PM7381

FREEDM™-32A672

FRAME ENGINE AND DATALINK MANAGER 32A672

PROGRAMMER'S GUIDE

PROPRIETARY AND CONFIDENTIAL

ADVANCE

ISSUE 1: JUNE 1999

CONTENTS

LIST OF FIGURES

1 INTRODUCTION

1.1 Scope

The FREEDM-32A672 Programmer's Guide is intended to describe the configurable features and operation of a FREEDM-32A672 from a programmer's perspective. This document may not cover all applications of the FREEDM-32A672. Please contact a PMC-Sierra Applications Engineer for specific uses not covered in this document.

This document is a supplement to the FREEDM-32A672 Longform Datasheet[1]. Both documents should be studied together to interface the FREEDM-32A672 to an embedded processor. In case of a discrepancy between the Programmer's Guide and the Longform Datasheet, the Longform Datasheet shall always be considered correct.

1.2 Target Audience

The target audience for this document is programmers who will be writing software to configure and control the FREEDM-32A672.

This document has been prepared for readers with some prior knowledge of the HDLC protocol.

1.3 Numbering Conventions

The following numbering conventions are used throughout this document:

binary	011 1010B, 011
decimal	129, 6, 12
hexadecimal	0x1FE2, 09FH

1.4 Register Description

Unless specified otherwise, FREEDM-32A672 registers are described using the convention **REGISTER_NAME** (address in FREEDM-32A672). There is only one register space that can be addressed on a FREEDM-32A672, and it consists of the normal mode microprocessor accessible registers.

1.4.1 Normal Mode Registers

Normal mode registers are used to configure, monitor and control the operation of the FREEDM-32A672. Registers must be accessed as 16-bit values with a DWORD aligned address. For all register descriptions, the hexadecimal register number indicates the address in the FREEDM-32A672 when accesses are made using the external microprocessor. A register value is accessed through an external microprocessor, and has the following characteristics:

- Writing values into unused register bits has no effect. However, to ensure software compatibility with future versions of the product, unused register bits should be written with logic zero. Reading back unused bits can produce either a logic one or a logic zero; hence, unused register bits should be masked off by software when read.
- Except where noted, all configuration bits that can be written into can also be read back. This allows the processor controlling the FREEDM-32A672 to determine the programming state of the block.
- Writable normal mode registers are cleared to logic zero upon reset unless otherwise noted.
- Writing into read-only normal mode register bit locations does not affect FREEDM-32A672 operation unless otherwise noted.
- Certain register bits are reserved. These bits are associated with megacell functions that are unused in this application. To ensure that the FREEDM-32A672 operates as intended, reserved register bits must only be written with their default values. Similarly, writing to reserved registers should be avoided.

2 REFERENCES

1. PMC-990263, PMC-Sierra, Inc., "Frame Engine and Data Link Manager 32A672" Longform Datasheet, February 1999, Issue 1.
2. PMC-990262, PMC-Sierra, Inc., "Frame Engine and Data Link Manager 32P672" Longform Datasheet, February 1999, Issue 1.
3. PMC-960758, PMC-Sierra, Inc., "Frame Engine and Data Link Manager" Longform Datasheet, May 1998, Issue 5.

3 FREEDM-32A672 OVERVIEW

3.1 FREEDM-32A672 Summary

The PM7381 FREEDM-32A672 Frame Engine and Datalink Manager is an advanced data link layer processor that is ideal for applications such as IETF PPP interfaces for routers, TDM switches, Frame Relay switches and multiplexors, ATM switches and multiplexors, Internet/Intranet access equipment, packet-based DSLAM equipment, and Packet over SONET. The FREEDM-32A672 implements HDLC processing for a maximum of 672 bi-directional channels. The functional blocks of the FREEDM-32A672 are illustrated in Figure 1.

As many as 32 bi-directional serial links can be connected to the FREEDM-32A672. These are processed by the Receive Channel Assigner (RCAS672) and Transmit Channel Assigner (TCAS672) blocks. The RCAS672 and TCAS672 can be interfaced to H-MVIP, channelised T1/J1/E1, or unchannelised links.

The data stream at each serial port can be assigned to one or more of the FREEDM-32A672 channels. There are as many as 672 receive channels and 672 transmit channels available for assignment to unchannelised links, or to time-slots within a channelised T1/J1/E1 or H-MVIP link.

When configured for 2.048 Mbps H-MVIP operation, the FREEDM-32A672 partitions the 32 physical links into 4 logical groups of 8 links. Links in each logical group share a common clock and a common type 0 frame pulse in each direction.

When configured for 8.192 Mbps H-MVIP operation, the FREEDM-32A672 partitions the 32 physical links into 8 logical groups of 4 links. All links configured for 8.192 Mbps H-MVIP operation will share a common type 0 frame pulse, a common frame pulse clock and a common data clock.

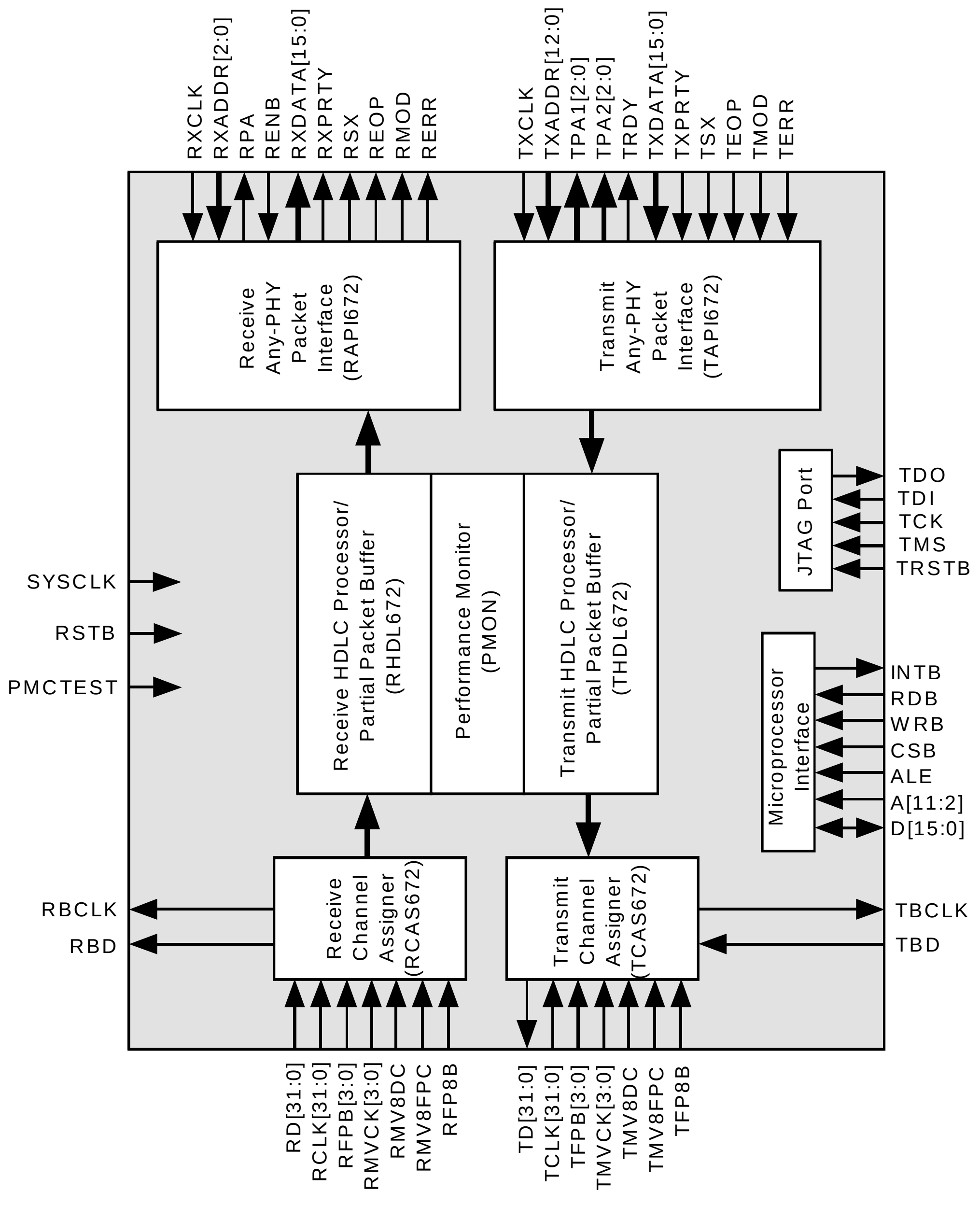
For channelised T1/J1/E1 links, the FREEDM-32A672 allows up to 672 bi-directional HDLC channels to be assigned to individual time-slots within a maximum of 32 independently timed T1/J1/E1 links.

For unchannelised links, the FREEDM-32A672 processes up to 32 bi-directional HDLC channels within 32 independently timed links.

Each data stream can be HDLC processed on a channelised basis within the Receive HDLC Processor / Partial Packet Buffer (RHDL672) and Transmit HDLC Processor / Partial Packet Buffer (THDL672). There is a 32 Kbyte buffer in the

RHDL672 and another 32 Kbyte buffer in the THDL672 that must be assigned to FREEDM-32A672 channels to serve as channel FIFO's. Each buffer is a group of 2048 blocks with 16 bytes per block, and a minimum of 3 blocks must be assigned to a channel during provisioning. This allows for flexible assignment of a channel FIFO based on the expected data rate for the channel.

Figure 1 – FREEDM-32A672 Block Diagram



Alternatively, the RHDL672 and THDL672 can provision a channel as transparent, in which case, the raw data stream is transferred without HDLC processing.

The Receive Any-PHY Interface (RAPI672) provides a low latency path for transferring data out of the partial packet buffer in the RHDL672 and onto the Receive Any-PHY Packet Interface (Rx APPI). The RAPI672 contains a FIFO block for latency control as well as to segregate the APPI timing domain from the SYSCLK timing domain. The RAPI672 contains the necessary logic to manage and respond to device polling from an upper layer device. The RAPI672 also provides the upper layer device with status information on a per packet basis.

The Transmit Any-PHY Interface (TAPI672) provides a low latency path for transferring data from the Transmit Any-PHY Packet Interface (Tx APPI) into the partial packet buffer in the THDL672. The TAPI672 contains a FIFO block for latency control as well as to segregate the APPI timing domain from the SYSCLK timing domain. The TAPI672 contains the necessary logic to manage and respond to channel polling from an upper layer device.

The PMON block provides performance monitor counts for a number of events. These counters can be read via the microprocessor interface and provides a means for the host software to accumulate performance statistics.

Channelised T1/J1/E1 and unchannelised links can be individually placed in line loopback. In addition to the line loopback capability, the FREEDM-32A672 provides a BERT port for attachment to BERT hardware. Under software control this port can be connected to any one of the 32 bi-directional links for additional diagnostic testing. The line loopback and BERT features are not supported for H-MVIP links. Finally, there is an internal diagnostic loopback configuration for each channel which can be used to diagnose FREEDM-32A672 operation on a per channel basis.

3.2 Any-PHY Packet Interface

The FREEDM-32A672 provides a low latency “Any-PHY” Packet Interface (APPI) to allow an external controller direct access into the 32 Kbyte partial packet buffers. Up to seven FREEDM-32A672 devices may share a single APPI. For each of the transmit and receive APPI, the external controller is the master of each FREEDM-32A672 device sharing the APPI from the point of view of device selection. The external controller is also the master for channel selection in the transmit direction. In the receive direction, however, each FREEDM-32A672 device retains control over selection of its respective channels. The transmit and receive APPI is made up of three groups of functional signals – polling, selection and data transfer. The polling signals are used by the external controller to interrogate the status of the transmit and receive 32 Kbyte partial packet buffers. The selection signals are used by the external controller to select a FREEDM-32A672 device, or a channel within a FREEDM-32A672 device, for data transfer. The data transfer signals provide a means of transferring data across the APPI between the external controller and a FREEDM-32A672 device.

In the receive direction, polling and selection are done at the device level. Polling is not decoupled from selection, as the receive address pins serve as both a device poll address and to select a FREEDM-32A672 device. In response to a positive poll, the external controller may select that FREEDM-32A672 device for data transfer. Once selected, the FREEDM-32A672 prepends an in-band channel address to each partial packet transfer across the receive APPI to associate the data with a channel. A FREEDM-32A672 must not be selected after a negative poll response.

In the transmit direction, polling is done at the channel level. Polling is completely decoupled from selection. To increase the polling bandwidth, up to two channels may be polled simultaneously. The polling engine in the external controller runs independently of other activity on the transmit APPI. In response to a positive poll, the external controller may commence partial packet data transfer across the transmit APPI for the successfully polled channel of a FREEDM-32A672 device. The external controller must prepend an in-band channel address to each partial packet transfer across the transmit APPI to associate the data with a channel.

Detailed information on configuring the RAPI672 and the TAPI672 can be found in section 6 of this document.

4 INTERRUPT ARCHITECTURE

This section provides an overview of the FREEDM-32A672 interrupt architecture. Detailed information on the individual interrupts is available in the Longform Datasheet[1].

The FREEDM-32A672 provides a number of individual interrupts which are identified as 'I' bits within the **FREEDM-32A672 Master Interrupt Status** (0x008) register. When an interrupt source becomes active, the 'I' bit is set and remains set until the **FREEDM-32A672 Master Interrupt Status** (0x008) register is read.

The FREEDM-32A672 provides interrupts to the microprocessor via the INTB pin of the FREEDM-32A672. The INTB pin is gated by the **FREEDM-32A672 Master Interrupt Enable** (0x004) register. This register contains 'E' bits which can mask the 'I' bit from causing an interrupt on the INTB pin of the FREEDM-32A672. When the 'E' and 'I' bits of an interrupt source are both high, then the INTB pin is active. When the 'E' bit is low, the interrupt source will not activate the INTB pin, regardless of the 'I' bit status. However, the 'I' bit remains valid when interrupts are disabled and may be polled to detect the various events.

The complete list of 'I' bits and 'E' bits is shown below:

'E' Bit	'I' Bit	Description
RFCSEE	RFCSEI	Receive FCS Error
RABRTE	RABRTI	Receive Abort
RPFEE	RPFEI	Receive Packet Format Error
RFOVRE	RFOVRI	Receive FIFO Overrun Error
TPRTYE	TPRTYI	Transmit Parity Error
TUNPVE	TUNPVI	Transmit Unprovisioned Error
TFOVRE	TFOVRI	Transmit FIFO Overflow Error
TFUDRE	TFUDRI	Transmit FIFO Underflow Error

5 CONFIGURING THE SERIAL LINKS

Each of the 32-bidirectional links is controlled via the RCAS672 and the TCAS672 blocks of the FREEDM-32A672. The RCAS672 controls the receive data stream while the TCAS672 controls the transmit data stream.

The RCAS672 extracts data bits from each of 32 receive links, based on the link configuration specified in the Normal Mode Register Space. The RCAS672 can align data to a link's gapped clock in channelised mode, extract unaligned data in unchannelised mode, or extract byte synchronized data in unchannelised mode.

The TCAS672 sources data bits onto each of 32 transmit links, based on the link configuration specified in the Normal Mode Register Space. The TCAS672 can provide data aligned to a link's gapped clock in channelised mode, provide unaligned data in unchannelised mode, or provide byte synchronized data in unchannelised mode.

Each of the serial link configurations is discussed separately in the following sections.

5.1 Channelised T1/J1 Links

The receive bit stream is input on RD[n], and the RCLK[n] input is a 1.544 MHz clock that provides bit timing. The clock is gapped to align time-slot 1 of the channelised T1/J1 receive link (see Figure 2).

The transmit bit stream is output on TD[n], and the TCLK[n] input is a 1.544 MHz clock that provides bit timing. The clock is gapped to align time-slot 1 of the channelised T1/J1 transmit link (see Figure 3).

In each direction, transmit or receive, a T1/J1 frame consists of 24 time-slots or bytes which are mapped to one or more channels of the FREEDM-32A672. The data stream of each direction is processed and clocked independently.

