PM7364

FREEDM™-32

FRAME ENGINE AND DATALINK MANAGER

DATA SHEET

ISSUE 6: AUGUST 2001

PUBLIC REVISION HISTORY

Issue No.	Issue Date	Details of Change
6	August 2001	Patent information added to legal footer.
5	May 1998	Document re-issue.
4	April 1998	Document re-issue.
3	October 1997	Document re-formatted.
2	April 23, 1997	Pin Diagram page replaced. Two entries added to Pin table diagram. Added AC, DC Timing section and 256 BGA mechanical package information.
1	July 24, 1996	Creation of Data Sheet

CONTENTS

LIST OF REGISTERS

LIST OF FIGURES

LIST OF TABLES

1 FEATURES

- Single-chip Peripheral Component Interconnect (PCI) Bus multi-channel HDLC controller.
- Supports up to 128 bi-directional HDLC channels assigned to a maximum of 32 channelised T1 or E1 links. The number of time-slots assigned to an HDLC channel is programmable from 1 to 24 (for T1) and from 1 to 31 (for E1).
- Supports up to 32 bi-directional HDLC channels each assigned to an unchannelised arbitrary rate link; subject to a maximum aggregate link clock rate of 64 MHz in each direction. Channels assigned to links 0 to 2 can have a clock rate of up to 45 MHz when SYSCLK is at or above 25 MHz and up to 52 MHz when SYSCLK is at 33 MHz. Channels assigned to links 3 to 31 can have a clock rate of up to 10 MHz.
- Supports up to two bi-directional HDLC channels each assigned to an unchannelised arbitrary rate link of up to 45 MHz when SYSCLK is at or above 25 MHz and up to 52 MHz when SYSCLK is at 33 MHz.
- Supports a mix of up to 32 channelised and unchannelised links; subject to the constraint of a maximum of 128 channels and a maximum aggregate link clock rate of 64 MHz in each direction.
- For each channel, the HDLC receiver performs flag sequence detection, bit de-stuffing, and frame check sequence validation. The receiver supports the validation of both CRC-CCITT and CRC-32 frame check sequences. The receiver also checks for packet abort sequences, octet aligned packet length and for minimum and maximum packet length.
- Alternatively, for each channel, the receiver supports a transparent mode where each octet is transferred transparently to host memory. For channelised links, the octets are aligned with the receive time-slots.
- For each channel, time-slots are selectable to be in 56 kbits/s format or 64 kbits/s clear channel format.
- For each channel, the HDLC transmitter performs flag sequence generation, bit stuffing, and, optionally, frame check sequence generation. The transmitter supports the generation of both CRC-CCITT and CRC-32 frame check sequences. The transmitter also aborts packets under the direction of the host or automatically when the channel underflows.

- Supports two levels of non-preemptive packet priority on each transmit channel. Low priority packets will not begin transmission until all high priority packets are transmitted.

- Alternatively, for each channel, the transmitter supports a transparent mode where each octet is inserted transparently from host memory. For channelised links, the octets are aligned with the transmit time-slots.

- Directly supports a 32-bit, 33 MHz PCI 2.1 interface for configuration, monitoring and transfer of packet data, with an on-chip DMA controller with scatter/gather capabilities.

- Provides 8 kbytes of on-chip memory for partial packet buffering in each direction. This memory can be configured to support a variety of different channel configurations from a single channel with 8 kbytes of buffering to 128 channels, each with a minimum of 48 bytes of buffering.

- Supports PCI burst sizes of up to 128 bytes for transfers of packet data.

- Pin compatible with PM7366 (FREEDM-8) device.

- Provides a standard 5 signal P1149.1 JTAG test port for boundary scan board test purposes.

- Supports 3.3 and 5 Volt PCI signaling environments.

- Low power CMOS technology.

- 256 pin enhanced ball grid array (SBGA) package (27 mm X 27 mm).

2 APPLICATIONS

- IETF PPP interfaces for routers
- Frame Relay interfaces for ATM or Frame Relay switches and multiplexors
- FUNI or Frame Relay service inter-working interfaces for ATM switches and multiplexors.
- D-channel processing in ISDN terminals and switches.
- Internet/Intranet access equipment.
- Packet-based DSLAM equipment.

