_3_FAE 3 /* The framer gets asynchronous after
detecting three consecutive frame alignment errors */
#define E3F_REFRAME_4_FAE 4 /* The framer gets asynchronous after
detecting four consecutive frame alignment errors */
#define E3F_REFRAME_DEFAULT E3F_REFRAME_3_FAE

#define E3F_REFRAME_BIPE_P (0x10B0|P_P)
#define E3F_REFRAME_BIPE_DISABLE 0
#define E3F_REFRAME_BIPE_ENABLE 1
#define E3F_REFRAME_BIPE_DEFAULT E3F_REFRAME_BIPE_DISABLE

#define E3F_OVHD_FORMAT_P (0x10B4|P_P)
#define E3F_OVHD_FORMAT_C_ONLY 0
#define E3F_OVHD_FORMAT_C_AND_J 1
#define E3F_OVHD_FORMAT_NONE 2
#define E3F_OVHD_FORMAT_DEFAULT E3F_OVHD_FORMAT_C_ONLY

#define E3F_PL_FORMAT_P (0x00DC|P_P)
#define E3F_PL_FORMAT_PPP 0
#define E3F_PL_FORMAT_ATM 1
```

```
#define HDLC_CRC_MODE_P (0x00E0|P_P)
#define HDLC_CRC_MODE_NOCRC 0 /* Disable CRC generation/check */
#define HDLC_CRC_MODE_CRC16 16 /* Select CRC-16 generation/check */
#define HDLC_CRC_MODE_CRC32 32 /* Select CRC-32 generation/check */
#define HDLC_CRC_MODE_DEFAULT HDLC_CRC_MODE_CRC16
#define HDLC_SHARED_FLG_P (0x00E4|P_P)
#define HDLC_SHARED_FLG_DISABLE 0 /* Disable shared flag transmission */
#define HDLC_SHARED_FLG_ENABLE 1 /* Enable shared flag transmission */
#define HDLC_SHARED_FLG_DEFAULT HDLC_SHARED_FLG_ENABLE

#define HDLC_MODE_P (0x00E8|P_P)
#define HDLC_MODE_BIT_HDLC 257 /* Bit-synchronous HDLC operation. */
#define HDLC_MODE_BIT_PPP 1 /* Bit-synchronous PPP operation. */
#define HDLC_MODE_BYTE_PPP 8 /* Byte-synchronous PPP operation. Setting
not valid for DS3 flows */
#define HDLC_MODE_BYTE_HDLC 127 /* Byte-synchronous HDLC operation. */
#define HDLC_MODE_TRANSP 0 /* Transparent mode: No flags, no CRC, no
bit/byte stuffing */
#define HDLC_MODE_DEFAULT HDLC_MODE_BIT_PPP

#define HDLC_MAX_FRMLEN_P (0x00EC|P_P)
#define HDLC_MAX_FRMLEN_DISABLED 0
#define HDLC_MAX_FRMLEN_DEFAULT HDLC_MAX_FRM_LEN_DISABLED

#define HDLC_MIN_FRMLEN_P (0x00F0|P_P)
#define HDLC_MIN_FRMLEN_DISABLED 0
#define HDLC_MIN_FRMLEN_DEFAULT HDLC_MIN_FRM_LEN_DISABLED

#define PPP_AFC_P (0x00F4|P_P)
#define PPP_AFC_DISABLE 0
#define PPP_AFC_ENABLE 1

#define PPP_PFC_P (0x00D8|P_P)
#define PPP_PFC_DISABLE 0
```

```
#define PPP_PFC_ENABLE 1

#define HDLC_IFTF_P (0x00F8|P_P)
#define HDLC_IFTF_FLAGS 0
#define HDLC_IFTF_ALL_ONES 1
#define HDLC_IFTF_DEFAULT HDLC_IFTF_FLAGS

#define ATM_MAPPING_P (0x00FC|P_P)
#define ATM_MAPPING_DIRECT 0 /* Direct mapped ATM.*/
#define ATM_MAPPING_PLCP 1 /* PLCP mapped ATM. Setting only
available with DS3 and E3 G.751 flow.*/
#define ATM_MAPPING_DEFAULT ATM_MAPPING_DIRECT