Figure 2 – Channelised T1/J1 Receive Link Timing

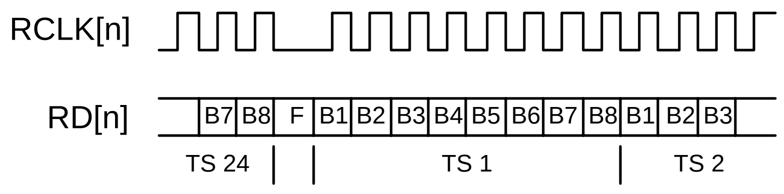
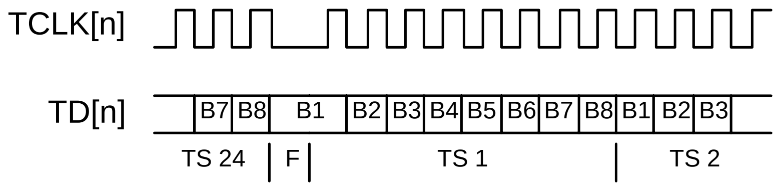


Figure 3 – Channelised T1/J1 Transmit Link Timing



To configure the RCAS672 and TCAS672 to interface to channelised T1/J1 links, the following bits are written:

Bit	Register	Value
MODE[2:0]	**RCAS Link #n Configuration** (0x180 – 0x1FC)	001
BSYNC	**RCAS Link #n Configuration** (0x180 – 0x188)	X
FTHRES[6:0]	**RCAS Framing Bit Threshold** (0x108)	0x25
MODE[2:0]	**TCAS Link #n Configuration** (0x480 – 0x4FC)	001
BSYNC	**TCAS Link #n Configuration** (0x480 – 0x488)	X
FTHRES[6:0]	**TCAS Framing Bit Threshold** (0x408)	0x25
FDATA[7:0]	**TCAS Idle Time-slot Fill Data** (0x40C)	0xFF

Notes:

- The **RCAS Link #n Configuration** register must be chosen from the range 0x180 through 0x1FC corresponding to the link being configured for channelised T1/J1 operation. The BSYNC bit is present only in registers with $0 \le n \le 2$. The BSYNC value is ignored for the channelised mode of operation.

- The **TCAS Link #n Configuration** register must be chosen from the range 0x480 through 0x4FC corresponding to the link being configured for channelised T1/J1 operation. The BSYNC bit is present only in registers with $0 \le n \le 2$. The BSYNC value is ignored for the channelised mode of operation.

- The FTHRES[6:0] value assumes a SYSCLK of 40Mhz. For other values of SYSCLK the framing threshold value is $1.5 * f_{SYSCLK} / 1.544\ MHz$. (This value ensures that the threshold is suitable for T1/J1 and E1 links).

- The FDATA[7:0] bits of the **TCAS Idle Time-slot Fill Data** (0x40C) register, and the FTHRES[6:0] bits of the **RCAS Framing Bit Threshold** (0x108) / **TCAS Framing Bit Threshold** (0x408) registers, affect all links of a

FREEDM-32A672. The programmer should ensure that these values are suitable for all links attached to a FREEDM-32A672.

5.2 Channelised E1 Links

The receive bit stream is input on RD[n], and the RCLK[n] input is a 2.048 MHz clock that provides bit timing. The clock is gapped to align time-slot 1 of the channelised E1 receive link (see Figure 4).

The transmit bit stream is output on TD[n], and the TCLK[n] input is a 2.048 MHz clock that provides bit timing. The clock is gapped to align time-slot 1 of the channelised E1 transmit link (see Figure 5).

In each direction, transmit or receive, an E1 frame consists of 31 time-slots or bytes which are mapped to one or more channels of the FREEDM-32A672. The data stream of each direction is processed and clocked independently.

Figure 4 – Channelised E1 Receive Link Timing

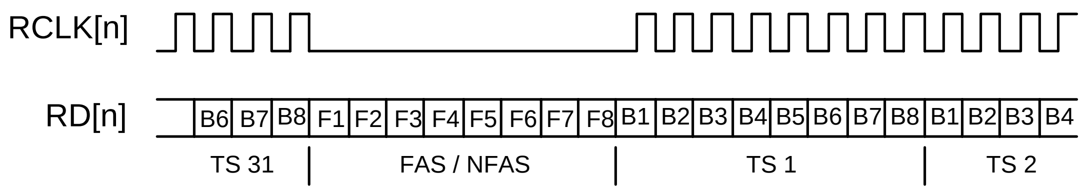
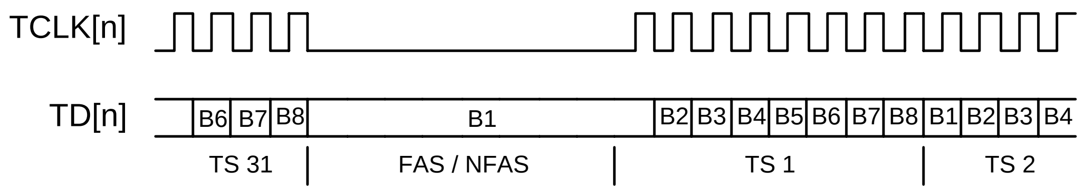

RCLK[n]
RD[n]
B6 B7 B8 F1 F2 F3 F4 F5 F6 F7 F8 B1 B2 B3 B4 B5 B6 B7 B8 B1 B2 B3 B4
TS 31
FAS / NFAS
TS 1
TS 2

Figure 5 – Channelised E1 Transmit Link Timing



To configure the RCAS672 and TCAS672 to interface to channelised E1 links, the following bits are written:

Bit	Register	Value
MODE[2:0]	**RCAS Link #n Configuration** (0x180 – 0x1FC)	010
BSYNC	**RCAS Link #n Configuration** (0x180 – 0x188)	X
FTHRES[6:0]	**RCAS Framing Bit Threshold** (0x108)	0x25

Bit	Register	Value
MODE[2:0]	**TCAS Link #n Configuration** (0x480 – 0x4FC)	010
BSYNC	**TCAS Link #n Configuration** (0x480 – 0x488)	X
FTHRES[6:0]	**TCAS Framing Bit Threshold** (0x408)	0x25
FDATA[7:0]	**TCAS Idle Time-slot Fill Data** (0x40C)	0xFF

Notes:

- The **RCAS Link #n Configuration** register must be chosen from the range 0x180 through 0x1FC corresponding to the link being configured for channelised E1 operation. The BSYNC bit is present only in registers with $0 \le n \le 2$. The BSYNC value is ignored for the channelised mode of operation.

- The **TCAS Link #n Configuration** register must be chosen from the range 0x480 through 0x4FC corresponding to the link being configured for channelised E1 operation. The BSYNC bit is present only in registers with $0 \le n \le 2$. The BSYNC value is ignored for the channelised mode of operation.

- Since channelised E1 links have a framing byte, its gap will always be longer than the gap for the T1/J1 framing bit. Therefore, the same FTHRES[6:0] value of 0x25 can be used. The FTHRES[6:0] value assumes a SYSCLK of 40Mhz. For other values of SYSCLK the framing threshold value is $1.5 * f_{SYSCLK} / 1.544$ *MHz.* (This value ensures the threshold is suitable for T1/J1 and E1 links).

- The FDATA[7:0] bits of the **TCAS Idle Time-slot Fill Data** (0x40C) register, and the FTHRES[6:0] bits of the **RCAS Framing Bit Threshold** (0x108) / **TCAS Framing Bit Threshold** (0x408) registers, affect all links of a FREEDM-32A672. The programmer should ensure that these values are suitable for all links attached to a FREEDM-32A672.

5.3 Unchannelised Links with Byte Synchronization

The bit stream of the unchannelised receive link that is attached to the RD[n] input is sampled on rising edges of the RCLK[n] input (see Figure 6). The receive data is byte aligned to the gapped RCLK[n] input, with the most significant bit of the byte clocked in following the gap.

The unchannelised transmit link is attached to the TD[n] output and is driven on falling edges of the TCLK[n] input (see Figure 7). The transmit data is byte aligned to the gapped TCLK[n] input, with the most significant bit of the byte clocked out during the gap.

This mode of operation is only available for links 0 through 2 of a FREEDM-32A672.

In each direction, transmit or receive, the data stream is processed and clocked independently.

Figure 6 – Unchannelised Receive Link Timing with Byte Synchronization

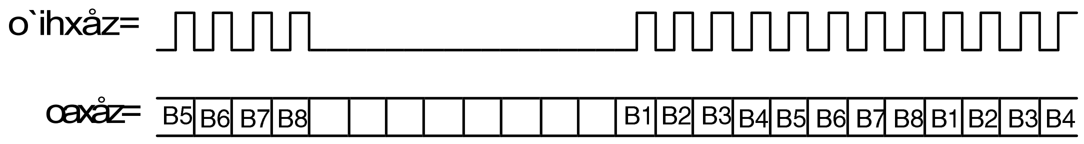
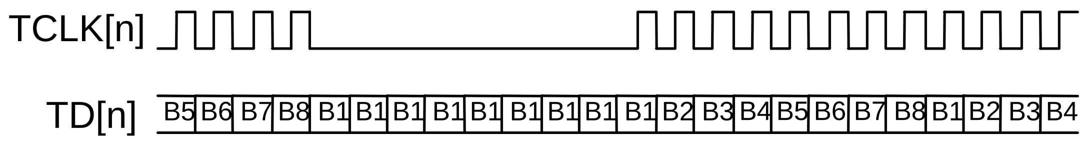


Figure 7 – Unchannelised Transmit Link Timing with Byte Synchronization


TCLK[n]
TD[n]
B5 B6 B7 B8 B1 B1 B1 B1 B1 B1 B1 B1 B1 B2 B3 B4 B5 B6 B7 B8 B1 B2 B3 B4

To configure the RCAS672 and TCAS672 to interface to byte synchronized unchannelised links, the following bits are written:

Bit	Register	Value
MODE[2:0]	**RCAS Link #n Configuration** (0x180 – 0x1FC)	000
BSYNC	**RCAS Link #n Configuration** (0x180 – 0x188)	1
FTHRES[6:0]	**RCAS Framing Bit Threshold** (0x108)	see note
MODE[2:0]	**TCAS Link #n Configuration** (0x480 – 0x4FC)	000
BSYNC	**TCAS Link #n Configuration** (0x480 – 0x488)	1
FTHRES[6:0]	**TCAS Framing Bit Threshold** (0x408)	see note
FDATA[7:0]	**TCAS Idle Time-slot Fill Data** (0x40C)	0xFF

Notes:

- The **RCAS Link #n Configuration** register must be chosen from the range 0x180 through 0x1FC corresponding to the link being configured. The unchannelised link with byte synchronization is only supported in registers with *0 ≤ n ≤ 2.*

- The **TCAS Link #n Configuration** register must be chosen from the range 0x480 through 0x4FC corresponding to the link being configured. The unchannelised link with byte synchronization is only supported in registers with *0 ≤ n ≤ 2.*
- The FTHRES[6:0] must be set based on the expected gap width of RCLK[n] or TCLK[n]. The reader should refer to the Longform Datasheet[1] on how to set this value.
- The FDATA[7:0] bits of the **TCAS Idle Time-slot Fill Data** (0x40C) register, and the FTHRES[6:0] bits of the **RCAS Framing Bit Threshold** (0x108) / **TCAS Framing Bit Threshold** (0x408) registers, affect all links of a FREEDM-32A672. The programmer should ensure that these values are suitable for all links attached to a FREEDM-32A672.

5.4 Unchannelised Links without Synchronization

The bit stream of the unchannelised receive link that is attached to the RD[n] input is sampled on rising edges of the RCLK[n] input (see Figure 8). The receive data is not aligned to the RCLK[n] input.

The unchannelised transmit link is attached to the TD[n] output and is driven on falling edges of the TCLK[n] input (see Figure 9). The transmit data is not aligned to the TCLK[n] input.

In each direction, transmit or receive, the data stream is processed and clocked independently.

Figure 8 – Unchannelised Receive Link Timing without Synchronization

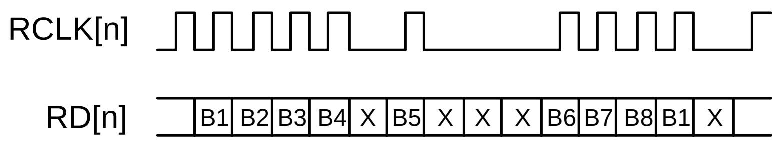
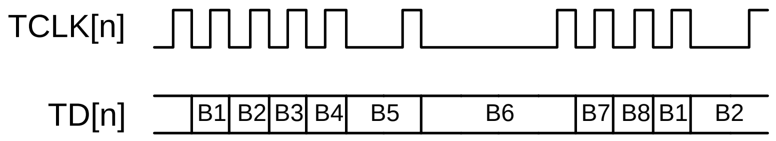


Figure 9 – Unchannelised Transmit Link Timing without Synchronization



To configure the RCAS672 and TCAS672 to interface to unchannelised links without synchronization, the following bits are written:

Bit	Register	Value
MODE[2:0]	**RCAS Link #n Configuration** (0x180 – 0x1FC)	000
BSYNC	**RCAS Link #n Configuration** (0x180 – 0x188)	0
FTHRES[6:0]	**RCAS Framing Bit Threshold** (0x108)	XXH
MODE[2:0]	**TCAS Link #n Configuration** (0x480 – 0x4FC)	000
BSYNC	**TCAS Link #n Configuration** (0x480 – 0x488)	0
FTHRES[6:0]	**TCAS Framing Bit Threshold** (0x408)	XXH
FDATA[7:0]	**TCAS Idle Time-slot Fill Data** (0x40C)	0xFF

Notes:

- The **RCAS Link #n Configuration** register must be chosen from the range 0x180 through 0x1FC corresponding to the link being configured for unchannelised operation.

- The **TCAS Link #n Configuration** register must be chosen from the range 0x480 through 0x4FC corresponding to the link being configured for unchannelised operation.

- The BSYNC bit must only be programmed for links where $0 \le n \le 2$. For other links, there is no BSYNC bit.

- The FTHRES[6:0] bits of the **RCAS Framing Bit Threshold** (0x108) / **TCAS Framing Bit Threshold** (0x408) registers have no effect on the operation of an unchannelised link with BSYNC low, or on unchannelised links with no BSYNC bit.

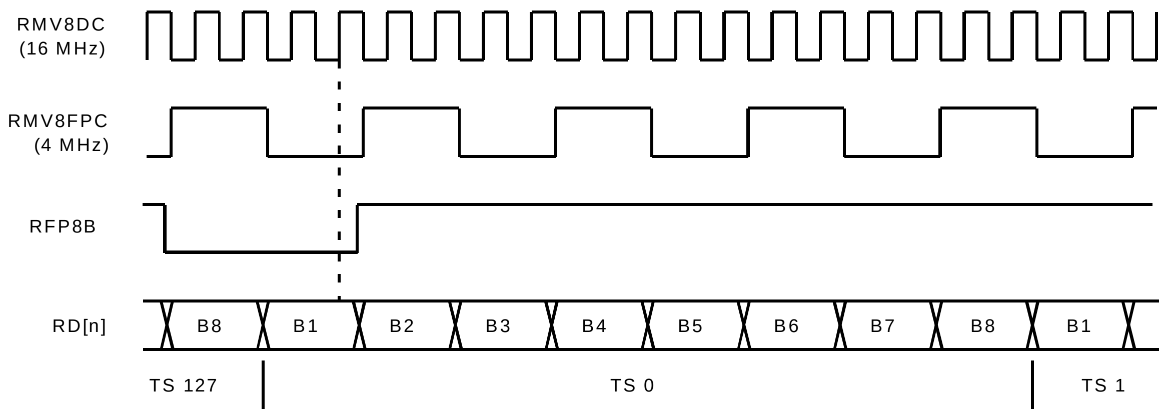
- The FDATA[7:0] bits of the **TCAS Idle Time-slot Fill Data** (0x40C) register, and the FTHRES[6:0] bits of the **RCAS Framing Bit Threshold** (0x108) / **TCAS Framing Bit Threshold** (0x408) registers, affect all links of a

FREEDM-32A672. The programmer should ensure that these values are suitable for all links attached to a FREEDM-32A672.

5.5 8.192 Mbps H-MVIP Links

The timing relationship of the receive data clock (RMV8DC), frame pulse clock (RMV8FPC), data (RD[n]) and frame pulse (RFP8B[n]) signals of a link configured for 8.192 Mbps H-MVIP operation with a type 0 frame pulse is shown in Figure 10. The falling edges of each RMV8FPC are aligned to a falling edge of the corresponding RMV8DC for 8.192 Mbps H-MVIP operation. The FREEDM-32A672 samples RFP8B low on the falling edge of RMV8FPC and references this point as the start of the next frame. The FREEDM-32A672 samples the data provided on RD[n] at the ¾ point of the data bit using the rising edge of RMV8DC as indicated for bit 1 (B1) of time-slot 0 (TS 0) in Figure 10. B1 is the most significant bit and B8 is the least significant bit of each octet. Time-slots can be ignored by setting the PROV bit in the corresponding word of the receive channel provision RAM in the RCAS672 block to low.