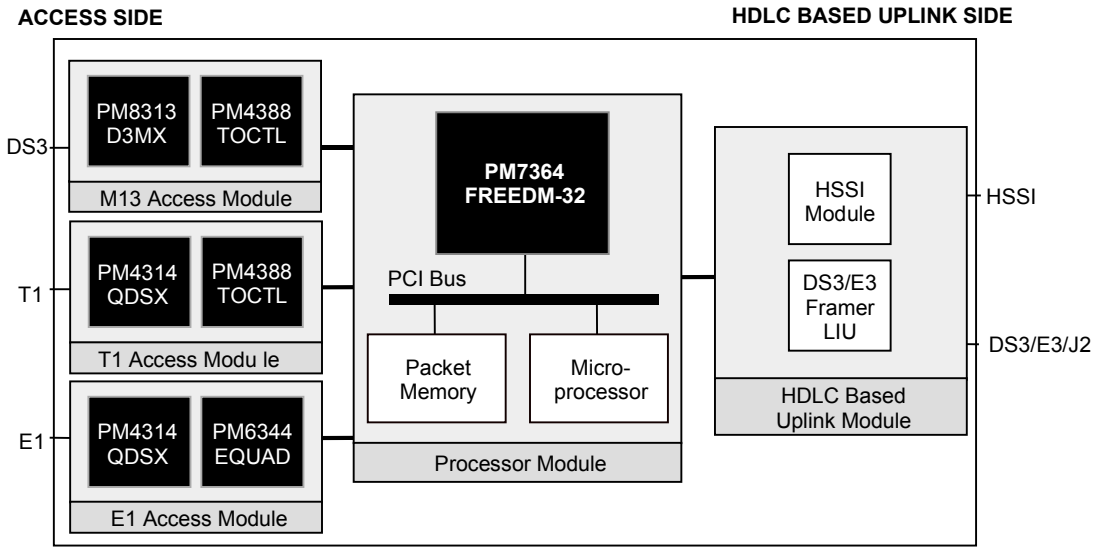
3 REFERENCES

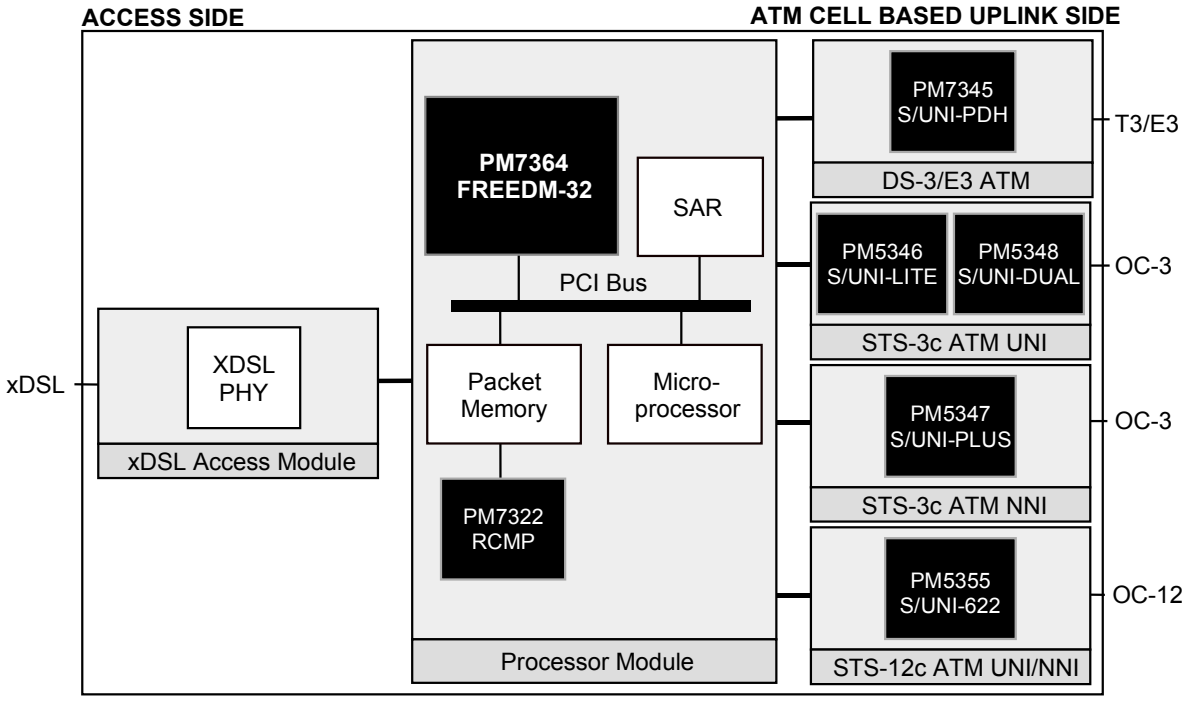
1. International Organization for Standardization, ISO Standard 3309-1993, "Information Technology - Telecommunications and information exchange between systems - High-level data link control (HDLC) procedures - Frame structure", December 1993.

2. RFC-1662 - "PPP in HDLC-like Framing" Internet Engineering Task Force, July 1994.

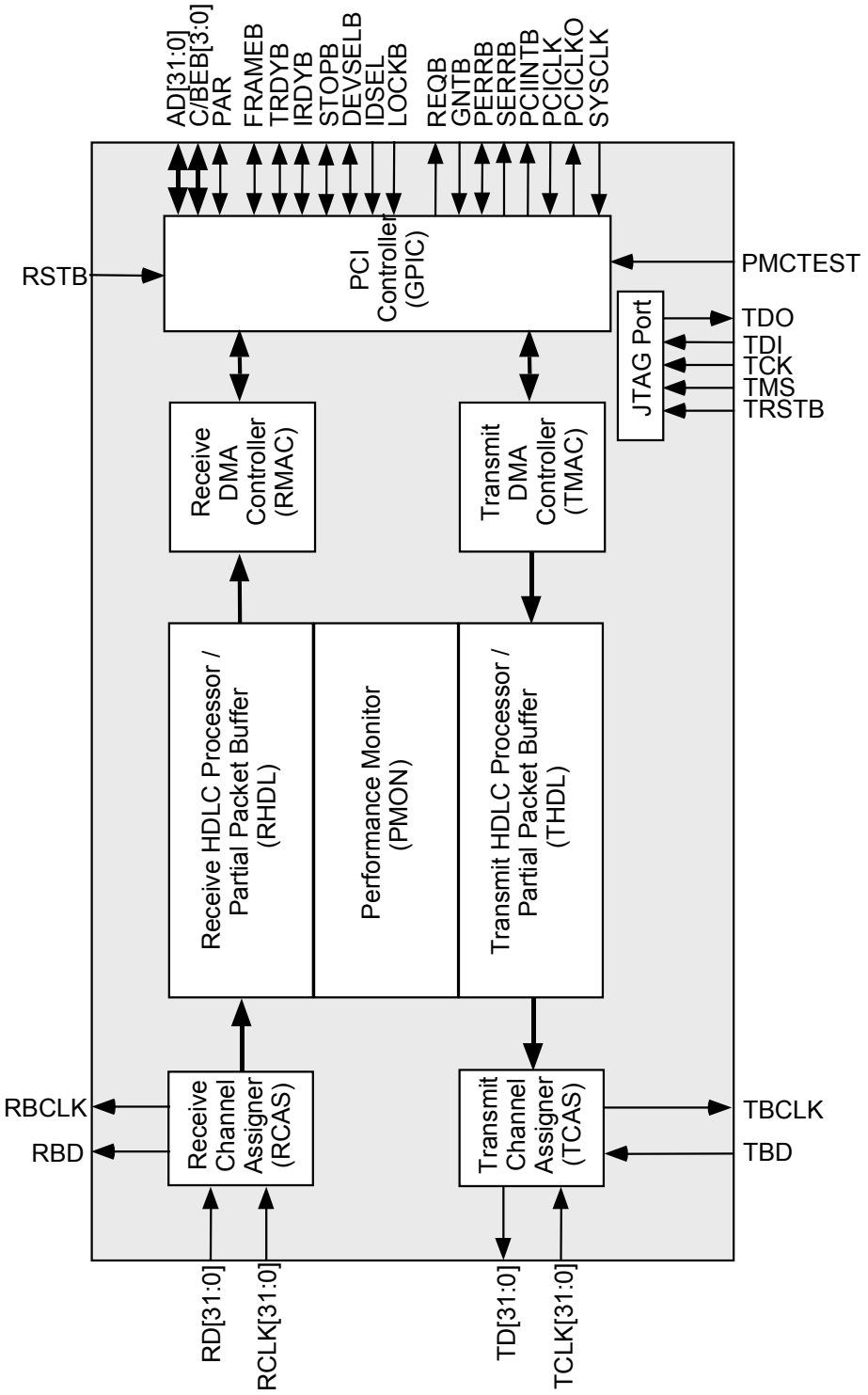
3. PCI Special Interest Group, PCI Local Bus Specification, June 1, 1995, Version 2.1.