#define ATM_SCRAMBLING_P (0x0100|P_P)
#define ATM_SCRAMBLING_DISABLE 0 /* Disable ATM payload scrambling */
#define ATM_SCRAMBLING_ENABLE 1 /* Enable ATM payload scrambling acc. to
I.432.1.*/
#define ATM_SCRAMBLING_DEFAULT ATM_SCRAMBLING_ENABLE

#define ATM_HEAD_CORR_P (0x0104|P_P)
#define ATM_HEAD_CORR_DISABLE 0 /* Disable header error correction.*/
#define ATM_HEAD_CORR_ENABLE 1 /* Enable single-bit header error correction.*/
#define ATM_HEAD_CORR_DEFAULT ATM_HEAD_CORR_ENABLE

#define ATM_ALPHA_P (0x0108|P_P)
#define ATM_ALPHA_DEFAULT 7

#define ATM_DELTA_P (0x010C|P_P)
#define ATM_DELTA_DEFAULT 6

#define BUF_FW_THRESH_P (0x0110|P_P)
#define BUF_FW_THRESH_DEFAULT 64
#define BUF_SIZE_P (0x0114|P_P)
#define BUF_SIZE_DEFAULT 0
```

#define TTI_RX_CHNG_CNT_P (0x0124|P_P)

#define TTI_RX_CHNG_CNT_DEFAULT 3

#define TTI_CRC_GENERATE_P (0x0128|P_P)

#define TTI_CRC_GENERATE_DISABLE 0 /* Disable CRC-7 checksum generation.*/

#define TTI_CRC_GENERATE_ENABLE 1 /* Enable CRC-7 checksum generation.*/

#define TTI_CRC_GENERATE_DEFAULT TTI_CRC_GENERATE_DISABLE

#define TTI_TX_DATA_08 (0x012C|P_08) /* 16 bytes of new transmit TTI Data (includes CRC) */

#define TTI_RX_CMP_DATA_08 (0x0130|P_08) /* 16 bytes of new receive TTI Data to compare with (includes CRC) */

#define TTI_RX_STATUS_P (0x0134|P_P)

#define TTI_RX_STATUS_INVALID 0 /* Received TTI data is unstable.*/

#define TTI_RX_STATUS_MATCH 1 /* New received TTI data matches preset receive data.*/

#define TTI_RX_STATUS_MISMATCH 2 /* New received TTI data does not match preset receive data.*/

#define TTI_RX_DATA_08 (0x0138|P_08) /* 16 bytes of new receive TTI Data (includes CRC) */

#define PTY_RX_CHNG_CNT_P (0x0144|P_P)

#define PTY_RX_CHNG_CNT_DEFAULT 3

#define PTY_TX_FIELD_P (0x0148|P_P)

#define PTY_TX_FIELD_UNEQUIPPED 0x00 /* values match the appropriate values from the MA byte */

#define PTY_TX_FIELD_NONSPECIFIC 0x08 /* see G.832 2.1.2 */

#define PTY_TX_FIELD_ATM 0x10

#define PTY_TX_FIELD_SDH_TU12 0x18

#define PTY_RX_CMP_FIELD_P (0x014C|P_P)

#define PTY_RX_CMP_FIELD_UNEQUIPPED 0x00 /* values match the appropriate values from the MA byte */

#define PTY_RX_CMP_FIELD_NONSPECIFIC 0x08 /* see G.832 2.1.2 */

#define PTY_RX_CMP_FIELD_ATM 0x10

#define PTY_RX_CMP_FIELD_SDH_TU12 0x18

#define PTY_RX_STATUS_P (0x0150|P_P)

#define PTY_RX_STATUS_UNSTABLE 0 /* Received PTY data is unstable.*/

#define PTY_RX_STATUS_MATCH 1 /* New received PTY data matches preset receive data.*/

#define PTY_RX_STATUS_MISMATCH 2 /* New received PTY data does not match preset receive data.*/

#define PTY_RX_FIELD_P (0x0154|P_P)

#define PTY_RX_FIELD_UNEQUIPPED 0x00 /* values match the appropriate values from the MA byte */

#define PTY_RX_FIELD_NONSPECIFIC 0x08 /* see G.832 2.1.2 */

#define PTY_RX_FIELD_ATM 0x10

#define PTY_RX_FIELD_SDH_TU12 0x18

#define MDL_RECEIVE_P (0x0160|P_P)

#define MDL_RECEIVE_DISABLE 0 /* Disable MDL Receive Channel.*/

#define MDL_RECEIVE_ENABLE 1 /* Enable MDL Receive Channel.*/

#define MDL_RECEIVE_DEFAULT MDL_RECEIVE_DISABLE

#define MDL_TRANSMIT_P (0x0164|P_P)

#define MDL_TRANSMIT_DISABLE 0 /* Disable MDL Transmit Channel.*/

#define MDL_TRANSMIT_ENABLE 1 /* Enable MDL Transmit Channel. */

#define MDL_TRANSMIT_DEFAULT MDL_TRANSMIT_DISABLE

#define MDL_MAX_FRM_LEN_P (0x0168|P_P)

#define MDL_MAX_FRM_LEN_DEFAULT 256

#define MDL_TX_DATA_08 (0x016C|P_08)

#define MDL_RX_DATA_08 (0x0174|P_08)