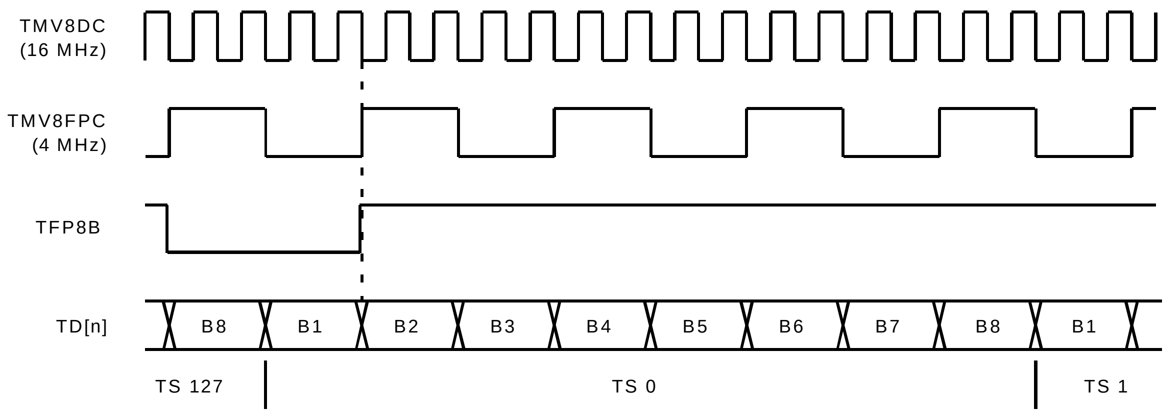
Figure 10 – Receive 8.192 Mbps H-MVIP Link Timing



The timing relationship of the transmit data clock (TMV8DC), frame pulse clock (TMV8FPC), data (TD[n]) and frame pulse (TFP8B) signals of a link configured for 8.192 Mbps H-MVIP operation with a type 0 frame pulse is shown in Figure 11. The falling edges of each TMV8FPC are aligned to a falling edge of the corresponding TMV8DC for 8.192 Mbps H-MVIP operation. The FREEDM-32A672 samples TFP8B low on the falling edge of TMV8FPC and references this point as the start of the next frame. The FREEDM-32A672 updates the data provided on TD[n] on every second falling edge of TMV8DC as indicated for bit 2 (B2) of time-slot 0 (TS 0) in Figure 11. The first bit of the next frame is updated on TD[n] on the falling TMV8DC clock edge for which TFP8B is also sampled low. B1 is the most significant bit and B8 is the least significant bit of each octet.

Time-slots that are not provisioned to belong to any channel (PROV bit in the corresponding word of the transmit channel provision RAM in the TCAS672 block set low) transmits the contents of the **TCAS Idle Time-slot Fill Data** (0x40C) register.

Figure 11 – Transmit 8.192 Mbps H-MVIP Link Timing



To configure the RCAS672 and TCAS672 to interface to 8.192 Mbps H-MVIP links, the following bits are written:

Bit	Register	Value
MODE[2:0]	**RCAS Link #n Configuration** (0x180 – 0x1FC)	111
BSYNC	**RCAS Link #n Configuration** (0x180 – 0x188)	X
FTHRES[6:0]	**RCAS Framing Bit Threshold** (0x108)	XXH
MODE[2:0]	**TCAS Link #n Configuration** (0x480 – 0x4FC)	111
BSYNC	**TCAS Link #n Configuration** (0x480 – 0x488)	X
FTHRES[6:0]	**TCAS Framing Bit Threshold** (0x408)	XXH
FDATA[7:0]	**TCAS Idle Time-slot Fill Data** (0x40C)	0xFF

Notes:

- When link 4m (0≤m≤7) is configured for operation in 8.192 Mbps H-MVIP mode (MODE[2:0]="111"), data cannot be received on links 4m+1, 4m+2 and 4m+3. However, links 4m+1, 4m+2 and 4m+3 must be configured for 8.192 Mbps H-MVIP mode for correct operation of the RCAS672. From a channel assignment point of view in the RCAS672 (Registers 0x100, 0x104), time-slots 0 through 31 of the H-MVIP link are treated as time-slots 0 through 31 of link 4m, time-slots 32 through 63 of the H-MVIP link are treated as time-slots 0 through 31 of link 4m+1, time-slots 64 through 95 of the H-MVIP link are treated as time-slots 0 through 31 of link 4m+2 and time-slots 96 through 127 of the H-MVIP link are treated as time-slots 0 through 31 of link 4m+3.

- When link 4m (0≤m≤7) is configured for operation in 8.192 Mbps H-MVIP mode, links 4m+1, 4m+2 and 4m+3 are driven with constant ones. However, links 4m+1, 4m+2 and 4m+3 must be configured for 8.192 Mbps H-MVIP mode for correct operation of the TCAS672. From a channel assignment point of view in the TCAS672 (Registers 0x40, 0x404), time-slots 0 through 31 of link 4m are mapped to time-slots 0 through 31 of the H-MVIP link, time-slots 0 through 31 of link 4m+1 are mapped to time-slots 32 through 63 of the H-MVIP link, time-slots 0 through 31 of link 4m+2 are mapped to time-slots 64 through 95 of the H-MVIP link and time-slots 0 through 31 of link 4m+3 are mapped to time-slots 96 through 127 of the H-MVIP link.

- The BSYNC value is ignored for links configured for H-MVIP.

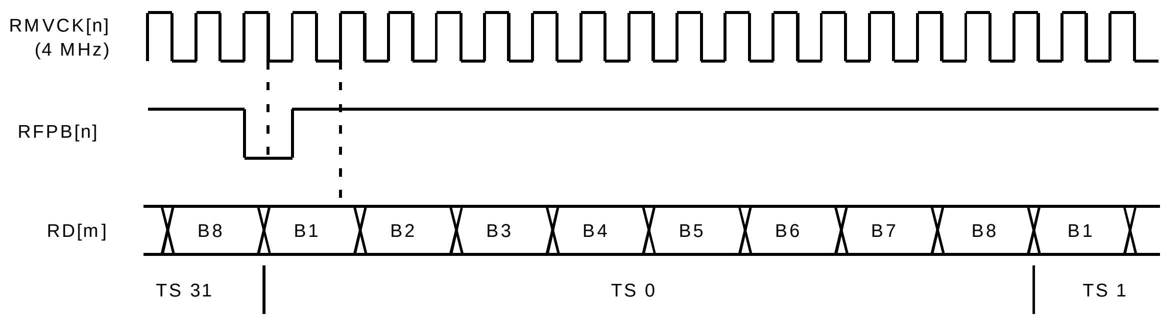
- The FTHRES[6:0] bits of the **RCAS Framing Bit Threshold** (0x108) / **TCAS Framing Bit Threshold** (0x408) registers have no effect on the operation of H-MVIP links.

- The FDATA[7:0] bits of the **TCAS Idle Time-slot Fill Data** (0x40C) register, and the FTHRES[6:0] bits of the **RCAS Framing Bit Threshold** (0x108) / **TCAS Framing Bit Threshold** (0x408) registers, affect all links of a FREEDM-32A672. The programmer should ensure that these values are suitable for all links attached to a FREEDM-32A672.

5.6 2.048 Mbps H-MVIP Links

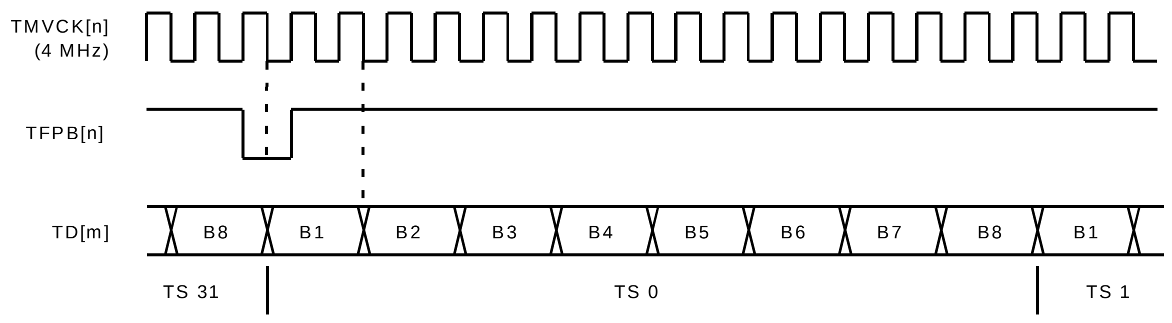
The timing relationship of the receive data clock (RMVCK[n]), data (RD[m], where 8n≤m≤8n+7) and frame pulse (RFPB[n]) signals of a link configured for 2.048 Mbps H-MVIP operation with a type 0 frame pulse is shown in Figure 12. The FREEDM-32A672 samples RFPB[n] low on the falling edge of the corresponding RMVCK[n] and references this point as the start of the next frame. The FREEDM-32A672 samples the data provided on RD[m] at the ¾ point of the data bit using the rising edge of the corresponding RMVCK[n] as indicated for bit 1 (B1) of time-slot 0 (TS 0) in Figure 12. B1 is the most significant bit and B8 is the least significant bit of each octet. Time-slots can be ignored by setting the PROV bit in the corresponding word of the receive channel provision RAM in the RCAS672 block to low.

Figure 12 – Receive 2.048 Mbps H-MVIP Link Timing



The timing relationship of the transmit data clock (TMVCK[n]), data (TD[m], where 8n≤m≤8n+7) and frame pulse (TFPB[n]) signals of a link configured for 2.048 Mbps H-MVIP operation with a type 0 frame pulse is shown in Figure 13. The FREEDM-32A672 samples TFPB[n] low on the falling edge of the corresponding TMVCK[n] and references this point as the start of the next frame. The FREEDM-32A672 updates the data provided on TD[m] on every second falling edge of the corresponding TMVCK[n] as indicated for bit 2 (B2) of time-slot 0 (TS 0) in Figure 13. The first bit of the next frame is updated on TD[n] on the falling TMVCK[n] clock edge for which TFPB[n] is also sampled low. B1 is the most significant bit and B8 is the least significant bit of each octet. Time-slots that are not provisioned to belong to any channel (PROV bit in the corresponding word of the transmit channel provision RAM in the TCAS672 block set low) transmits the contents of the **TCAS Idle Time-slot Fill Data** (0x40C) register.

Figure 13 – Transmit 2.048 Mbps H-MVIP Link Timing



To configure the RCAS672 and TCAS672 to interface to 2.048 Mbps H-MVIP links, the following bits are written:

Bit	Register	Value
MODE[2:0]	**RCAS Link #n Configuration** (0x180 – 0x1FC)	011

Bit	Register	Value
BSYNC	**RCAS Link #n Configuration** (0x180 – 0x188)	X
FTHRES[6:0]	**RCAS Framing Bit Threshold** (0x108)	XXH
MODE[2:0]	**TCAS Link #n Configuration** (0x480 – 0x4FC)	011
BSYNC	**TCAS Link #n Configuration** (0x480 – 0x488)	X
FTHRES[6:0]	**TCAS Framing Bit Threshold** (0x408)	XXH
FDATA[7:0]	**TCAS Idle Time-slot Fill Data** (0x40C)	0xFF

Notes:

- The **RCAS Link #n Configuration** register must be chosen from the range 0x180 through 0x1FC corresponding to the link being configured for 2.048 Mbps H-MVIP operation.
- The **TCAS Link #n Configuration** register must be chosen from the range 0x480 through 0x4FC corresponding to the link being configured for 2.048 Mbps H-MVIP operation.
- The BSYNC value is ignored for links configured for H-MVIP.
- The FTHRES[6:0] bits of the **RCAS Framing Bit Threshold** (0x108) / **TCAS Framing Bit Threshold** (0x408) registers have no effect on the operation of H-MVIP links.
- The FDATA[7:0] bits of the **TCAS Idle Time-slot Fill Data** (0x40C) register, and the FTHRES[6:0] bits of the **RCAS Framing Bit Threshold** (0x108) / **TCAS Framing Bit Threshold** (0x408) registers, affect all links of a FREEDM-32A672. The programmer should ensure that these values are suitable for all links attached to a FREEDM-32A672.

6 CONFIGURING THE ANY-PHY PACKET INTERFACE

The RAPI672 and TAPI672 blocks must be configured via the normal mode registers in order to enable the transferring of data between the partial packet buffers and the receive and transmit APPI.

6.1 Configuring the Receive Any-PHY Packet Interface (RAPI672)

The RAPI672 is configured by programming bits within the **RAPI Control** (0x580) register. The values programmed affect all receive channels. The default configuration is as follows:

Bit	Register	Value
BADDR[2:0]	**RAPI Control** (0x580)	111
ALL1ENB	**RAPI Control** (0x580)	1
Reserved	**RAPI Control** (0x580)	0
STATEN	**RAPI Control** (0x580)	0
ENABLE	**RAPI Control** (0x580)	0

The default indicates that the RAPI672 is disabled from responding to device selection.

Activation of the RAPI672

By default, the RAPI672 is disabled from responding to device selection. The ENABLE bit must be set to enable normal operation of the RAPI672. The encoding of this bit is:

ENABLE	Function
0	The RAPI672 will not respond to device selection.
1	The RAPI672 operates normally, and will respond to device selection.

Base Address of the Receive APPI

The base address bits (BADDR[2:0]) configure the address space occupied by the FREEDM-32A672 device for purposes of responding to receive polling and receive device selection. During polling, the BADDR[2:0] bits are used to respond to polling via the RXADDR[2:0] pins. During device selection, the

BADDR[2:0] are used to select a FREEDM-32A672 device, enabling it to accept data on the receive APPI. During data transfer, the RXDATA[15:13] pins of the prepended channel address reflect the BADDR[2:0] bits.

All Ones Enable

The All Ones Enable bit (ALL1ENB) permits the FREEDM-32A672 to respond to receive polling and device selection when BADDR[2:0] = '111'. The encoding of this bit is:

ALL1ENB	Function
0	The FREEDM-32A672 responds to receive polling and device selection when BADDR[2:0] = RXADDR[2:0] = '111'.
1	The FREEDM-32A672 regards the all-ones address as a null address and does not respond to receive polling and device selection when BADDR[2:0] = '111', regardless of the value of RXADDR[2:0].

Status Enable

The FREEDM-32A672 can be programmed to overwrite the receive data signal pins, RXDATA[7:0], of the final word of each packet transfer (REOP pin is high) with the status of packet reception when that packet is errored (RERR pin is high). The RAPI672 Status Enable bit enables this feature as follows:

STATEN	Function
0	The RAPI672 does not report detailed status information for an errored packet.
1	The RAPI672 overwrites RXDATA[7:0] of the final word of an errored packet with status information for that packet.

When STATEN = 1 and the REOP and RERR pins are both high, the status information is bit mapped on RXDATA[7:0] as follows:

Bit	Status
RXDATA[0] = 1	Channel FIFO overrun
RXDATA[1] = 1	Maximum packet length violation
RXDATA[2] = 1	FCS error
RXDATA[3] = 1	Non-octet aligned

Bit	Status
RXDATA[4] = 1	HDLC packet abort
RXDATA[7:5] = XH	Reserved

6.2 Configuring the Transmit Any-PHY Packet Interface (TAPI672)

The TAPI672 is configured by programming bits within the **TAPI Control** (0x600) register. The values programmed affect all transmit channels. The default configuration is as follows:

Bit	Register	Value
BADDR[2:0]	**TAPI Control** (0x600)	111
ALL1ENB	**TAPI Control** (0x600)	1
Reserved[1:0]	**TAPI Control** (0x600)	00
ENABLE	**TAPI Control** (0x600)	0
BLEN[7:0]	**TAPI Indirect Channel Data Register** (0x608)	0x00

The default indicates that data provided to the TAPI672 by the transmit APPI will be ignored.

Activation of the TAPI672

By default, data provided to the TAPI672 by the transmit APPI is ignored. The ENABLE bit must be set to enable normal operation of the TAPI672. The encoding of this bit is:

ENABLE	Function
0	The state machines in the TAPI672 are held in their idle state. The TAPI672 will complete the current data transfer and will respond to any further transactions on the transmit APPI normally (by setting TRDY high), but data provided will be ignored.
1	The TAPI672 operates normally. Data can be transferred from the transmit APPI to the partial packet buffer in the THDL672.

Base Address of the Receive APPI

The base address bits (BADDR[2:0]) configure the address space occupied by the FREEDM-32A672 device for purposes of responding to transmit polling and transmit data transfers. During polling, the TXADDR[12:10] pins are compared with the BADDR[2:0] bits to determine if the poll address identified by TXADDR[9:0] is intended for a channel in this FREEDM-32A672 device. During data transmission, the TXDATA[15:13] pins of the prepended channel address are compared with the BADDR[2:0] bits to determine if the data to follow is intended for this FREEDM-32A672 device.

All Ones Enable

The All Ones Enable bit (ALL1ENB) permits the FREEDM-32A672 to respond to transmit polling and device selection when BADDR[2:0] = '111'. The encoding of this bit is:

ALL1ENB	Function
0	The FREEDM-32A672 responds to transmit polling when BADDR[2:0] = TXADDR[12:10] = '111' and device selection when BADDR[2:0] = TXDATA[15:13] = '111'.
1	The FREEDM-32A672 regards the all-ones address as a null address and does not respond to transmit polling and device selection when BADDR[2:0] = '111', regardless of the values of TXADDR[12:10] and TXDATA[15:13].

Channel Burst Length

The channel burst length (BLEN[7:0]) bits report the data transfer burst length read from the TAPI672 channel provision RAM after an indirect read operation has completed. The data transfer burst length specifies the length (in bytes, less one) of burst data transfers on the transmit APPI which are not terminated by the assertion of TEOP. The data transfer burst length can be specified on a per-channel basis with burst lengths of up to 256 bytes. The data transfer burst length to be written to the channel provision RAM in an indirect write operation must be set up in this register before triggering the write. BLEN[7:0] reflects the value written until the completion of a subsequent indirect read operation.