4 APPLICATION EXAMPLES





5 BLOCK DIAGRAM



6 DESCRIPTION

The PM7364 FREEDM-32 Frame Engine and Datalink Manager device is a monolithic integrated circuit that implements HDLC processing, and PCI Bus memory management functions for a maximum of 128 bi-directional channels.

For channelised links, the FREEDM-32 allows up to 128 bi-directional HDLC channels to be assigned to individual time-slots within a maximum of 32 independently timed T1 or E1 links. The channel assignment supports the concatenation of time-slots (N x DS0) up to a maximum of 24 concatenated time-slots for a T1 link and 31 concatenated time-slots for an E1 link. Time-slots assigned to any particular channel need not be contiguous within the T1 or E1 link.

For unchannelised links, the FREEDM-32 processes up to 32 bi-directional HDLC channels within 32 independently timed links. The links can be of arbitrary frame format. When limited to two unchannelised links, each link can be rated at up to 45 MHz when SYSCLK is at 25 MHz and at up to 52 MHz when SYSCLK is at 33 MHz. For lower rate unchannelised links, the FREEDM-32 processes up to 32 links, where the aggregate clock rate of all the links is limited to 64 MHz, links 0 to 2 can have a clock rate of up to 45 MHz when SYSCLK is at or above 25 MHz and up to 52 MHz when SYSCLK is at 33 MHz and links 3 to 31 can have a clock rate of up to 10 MHz.

The FREEDM-32 supports mixing of up to 32 channelised and unchannelised links. The total number of channels in each direction is limited to 128. The aggregate clock rate over all 32 possible links is limited to 64 MHz.

In the receive direction, the FREEDM-32 performs channel assignment and packet extraction and validation. For each provisioned HDLC channel, the FREEDM-32 delineates the packet boundaries using flag sequence detection, and performs bit de-stuffing. Sharing of opening and closing flags, as well as, sharing of zeros between flags are supported. The resulting packet data is placed into the internal 8 kbyte partial packet buffer RAM. The partial packet buffer acts as a logical FIFO for each of the assigned channels. Partial packets are DMA'd out of the RAM, across the PCI bus and into host packet memory. The FREEDM-32 validates the frame check sequence for each packet, and verifies that the packet is an integral number of octets in length and is within a programmable minimum and maximum length. The receive packet status is updated before linking the packet into a receive ready queue. The FREEDM-32 alerts the PCI Host that there are packets in a receive ready queue by, optionally, asserting an interrupt on the PCI bus.

Alternatively, in the receive direction, the FREEDM-32 supports a transparent operating mode. For each provisioned transparent channel, the FREEDM-32 directly transfers the received octets into host memory verbatim. If the transparent channel is assigned to a channelised link, then the octets are aligned to the received time-slots.

In the transmit direction, the PCI Host provides packets to transmit using a transmit ready queue. For each provisioned HDLC channel, the FREEDM-32 DMA's partial packets across the PCI bus and into the transmit partial packet buffer. The partial packets are read out of the packet buffer by the FREEDM-32 and frame check sequence is optionally calculated and inserted at the end of each packet. Bit stuffing is performed before being assigned to a particular link. The flag sequence is automatically inserted when there is no packet data for a particular channel. Sequential packets are optionally separated by two flags (an opening flag and a closing flag) or a single flag (combined opening and closing flag). Zeros between flags are not shared. PCI bus latency may cause one or more channels to underflow, in which case, the packets are aborted, and the host is notified. For normal traffic, an abort sequence is generated, followed by inter-frame time fill characters (flags or all-ones bytes) until a new packet is sourced from the PCI host. No attempt is made to automatically re-transmit an aborted packet.

Alternatively, in the transmit direction, the FREEDM-32 supports a transparent operating mode. For each provisioned transparent channel, the FREEDM-32 directly inserts the transmitted octets from host memory. If the transparent channel is assigned to a channelised link, then the octets are aligned to the transmitted time-slots. If a channel underflows due to excessive PCI bus latency, an abort sequence is generated, followed by inter-frame time fill characters (flags or all-ones bytes) to indicate idle channel. Data resumes immediately when the FREEDM-32 receives new data from the host.