```
#define MDL_RX_STATUS_P (0x0178|P_P)
#define MDL_RX_STATUS_VALID_FRAME 0 /* Valid Frame */
#define MDL_RX_STATUS_CRC_ERR 1 /* CRC error*/
#define MDL_RX_STATUS_ABORT_ERR 2 /* Aborted frame*/
#define MDL_RX_STATUS_MAXFL_ERR 3 /* Frame exceeded maximum length*/
#define MDL_RX_STATUS_MINFL_ERR 4 /* Frame length below minimum length */
#define MDL_RX_STATUS_ILEN_ERR 5 /* Frame length not multiple of 8 bits */

#define MDL_RX_ERR_RPT_P (0x017C|P_P)
#define MDL_RX_ERR_RPT_DISABLE 0
#define MDL_RX_ERR_RPT_ENABLE 1
#define MDL_RX_ERR_RPT_DEFAULT MDL_RX_ERR_RPT_DISABLE

#define FEAC_CODEWORD_P (0x0180|P_P)
#define FEAC_CODEWORD_DS3_SA 0x3238 /* highbyte contains the FEAC pattern */
#define FEAC_CODEWORD_DS3_LOS 0x1C30 /* lowbyte contains a relative priority */
#define FEAC_CODEWORD_DS3_OOF 0x0028 /* that manages the precedence of
FEAC codes */
#define FEAC_CODEWORD_DS3_AIS 0x2C20
#define FEAC_CODEWORD_DS3_IDLE 0x3418
#define FEAC_CODEWORD_DS3_NSA 0x1E10
#define FEAC_CODEWORD_COM_NSA 0x3A08
#define FEAC_CODEWORD_DS3LP_ON 0x8F40
#define FEAC_CODEWORD_DS3LP_OFF 0xB940
#define FEAC_CODEWORD_NONE 0xFF00

#define                    FEAC_CODEWORD_USER_DEF(pattern,prio)
((((pattern)<<8)&0xFF00)|((prio)&0x00FF))

#define FEAC_CODE_CNT_P (0x0184|P_P)
#define FEAC_CODE_CNT_INFINITE 0
#define FEAC_CODE_CNT_DEFAULT 10
```

```
#define FEAC_DS3_LOOP_P (0x0188|P_P)
#define FEAC_DS3_LOOP_INACTIVE 0
#define FEAC_DS3_LOOP_ACTIVE 1

#define FRM_TX_SIGNAL_P (0x0190|P_P)
#define FRM_TX_SIGNAL_NORMAL 0x0000
#define FRM_TX_SIGNAL_AIS 0xAAAA
#define FRM_TX_SIGNAL_IDLE 0xCCCC
#define FRM_TX_SIGNAL_DEFAULT FRM_TX_SIGNAL_NORMAL