The BLEN[7:0] value must be set according to the indirect channel transfer size of the THDL672 block (XFER[3:0] in the **THDL Indirect Channel Data #3** (0x38C) register) using the following equation:

$$BLEN[7:0] = (XFER[3:0] + 1)\times 16 - 1.$$

A description of the XFER[3:0] bits can be found in section 7.5. The relationship between XFER[3:0] and BLEN[7:0] is shown in the following table.

XFER[3:0]	**BLEN[7:0]**
0000	0x0F
0001	0x1F
0010	0x2F
0011	0x3F
0100	0x4F
0101	0x5F
0110	0x6F
0111	0x7F
1000	0x8F
1001	0x9F
1010	0xAF
1011	0xBF
1100	0xCF
1101	0xDF
1110	0xEF
1111	0xFF

7 HDLC AND CHANNEL FIFO CONFIGURATION

The FREEDM-32A672 processes the data stream in the receive direction via the RHDL672 block and it processes the data stream in the transmit direction via the THDL672 block. Each of these blocks must be configured via the normal mode registers.

7.1 Configuring the RHDL672

The RHDL672 is configured by programming bits within the **RHDL Configuration** (0x220) register and the **RHDL Maximum Packet Length** (0x224) register. The values programmed affect all receive channels. The default configuration is as follows:

Bit	Register	Value
LENCHK	**RHDL Configuration** (0x220)	0
TSTD	**RHDL Configuration** (0x220)	0
MAX[15:0]	**RHDL Maximum Packet Length** (0x224)	0xFFFF

The default indicates no maximum packet length checking and datacom bit ordering.

Maximum Packet Length

The RHDL672 may be configured to abort packets which exceed the maximum length of *n* where *0 ≤ n ≤ 0xFFFF*. The following bits are written to enable or disable this feature:

LENCHK	MAX[15:0]	Function
0	0xFFFF	Receive packets are not checked for maximum size and MAX[15:0] must be set to 0xFFFF.
1	n	Receive packets with total length, including address, control, information and FCS fields, greater than MAX[15:0] bytes are aborted and the remainder of the frame discarded.

Datacom/Telecom Bit Order

The RHDL672 may be configured to reverse the order of bits in the HDLC data transferred across the receive APPI. The following bit is written to specify the order of bits:

TSTD	Function
0	Datacom standard: least significant bit of each byte on the receive APPI bus (AD[0] and AD[8]) is the first HDLC bit received. Normally, when HDLC processing is enabled, the TSTD bit must be set to zero.
1	Telecom standard: most significant bit of each byte on the receive APPI bus (AD[7] and AD[15]) is the first HDLC bit received.

7.2 Configuring the THDL672

The THDL672 is configured by programming bits within the **THDL Configuration** (0x3B0) register. The values programmed affect all transmit channels. The default configuration is as follows:

Bit	Register	Value
Reserved[3:0]	**THDL Configuration** (0x3B0)	0x0
Reserved[4]	**THDL Configuration** (0x3B0)	0
TSTD	**THDL Configuration** (0x3B0)	0
BIT8	**THDL Configuration** (0x3B0)	0

The default indicates that data is formatted in datacom bit ordering.

Datacom/Telecom Bit Order

The THDL672 may be configured to reverse the order of bits in the HDLC data transferred on the transmit APPI. The following bit is written to specify the order of bits:

TSTD	Function
0	Datacom standard: least significant bit of each byte on the transmit APPI bus (AD[0] and AD[8]) is the first HDLC bit transmitted. Normally, when HDLC processing is enabled, the TSTD bit must be set to zero.

TSTD	Function
1	Telecom standard: most significant bit of each byte on the transmit APPI bus (AD[7] and AD[15]) is the first HDLC bit transmitted.

BIT8

The BIT8 field affects channels of the THDL672 that are configured with 7BIT set. The BIT8 value specifies the data bit transmitted on the least significant bit of each octet.

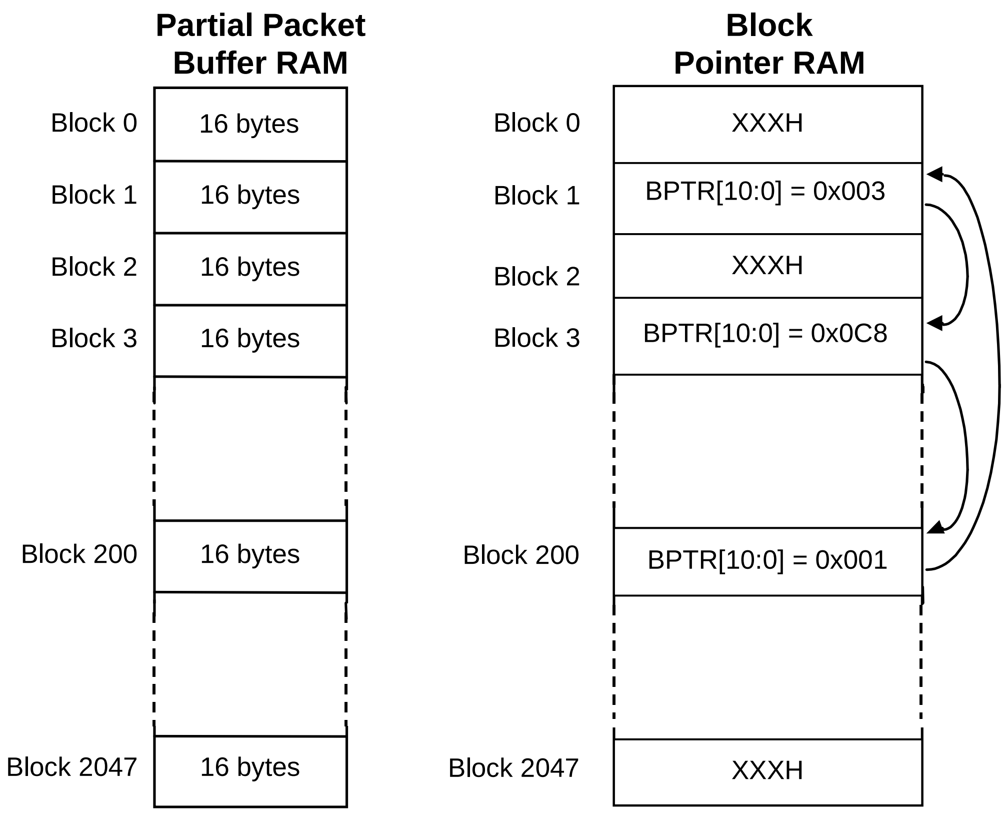
BIT8	Function
0	Channels configured for 7BIT will transmit a zero on the least significant bit of each octet.
1	Channels configured for 7BIT will transmit a one on the least significant bit of each octet.

7.3 Programming a Channel FIFO

A Channel FIFO is created from 3 or more blocks of internal RAM, and each block holds 16 bytes of packet data. There is a total of 2048 blocks (32 Kbytes) available to assign among the receive channels, and another 2048 blocks (32 Kbytes) available to assign among the transmit channels.

A FIFO is created by assigning a circular linked list of blocks as shown in Figure 14. This shows a channel FIFO consisting of 3 blocks. The quantity of buffers and the arrangement of links is chosen by the programmer, and the selection of blocks can be arbitrary. The programmer must ensure that a block is not assigned to more than one circularly linked list.

Figure 14 – Specifying a Channel FIFO



7.3.1 Receive Channel FIFO

A receive channel FIFO is programmed by repeating the following procedure for each block within the circularly linked list:

1. Poll the BUSY bit of the **RHDL Indirect Block Select** (0x210) register until it is zero. This ensures that a previous indirect RAM access has completed and that a new indirect RAM access can be started.

2. Write the following register with the next block in the circular linked list, or exit if all links have been programmed:

Bit	Register	Value
BPTR[10:0]	**RHDL Indirect Block Data** (0x214)	0 through 0x7FF are valid

Bit	Register	Value
Reserved	**RHDL Indirect Block Data** (0x214)	0

3. Specify the block and update the internal block pointer RAM by writing the following register. Proceed to step 1.

Bit	Register	Value
BLOCK[10:0]	**RHDL Indirect Block Select** (0x210)	0 through 0x7FF are valid
Reserved	**RHDL Indirect Block Select** (0x210)	0
BRWB	**RHDL Indirect Block Select** (0x210)	0
BUSY	**RHDL Indirect Block Select** (0x210)	X

7.3.2 Transmit Channel FIFO

A transmit channel FIFO is programmed by repeating the following procedure for each block within the circularly linked list:

1. Poll the BUSY bit of the **THDL Indirect Block Select** (0x3A0) register until it is zero. This ensures that a previous indirect RAM access has completed and that a new indirect RAM access can be started.

2. Write the following register with the next block in the circular linked list, or exit if all links have been programmed:

Bit	Register	Value
BPTR[10:0]	**THDL Indirect Block Data** (0x3A4)	0 through 0x7FF are valid
Reserved[0]	**THDL Indirect Block Data** (0x3A4)	0
Reserved[1]	**THDL Indirect Block Data** (0x3A4)	0

3. Specify the block and update the internal block pointer RAM by writing the following register. Proceed to step 1.

Bit	Register	Value
BLOCK[10:0]	**THDL Indirect Block Select** (0x3A0)	0 through 0x7FF are valid
Reserved	**THDL Indirect Block Select** (0x3A0)	0
BRWB	**THDL Indirect Block Select** (0x3A0)	0
BUSY	**THDL Indirect Block Select** (0x3A0)	X

7.4 RHDL672 Channel Configuration

The RHDL672 provides configurable options for each receive channel as identified in the following register fields:

Bit	Register
DELIN	**RHDL Indirect Channel Data Register #1** (0x204)
STRIP	**RHDL Indirect Channel Data Register #1** (0x204)
XFER[3:0]	**RHDL Indirect Channel Data Register #2** (0x208)
OFFSET[1:0]	**RHDL Indirect Channel Data Register #2** (0x208)
CRC[1:0]	**RHDL Indirect Channel Data Register #2** (0x208)
INVERT	**RHDL Indirect Channel Data Register #2** (0x208)
PRIORITY	**RHDL Indirect Channel Data Register #2** (0x208)
7BIT	**RHDL Indirect Channel Data Register #2** (0x208)

Note: When writing to **RHDL Indirect Channel Data Register #1** (0x204), the reserved bit (bit 11) must be set low for correct operation of the FREEDM-32A672.

Delineation

The data bits from the RCAS672 can be written directly to the Partial Packet Buffer or processed for flag sequence delineation, bit de-stuffing and CRC verification. The following bit enables or disables this feature:

DELIN	Function
0	Data is written to the Partial Packet Buffer without any HDLC processing (no flag sequence delineation, bit de-stuffing nor CRC verification) on the incoming stream.
1	Data is processed for flag sequence delineation, bit de-stuffing and optionally, CRC verification (CRC verification depends on CRC[1:0] value).

Strip FCS Bit

The indirect frame check sequence discard bit (STRIP) enables the RHDL672 to remove the FCS data before writing to the channel FIFO. STRIP is ignored when DELIN is low or when CRC[1:0] = 00B. This feature is configured as follows:

STRIP	Function
0	Includes FCS data with the data stream written to the channel FIFO.
1	Removes the FCS data from the data stream written to the channel FIFO.

DMA Transfer Size

The indirect channel transfer size configures the amount of data transferred in each transaction. When the channel FIFO depth reaches the depth specified by XFER[3:0] or when an end-of-packet exists in the FIFO, a poll of this FREEDM-32A672 device will indicate that data exists and is ready to be transferred across the receive APPI. Specifying a large transfer size may affect APPI bus access latencies for other channels. The following bits specify the channel transfer size:

XFER[3:0]	Function
0 through 15 are valid	Specifies the data transfer size in blocks: Blocks = XFER[3:0] +1, and there are 16 bytes per block.

Notes:

- XFER[3:0] should be set such that the number of blocks transferred is at least two fewer than the total allocated to the associated channel.

Insertion of Offset Bytes

The RHDL672 can be configured to insert offset bytes into the data stream before writing the data stream to the channel FIFO. The offset bytes are placed before each packet and their value is undefined. The following configuration options are available:

OFFSET[1:0]	Function
00	RHDL672 does not insert offset bytes
01	RHDL672 inserts 1 offset byte per packet
10	RHDL672 inserts 2 offset bytes per packet
11	RHDL672 inserts 3 offset bytes per packet

CRC Algorithm

The RHDL672 can perform CRC verification of the incoming data stream. The available options are as follows:

CRC[1:0]	DELIN	Function
X	0	No CRC verification
00	1	No CRC verification
01	1	CRC-CCITT verification
10	1	CRC-32 verification
11	1	Reserved

HDLC Data Inversion

The INVERT bit configures the RHDL672 to logically invert the incoming HDLC stream from the RCAS672 before processing it. The bit is specified as follows:

INVERT	Function
0	HDLC stream is not inverted.
1	HDLC stream is inverted.

Specifying Receive Channel Priority

All receive channels that must transfer data from their channel FIFO to packet memory contend for access to the receive APPI bus. The PRIORITY bit allows specified channels to have priority access to the receive APPI bus. The bit encoding is as follows:

PRIORITY	Function
0	This channel is serviced after channels with PRIORITY=1.
1	This channel is serviced before channels with PRIORITY=0.

Handling of Robbed bit Signaling

The 7BIT enable configures the RHDL672 to ignore the least significant bit of each octet (last bit of each octet received) in the corresponding link RD[n]. This bit is encoded as follows:

7BIT	Function
0	The entire receive data stream is processed.
1	The least significant bit (last bit of each octet received) is ignored.

7.5 THDL672 Channel Configuration

The THDL672 provides configurable options for each transmit channel as identified in the following register fields:

Bit	Register
DELIN	**THDL Indirect Channel Data Register #1** (0x384)
CRC[1:0]	**THDL Indirect Channel Data Register #1** (0x384)
FLEN[10:0]	**THDL Indirect Channel Data Register #2** (0x388)
DFCS	**THDL Indirect Channel Data Register #2** (0x388)
INVERT	**THDL Indirect Channel Data Register #2** (0x388)
7BIT	**THDL Indirect Channel Data Register #2** (0x388)
XFER[3:0]	**THDL Indirect Channel Data Register #3** (0x38C)
FLAG[2:0]	**THDL Indirect Channel Data Register #3** (0x38C)
LEVEL[3:0]	**THDL Indirect Channel Data Register #3** (0x38C)
IDLE	**THDL Indirect Channel Data Register #3** (0x38C)
TRANS	**THDL Indirect Channel Data Register #3** (0x38C)

Note: When writing to **THDL Indirect Channel Data Register #1** (0x384), the reserved bit (bit 11) must be set low for correct operation of the FREEDM-32A672. When writing to **THDL Indirect Channel Data Register #2** (0x388), the reserved bits (bits 11 and 14) must be set low for correct operation of the FREEDM-32A672.

Frame Delineation

The transmit packet data from packet memory can be written directly to the outgoing data stream or processed for flag sequence insertion, bit stuffing and CRC generation. The following bit enables or disables this feature:

DELIN	Function
0	Data is written directly to the outgoing data stream without any HDLC processing (no flag sequence insertion, bit stuffing nor CRC generation).
1	Data is processed for flag sequence insertion, bit stuffing and optionally, CRC generation (CRC generation depends on CRC[1:0] value).

CRC Algorithm

The THDL672 can perform CRC generation on the outgoing data stream. The available options are as follows:

CRC[1:0]	DELIN	Function
X	0	No CRC generation
00	1	No CRC generation
01	1	CRC-CCITT generation
10	1	CRC-32 generation
11	1	Reserved

Channel FIFO Length

The indirect FIFO length (FLEN[10:0]) is the number of blocks, less one, that is provisioned to the circular channel FIFO specified by the FPTR[10:0] block pointer.

FLEN[10:0]	Function
0 through 2047 are valid	Specifies the Channel FIFO size in blocks, where Blocks = FLEN[10:0] + 1, and each block is 16 bytes.

Inverting the FCS

The diagnose frame check sequence bit (DFCS) specifies whether the FCS field inserted into the transmit data stream is inverted. This is provided for diagnostic purposes and is programmed as follows:

DFCS	Function
0	FCS field in the outgoing HDLC stream is not inverted.
1	FCS field in the outgoing HDLC stream is logically inverted.

Robbed Bit Signaling

The least significant stuff enable bit (7BIT) configures the THDL672 to stuff the least significant bit of each octet assigned to the transmit channel in the corresponding transmit link (TD[n]).

7BIT	Function
0	The entire octet contains valid data and BIT8 is ignored.
1	The least significant bit (last bit of each octet transmitted) does not contain channel data and is forced to the value configured by the BIT8 register bit.