The FREEDM-32 is configured, controlled and monitored using the PCI bus interface. The FREEDM-32 is implemented in low power CMOS technology. It has TTL compatible inputs and outputs and is packaged in a 256 pin enhanced ball grid array (SBGA) package.

7 PIN DIAGRAM

The FREEDM-32 is manufactured in a 256 pin enhanced ball grid array package.

	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	
A	VSS	VSS	VSS	RCLK[8]	RCLK[10]	RD[12]	RD[14]	VSS	RD[17]	RD[19]	VSS	VSS	RCLK[21]	RCLK[23]	RCLK[25]	RD[27]	RD[29]	VSS	VSS	VSS	A
B	VSS	VDD	VDD	RD[8]	RD[10]	RCLK[11]	RCLK[13]	RCLK[15]	RCLK[16]	RCLK[18]	RD[20]	RCLK[20]	RCLK[22]	RD[24]	RD[26]	RCLK[27]	RCLK[29]	VDD	VDD	VSS	B
C	VSS	VDD	VDD	RD[7]	RD[9]	RD[11]	RCLK[12]	RCLK[14]	RD[16]	RD[18]	RCLK[19]	RD[21]	RD[23]	RD[25]	RCLK[26]	RCLK[28]	PCICLK	VDD	VDD	VSS	C
D	RD[5]	RCLK[5]	RCLK[6]	EN5V	RCLK[7]	RCLK[9]	VDD	RD[13]	RD[15]	RCLK[17]	VDD	RD[22]	RCLK[24]	VDD	RD[28]	PCICLKO	VBIAS[2]	GNTB	AD[31]	AD[30]	D
E	RD[3]	RCLK[3]	RCLK[4]	RD[6]													REQB	AD[29]	AD[27]	AD[26]	E
F	RCLK[1]	RD[2]	RCLK[2]	RD[4]													AD[28]	AD[25]	AD[24]	CBEB[3]	F
G	RBCLK	RD[0]	RD[1]	VDD													VDD	IDSEL	AD[22]	AD[21]	G
H	VBIAS[1]	SYSCLK	RBD	RCLK[0]													AD[23]	AD[20]	AD[18]	VSS	H
J	VSS	TCK	TMS	TRSTB													AD[19]	AD[17]	AD[16]	CBEB[2]	J
K	VSS	TDI	TDO	VDD						BOTTOM VIEW							FRAMEB	IRDYB	TRDYB	DEVSELB	K
L	TD[0]	TCLK[0]	TD[1]	TCLK[1]													VDD	STOPB	LOCKB	VSS	L
M	TD[2]	TCLK[2]	TD[3]	TD[4]													CBEB[1]	SERRB	PERRB	VSS	M
N	VSS	TCLK[3]	TCLK[4]	TD[6]													AD[11]	AD[14]	AD[15]	PAR	N
P	TD[5]	TCLK[5]	TCLK[6]	VDD													VDD	AD[10]	AD[12]	AD[13]	P
R	TD[7]	TCLK[7]	TD[8]	TCLK[9]													AD[5]	CBEB[0]	AD[8]	AD[9]	R
T	TCLK[8]	TD[9]	TD[10]	TCLK[11]													AD[1]	AD[4]	AD[6]	AD[7]	T
U	TCLK[10]	TD[11]	TD[12]	NC	TD[13]	TD[15]	VDD	TCLK[18]	TD[21]	VDD	TCLK[25]	TD[28]	TD[30]	VDD	PMCTEST	RCLK[30]	VBIAS[3]	AD[0]	AD[2]	AD[3]	U
V	VSS	VDD	VDD	TCLK[12]	TCLK[14]	TCLK[16]	TD[18]	TD[20]	TD[22]	TCLK[23]	TD[25]	TD[27]	TCLK[28]	TCLK[30]	TBD	PCIINTB	RD[30]	VDD	VDD	VSS	V
W	VSS	VDD	VDD	TCLK[13]	TCLK[15]	TD[17]	TD[19]	TCLK[20]	TCLK[22]	TD[23]	TCLK[24]	TCLK[26]	TCLK[27]	TCLK[29]	TCLK[31]	RSTB	RD[31]	VDD	VDD	VSS	W
Y	VSS	VSS	VSS	TD[14]	TD[16]	TCLK[17]	TCLK[19]	TCLK[21]	VSS	VSS	TD[24]	TD[26]	VSS	TD[29]	TD[31]	TBCLK	RCLK[31]	VSS	VSS	VSS	Y
	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	

8 PIN DESCRIPTION

Table 1 – Line Side Interface Signals (132)

Pin Name	Type	Pin No.	Function
RCLK[0] RCLK[1] RCLK[2] RCLK[3] RCLK[4] RCLK[5] RCLK[6] RCLK[7] RCLK[8] RCLK[9] RCLK[10] RCLK[11] RCLK[12] RCLK[13] RCLK[14] RCLK[15] RCLK[16] RCLK[17] RCLK[18] RCLK[19] RCLK[20] RCLK[21] RCLK[22] RCLK[23] RCLK[24] RCLK[25] RCLK[26] RCLK[27] RCLK[28] RCLK[29] RCLK[30] RCLK[31]	Input	H17 F20 F18 E19 E18 D19 D18 D16 A17 D15 A16 B15 C14 B14 C13 B13 B12 D11 B11 C10 B9 A8 B8 A7 D8 A6 C6 B5 C5 B4 U5 Y4	The receive line clock signals (RCLK[31:0]) contain the recovered line clock for the 32 independently timed links. Processing of the receive links is on a priority basis, in descending order from RCLK[0] to RCLK[31]. Therefore, the highest rate link should be connected to RCLK[0] and the lowest to RCLK[31]. RD[31:0] is sampled on the rising edge of the corresponding RCLK[31:0] clock. For channelised T1 or E1 links, RCLK[n] must be gapped during the framing bit (for T1 interfaces) or during time-slot 0 (for E1 interfaces) of the RD[n] stream. The FREEDM-32 uses the gapping information to determine the time-slot alignment in the receive stream. RCLK[31:0] is nominally a 50% duty cycle clock of 1.544 MHz for T1 links and 2.048 MHz for E1 links. For unchannelised links, RCLK[n] must be externally gapped during the bits or time-slots that are not part of the transmission format payload (i.e. not part of the HDLC packet). RCLK[2:0] is nominally a 50% duty cycle clock between 0 and 52 MHz. RCLK[31:3] is nominally a 50% duty cycle clock between 0 and 10 MHz.