#define RDI_ON_LOS_DEFCT_P (0x01A0|P_P)
#define RDI_ON_LOS_DEFCT_DISABLE 0
#define RDI_ON_LOS_DEFCT_ENABLE 1
#define RDI_ON_LOS_DEFCT_DEFAULT RDI_ON_LOS_DEFCT_ENABLE

#define RDI_ON_OOF_DEFCT_P (0x01A8|P_P)
#define RDI_ON_OOF_DEFCT_DISABLE 0
#define RDI_ON_OOF_DEFCT_ENABLE 1
#define RDI_ON_OOF_DEFCT_DEFAULT RDI_ON_OOF_DEFCT_ENABLE

#define RDI_ON_LOF_FAIL_P (0x01AC|P_P)
#define RDI_ON_LOF_FAIL_DISABLE 0
#define RDI_ON_LOF_FAIL_ENABLE 1
#define RDI_ON_LOF_FAIL_DEFAULT RDI_ON_LOF_FAIL_DISABLE

#define RDI_ON_AIS_DEFCT_P (0x01B0|P_P)
#define RDI_ON_AIS_DEFCT_DISABLE 0
#define RDI_ON_AIS_DEFCT_ENABLE 1
#define RDI_ON_AIS_DEFCT_DEFAULT RDI_ON_AIS_DEFCT_ENABLE

#define RDI_ON_LCD_P (0x01D0|P_P)
#define RDI_ON_LCD_DISABLE 0
#define RDI_ON_LCD_ENABLE 1
#define RDI_ON_LCD_DEFAULT RDI_ON_LCD_ENABLE
```

```
#define RDI_ON_TIM_P (0x01E0|P_P)
#define RDI_ON_TIM_DISABLE 0
#define RDI_ON_TIM_ENABLE 1
#define RDI_ON_TIM_DEFAULT RDI_ON_TIM_DISABLE

#define RDI_ON_UNEQ_P (0x01E4|P_P)
#define RDI_ON_UNEQ_DISABLE 0
#define RDI_ON_UNEQ_ENABLE 1
#define RDI_ON_UNEQ_DEFAULT RDI_ON_UNEQ_DISABLE

#define RDI_ON_PLM_P (0x01E8|P_P)
#define RDI_ON_PLM_DISABLE 0
#define RDI_ON_PLM_ENABLE 1
#define RDI_ON_PLM_DEFAULT RDI_ON_PLM_DISABLE

#define IF_SYSTEM_P (0x01F0|P_P)
#define IF_SYSTEM_DISABLE 0
#define IF_SYSTEM_ENABLE 1
#define IF_SYSTEM_DEFAULT IF_SYSTEM_DISABLE

#define IF_LINE_P (0x01F4|P_P)
#define IF_LINE_DISABLE 0
#define IF_LINE_ENABLE 1
#define IF_LINE_DEFAULT IF_LINE_DISABLE

#define IF_BREAKOUT_P (0x01F8|P_P)
#define IF_BREAKOUT_DISABLE 0
#define IF_BREAKOUT_ENABLE 1
#define IF_BREAKOUT_DEFAULT IF_BREAKOUT_DISABLE

#define IF_OVERHEAD_P (0x01FC|P_P)
#define IF_OVERHEAD_DISABLE 0
#define IF_OVERHEAD_ENABLE 1
```

```
#define IF_OVERHEAD_DEFAULT IF_OVERHEAD_DISABLE

#define FRM_LOS_DEFECT_P (0x0200|P_P)
#define FRM_LOS_DEFECT_CLEARED 0
#define FRM_LOS_DEFECT_DECLARED 1

#define FRM_LOS_FAILURE_P (0x0204|P_P)
#define FRM_LOS_FAILURE_CLEARED 0
#define FRM_LOS_FAILURE_DECLARED 1