DMA Transfer Size

The indirect channel transfer size specifies the amount of data that the partial packet processor requests from the TAPI672 block. When the channel FIFO free space reaches or exceeds the limit specified by XFER[3:0], the partial packet processor will inform the TAPI672 so that a poll on that channel reflects that the channel FIFO is able to accept XFER[3:0] + 1 blocks of data. Specifying a large transfer size may affect APPI bus access latencies for other channels. The following bits specify the channel transfer size:

XFER[3:0]	Function
0 through 15 are valid	Specifies the data transfer size in blocks, where Blocks = XFER[3:0] + 1, and each block is 16 bytes.

Notes:

- To prevent lockup, the channel transfer size (XFER[3:0]) can be configured to be less than or equal to the start transmission level set by LEVEL[3:0] and TRANS. Alternatively, the channel transfer size can be set such that the total number of blocks in the logical channel FIFO minus the start transmission level is an integer multiple of the channel transfer size.

Specifying The Number of Flag or Idle Bytes Inserted Between Frames

The THDL672 can be configured to insert either flag or idle bytes (8 bits of one's) into the data stream between HDLC packets. The number of these is programmed as follows:

FLAG[2:0]	Minimum Number of Flag/Idle Bytes
000	1 flag / 0 Idle byte
001	2 flags / 0 idle byte
010	4 flags / 2 idle bytes
011	8 flags / 6 idle bytes
100	16 flags / 14 idle bytes
101	32 flags / 30 idle bytes
110	64 flags / 62 idle bytes
111	128 flags / 126 idle bytes

Interframe Time Fill

The IDLE bit specifies the byte pattern inserted in the data stream between HDLC packets.

IDLE	Function
0	Flag bytes are inserted between HDLC packets.
1	HDLC idle (all one's bit with no bit-stuffing) is inserted between HDLC packets.

Specifying the Channel FIFO's Starving Level and Start Transmit Level

The HDLC processor starts transmitting a packet when the channel FIFO free space is less than or equal to the level specified in the appropriate Start

Transmission Level column of the following table or when an end of a packet is stored in the channel FIFO.

When the channel FIFO free space is less than or equal to than the level specified in the Starving Trigger Level column of the following table and the HDLC processor is transmitting a packet and an end of a packet is not stored in the channel FIFO, the partial packet buffer makes expedited requests to the TAPI672 to retrieve XFER[3:0] + 1 blocks of data.

The starving trigger level and start transmission level are programmed via the LEVEL[3:0] and the TRANS field as follows:

LEVEL[3:0]	**Starving Trigger Level**	**Start Transmission Level (TRANS=0)**	**Start Transmission Level (TRANS=1)**
0000	2 Blocks (32 bytes free)	1 Block (16 bytes free)	1 Block (16 bytes free)
0001	3 Blocks (48 bytes free)	2 Blocks (32 bytes free)	1 Block (16 bytes free)
0010	4 Blocks (64 bytes free)	3 Blocks (48 bytes free)	2 Blocks (32 bytes free)
0011	6 Blocks (96 bytes free)	4 Blocks (64 bytes free)	3 Blocks (48 bytes free)
0100	8 Blocks (128 bytes free)	6 Blocks (96 bytes free)	4 Blocks (64 bytes free)
0101	12 Blocks (192 bytes free)	8 Blocks (128 bytes free)	6 Blocks (96 bytes free)
0110	16 Blocks (256 bytes free)	12 Blocks (192 bytes free)	8 Blocks (128 bytes free)
0111	24 Blocks (384 bytes free)	16 Blocks (256 bytes free)	12 Blocks (192 bytes free)
1000	32 Blocks (512 bytes free)	24 Blocks (384 bytes free)	16 Blocks (256 bytes free)
1001	48 Blocks (768 bytes free)	32 Blocks (512 bytes free)	24 Blocks (384 bytes free)
1010	64 Blocks (1 Kbytes free)	48 Blocks (768 bytes free)	32 Blocks (512 bytes free)
1011	96 Blocks (1.5 Kbytes free)	64 Blocks (1 Kbytes free)	48 Blocks (768 bytes free)

LEVEL[3:0]	Starving Trigger Level	Start Transmission Level (TRANS=0)	Start Transmission Level (TRANS=1)
1100	192 Blocks (3 Kbytes free)	128 Blocks (2 Kbytes free)	96 Blocks (1.5 Kbytes free)
1101	384 Blocks (6 Kbytes free)	256 Blocks (4 Kbytes free)	192 Blocks (2 Kbytes free)
1110	768 Blocks (12 Kbytes free)	512 Blocks (8 Kbytes free)	384 Blocks (4 Kbytes free)
1111	1536 Blocks (24 Kbytes free)	1024 Blocks (16 Kbytes free)	768 Blocks (8 Kbytes free)

8 FREEDM-32A672 OPERATIONAL PROCEDURES

8.1 Device Identification, Location and System Resource Assignment

This section describes the software interaction required to identify a FREEDM-32A672 device on the APPI bus, and to map the Normal Mode Registers in the microprocessor memory map.

Identifying and Locating a FREEDM-32A672

The software can identify a FREEDM-32A672 attached to an APPI bus by reading the TYPE[3:0] bits in the **FREEDM-32A672 Master Reset** (0x000) register. The default value of TYPE[3:0] = 0001B indicates that the device is the FREEDM-32A672 member of the FREEDM family of products.

For purposes of responding to receive polling and receive device selection, the address space occupied by each FREEDM-32A672 on the receive APPI needs to be configured using the base address bits (BADDR[2:0]) in the **RAPI Control** (0x580) register. Similarly, for purposes of responding to transmit polling and transmit data transfers, the address space occupied by each FREEDM-32A672 on the transmit APPI needs to be configured using the base address bits (BADDR[2:0]) in the **TAPI Control** (0x600) register. Note that up to seven FREEDM-32A672 devices may share a single APPI bus (one address is reserved as a null address), with an external controller acting as bus master.

In addition, the software can identify the version level of the FREEDM-32A672 with the ID[7:0] bits in the **FREEDM-32A672 Master Reset** (0x000) register. This may be useful to distinguish between future versions of the FREEDM-32A672.

Memory Mapping the Register Space

During power-up, the Normal Mode Register space needs to be mapped to the microprocessor. This register space is located in the FREEDM-32A672 and is accessed through the microprocessor interface. All registers are 16 bits wide but are DWORD aligned in the microprocessor memory map.

8.2 Reset

This section describes the procedure to reset the FREEDM-32A672 via software. The FREEDM-32A672 is powered on in an inactive state and should be reset via software following a hardware reset, or as required by the embedded processor. The reset procedure is normally followed by the initialization procedure.

The steps to reset a FREEDM-32A672 are:

1. If the FREEDM-32A672 was active before the reset procedure, then the deactivation procedure must be done (see section 8.6).

2. The RESET bit in the **FREEDM-32A672 Master Reset** (0x000) register must be written high and then written low.

This reset procedure has the following effects:

- The RESET bit allows the FREEDM-32A672 to be reset under software control. If the RESET bit is a logic one, the entire FREEDM-32A672 except the microprocessor interface is held in reset. This bit is not self-clearing. Therefore, a logic zero must be written to bring the FREEDM-32A672 out of reset. Holding the FREEDM-32A672 in a reset state places it into a low power, stand-by mode. A hardware reset clears the RESET bit, thus negating the software reset.

- All Normal Mode registers are set to their default values.

- None of the channel provisioning, or the Channel FIFO configuration is preserved under software reset.

8.3 Initialization

This section describes the procedure to initialize the FREEDM-32A672. The initialization procedure normally follows the software reset procedure and is followed by the activation procedure.

The steps to initialize a FREEDM-32A672 are:

1. Configure the serial links. The register accesses are described in section 5.

2. Assign base addresses for the receive and the transmit APPI. The register accesses are described in sections 6.1 and 6.2.

3. Configure HDLC processing of the RHDL672 and the THDL672 blocks. The register accesses are described in sections 7.1 and 7.2.

8.4 Activation Procedure

The activation procedure is required to place the FREEDM-32A672 in a state after which the software may service FREEDM-32A672 interrupts, provision/unprovision channels, and monitor the status of the FREEDM-32A672.

The activation procedure normally follows the initialization procedure.

The steps to activate a FREEDM-32A672 are:

1. Enable interrupt 'E' bits as described in section 4.
2. Enable data transfer across the receive and transmit APPI by setting the ENABLE bits to one in the RAPI672 and TAPI672 registers as described in sections 6.1 and 6.2.
3. The SYSCLKA, TDBA, RFP8A, TFP8A, RFPA[3:0], TFPA[3:0], RXCLKA, and TXCLKA bits in the **FREEDM-32A672 Master Clock/Frame Pulse/BERT Activity Monitor and Accumulation Trigger** (0x00C) register should be read periodically to detect for stuck at conditions. The SYSCLKA bit must be read high for proper operation of the FREEDM-32A672. A low value indicates a failure in clocking that is provided at the SYSCLK input pin of the FREEDM-32A672. Similarly, a low value in the other register bits indicates a failure in clocking that is provided by the corresponding input pin.
4. The TLGA[7:0] and the RLGA[7:0] bits in the **FREEDM-32A672 Master Link Activity Monitor** (0x010) register should be read periodically to detect for stuck at conditions. The bits which correspond to links attached to the FREEDM-32A672 should be read high. A low value indicates a failure in clocking that is provided by the corresponding RCLK[n:m], RMVCK[n], RMV8DC, TCLK[n:m], TMVCK[n], or TMV8DC input pins.[1]

8.5 Deactivation Procedure

The deactivation procedure is required to place the FREEDM-32A672 in a state in which it will not interrupt the embedded processor, or transfer data across the APPI. This procedure should occur after the FREEDM-32A672 actively transfers packets, or to gracefully shut down the FREEDM-32A672.

The steps to deactivate a FREEDM-32A672 are:

1. Disable interrupt 'E' bits as described in section 4.
2. Disable data transfer across the receive and transfer APPI by programming the ENABLE bits to zero in the RAPI672 and TAPI672 registers as described in sections 6.1 and 6.2.

[1]Each RLGA[7:0] or TLGA[7:0] bit corresponds to a group of 4 non H-MVIP links (in addition to the H-MVIP links). If less than four links are configured, then the unused RCLK or TCLK inputs should be tied to the same clock as one of the configured links in this group.

3. Continue by performing the software reset procedure.

8.6 Provisioning a Channel

The provisioning procedure normally follows the activation procedure and enables the FREEDM-32A672 to receive and/or transmit packets.

8.6.1 Receive Channel Provisioning

The steps to provision a receive channel *RCC*, where *0 ≤ RCC ≤ 671* are:

1. Disable FREEDM-32A672 processing of the channel's data stream to allow for graceful provisioning. Write the following bits:

Bit	Register	Value
DCHAN[9:0]	**RCAS Channel Disable** (0x10C)	*RCC*
CHDIS	**RCAS Channel Disable** (0x10C)	1

2. Program the Channel FIFO as described in section 7.3.1.

3. Poll the BUSY bit of the **RHDL Indirect Channel Select** (0x200) register until it is zero. This ensures that a previous indirect RAM access has completed and that a new indirect RAM access can be started.

4. Specify the HDLC configuration for this channel by writing appropriate bits in the **RHDL Indirect Channel Data Register #1** (0x204) and the **RHDL Indirect Channel Data Register #2** (0x208) as described in section 7.4. When writing the **RHDL Indirect Channel Data Register #1**, ensure that the PROV bit is set, and ensure that the FPTR[10:0] bits identify a block within the circular linked list of buffers of step 2.

5. Specify the RHDL672 channel to provision by writing the following register. Then poll the BUSY bit to ensure that it is low before proceeding to step 6.

Bit	Register	Value
CHAN[9:0]	**RHDL Indirect Channel Select** (0x200)	*RCC*
CRWB	**RHDL Indirect Channel Select** (0x200)	0
BUSY	**RHDL Indirect Channel Select** (0x200)	X

6. Poll the BUSY bit of the **RCAS Indirect Link and Time-slot Select** (0x100) register until it is zero. This ensures that a previous indirect RAM access has completed and that a new indirect RAM access can be started.

7. Specify the RCAS672 channel that is provisioned. Write the following register:

Bit	Register	Value
CHAN[9:0]	**RCAS Indirect Channel Data** (0x104)	*RCC*
PROV	**RCAS Indirect Channel Data** (0x104)	1
CDLBEN	**RCAS Indirect Channel Data** (0x104)	0

8. For a **channelised** link, specify the time-slots which are assigned for processing on this channel by writing the following register once for each time-slot that is assigned to the channel. Valid values for TSLOT[4:0] are 1 through 24 for a T1/J1 link, 1 through 31 for an E1 link, and 0 through 31 for an H-MVIP link. For an **unchannelised** link, TSLOT[4:0] must only have the value 0, and this register is written just once. Each write must be followed by a read to determine whether the BUSY bit (bit15) is low, and to ensure that the indirect RAM has been updated.

Bit	Register	Value
TSLOT[4:0]	**RCAS Indirect Link and Time-slot Select** (0x100)	see above
LINK[4:0]	**RCAS Indirect Link and Time-slot Select** (0x100)	0 through 31 are valid
Reserved[1:0]	**RCAS Indirect Link and Time-slot Select** (0x100)	00
RWB	**RCAS Indirect Link and Time-slot Select** (0x100)	0
BUSY	**RCAS Indirect Link and Time-slot Select** (0x100)	X

9. Enable FREEDM-32A672 processing of the channel data stream to allow for graceful provisioning. Write the following bits:

Bit	Register	Value
DCHAN[9:0]	**RCAS Channel Disable** (0x10C)	*RCC*
CHDIS	**RCAS Channel Disable** (0x10C)	0

Warning:

- The RCAS Channel Disable bit (CHDIS) is only applicable to one channel at a time. In other words, the receive channel provisioning procedure needs to be run once for each channel.
- The programmer must ensure that the channel has not been provisioned, or has been unprovisioned before doing the provisioning procedure. The reset procedure has the effect of unprovisioning all channels of the FREEDM-32A672.
- Continuous polling of a register in a tight loop involves multiple microprocessor memory read transactions and may have an adverse effect on the microprocessor bus bandwidth available for other activities. The recommended method of polling the BUSY bit is to read the register on expiration of a system timer, or after a number of CPU clock ticks. Recommended time intervals are in the range 0.1 msec through 1 msec.
- A Channel is not provisioned until the BUSY bit toggles low.

8.6.2 Transmit Channel Provisioning

The steps to provision a transmit channel *TCC*, where $0 \leq TCC \leq 671$ are:

1. Disable FREEDM-32A672 processing of the channel's data stream to allow for graceful provisioning. Write the following bits:

Bit	Register	Value
DCHAN[9:0]	**TCAS Channel Disable** (0x410)	*TCC*
CHDIS	**TCAS Channel Disable** (0x410)	1

2. Program the Channel FIFO as described in section 7.3.2 for a transmit channel.
3. Poll the BUSY bit of the **THDL Indirect Channel Select** (0x380) register until it is zero. This ensures that a previous indirect RAM access has completed and that a new indirect RAM access can be started.
4. Specify the HDLC configuration for this transmit channel by writing the **THDL Indirect Channel Data Register #1** (0x384), **THDL Indirect Channel Data Register #2** (0x388) and the **THDL Indirect Channel Data Register #3** (0x38C) as described in section 7.5. In writing the **THDL Indirect Channel**

Data Register #1, ensure the PROV bit is set, and ensure the FPTR[10:0] bits identify a block within the circular linked list of buffers of step 2.

5. Specify the THDL672 channel that is provisioned by writing the following register. Then poll the BUSY bit to ensure it is low before proceeding with step 6.

Bit	Register	Value
CHAN[9:0]	**THDL Indirect Channel Select** (0x380)	*TCC*
CRWB	**THDL Indirect Channel Select** (0x380)	0
BUSY	**THDL Indirect Channel Select** (0x380)	X

6. Poll the BUSY bit of the **TCAS Indirect Link and Time-slot Select** (0x400) register until it is zero. This ensures that a previous indirect RAM access has completed and that a new indirect RAM access can be started.

7. Specify the TCAS672 channel that is provisioned. Write the following register:

Bit	Register	Value
CHAN[9:0]	**TCAS Indirect Channel Data** (0x404)	*TCC*
PROV	**TCAS Indirect Channel Data** (0x404)	1

8. For a **channelised** link, specify the time-slots which are assigned for processing on this channel by writing the following register once for each time-slot that is assigned to the channel. Valid values for TSLOT[4:0] are 1 through 24 for a T1/J1 link, 1 through 31 for an E1 link, and 0 through 31 for an H-MVIP link. For an **unchannelised** link, TSLOT[4:0] must only have the value 0, and this register is written just once. Each write must be followed by a read to determine whether the BUSY bit (bit15) is low, and to ensure that the indirect RAM has been updated.