Pin Name	Type	Pin No.	Function
RD[0] RD[1] RD[2] RD[3] RD[4] RD[5] RD[6] RD[7] RD[8] RD[9] RD[10] RD[11] RD[12] RD[13] RD[14] RD[15] RD[16] RD[17] RD[18] RD[19] RD[20] RD[21] RD[22] RD[23] RD[24] RD[25] RD[26] RD[27] RD[28] RD[29] RD[30] RD[31]	Input	G19 G18 F19 E20 F17 D20 E17 C17 B17 C16 B16 C15 A15 D13 A14 D12 C12 A12 C11 A11 B10 C9 D9 C8 B7 C7 B6 A5 D6 A4 V4 W4	The receive data signals (RD[31:0]) contain the recovered line data for the 32 independently timed links in normal mode (PMCTEST set low). Processing of the receive links is on a priority basis, in descending order form RD[0] to RD[31]. Therefore, the highest rate link should be connected to RD[0] and the lowest to RD[31]. For channelised links, RD[n] contains the 24 (T1) or 31 (E1) time-slots that comprise the channelised link. RCLK[n] must be gapped during the T1 framing bit position or the E1 frame alignment signal (time-slot 0). The FREEDM-32 uses the location of the gap to determine the channel alignment on RD[n]. For unchannelised links, RD[n] contains the HDLC packet data. For certain transmission formats, RD[n] may contain place holder bits or time-slots. RCLK[n] must be externally gapped during the place holder positions in the RD[n] stream. The FREEDM-32 supports a maximum data rate of 10 Mbit/s on an individual RD[31:3] link and a maximum data rate of 52 Mbit/s on RD[2:0]. RD[31:0] is sampled on the rising edge of the corresponding RCLK[31:0] clock.
RBD	Tristate Output	H18	The receive BERT data signal (RBD) contains the receive bit error rate test data. RBD reports the data on the selected one of the receive data signals (RD[31:0]) and is updated on the falling edge of RBCLK. RBD may be tri-stated by setting the RBEN bit in the FREEDM-32 Master BERT Control register low.

Pin Name	Type	Pin No.	Function
RBCLK	Tristate Output	G20	The receive BERT clock signal (RBCLK) contains the receive bit error rate test clock. RBCLK is a buffered version of the selected one of the receive clock signals (RCLK[31:0]). RBCLK may be tri-stated by setting the RBEN bit in the FREEDM-32 Master BERT Control register low.
TCLK[0] TCLK[1] TCLK[2] TCLK[3] TCLK[4] TCLK[5] TCLK[6] TCLK[7] TCLK[8] TCLK[9] TCLK[10] TCLK[11] TCLK[12] TCLK[13] TCLK[14] TCLK[15] TCLK[16] TCLK[17] TCLK[18] TCLK[19] TCLK[20] TCLK[21] TCLK[22] TCLK[23] TCLK[24] TCLK[25] TCLK[26] TCLK[27] TCLK[28] TCLK[29] TCLK[30] TCLK[31]	Input	L19 L17 M19 N19 N18 P19 P18 R19 T20 R17 U20 T17 V17 W17 V16 W16 V15 Y15 U13 Y14 W13 Y13 W12 V11 W10 U10 W9 W8 V8 W7 V7 W6	The transmit line clock signals (TCLK[31:0]) contain the transmit clocks for the 32 independently timed links. Processing of the transmit links is on a priority basis, in descending order from TCLK[0] to TCLK[31]. Therefore, the highest rate link should be connected to TCLK[0] and the lowest to TCLK[31]. TD[31:0] is updated on the falling edge of the corresponding TCLK[31:0] clock. For channelised T1 or E1 links, TCLK[n] must be gapped during the framing bit (for T1 interfaces) or during time-slot 0 (for E1 interfaces) of the TD[n] stream. The FREEDM-32 uses the gapping information to determine the time-slot alignment in the transmit stream. For unchannelised links, TCLK[n] must be externally gapped during the bits or time-slots that are not part of the transmission format payload (i.e. not part of the HDLC packet). TCLK[31:3] is nominally a 50% duty cycle clock between 0 and 10 MHz. TCLK[2:0] is nominally a 50% duty cycle clock between 0 and 52 MHz. Typical values for TCLK[31:0] include 1.544 MHz (for T1 links) and 2.048 MHz (for E1 links).