#define FRM_OOF_DEFECT_P (0x0208|P_P)
#define FRM_OOF_DEFECT_CLEARED 0
#define FRM_OOF_DEFECT_DECLARED 1

#define FRM_LOF_FAILURE_P (0x020C|P_P)
#define FRM_LOF_FAILURE_CLEARED 0
#define FRM_LOF_FAILURE_DECLARED 1

#define FRM_AIS_DEFECT_P (0x0210|P_P)
#define FRM_AIS_DEFECT_CLEARED 0
#define FRM_AIS_DEFECT_DECLARED 1

#define FRM_AIS_FAILURE_P (0x0214|P_P)
#define FRM_AIS_FAILURE_CLEARED 0
#define FRM_AIS_FAILURE_DECLARED 1

#define FRM_IDLE_RCVD_P (0x0218|P_P)
#define FRM_IDLE_RCVD_CLEARED 0
#define FRM_IDLE_RCVD_DECLARED 1

#define FRM_RDI_DEFECT_P (0x021C|P_P)
#define FRM_RDI_DEFECT_CLEARED 0
#define FRM_RDI_DEFECT_DECLARED 1
```

```
#define FRM_RDI_FAILURE_P (0x0220|P_P)
#define FRM_RDI_FAILURE_CLEARED 0
#define FRM_RDI_FAILURE_DECLARED 1

#define FEAC_ALARM_P (0x0224|P_P)
#define FEAC_ALARM_DS3_SA 0x32FF
#define FEAC_ALARM_DS3_LOS 0x1CFF
#define FEAC_ALARM_DS3_OOF 0x00FF
#define FEAC_ALARM_DS3_AIS 0x2CFF
#define FEAC_ALARM_DS3_IDLE 0x34FF
#define FEAC_ALARM_DS3_NSA 0x1EFF
#define FEAC_ALARM_COM_NSA 0x3AFF
#define FEAC_ALARM_NONE 0xFFFF

#define PLCP_OOF_DEFECT_P (0x0228|P_P)
#define PLCP_OOF_DEFECT_CLEARED 0
#define PLCP_OOF_DEFECT_DECLARED 1

#define PLCP_RAI_DEFECT_P (0x022C|P_P)
#define PLCP_RAI_DEFECT_CLEARED 0
#define PLCP_RAI_DEFECT_DECLARED 1

#define G804_LCD_P (0x0230|P_P)
#define G804_LCD_CLEARED 0
#define G804_LCD_DECLARED 1

#define SSM_RX_CHNG_CNT_P (0x0244|P_P)

#define SSM_TX_DATA_P (0x0248|P_P)

#define SSM_RX_CMP_DATA_P (0x024C|P_P)

#define SSM_RX_STATUS_P (0x0250|P_P)
#define SSM_RX_STATUS_UNSTABLE 0 /* Received SSM data is unstable.*/
```

#define SSM_RX_STATUS_MATCH 1 /* New received SSM data matches preset receive data.*/

#define SSM_RX_STATUS_MISMATCH 2 /* New received SSM data does not match preset receive data.*/

#define SSM_RX_DATA_P (0x0254|P_P)

#define FEAC_ON_LOS_FAIL_P (0x0260|P_P)
#define FEAC_ON_LOS_FAIL_DISABLE 0
#define FEAC_ON_LOS_FAIL_ENABLE 1

#define FEAC_ON_LOF_FAIL_P (0x0264|P_P)
#define FEAC_ON_LOF_FAIL_DISABLE 0
#define FEAC_ON_LOF_FAIL_ENABLE 1

#define FEAC_ON_AIS_FAIL_P (0x0268|P_P)
#define FEAC_ON_AIS_FAIL_DISABLE 0
#define FEAC_ON_AIS_FAIL_ENABLE 1

#define FEAC_ON_IDLE_RCV_P (0x026C|P_P)
#define FEAC_ON_IDLE_RCV_DISABLE 0
#define FEAC_ON_IDLE_RCV_ENABLE 1

#define BERT_CONTROL_P (0x0270|P_P) /* start/stop BERT (transmit/receive) */
#define BERT_CONTROL_START 0
#define BERT_CONTROL_STOP 1

#define BERT_PATTERN_P (0x0274|P_P) /* selects the pattern */
#define BERT_PATTERN_2P15 0
#define BERT_PATTERN_2P20 1
#define BERT_PATTERN_2P20_QRSS 2
#define BERT_PATTERN_2P23 3
#define BERT_PATTERN_ALL_ONES 4
#define BERT_PATTERN_ALL_ZEROS 5