Bit	Register	Value
TSLOT[4:0]	**TCAS Indirect Link and Time-slot Select** (0x400)	see above
LINK[4:0]	**TCAS Indirect Link and Time-slot Select** (0x400)	0 through 31 are valid
Reserved[1:0]	**TCAS Indirect Link and Time-slot Select** (0x400)	00
RWB	**TCAS Indirect Link and Time-slot Select** (0x400)	0

Bit	Register	Value
BUSY	**TCAS Indirect Link and Time-slot Select** (0x400)	X

9. Poll the BUSY bit of the **TAPI Indirect Channel Provisioning** (0x604) register until it is zero. This ensures that a previous indirect RAM access has completed and that a new indirect RAM access can be started.

10. Specify the channel burst length and enable channel provisioning by writing the following register. The BLEN[7:0] bits need to be set according to the XFER[3:0] value of the THDL672 as described in section 6.2.

Bit	Register	Value
BLEN[7:0]	**TAPI Indirect Channel Data Register** (0x608)	see above
PROV	**TAPI Indirect Channel Data Register** (0x608)	1

11. Specify the TAPI672 channel to provision. Write the following register fields, then poll the BUSY bit to ensure that the provisioning process has completed.

Bit	Register	Value
CHAN[9:0]	**TAPI Indirect Channel Provisioning** (0x604)	*TCC*
RWB	**TAPI Indirect Channel Provisioning** (0x604)	0
BUSY	**TAPI Indirect Channel Provisioning** (0x604)	X

12. Enable FREEDM-32A672 processing of the channel data stream to allow for graceful provisioning. Write the following bits:

Bit	Register	Value
DCHAN[9:0]	**TCAS Channel Disable** (0x410)	*TCC*
CHDIS	**TCAS Channel Disable** (0x410)	0

Warning:

- The TCAS Channel Disable bit (CHDIS) is only applicable to one channel at a time. In other words, the transmit channel provisioning procedure needs to be run once for each channel.

- The programmer must ensure that the channel has not been provisioned, or has been unprovisioned before doing the provisioning procedure. The reset procedure has the affect of unprovisioning all channels of the FREEDM-32A672.

- Continuous polling of a register in a tight loop involves multiple microprocessor memory read transactions and may have an adverse effect on the microprocessor bus bandwidth available for other activities. The recommended method of polling the BUSY bit is to read the register on expiration of a system timer, or after a number of CPU clock ticks. Recommended time intervals are in the range 0.1 msec through 1 msec.

- A Channel is not provisioned until the BUSY bit toggles low.

8.7 Unprovisioning a Channel

The unprovisioning procedure is normally applied to channels that are provisioned.

8.7.1 Receive Channel Unprovisioning

The steps to unprovision a receive channel *RCC*, where *0 ≤ RCC ≤ 671* are:

1. Disable FREEDM-32A672 processing of the channel's data stream to allow for graceful unprovisioning. Write the following bits:

Bit	Register	Value
DCHAN[9:0]	**RCAS Channel Disable** (0x10C)	*RCC*
CHDIS	**RCAS Channel Disable** (0x10C)	1

2. Poll the BUSY bit of the **RCAS Indirect Link and Time-slot Select** (0x100) register until it is zero. This ensures that a previous indirect RAM access has completed and that a new indirect RAM access can be started.

3. Specify the RCAS672 channel to unprovision by writing the following register:

Bit	Register	Value
CHAN[9:0]	**RCAS Indirect Channel Data** (0x104)	*RCC*
PROV	**RCAS Indirect Channel Data** (0x104)	0
CDLBEN	**RCAS Indirect Channel Data** (0x104)	X

4. For a **channelised** link, specify the time-slots which are unassigned on this channel by writing the following register once for each time-slot that is

unassigned. Valid values for TSLOT[4:0] are 1 through 24 for a T1/J1 link, 1 through 31 for an E1 link, and 0 through 31 for an H-MVIP link. For an **unchannelised** link, TSLOT[4:0] must only have the value 0, and this register is written just once. Each write must be followed by a read to determine whether the BUSY bit (bit15) is low, and to ensure that the indirect RAM has been updated.

Bit	Register	Value
TSLOT[4:0]	**RCAS Indirect Link and Time-slot Select** (0x100)	see above
LINK[4:0]	**RCAS Indirect Link and Time-slot Select** (0x100)	0 through 31 are valid
Reserved[1:0]	**RCAS Indirect Link and Time-slot Select** (0x100)	00
RWB	**RCAS Indirect Link and Time-slot Select** (0x100)	0
BUSY	**RCAS Indirect Link and Time-slot Select** (0x100)	X

5. Poll the BUSY bit of the **RHDL Indirect Channel Select** (0x200) register until it is zero. This ensures that a previous indirect RAM access has completed and that a new indirect RAM access can be started.

6. Read the RHDL672 channel data by writing the following register. Then poll the BUSY bit to ensure it is low before proceeding with step 7.

Bit	Register	Value
CHAN[9:0]	**RHDL Indirect Channel Select** (0x200)	*RCC*
CRWB	**RHDL Indirect Channel Select** (0x200)	1
BUSY	**RHDL Indirect Channel Select** (0x200)	X

7. Read the RHDL672 indirect channel data and check that the TAVAIL bit of the **RHDL Indirect Channel Data #1** (0x204) register is zero. This ensures that the last data transfer across the receive APPI for this channel has completed. If the TAVAIL bit is zero, proceed to step 8, otherwise, return to step 6.

8. Write the **RHDL Indirect Channel Data #1** (0x204) register with PROV modified to zero, while keeping the same FPTR[10:0] bits.

9. Specify the RHDL672 channel to unprovision by writing the following register. Then poll the BUSY bit to ensure that it is low before proceeding with step 10.

Bit	Register	Value
CHAN[9:0]	**RHDL Indirect Channel Select** (0x200)	*RCC*
CRWB	**RHDL Indirect Channel Select** (0x200)	0
BUSY	**RHDL Indirect Channel Select** (0x200)	X

10. Enable FREEDM-32A672 processing of the unprovisioned channel. Write the following bits:

Bit	Register	Value
DCHAN[9:0]	**RCAS Channel Disable** (0x10C)	*RCC*
CHDIS	**RCAS Channel Disable** (0x10C)	0

Warning:

- The RCAS Channel Disable bit (CHDIS) is only applicable to one channel at a time. In other words, the receive channel unprovisioning procedure needs to be run once for each channel.
- Continuous polling of a register in a tight loop involves multiple microprocessor memory read transactions and may have an adverse effect on the microprocessor bus bandwidth available for other activities. The recommended method of polling the BUSY bit is to read the register on expiration of a system timer, or after a number of CPU clock ticks. Recommended time intervals are in the range 0.1 msec through 1 msec.
- A Channel is not unprovisioned until the BUSY bit toggles low.

8.7.2 Transmit Channel Unprovisioning

The steps to unprovision a transmit channel *TCC*, where $0 \leq TCC \leq 671$ are:

1. Poll the BUSY bit of the **TAPI Indirect Channel Provisioning** (0x604) register until it is zero. This ensures that a previous indirect RAM access has completed and that a new indirect RAM access can be started.
2. Enable channel unprovisioning by writing the following register:

Bit	Register	Value
PROV	**TAPI Indirect Channel Data Register** (0x608)	0

3. Specify the TAPI672 channel to unprovision. Write the following register fields, then poll the BUSY bit to ensure that the unprovisioning process has completed.

Bit	Register	Value
CHAN[9:0]	**TAPI Indirect Channel Provisioning** (0x604)	*TCC*
RWB	**TAPI Indirect Channel Provisioning** (0x604)	0
BUSY	**TAPI Indirect Channel Provisioning** (0x604)	X

4. Disable FREEDM-32A672 processing of the channel's data stream to allow for graceful unprovisioning. Write the following bits:

Bit	Register	Value
DCHAN[9:0]	**TCAS Channel Disable** (0x410)	*TCC*
CHDIS	**TCAS Channel Disable** (0x410)	1

5. Poll the BUSY bit of the **TCAS Indirect Link and Time-slot Select** (0x400) register until it is zero. This ensures that a previous indirect RAM access has completed and that a new indirect RAM access can be started.

6. Specify the TCAS672 channel to be unprovisioned. Write the following register:

Bit	Register	Value
CHAN[9:0]	**TCAS Indirect Channel Data** (0x404)	*TCC*
PROV	**TCAS Indirect Channel Data** (0x404)	0

7. For a **channelised** link, specify the time-slots which are unassigned for processing on this channel by writing the following register once for each time-slot that is unassigned. Valid values for TSLOT[4:0] are 1 through 24 for a T1/J1 link, 1 through 31 for an E1 link, and 0 through 31 for an H-MVIP link. For an **unchannelised** link, TSLOT[4:0] must only have the value 0, and this register is written just once. Each write must be followed by a read to determine whether the BUSY bit (bit15) is low, and to ensure that the indirect RAM has been updated.

Bit	Register	Value
TSLOT[4:0]	**TCAS Indirect Link and Time-slot Select** (0x400)	see above
LINK[4:0]	**TCAS Indirect Link and Time-slot Select** (0x400)	0 through 31 are valid
Reserved[1:0]	**TCAS Indirect Link and Time-slot Select** (0x400)	00
RWB	**TCAS Indirect Link and Time-slot Select** (0x400)	0
BUSY	**TCAS Indirect Link and Time-slot Select** (0x400)	X

8. Poll the BUSY bit of the **THDL Indirect Channel Select** (0x380) register until it is zero. This ensures that a previous indirect RAM access has completed and that a new indirect RAM access can be started.

9. Read the THDL672 channel data by writing the following register. Then poll the BUSY bit to ensure that it is low before proceeding with step 9.

Bit	Register	Value
CHAN[9:0]	**THDL Indirect Channel Select** (0x380)	*TCC*
CRWB	**THDL Indirect Channel Select** (0x380)	1
BUSY	**THDL Indirect Channel Select** (0x380)	X

10. Read the **THDL Indirect Channel Data #1** (0x384) register. Then write this register with PROV modified to zero, while keeping the same FPTR[10:0] bits.

11. Specify the THDL672 channel to unprovision by writing the following register. Then poll the BUSY bit to ensure that it is low before proceeding with step 11.

Bit	Register	Value
CHAN[9:0]	**THDL Indirect Channel Select** (0x380)	*TCC*
CRWB	**THDL Indirect Channel Select** (0x380)	0
BUSY	**THDL Indirect Channel Select** (0x380)	X

12. Enable FREEDM-32A672 processing of the channel. Write the following bits:

Bit	Register	Value
DCHAN[9:0]	**TCAS Channel Disable** (0x410)	*TCC*
CHDIS	**TCAS Channel Disable** (0x410)	0

Warning:

- The TCAS Channel Disable bit (CHDIS) is only applicable to one channel at a time. In other words, the transmit channel unprovisioning procedure needs to be run once for each channel.
- Continuous polling of a register in a tight loop involves multiple microprocessor memory read transactions and may have an adverse effect on the microprocessor bus bandwidth available for other activities. The recommended method of polling the BUSY bit is to read the register on expiration of a system timer, or after a number of CPU clock ticks. Recommended time intervals are in the range 0.1 msec through 1 msec.
- A Channel is not unprovisioned until the BUSY bit toggles low.

8.8 Receive Sequence

The software is not required to receive packets when interfacing to the RAPI672. Data transfer functions for the FREEDM-32A672 are performed by an external controller.

In the receive direction, the external controller transfers partial packets out of the internal 32 Kbyte partial packet buffer RAM in the RHDL672, across the receive APPI bus, and into host packet memory. Please refer to the Longform Datasheet[1] for detailed information on the operation and timing of the receive APPI.

8.9 Transmit Sequence

The software is not required to transmit packets when interfacing to the TAPI672. Data transfer functions for the FREEDM-32A672 are performed by an external controller.

In the transmit direction, the external controller provides packets to transmit using the transmit APPI. For each provisioned HDLC channel, an external controller transfers partial packets across the transfer APPI, and into the internal 32 Kbyte partial packet buffer RAM in the THDL672. Please refer to the

Longform Datasheet[1] for detailed information on the operation and timing of the transmit APPI.

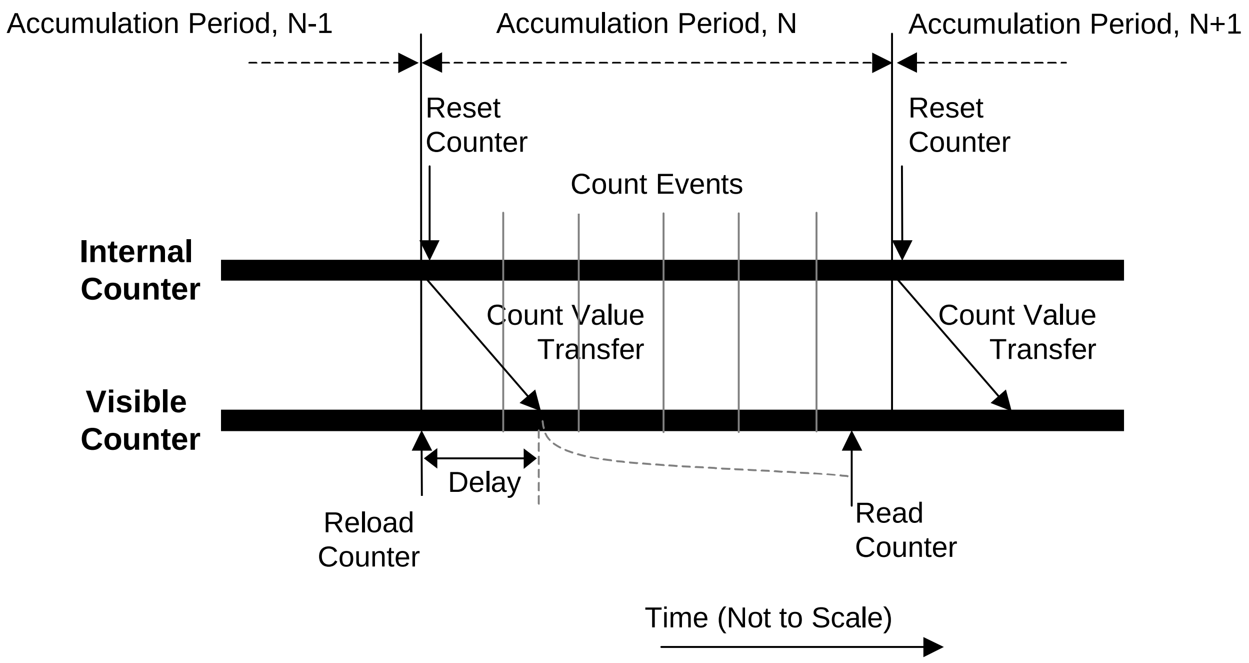
8.10 Performance Counters

The FREEDM-32A672 provides four count registers within the Normal Mode Register Space. These are as follows:

Bit	Register
OF[15:0]	**PMON Receive FIFO Overflow Count** (0x504)
UF[15:0]	**PMON Transmit FIFO Underflow Count** (0x508)
C1[15:0]	**PMON Configurable Count #1** (0x50C)
C2[15:0]	**PMON Configurable Count #2** (0x510)

The software must poll these counters to prevent overflow. Figure 15 illustrates the sequence of events when the counters are polled. The **PMON Status** (0x500) register provides status bits which indicate whether any of the four internal holding counters has overflowed.

Figure 15 – Event Sequence for Polling of Counters



The software initiates a counter reload by writing to the **FREEDM-32A672 Master Clock/Frame Pulse/BERT Activity Monitor and Accumulation Trigger** (0x00C) register. There is a small delay to transfer data from internal counters to the visible counters. The recommended polling strategy is to read the counters first before initiating a reload. Using this strategy, the transfer latency can be ignored.

Counters are normally configured during initialization. The first configurable count register is assigned by setting one of the following register bits, while setting all other bits to zero:

Bit	Register
RSPE1EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)
RFCSE1EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)
RABRT1EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)
RLENE1EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)
RP1EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)
TABRT1EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)
TP1EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)

The second configurable count register is assigned by setting one of the following register bits, while setting all other bits to zero:

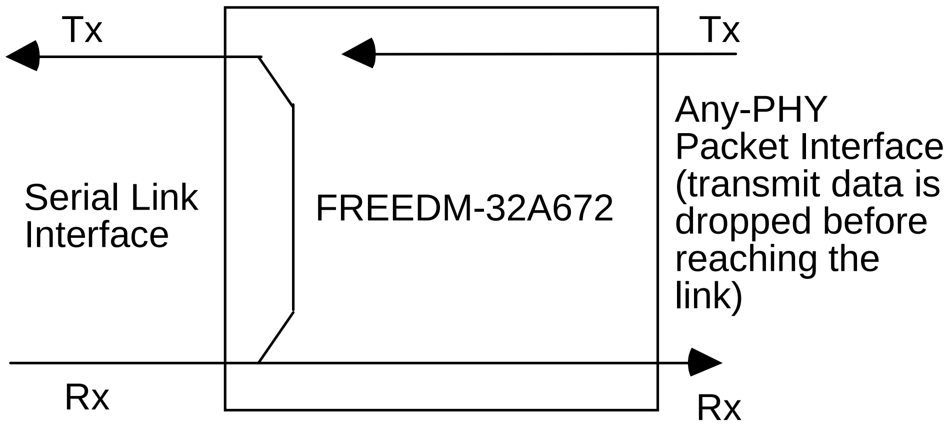
Bit	Register
RSPE2EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)
RFCSE2EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)
RABRT2EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)

Bit	Register
RLENE2EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)
RP2EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)
TABRT2EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)
TP2EN	**FREEDM-32A672 Master Performance Monitor Control** (0x024)

8.11 Line Loopback

Serial ports configured for non H-MVIP traffic can be placed in line loopback. In this configuration, receive data from the serial link is looped back to the transmit serial link as illustrated in Figure 16. Line loopback is not supported for H-MVIP traffic.