Pin Name	Type	Pin No.	Function
TD[0] TD[1] TD[2] TD[3] TD[4] TD[5] TD[6] TD[7] TD[8] TD[9] TD[10] TD[11] TD[12] TD[13] TD[14] TD[15] TD[16] TD[17] TD[18] TD[19] TD[20] TD[21] TD[22] TD[23] TD[24] TD[25] TD[26] TD[27] TD[28] TD[29] TD[30] TD[31]	Output	L20 L18 M20 M18 M17 P20 N17 R20 R18 T19 T18 U19 U18 U16 Y17 U15 Y16 W15 V14 W14 V13 U12 V12 W11 Y10 V10 Y9 V9 U9 Y7 U8 Y6	The transmit data signals (TD[31:0]) contains the transmit data for the 32 independently timed links in normal mode (PMCTEST set low). Processing of the transmit links is on a priority basis, in descending order from TD[0] to TD[31]. Therefore, the highest rate link should be connected to TD[0] and the lowest to TD[31]. For channelised links, TD[n] contains the 24 (T1) or 31 (E1) time-slots that comprise the channelised link. TCLK[n] must be gapped during the T1 framing bit position or the E1 frame alignment signal (time-slot 0). The FREEDM-32 uses the location of the gap to determine the channel alignment on TD[n]. For unchannelised links, TD[n] contains the HDLC packet data. For certain transmission formats, TD[n] may contain place holder bits or time-slots. TCLK[n] must be externally gapped during the place holder positions in the TD[n] stream. The FREEDM-32 supports a maximum data rate of 10 Mbit/s on an individual TD[31:3] link and a maximum data rate of 52 Mbit/s on TD[2:0] TD[31:0] is updated on the falling edge of the corresponding TCLK[31:0] clock.
TBD	Input	V6	The transmit BERT data signal (TBD) contains the transmit bit error rate test data. When the TBERTEN bit in the BERT Control register is set high, the data on TBD is transmitted on the selected one of the transmit data signals (TD[31:0]). TBD is sampled on the rising edge of TBCLK.

Pin Name	Type	Pin No.	Function
TBCLK	Tristate Output	Y5	The transmit BERT clock signal (TBCLK) contains the transmit bit error rate test clock. TBCLK is a buffered version of the selected one of the transmit clock signals (TCLK[31:0]). TBCLK may be tri-stated by setting the TBEN bit in the FREEDM-32 Master BERT Control register low.

Table 2 – PCI Host Interface Signals (51)

Pin Name	Type	Pin No.	Function
PCICLK	Input	C4	The PCI clock signal (PCICLK) provides timing for PCI bus accesses. PCICLK is a nominally 50% duty cycle, 0 to 33 MHz clock.
PCICLKO	Output	D5	The PCI clock output signal (PCICLKO) is a buffered version of the PCICLK. PCICLKO may be used to drive the SYSCLK input.
AD[0] AD[1] AD[2] AD[3] AD[4] AD[5] AD[6] AD[7] AD[8] AD[9] AD[10] AD[11] AD[12] AD[13] AD[14] AD[15] AD[16] AD[17] AD[18] AD[19] AD[20] AD[21] AD[22] AD[23] AD[24] AD[25] AD[26] AD[27] AD[28] AD[29] AD[30]	I/O	U3 T4 U2 U1 T3 R4 T2 T1 R2 R1 P3 N4 P2 P1 N3 N2 J2 J3 H2 J4 H3 G1 G2 H4 F2 F3 E1 E2 F4 E3 D1	The PCI address and data bus (AD[31:0]) carries the PCI bus multiplexed address and data. During the first clock cycle of a transaction, AD[31:0] contains a physical byte address. During subsequent clock cycles of a transaction, AD[31:0] contains data. A transaction is defined as an address phase followed by one or more data phases. When Little-Endian byte formatting is selected, AD[31:24] contain the most significant byte of a DWORD while AD[7:0] contain the least significant byte. When Big-Endian byte formatting is selected. AD[7:0] contain the most significant byte of a DWORD while AD[31:24] contain the least significant byte. When the FREEDM-32 is the initiator, AD[31:0] is an output bus during the first (address) phase of a transaction. For write transactions, AD[31:0] remains an output bus for the data phases of the transaction. For read transactions, AD[31:0] is an input bus during the data phases. When the FREEDM-32 is the target, AD[31:0] is an input bus during the first (address) phase of a transaction. For write transactions, AD[31:0] remains an input bus during the data phases of the transaction. For read transactions, AD[31:0] is an output bus during the data phases. When the FREEDM-32 is not involved in the current transaction, AD[31:0] is tri-stated.

Pin Name	Type	Pin No.	Function
AD[31]	I/O	D2	As an output bus, AD[31:0] is updated on the rising edge of PCICLK. As an input bus, AD[31:0] is sampled on the rising edge of PCICLK.
C/BEB[0] C/BEB[1] C/BEB[2] C/BEB[3]	I/O	R3 M4 J1 F1	The PCI bus command and byte enable bus (C/BEB[3:0]) contains the bus command or the byte valid indications. During the first clock cycle of a transaction, C/BEB[3:0] contains the bus command code. For subsequent clock cycles, C/BEB[3:0] identifies which bytes on the AD[31:0] bus carry valid data. C/BEB[3] is associated with byte 3 (AD[31:24]) while C/BEB[0] is associated with byte 0 (AD[7:0]). When C/BEB[n] is set high, the associated byte is invalid. When C/BEB[n] is set low, the associated byte is valid. When the FREEDM-32 is the initiator, C/BEB[3:0] is an output bus. When the FREEDM-32 is the target, C/BEB[3:0] is an input bus. When the FREEDM-32 is not involved in the current transaction, C/BEB[3:0] is tri-stated. As an output bus, C/BEB[3:0] is updated on the rising edge of PCICLK. As an input bus, C/BEB[3:0] is sampled on the rising edge of PCICLK.

Pin Name	Type	Pin No.	Function
PAR	I/O	N1	The parity signal (PAR) indicates the parity of the AD[31:0] and C/BEB[3:0] buses. Even parity is calculated over all 36 signals in the buses regardless of whether any or all the bytes on the AD[31:0] are valid. PAR always reports the parity of the previous PCICLK cycle. Parity errors detected by the FREEDM-32 are indicated on output PERRB and in the FREEDM-32 Interrupt Status register. When the FREEDM-32 is the initiator, PAR is an output for writes and an input for reads. When the FREEDM-32 is the target, PAR is an input for writes and an output for reads. When the FREEDM-32 is not involved in the current transaction, PAR is tri-stated. As an output signal, PAR is updated on the rising edge of PCICLK. As an input signal, PAR is sampled on the rising edge of PCICLK.
FRAMEB	I/O	K4	The active low cycle frame signal (FRAMEB) identifies a transaction cycle. When FRAMEB transitions low, the start of a bus transaction is indicated. FRAMEB remains low to define the duration of the cycle. When FRAMEB transitions high, the last data phase of the current transaction is indicated. When the FREEDM-32 is the initiator, FRAMEB is an output. When the FREEDM-32 is the target, FRAMEB is an input. When the FREEDM-32 is not involved in the current transaction, FRAMEB is tri-stated. As an output signal, FRAMEB is updated on the rising edge of PCICLK. As an input signal, FRAMEB is sampled on the rising edge of PCICLK.