```
#define BERT_PATTERN_TOGGLE 6
#define BERT_PATTERN_GEN_RAND 7
#define BERT_PATTERN_GEN_FIXED 8
#define BERT_PATTERN_DEFAULT BERT_PATTERN_TOGGLE
#define BERT_INTERVAL_P (0x0278|P_P)  /* selects the measurement interval */

#define BERT_GEN_LENGTH_P (0x027C|P_P)  /* Transmit/Receive pattern length */

#define BERT_GEN_FBTAP_P (0x0280|P_P)  /* Transmit/Receive feedback length */

#define BERT_GEN_FIXPAT_32 (0x0284|P_32)  /* set the pattern word */

#define BERT_GEN_INVERT_P (0x0288|P_P)  /* Invert transmit output */
#define BERT_GEN_INVERT_DISABLE 0 /* No inversion */
#define BERT_GEN_INVERT_ENABLE 1 /* Invert pattern generator output */

#define BERT_GEN_TXER_P          (0x28C|P_P)   /* Transmit error insertion rate */
#define BERT_GEN_TXER_NONE           0         /* No error */
#define BERT_GEN_TXER_3              3         /* 1 in 10^3 */
#define BERT_GEN_TXER_4              4         /* 1 in 10^4 */
#define BERT_GEN_TXER_5              5         /* 1 in 10^5 */
#define BERT_GEN_TXER_6              6         /* 1 in 10^6 */
#define BERT_GEN_TXER_7              7         /* 1 in 10^7 */
#define BERT_GEN_TXER_SINGLE         8         /* single bit error */

#define DEVICE_STATUS_32 (0x02F0|P_32)

#define GPIO_0_CTRL_P (0x0F00|P_P)
#define GPIO_1_CTRL_P (0x0F04|P_P)
#define GPIO_2_CTRL_P (0x0F08|P_P)
#define GPIO_3_CTRL_P (0x0F0C|P_P)
#define GPIO_4_CTRL_P (0x0F10|P_P)
#define GPIO_5_CTRL_P (0x0F14|P_P)
#define GPIO_6_CTRL_P (0x0F18|P_P)
```

```
#define GPIO_7_CTRL_P (0x0F1C|P_P)
#define GPIO_ALL_CTRL_P (0x0F20|P_P)

#define GPIO_CTRL_IN 1
#define GPIO_CTRL_IN_PU 2
#define GPIO_CTRL_IN_PD 3
#define GPIO_CTRL_OUT 4
#define GPIO_CTRL_OUT_OD 5
#define GPIO_CTRL_OUT_OD_PU 6
#define GPIO_CTRL_LED_STATUS 255
#define GPIO_CTRL_DEFAULT GPIO_CTRL_IN_PU

#define GPIO_VALUE_P (0x0F30|P_P)
#define GPIO_MASK_P (0x0F34|P_P)

#define GPIO_0_LED_STAT_32 (0x0F40|P_32)
#define GPIO_1_LED_STAT_32 (0x0F44|P_32)
#define GPIO_2_LED_STAT_32 (0x0F48|P_32)
#define GPIO_3_LED_STAT_32 (0x0F4C|P_32)
#define GPIO_4_LED_STAT_32 (0x0F50|P_32)
#define GPIO_5_LED_STAT_32 (0x0F54|P_32)
#define GPIO_6_LED_STAT_32 (0x0F58|P_32)
#define GPIO_7_LED_STAT_32 (0x0F5C|P_32)