Figure 16 – Line Loopback


Tx
Tx
Any-PHY Packet Interface (transmit data is dropped before reaching the link)
Serial Link Interface
FREEDM-32A672
Rx
Rx

Non H-MVIP serial ports can be placed in line loopback by setting the appropriate bit within one of the following registers. There are 32 bits corresponding to the 32 serial ports.

Bit	Register
LLBEN[15:0]	**FREEDM-32A672 Master Line Loopback #1** (0x014)
LLBEN[31:16]	**FREEDM-32A672 Master Line Loopback #2** (0x018)

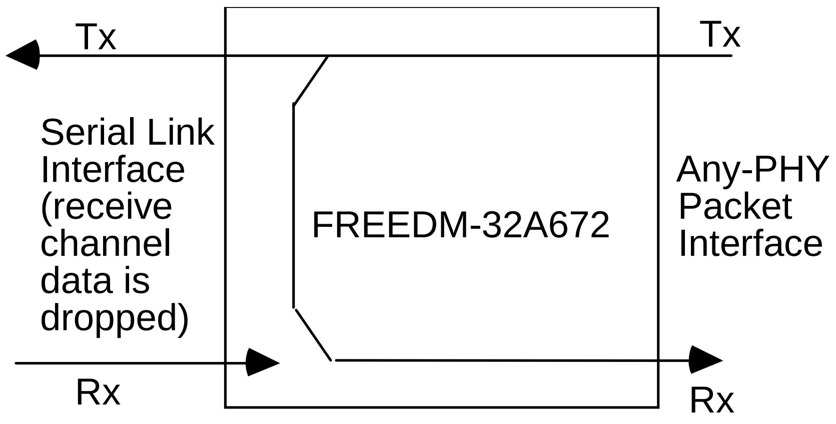
Note: The software should unprovision channels associated with the link that is placed in line loopback mode before placing the link in line loopback. This will prevent the data stream at the serial link from passing through the FREEDM-32A672 to the receive APPI.

8.12 Diagnostic Loopback

Each channel of the FREEDM-32A672 can be placed in a diagnostic loopback mode. In this configuration, the transmit data stream is looped back to the receive data stream as illustrated in Figure 17. The pair of transmit/receive channels is configured in diagnostic loopback mode by provisioning both the transmit and the receive channels as specified in section 8.7, except with the CDLBEN bit set high within the **RCAS Indirect Channel Data** (0x104) register.

In diagnostic loopback mode, the transmit channel data is looped back as well as driven onto the transmit serial link. The channel data from the receive serial link is dropped. The TCLK[n] input pin provides the bit timing for the diagnostic loopback mode.

Figure 17 – Diagnostic Loopback



Note: For the diagnostic loopback to pass data through the FREEDM-32A672, the receive line clock must be provided at the RCLK[n] input pin for the link that is placed in diagnostic loopback. The data rate for the diagnostic loopback is the same as the clock rate provided at this pin.

8.13 BERT Port

The FREEDM-32A672 provides pins to transmit/receive a BERT data stream on serial links configured for non H-MVIP traffic. The following register is written to enable the BERT port and transmit/receive the BERT data stream on serial port *n*, where *0 ≤ n ≤ 31.*

Bit	Register	Value
RBSEL[4:0]	**FREEDM-32A672 Master BERT Control** (0x020)	*n*
RBEN	**FREEDM-32A672 Master BERT Control** (0x020)	1
TBSEL[4:0]	**FREEDM-32A672 Master BERT Control** (0x020)	*n*
TBEN	**FREEDM-32A672 Master BERT Control** (0x020)	1

The BERT port is disabled by programming the following register:

Bit	Register	Value
RBSEL[4:0]	**FREEDM-32A672 Master BERT Control** (0x020)	X
RBEN	**FREEDM-32A672 Master BERT Control** (0x020)	0
TBSEL[4:0]	**FREEDM-32A672 Master BERT Control** (0x020)	X
TBEN	**FREEDM-32A672 Master BERT Control** (0x020)	0

The TDBA bit of the **FREEDM-32A672 Master Clock/Frame Pulse/BERT Activity Monitor and Accumulation Trigger** (0x00C) can be read by software to determine whether transmit data is present at the BERT port.

Note: The FREEDM-32A672 channels which are assigned to the same link as the BERT port should be unprovisioned to prevent the receive BERT data stream from passing through the FREEDM-32A672 to the receive APPI.

APPENDIX A – REGISTER LEVEL CHANGES

The following table is a comparison of the normal mode registers at the register level among the FREEDM-32, the FREEDM-32P672 and the FREEDM-32A672. Registers in bold indicate differences at the register level among the members of the FREEDM family listed in the table. Table entries that are "N/A" indicate that the register is not applicable in the corresponding FREEDM device. Please see Appendix D for differences at the bit level for the normal mode registers.

Register	FREEDM-32 PCI Offset	FREEDM-32P672 PCI Offset	FREEDM-32A672 Address
FREEDM-32xxxx Master Reset	0x000	0x000	0x000
FREEDM-32xxxx Master Interrupt Enable	0x004	0x004	0x004
FREEDM-32xxxx Master Interrupt Status	0x008	0x008	0x008
FREEDM-32xxxx Master Clock/Frame Pulse/BERT Activity Monitor and Accumulation Trigger	0x00C	0x00C	0x00C
FREEDM-32xxxx Master Link Activity Monitor	0x010	0x010	0x010
FREEDM-32xxxx Master Line Loopback #1	0x014	0x014	0x014
FREEDM-32xxxx Master Line Loopback #2	0x018	0x018	0x018
Reserved	**0x01C**	**N/A**	**N/A**
FREEDM-32xxxx Reserved	**N/A**	**0x01C**	**0x01C**
FREEDM-32xxxx Master BERT Control	0x020	0x020	0x020
FREEDM-32xxxx Master Performance Monitor Control	0x024	0x024	0x024

Register	FREEDM-32 PCI Offset	FREEDM-32P672 PCI Offset	FREEDM-32A672 Address
Reserved	**0x028 – 0x03C**	**0x028 – 0x07C**	**0x028 – 0x0FC**
GPIC Control	**0x040**	**0x080**	**N/A**
GPIC Reserved	**0x044 – 0x07C**	**0x084 – 0x0FC**	**N/A**
Reserved	**0x080 – 0x0FC**	**N/A**	**N/A**
RCAS Indirect Channel and Time-slot Select	0x100	0x100	0x100
RCAS Indirect Channel Data	0x104	0x104	0x104
RCAS Framing Bit Threshold	0x108	0x108	0x108
RCAS Channel Disable	0x10C	0x10C	0x10C
RCAS Reserved	0x110 – 0x17C	0x110 – 0x17C	0x110 – 0x17C
RCAS Link #0 through #31 Configuration	0x180 – 0x1FC	0x180 – 0x1FC	0x180 – 0x1FC
RHDL Indirect Channel Select	0x200	0x200	0x200
RHDL Indirect Channel Data Register #1	0x204	0x204	0x204
RHDL Indirect Channel Data Register #2	0x208	0x208	0x208
RHDL Reserved	0x20C	0x20C	0x20C
RHDL Indirect Block Select	0x210	0x210	0x210
RHDL Indirect Block Data Register	0x214	0x214	0x214
RHDL Reserved	0x218 – 0x21C	0x218 – 0x21C	0x218 – 0x21C
RHDL Configuration	0x220	0x220	0x220
RHDL Maximum Packet Length	0x224	0x224	0x224

Register	FREEDM-32 PCI Offset	FREEDM-32P672 PCI Offset	FREEDM-32A672 Address
RHDL Reserved	0x228 – 0x23C	0x228 – 0x23C	0x228 – 0x23C
Reserved	**0x240 – 0x27C**	**0x240 – 0x27C**	**0x240 – 0x37C**
RMAC Control	**0x280**	**0x280**	**N/A**
RMAC Indirect Channel Provisioning	**0x284**	**0x284**	**N/A**
RMAC Packet Descriptor Table Base LSW	**0x288**	**0x288**	**N/A**
RMAC Packet Descriptor Table Base MSW	**0x28C**	**0x28C**	**N/A**
RMAC Queue Base LSW	**0x290**	**0x290**	**N/A**
RMAC Queue Base MSW	**0x294**	**0x294**	**N/A**
RMAC Packet Descriptor Reference Large Buffer Free Queue Start	**0x298**	**0x298**	**N/A**
RMAC Packet Descriptor Reference Large Buffer Free Queue Write	**0x29C**	**0x29C**	**N/A**
RMAC Packet Descriptor Reference Large Buffer Free Queue Read	**0x2A0**	**0x2A0**	**N/A**
RMAC Packet Descriptor Reference Large Buffer Free Queue End	**0x2A4**	**0x2A4**	**N/A**
RMAC Packet Descriptor Reference Small Buffer Free Queue Start	**0x2A8**	**0x2A8**	**N/A**
RMAC Packet Descriptor Reference Small Buffer Free Queue Write	**0x2AC**	**0x2AC**	**N/A**

Register	FREEDM-32 PCI Offset	FREEDM-32P672 PCI Offset	FREEDM-32A672 Address
RMAC Packet Descriptor Reference Small Buffer Free Queue Read	0x2B0	0x2B0	N/A
RMAC Packet Descriptor Reference Small Buffer Free Queue End	0x2B4	0x2B4	N/A
RMAC Packet Descriptor Reference Ready Queue Start	0x2B8	0x2B8	N/A
RMAC Packet Descriptor Reference Ready Queue Write	0x2BC	0x2BC	N/A
RMAC Packet Descriptor Reference Ready Queue Read	0x2C0	0x2C0	N/A
RMAC Packet Descriptor Reference Ready Queue End	0x2C4	0x2C4	N/A
RMAC Reserved	0x2C8 – 0x2FC	0x2C8 – 0x2FC	N/A
TMAC Control	0x300	0x300	N/A
TMAC Indirect Channel Provisioning	0x304	0x304	N/A
TMAC Descriptor Table Base LSW	0x308	0x308	N/A
TMAC Descriptor Table Base MSW	0x30C	0x30C	N/A
TMAC Queue Base LSW	0x310	0x310	N/A
TMAC Queue Base MSW	0x314	0x314	N/A
TMAC Descriptor Reference Free Queue Start	0x318	0x318	N/A

Register	FREEDM-32 PCI Offset	FREEDM-32P672 PCI Offset	FREEDM-32A672 Address
TMAC Descriptor Reference Free Queue Write	**0x31C**	**0x31C**	**N/A**
TMAC Descriptor Reference Free Queue Read	**0x320**	**0x320**	**N/A**
TMAC Descriptor Reference Free Queue End	**0x324**	**0x324**	**N/A**
TMAC Descriptor Reference Ready Queue Start	**0x328**	**0x328**	**N/A**
TMAC Descriptor Reference Ready Queue Write	**0x32C**	**0x32C**	**N/A**
TMAC Descriptor Reference Ready Queue Read	**0x330**	**0x330**	**N/A**
TMAC Descriptor Reference Ready Queue End	**0x334**	**0x334**	**N/A**
TMAC Reserved	**0x338 – 0x37C**	**0x338 – 0x37C**	**N/A**
THDL Indirect Channel Select	0x380	0x380	0x380
THDL Indirect Channel Data #1	0x384	0x384	0x384
THDL Indirect Channel Data #2	0x388	0x388	0x388
THDL Indirect Channel Data #3	0x38C	0x38C	0x38C
THDL Reserved	0x390 – 0x39C	0x390 – 0x39C	0x390 – 0x39C
THDL Indirect Block Select	0x3A0	0x3A0	0x3A0

Register	**FREEDM-32 PCI Offset**	**FREEDM-32P672 PCI Offset**	**FREEDM-32A672 Address**
THDL Indirect Block Data	0x3A4	0x3A4	0x3A4
THDL Reserved	0x3A8 – 0x3AC	0x3A8 – 0x3AC	0x3A8 – 0x3AC
THDL Configuration	0x3B0	0x3B0	0x3B0
THDL Reserved	0x3B4 – 0x3BC	0x3B4 – 0x3BC	0x3B4 – 0x3BC
Reserved	0x3C0 – 0X3FC	0x3C0 – 0X3FC	0x3C0 – 0X3FC
TCAS Indirect Channel and Time-slot Select	0x400	0x400	0x400
TCAS Indirect Channel Data	0x404	0x404	0x404
TCAS Framing Bit Threshold	0x408	0x408	0x408
TCAS Idle Time-slot Fill Data	0x40C	0x40C	0x40C
TCAS Channel Disable	0x410	0x410	0x410
TCAS Reserved	0x414 – 0x47C	0x414 – 0x47C	0x414 – 0x47C
TCAS Link #0 through #31 Configuration	0x480 – 0x4FC	0x480 – 0x4FC	0x480 – 0x4FC
PMON Status	0x500	0x500	0x500
PMON Receive FIFO Overflow Count	0x504	0x504	0x504
PMON Transmit FIFO Underflow Count	0x508	0x508	0x508
PMON Configurable Count #1	0x50C	0x50C	0x50C
PMON Configurable Count #2	0x510	0x510	0x510
PMON Reserved	0x514 – 0x51C	0x514 – 0x51C	0x514 – 0x51C
Reserved	**0x520 – 0x7FC**	**0x520 – 0x7FC**	**0x520 – 0x57C**

Register	FREEDM-32 PCI Offset	FREEDM-32P672 PCI Offset	FREEDM-32A672 Address
RAPI Control	N/A	N/A	0x580
RAPI Reserved	N/A	N/A	0x584 – 0x5BC
Reserved	N/A	N/A	0x5C0 – 0x5FC
TAPI Control	N/A	N/A	0x600
TAPI Indirect Channel Provisioning	N/A	N/A	0x604
TAPI Indirect Channel Data Register	N/A	N/A	0x608
TAPI Reserved	N/A	N/A	0x60C – 0x63C
Reserved	N/A	N/A	0x640 – 0x7FC

Note: There are no PCI Configuration registers in the FREEDM-32A672 device.

APPENDIX B – NEW NORMAL MODE REGISTERS

The following registers are new for the FREEDM-32A672. The new registers are used to configure and control the new RAPI672 and TAPI672 blocks. Please refer to the Longform Datasheet[1] for detailed descriptions of these registers.

FREEDM-32A672 Address	Register
0x580	RAPI Control
0x584 – 0x5BC	RAPI Reserved
0x600	TAPI Control
0x604	TAPI Indirect Channel Provisioning
0x608	TAPI Indirect Channel Data Register
0x60C – 0x63C	TAPI Reserved

APPENDIX C – NON-APPLICABLE NORMAL MODE REGISTERS

The following FREEDM-32 registers are no longer applicable for the FREEDM-32A672. These registers were used to configure and control the RMAC, TMAC and GPIC of the FREEDM-32.

FREEDM-32 PCI Offset	Register
0x040	GPIC Control
0x044 - 0x07C	GPIC Reserved
0x280	RMAC Control
0x284	RMAC Indirect Channel Provisioning
0x288	RMAC Packet Descriptor Table Base LSW
0x28C	RMAC Packet Descriptor Table Base MSW
0x290	RMAC Queue Base LSW
0x294	RMAC Queue Base MSW
0x298	RMAC Packet Descriptor Reference Large Buffer Free Queue Start
0x29C	RMAC Packet Descriptor Reference Large Buffer Free Queue Write
0x2A0	RMAC Packet Descriptor Reference Large Buffer Free Queue Read
0x2A4	RMAC Packet Descriptor Reference Large Buffer Free Queue End
0x2A8	RMAC Packet Descriptor Reference Small Buffer Free Queue Start
0x2AC	RMAC Packet Descriptor Reference Small Buffer Free Queue Write
0x2B0	RMAC Packet Descriptor Reference Small Buffer Free Queue Read
0x2B4	RMAC Packet Descriptor Reference Small Buffer Free Queue End
0x2B8	RMAC Packet Descriptor Reference Ready Queue Start

FREEDM-32 PCI Offset	Register
0x2BC	RMAC Packet Descriptor Reference Ready Queue Write
0x2C0	RMAC Packet Descriptor Reference Ready Queue Read
0x2C4	RMAC Packet Descriptor Reference Ready Queue End
0x2C8 - 0x2FC	RMAC Reserved
0x300	TMAC Control
0x304	TMAC Indirect Channel Provisioning
0x308	TMAC Descriptor Table Base LSW
0x30C	TMAC Descriptor Table Base MSW
0x310	TMAC Queue Base LSW
0x314	TMAC Queue Base MSW
0x318	TMAC Descriptor Reference Free Queue Start
0x31C	TMAC Descriptor Reference Free Queue Write
0x320	TMAC Descriptor Reference Free Queue Read
0x324	TMAC Descriptor Reference Free Queue End
0x328	TMAC Descriptor Reference Ready Queue Start
0x32C	TMAC Descriptor Reference Ready Queue Write
0x330	TMAC Descriptor Reference Ready Queue Read
0x334	TMAC Descriptor Reference Ready Queue End
0x338 - 0x37C	TMAC Reserved

APPENDIX D – NORMAL MODE REGISTER BIT CHANGES

The following normal mode registers have changed at the bit level from the FREEDM-32 to the FREEDM-32A672. Unless otherwise specified, register names, locations and comments refer to FREEDM-32A672 registers.