Pin Name	Type	Pin No.	Function
TRDYB	I/O	K2	The active low target ready signal (TRDYB) indicates when the target is ready to start or continue with a transaction. TRDYB works in conjunction with IRDYB to complete transaction data phases. During a transaction in progress, TRDYB is set high to indicate that the target cannot complete the current data phase and to force a wait state. TRDYB is set low to indicate that the target can complete the current data phase. The data phase is completed when TRDYB is set low and the initiator ready signal (IRDYB) is also set low. When the FREEDM-32 is the initiator, TRDYB is an input. When the FREEDM-32 is the target, TRDYB is an output. During accesses to FREEDM-32 registers, TRDYB is set high to extend data phases over multiple PCICLK cycles. When the FREEDM-32 is not involved in the current transaction, TRDYB is tri-stated. As an output signal, TRDYB is updated on the rising edge of PCICLK. As an input signal, TRDYB is sampled on the rising edge of PCICLK.

Pin Name	Type	Pin No.	Function
IRDYB	I/O	K3	The active low initiator ready (IRDYB) signal is used to indicate whether the initiator is ready to start or continue with a transaction. IRDYB works in conjunction with TRDYB to complete transaction data phases. When IRDYB is set high and a transaction is in progress, the initiator is indicating it cannot complete the current data phase and is forcing a wait state. When IRDYB is set low and a transaction is in progress, the initiator is indicating it has completed the current data phase. The data phase is completed when IRDYB is set low and the target ready signal (IRDYB) is also set low. When the FREEDM-32 is the initiator, IRDYB is an output. When the FREEDM-32 is the target, IRDYB is an input. When the FREEDM-32 is not involved in the current transaction, IRDYB is tri-stated. IRDYB is updated on the rising edge of PCICLK or sampled on the rising edge of PCICLK depending on whether it is an output or an input.

Pin Name	Type	Pin No.	Function
STOPB	I/O	L3	The active low stop signal (STOPB) requests the initiator to stop the current bus transaction. When STOPB is set high by a target, the initiator continues with the transaction. When STOPB is set low, the initiator will stop the current transaction. When the FREEDM-32 is the initiator, STOPB is an input. When STOPB is sampled low, the FREEDM-32 will terminate the current transaction in the next PCICLK cycle. When the FREEDM-32 is the target, STOPB is an output. The FREEDM-32 only issues transaction stop requests when responding to reads and writes to configuration space (disconnecting after 1 DWORD transferred) or if an initiator introduces wait states during a transaction. When the FREEDM-32 is not involved in the current transaction, STOPB is tri-stated. STOPB is updated on the rising edge of PCICLK or sampled on the rising edge of PCICLK depending on whether it is an output or an input.
IDSEL	Input	G3	The initialization device select signal (IDSEL) enables read and write access to the PCI configuration registers. When IDSEL is set high during the address phase of a transaction and the C/BEB[3:0] code indicates a register read or write, the FREEDM-32 performs a PCI configuration register transaction and asserts the DEVSELB signal in the next PCICLK period. IDSEL is sampled on the rising edge of PCICLK.

Pin Name	Type	Pin No.	Function
DEVSELB	I/O	K1	The active low device select signal (DEVSELB) indicates that a target claims the current bus transaction. During the address phase of a transaction, all targets decode the address on the AD[31:0] bus. When a target, recognizes the address as its own, it sets DEVSELB low to indicate to the initiator that the address is valid. If no target claims the address in six bus clock cycles, the initiator assumes that the target does not exist or cannot respond and aborts the transaction. When the FREEDM-32 is the initiator, DEVSELB is an input. If no target responds to an address in six PCICLK cycles, the FREEDM-32 will abort the current transaction and alerts the PCI Host via an interrupt. When the FREEDM-32 is the target, DEVSELB is an output. DELSELB is set low when the address on AD[31:0] is recognised. When the FREEDM-32 is not involved in the current transaction, DEVSELB is tri-stated. FREEDM-32 is updated on the rising edge of PCICLK or sampled on the rising edge of PCICLK depending on whether it is an output or an input.
LOCKB	Input	L2	The active low bus lock signal (LOCKB) locks a target device. When LOCKB and FRAME are set low, and the FREEDM-32 is the target, an initiator is locking the FREEDM-32 as an "owned" target. Under these circumstances, the FREEDM-32 will reject all transaction with other initiators. The FREEDM-32 will continue to reject other initiators until its owner releases the lock by forcing both FRAMEB and LOCKB high. As a initiator, the FREEDM-32 will never lock a target. LOCKB is sampled using the rising edge of PCICLK.

Pin Name	Type	Pin No.	Function
REQB	Output	E4	The active low PCI bus request signal (REQB) requests an external arbiter for control of the PCI bus. REQB is set low when the FREEDM-32 desires access to the host memory. REQB is set high when access is not desired. REQB is updated on the rising edge of PCICLK.
GNTB	Input	D3	The active low PCI bus grant signal (GNTB) indicates the granting of control over the PCI in response to a bus request via the REQB output. When GNTB is set high, the FREEDM-32 does not have control over the PCI bus. When GNTB is set low, the external arbiter has granted the FREEDM-32 control over the PCI bus. However, the FREEDM-32 will not proceed until the FRAMEB signal is sampled high, indicating no current transactions are in progress. GNTB is sampled on the rising edge of PCICLK.
PCIINTB	OD Output	V5	The active low PCI interrupt signal (PCIINTB) is set low when a FREEDM-32 interrupt source is active, and that source is unmasked. The FREEDM-32 may be enabled to report many alarms or events via interrupts. PCIINTB returns high when the interrupt is acknowledged via an appropriate register access. PCIINTB is an open drain output and is updated on the rising edge of PCICLK.