#define GPIO_0_LED_STAT_DEFAULT 0x00000800  /* OOS */
#define GPIO_1_LED_STAT_DEFAULT 0x00400000  /* local loop */
#define GPIO_2_LED_STAT_DEFAULT 0x00000080  /* remote loop */
#define GPIO_3_LED_STAT_DEFAULT 0x00020000  /* OOF defect */
#define GPIO_4_LED_STAT_DEFAULT 0x00000001  /* no alarm */
#define GPIO_5_LED_STAT_DEFAULT 0x00040000  /* RDI defect */
#define GPIO_6_LED_STAT_DEFAULT 0x00080000  /* LOS defect */
#define GPIO_7_LED_STAT_DEFAULT 0x00010000  /* AIS defect */

#define STAT_FRAMR_TOTAL_P (0x1000|P_P)
```

```
#define STAT_FRAMR_TOTAL_NO_RPT 0
#define STAT_FRAMR_TOTAL_RPT 1
#define STAT_FRAMR_TOTAL_RPT_CLEAR 2
#define STAT_FRAMR_TOTAL_DEFAULT STAT_FRAMR_TOTAL_NO_RPT

#define STAT_PLCP_TOTAL_P (0x1004|P_P)
#define STAT_PLCP_TOTAL_NO_RPT 0
#define STAT_PLCP_TOTAL_RPT 1
#define STAT_PLCP_TOTAL_RPT_CLEAR 2
#define STAT_PLCP_TOTAL_DEFAULT STAT_PLCP_TOTAL_NO_RPT

#define STAT_ATM_TOTAL_P (0x1008|P_P)
#define STAT_ATM_TOTAL_NO_RPT 0
#define STAT_ATM_TOTAL_RPT 1
#define STAT_ATM_TOTAL_RPT_CLEAR 2
#define STAT_ATM_TOTAL_DEFAULT STAT_ATM_TOTAL_NO_RPT

#define STAT_HDLC_TOTAL_P (0x100C|P_P)
#define STAT_HDLC_TOTAL_NO_RPT 0
#define STAT_HDLC_TOTAL_RPT 1
#define STAT_HDLC_TOTAL_RPT_CLEAR 2
#define STAT_HDLC_TOTAL_DEFAULT STAT_PPP_TOTAL_NO_RPT

#define STAT_BERT_TOTAL_P (0x1010|P_P)
#define STAT_BERT_TOTAL_NO_RPT 0
#define STAT_BERT_TOTAL_RPT 1
#define STAT_BERT_TOTAL_RPT_CLEAR 2
#define STAT_BERT_TOTAL_DEFAULT STAT_BERT_TOTAL_NO_RPT

#define STAT_FRAMR_1SEC_P (0x1080|P_P)
#define STAT_FRAMR_1SEC_NO_RPT 0
#define STAT_FRAMR_1SEC_RPT 1
#define STAT_FRAMR_1SEC_DEFAULT STAT_FRAMR_1SEC_NO_RPT
```

```
#define STAT_PLCP_1SEC_P (0x1084|P_P)
#define STAT_PLCP_1SEC_NO_RPT 0
#define STAT_PLCP_1SEC_RPT 1
#define STAT_PLCP_1SEC_DEFAULT STAT_PLCP_1SEC_NO_RPT

#define STAT_ATM_1SEC_P (0x1088|P_P)
#define STAT_ATM_1SEC_NO_RPT 0
#define STAT_ATM_1SEC_RPT 1
#define STAT_ATM_1SEC_DEFAULT STAT_ATM_1SEC_NO_RPT

#define STAT_HDLC_1SEC_P (0x108C|P_P)
#define STAT_HDLC_1SEC_NO_RPT 0
#define STAT_HDLC_1SEC_RPT 1
#define STAT_HDLC_1SEC_DEFAULT STAT_PPP_1SEC_NO_RPT

#define STAT_BERT_1SEC_P (0x1090|P_P)
#define STAT_BERT_1SEC_NO_RPT 0
#define STAT_BERT_1SEC_RPT 1
#define STAT_BERT_1SEC_DEFAULT STAT_BERT_1SEC_NO_RPT