Register 0x000 : FREEDM-32A672 Master Reset

Bit	**FREEDM-32A672**		**FREEDM-32**		**Comments**
	Function	Default	Function	Default	
15	Reset	0	Reset	0	Now also forces the APPI outputs tristate.
11	TYPE[3]	0	Unused	X	New Device Type bits allow software to identify the device as the FREEDM-32A672 member of the FREEDM family of products.
10	TYPE[2]	0	Unused	X	
9	TYPE[1]	0	Unused	X	
8	TYPE[0]	1	Unused	X	
7	ID[7]	0	Unused	X	New Device ID bits allow software to identify the version level of FREEDM-32A672.
6	ID[6]	0	Unused	X	
5	ID[5]	0	Unused	X	
4	ID[4]	0	Unused	X	
3	ID[3]	0	Unused	X	
2	ID[2]	0	Unused	X	
1	ID[1]	0	Unused	X	
0	ID[0]	0	Unused	X	

Register 0x004 : FREEDM-32A672 Master Interrupt Enable

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
14	TFOVRE	0	IOCE	0	SERRE, PERRE, IOCE, and all queue-related interrupt enable bits are not used because queues and the PCI bus are not used for the FREEDM-32A672. New interrupt enable bits are TPRTYE, TUNPVE, and TFOVRE.
13	TUNPVE	0	TDFQEE	0	
12	TPRTYE	0	TDQRDYE	0	
11	Unused	X	TDQFE	0	
10	Unused	X	RPDRQEE	0	
9	Unused	X	RPDFQEE	0	
8	Unused	X	RPQRDYE	0	
7	Unused	X	RPQLFE	0	
6	Unused	X	RPQSFE	0	
1	Unused	X	PERRE	0	
0	Unused	X	SERRE	0	

Register 0x008 : FREEDM-32A672 Master Interrupt Status

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
14	TFOVRI	0	IOCI	0	SERRI, PERRI, IOCI, and all queue-related interrupt status bits are not used because queues and the PCI bus are not used for the FREEDM-32A672. New interrupt status bits are TPRTYI, TUNPVI, and TFOVRI.
13	TUNPVI	0	TDFQEI	0	
12	TPRTYI	0	TDQRDYI	0	
11	Unused	X	TDQFI	0	
10	Unused	X	RPDRQEI	0	
9	Unused	X	RPDFQEI	0	
8	Unused	X	RPQRDYI	0	
7	Unused	X	RPQLFI	0	
6	Unused	X	RPQSFI	0	
1	Unused	X	PERRI	0	
0	Unused	X	SERRI	0	

Register 0x00C : FREEDM-32A672 Master Clock / Frame Pulse /BERT Activity Monitor and Accumulation Trigger

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
13	TXCLKA	X	Unused	X	Any-PHY transmit clock active bit.
12	RXCLKA	X	Unused	X	Any-PHY receive clock active bit.
11	TFPA[3]	X	Unused	X	Transmit 2.048 Mbps H-MVIP frame pulse activity bits.
10	TFPA[2]	X	Unused	X	
9	TFPA[1]	X	Unused	X	
8	TFPA[0]	X	Unused	X	
7	RFPA[3]	X	Unused	X	Receive 2.048 Mbps H-MVIP frame pulse activity bits.
6	RFPA[2]	X	Unused	X	
5	RFPA[1]	X	Unused	X	
4	RFPA[0]	X	Unused	X	
3	TFP8A	X	Unused	X	Transmit 8.192 Mbps H-MVIP frame pulse activity bit.
2	RFP8A	X	Unused	X	Receive 8.192 Mbps H-MVIP frame pulse activity bit.

Register 0x010 : FREEDM-32A672 Master Link Activity Monitor

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
15	TLGA[7]	X	TLGA[7]	X	Now also monitors TMV8DC input.
11	TLGA[3]	X	TLGA[3]	X	Now also monitors TMVCK[3] input.
10	TLGA[2]	X	TLGA[2]	X	Now also monitors TMVCK[2] input.
9	TLGA[1]	X	TLGA[1]	X	Now also monitors TMVCK[1] input.
8	TLGA[0]	X	TLGA[0]	X	Now also monitors TMVCK[0] input.
7	RLGA[7]	X	RLGA[7]	X	Now also monitors RMVCK[3] and RMV8DC inputs.
6	RLGA[6]	X	RLGA[6]	X	
5	RLGA[5]	X	RLGA[5]	X	Now also monitors RMVCK[2] and RMV8DC inputs.
4	RLGA[4]	X	RLGA[4]	X	
3	RLGA[3]	X	RLGA[3]	X	Now also monitors RMVCK[1] and RMV8DC inputs.
2	RLGA[2]	X	RLGA[2]	X	
1	RLGA[1]	X	RLGA[1]	X	Now also monitors RMVCK[0] and RMV8DC inputs.
0	RLGA[0]	X	RLGA[0]	X	

Register 0x01C : FREEDM-32A672 Reserved

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
0	Reserved	0	Not Specified	X	Reserved bit must be set low for correct operation of FREEDM-32A672.

Register 0x100 : RCAS Indirect Link and Time-slot Select

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
12	Reserved	0	LINK[4]	0	Reserved bits must be set low for correct operation of FREEDM-32A672.
11	Reserved	0	LINK[3]	0	
10	LINK[4]	0	LINK[2]	0	
9	LINK[3]	0	LINK[1]	0	
8	LINK[2]	0	LINK[0]	0	
7	LINK[1]	0	Unused	X	
6	LINK[0]	0	Unused	X	

Register 0x104 : RCAS Indirect Channel Data

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
15	CDLBEN	0	Unused	X	
14	PROV	0	Unused	X	
9	CHAN[9]	0	CDLBEN	0	Increase in size of CHAN from 7 bits to 10 bits as the result of increase in HDLC channels from 128 to 672.
8	CHAN[8]	0	PROV	0	
7	CHAN[7]	0	Unused	X	

Register 0x108 : RCAS Framing Bit Threshold

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
6	FTHRES[6]	0	FTHRES[6]	0	
5	FTHRES[5]	1	FTHRES[5]	1	
4	FTHRES[4]	0	FTHRES[4]	1	Change of default value.
3	FTHRES[3]	0	FTHRES[3]	1	Change of default value.
2	FTHRES[2]	1	FTHRES[2]	1	
1	FTHRES[1]	0	FTHRES[1]	1	Change of default value.
0	FTHRES[0]	1	FTHRES[0]	1	

Register 0x10C : RCAS Channel Disable

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
9	DCHAN[9]	0	Unused	X	Increase in size of DCHAN from 7 bits to 10 bits as the result of increase in HDLC channels from 128 to 672.
8	DCHAN[8]	0	Unused	X	
7	DCHAN[7]	0	Unused	X	

Registers 0x180 – 0x188 : RCAS Links #0 to #2 Configuration

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
4	BSYNC	0	Unused	X	
2	MODE[2]	0	BSYNC	0	New mode select bits; see table below. E1 is no longer used. CEN is no longer used.
1	MODE[1]	0	E1	0	
0	MODE[0]	0	CEN	0	

Registers 0x18C – 0x1FC : RCAS Links #3 to #31 Configuration

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
2	MODE[2]	0	Unused	X	New mode select bits; see table below. E1 is no longer used. CEN is no longer used.
1	MODE[1]	0	E1	0	
0	MODE[0]	0	CEN	0	

The mode select bits are encoded as follows to configure the serial links of the FREEDM-32A672:

MODE[2:0]	Link Configuration
000	Unchannelised
001	Channelised T1/J1 (24 time slots labeled 1-24)
010	Channelised E1 (31 time slots labeled 1-31)
011	2 Mbps H-MVIP (32 time slots labeled 0-31)
100	Reserved
101	Reserved
110	Reserved
111	8 Mbps H-MVIP (128 time slots mapped to time-slots 0 through 31 of links 4m, 4m+1, 4m+2 and 4m+3)

Register 0x200 : RHDL Indirect Channel Select

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
9	CHAN[9]	0	Unused	X	Increase in size of CHAN from 7 bits to 10 bits as the result of increase in HDLC channels from 128 to 672.
8	CHAN[8]	0	Unused	X	
7	CHAN[7]	0	Unused	X	

Register 0x204 : RHDL Indirect Channel Data #1

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
14	STRIP	0	CRC[1]	0	CRC[1] moved to bit 11 of Register 0x208 of the FREEDM-32A672.
13	DELIN	0	CRC[0]	0	CRC[0] moved to bit 10 of Register 0x208 the FREEDM-32A672.
12	TAVAIL	X	STRIP	0	
11	Reserved	X	DELIN	0	Reserved bit must be set low for correct operation of the FREEDM-32A672.
10	FPTR[10]	X	TAVAIL	X	Increase in size of FPTR from 9 bits to 11 bits as the result of increase in addressable descriptors from 512 to 2048.
9	FPTR[9]	X	Unused	X	

Register 0x208 : RHDL Indirect Channel Data #2

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
11	CRC[1]	0	Unused	X	CRC[1] moved from bit 14 of Register 0x204 of the FREEDM-32.
10	CRC[0]	0	Unused	X	CRC[0] moved from bit 13 of Register 0x204 of the FREEDM-32.
3	XFER[3]	0	Unused	X	XFER increased from 3 bits to 4 bits to support larger data transfers.

Register 0x210 : RHDL Indirect Block Select

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
11	Reserved	X	Unused	X	Reserved bit must be set low for correct operation of FREEDM-32A672.
10	BLOCK[10]	X	Unused	X	BLOCK increased from 9 bits to 11 bits as the result of increase in addressable blocks from 512 to 2048.
9	BLOCK[9]	X	Unused	X	

Register 0x214 : RHDL Indirect Block Data

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
11	Reserved	X	Unused	X	Reserved bit must be set low for correct operation of FREEDM-32A672.
10	BPTR[10]	X	Unused	X	BPTR increased from 9 bits to 11 bits as a result of increase in addressable blocks from 512 to 2048.
9	BPTR[9]	X	Unused	X	

Register 0x220 : RHDL Configuration

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
2	Unused	X	Reserved[2]	1	These reserved bits are no longer used in the FREEDM-32A672.
1	Unused	X	Reserved[1]	1	
0	Unused	X	Reserved[0]	1	

Register 0x380 : THDL Indirect Channel Select

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
9	CHAN[9]	0	Unused	X	Increase in size of CHAN from 7 bits to 10 bits as the result of increase in HDLC channels from 128 to 672.
8	CHAN[8]	0	Unused	X	
7	CHAN[7]	0	Unused	X	

Register 0x384 : THDL Indirect Channel Data #1

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
12	DELIN	X	IDLE	0	IDLE moved to bit 14 of Register 0x38C of the FREEDM-32A672.
11	Reserved	X	DELIN	0	Reserved bit must be set low for correct operation of FREEDM-32A672.
10	FPTR[10]	0	Unused	X	Increase in size of FPTR from 9 bits to 11 bits as the result of increase in addressable descriptors from 512 to 2048.
9	FPTR[9]	0	Unused	X	

Register 0x388 : THDL Indirect Channel Data #2

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
14	Reserved	0	PRIORITYB	0	Reserved bits must be set low for correct operation of FREEDM-32A672. PRIORITYB is not used.
11	Reserved	0	Unused	X	
10	FLEN[10]	0	Unused	X	Increase in size of FLEN from 9 bits to 11 bits as the result of increase in addressable descriptors from 512 to 2048.
9	FLEN[9]	0	Unused	X	

Register 0x38C : THDL Indirect Channel Data #3

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
14	IDLE	0	Unused	X	IDLE moved from bit 12 of Register 0x384 of the FREEDM-32.
3	XFER[3]	0	Unused	X	XFER increased from 3 bits to 4 bits to support larger data transfers.

Register 0x3A0 : THDL Indirect Block Select

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
11	Reserved	X	Unused	X	Reserved bit must be set low for correct operation of FREEDM-32A672.
10	BLOCK[10]	0	Unused	X	BLOCK increased from 9 bits to 11 bits as the result of increase in addressable blocks from 512 to 2048.
9	BLOCK[9]	0	Unused	X	

Register 0x3A4 : THDL Indirect Block Data

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
11	Reserved	X	Unused	X	Reserved bit must be set low for correct operation of FREEDM-32A672.
10	BPTR[10]	0	Unused	X	BPTR increased from 9 bits to 11 bits as the result of increase in addressable blocks from 512 to 2048.
9	BPTR[9]	0	Unused	X	

Register 0x3B0 : THDL Configuration

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
7	Reserved	0	BURSTEN	0	Reserved bits must be set low for correct operation of FREEDM-32A672. The DMA burst length feature is not used.
3	Reserved	0	Unused	X	
2	Reserved	0	BURST[2]	0	
1	Reserved	0	BURST[1]	0	
0	Reserved	0	BURST[0]	0	

Register 0x400 : TCAS Indirect Link and Time-slot Select

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
12	Reserved	0	LINK[4]	0	Reserved bits must be set low for correct operation of the FREEDM-32A672.
11	Reserved	0	LINK[3]	0	
10	LINK[4]	0	LINK[2]	0	
9	LINK[3]	0	LINK[1]	0	
8	LINK[2]	0	LINK[0]	0	
7	LINK[1]	0	Unused	X	
6	LINK[0]	0	Unused	X	

Register 0x404 : TCAS Indirect Channel Data

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
15	PROV	0	Unused	X	
9	CHAN[9]	0	Unused	X	Increase in size of CHAN from 7 bits to 10 bits as the result of increase in HDLC channels from 128 to 672.
8	CHAN[8]	0	PROV	0	
7	CHAN[7]	0	Unused	X	

Register 0x408 : TCAS Framing Bit Threshold

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
6	FTHRES[6]	0	FTHRES[6]	0	
5	FTHRES[5]	1	FTHRES[5]	0	Change of default value.
4	FTHRES[4]	0	FTHRES[4]	1	Change of default value.
3	FTHRES[3]	0	FTHRES[3]	1	Change of default value.
2	FTHRES[2]	1	FTHRES[2]	1	
1	FTHRES[1]	0	FTHRES[1]	1	Change of default value.
0	FTHRES[0]	1	FTHRES[0]	1	

Register 0x410 : TCAS Channel Disable

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
9	DCHAN[9]	0	Unused	X	Increase in size of DCHAN from 7 bits to 10 bits as the result of increase in HDLC channels from 128 to 672.
8	DCHAN[8]	0	Unused	X	
7	DCHAN[7]	0	Unused	X	

Registers 0x480 – 0x488 : TCAS Links #0 to #2 Configuration

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
4	BSYNC	0	Unused	X	
2	MODE[2]	0	BSYNC	0	New mode select bits; see table below. E1 is no longer used. CEN is no longer used.
1	MODE[1]	0	E1	0	
0	MODE[0]	0	CEN	0	

Registers 0x48C – 0x4FC : TCAS Links #3 to #31 Configuration

Bit	FREEDM-32A672		FREEDM-32		Comments
	Function	Default	Function	Default	
2	MODE[2]	0	Unused	X	New mode select bits; see table below. E1 is no longer used. CEN is no longer used.
1	MODE[1]	0	E1	0	
0	MODE[0]	0	CEN	0	

The mode select bits are encoded as follows to configure the serial links of the FREEDM-32A672:

MODE[2:0]	Link Configuration
000	Unchannelised
001	Channelised T1/J1 (24 time slots labeled 1-24)
010	Channelised E1 (31 time slots labeled 1-31)
011	2 Mbps H-MVIP (32 time slots labeled 0-31)
100	Reserved
101	Reserved
110	Reserved
111	8 Mbps H-MVIP (128 time slots mapped to time-slots 0 through 31 of links 4m, 4m+1, 4m+2 and 4m+3)

CONTACTING PMC-SIERRA, INC.

PMC-Sierra, Inc.
105-8555 Baxter Place Burnaby, BC
Canada V5A 4V7

Tel: (604) 415-6000

Fax: (604) 415-6200

Document Information:	document@pmc-sierra.com
Corporate Information:	info@pmc-sierra.com
Application Information:	apps@pmc-sierra.com
	(604) 415-4533
Web Site:	http://www.pmc-sierra.com

PMC-990639 (A1) date: June 1999