#define STAT_FRAMR_T1231_P (0x1100|P_P)
#define STAT_FRAMR_T1231_NO_RPT 0
#define STAT_FRAMR_T1231_RPT 1
#define STAT_FRAMR_T1231_RPT_CLEAR 2
#define STAT_FRAMR_T1231_DEFAULT STAT_FRAMR_T1231_NO_RPT

#define STAT_TIMESTAMP_P (0x17FC|P_P)

#define RPT_FRAMR_TOTAL_32 (0x1800|P_32)
#define RPT_FRAMR_1SEC_32 (0x1880|P_32)
#define RPT_PLCP_TOTAL_32 (0x1804|P_32)
#define RPT_PLCP_1SEC_32 (0x1884|P_32)
#define RPT_ATM_TOTAL_32 (0x1808|P_32)
#define RPT_ATM_1SEC_32 (0x1888|P_32)
```

```
#define RPT_HDLC_TOTAL_32 (0x180C|P_32)
#define RPT_HDLC_1SEC_32 (0x188C|P_32)
#define RPT_BERT_TOTAL_32 (0x1810|P_32)
#define RPT_BERT_1SEC_32 (0x1890|P_32)
#define RPT_FRAMR_T1231_32 (0x1900|P_32)

#define READ_BLOCK (0x10A0|P_P)
#define READ_BLOCK_MODULE_CFG 0
#define READ_BLOCK_LINK_MAINT 1

#define DJAT_RECEIVE_P (0x10B0|P_P)
#define DJAT_RECEIVE_DISABLE 0
#define DJAT_RECEIVE_ENABLE 1
#define DJAT_RECEIVE_DEFAULT DJAT_RECEIVE_DISABLE

#define DJAT_RX_REF_P (0x10B4|P_P)
#define DJAT_RX_REF_RSYNC_PLL 0
#define DJAT_RX_REF_RCLK_PLL 1
#define DJAT_RX_REF_DEFAULT DJAT_RX_REF_RCLK_PLL

#define DJAT_RX_LOSMODE_P (0x10B8|P_P)
#define DJAT_RX_LOSMODE_AUTO_REF1 0
#define DJAT_RX_LOSMODE_AUTO_REF2 1
#define DJAT_RX_LOSMODE_NON_AUTO 2
#define DJAT_RX_LOSMODE_DEFAULT DJAT_RX_LOSMODE_AUTO_REF2

#define DJAT_TRANSMIT_P (0x10BC|P_P)
#define DJAT_TRANSMIT_DISABLE 0
#define DJAT_TRANSMIT_ENABLE 1
#define DJAT_TRANSMIT_DEFAULT DJAT_TRANSMIT_DISABLE

#define DJAT_TX_LOSMODE_P (0x10C0|P_P)
#define DJAT_TX_LOSMODE_AUTO_REF1 0
#define DJAT_TX_LOSMODE_AUTO_REF2 1
```

```
#define DJAT_TX_LOSMODE_NON_AUTO 2
#define DJAT_TX_LOSMODE_DEFAULT DJAT_TX_LOSMODE_AUTO_REF2

#define FRM_TX_RDI_P (0x10D0|P_P)
#define FRM_TX_RDI_INACTIVE 0
#define FRM_TX_RDI_ACTIVE 1
#define FRM_TX_RDI_DEFAULT FRM_TX_RDI_INACTIVE

#define PLL_RX_CHARAC_P (0x10E0|P_P)
#define PLL_RX_CHARAC_P_ELEMENT 0
#define PLL_RX_CHARAC_PI_ELEMENT 1
#define PLL_RX_CHARAC_DEFAULT PLL_RX_CHARAC_P_ELEMENT

#define PLL_TX_CHARAC_P (0x10F0|P_P)
#define PLL_TX_CHARAC_P_ELEMENT 0
#define PLL_TX_CHARAC_PI_ELEMENT 1
#define PLL_TX_CHARAC_DEFAULT PLL_TX_CHARAC_P_ELEMENT
```

w w w . i n f i n e o n . c o m

Published by Infineon Technologies AG