Pin Name	Type	Pin No.	Function
PERRB	I/O	M2	The active low parity error signal (PERRB) indicates a parity error over the AD[31:0] and C/BEB[3:0] buses. Parity error is signalled when even parity calculations do not match the PAR signal. PERRB is set low at the cycle immediately following an offending PAR cycle. PERRB is set high when no parity error is detected. PERRB is enabled by setting the PERREN bit in the Control/Status register in the PCI Configuration registers space. Regardless of the setting of PERREN, parity errors are always reported by the PERR bit in the Control/Status register in the PCI Configuration registers space. PERRB is updated on the rising edge of PCICLK.
SERRB	OD Output	M3	The active low system error signal (SERRB) indicates an address parity error. Address parity errors are detected when the even parity calculations during the address phase do not match the PAR signal. When the FREEDM-32 detects a system error, SERRB is set low for one PCICLK period. SERRB is enabled by setting the SERREN bit in the Control/Status register in the PCI Configuration registers space. Regardless of the setting of SERREN, parity errors are always reported by the SERR bit in the Control/Status register in the PCI Configuration registers space. SERRB is an open drain output and is updated on the rising edge of PCICLK.

Table 3 – Miscellaneous Interface Signals (13)

Pin Name	Type	Pin No.	Function
SYSCLK	Input	H19	The system clock (SYSCLK) provides timing for the core logic. SYSCLK is nominally a 50% duty cycle 25 MHz to 33 MHz clock.
RSTB	Input	W5	The active low reset signal (RSTB) signal provides an asynchronous FREEDM-32 reset. RSTB is an asynchronous input. When RSTB is set low, all FREEDM-32 registers are forced to their default states. In addition, TD[31:0] are forced high and all PCI output pins are forced tri-state and will remain high or tri-stated, respectively, until RSTB is set high.
PMCTEST	Input	U6	The PMC production test enable signal (PMCTEST) places the FREEDM-32 is test mode. When PMCTEST is set high, production test vectors can be executed to verify manufacturing via the test mode interface signals TA[10:0], TA[11]/TRS, TRDB, TWRB and TDAT[15:0]. PMCTEST must be tied low in normal operation.
TCK	Input	J19	The test clock signal (TCK) provides timing for test operations that can be carried out using the IEEE P1149.1 test access port. TMS and TDI are sampled on the rising edge of TCK. TDO is updated on the falling edge of TCK.
TMS	Input	J18	The test mode select signal (TMS) controls the test operations that can be carried out using the IEEE P1149.1 test access port. TMS is sampled on the rising edge of TCK. TMS has an integral pull up resistor.
TDI	Input	K19	The test data input signal (TDI) carries test data into the FREEDM-32 via the IEEE P1149.1 test access port. TDI is sampled on the rising edge of TCK. TDI has an integral pull up resistor.

Pin Name	Type	Pin No.	Function
TDO	Tristate Output	K18	The test data output signal (TDO) carries test data out of the FREEDM-32 via the IEEE P1149.1 test access port. TDO is updated on the falling edge of TCK. TDO is a tri-state output which is inactive except when scanning of data is in progress.
TRSTB	Input	J17	The active low test reset signal (TRSTB) provides an asynchronous FREEDM-32 test access port reset via the IEEE P1149.1 test access port. TRSTB is an asynchronous input with an integral pull up resistor. Note that when TRSTB is not being used, it must be connected to the RSTB input.
VBIAS[3:1]	Input	U4 D4 H20	The bias signals (VBIAS[3:1]) provide 5 Volt bias to input and I/O pads to allow the FREEDM-32 to tolerate connections to 5 Volt devices. To avoid damage to the device, the VBIAS[3:1] signals must be connected together externally and must at all times be kept at a voltage that is equal to or higher than the VDD[28:1] power supplies. In a 3.3V operating environment, VBIAS[3:1] and VDD[28:1] may be connected together. In a 5V operating environment, VBIAS[3:1] should be powered up to 5V before VDD[28:1] are powered up to 3.3V.
EN5V	Input	D17	The 5 Volt PCI signalling enable signal (EN5V) causes the PCI Host Interface Signals to operate in the 5V PCI signalling environment when set high and the 3.3V PCI signalling environment when set low. EN5V is an asynchronous input with an integral pull up resistor.
NC	Open	U17	This pin must be left unconnected.

Table 4 – Production Test Interface Signals (0 - Multiplexed)

Pin Name	Type	Pin No.	Function
TA[0] TA[1] TA[2] TA[3] TA[4] TA[5] TA[6] TA[7] TA[8] TA[9] TA[10]	Input		The test mode address bus (TA[10:0]) selects specific registers during production test (PMCTEST set high) read and write accesses. TA[10:0] replace RD[20:10] when PMCTEST is set high.
TA[11]/TRS	Input		The test register select signal (TA[11]/TRS) selects between normal and test mode register accesses during production test (PMCTEST set high). TRS is set high to select test registers and is set low to select normal registers. TA[11]/TRS replaces RD[21] when PMCTEST is set high.
TRDB	Input		The test mode read enable signal (TRDB) is set low during FREEDM-32 register read accesses during production test (PMCTEST set high). The FREEDM-32 drives the test data bus (TDAT[15:0]) with the contents of the addressed register while TRDB is low. TRDB replaces RD[22] when PMCTEST is set high.
TWRB	Input		The test mode write enable signal (TWRB) is set low during FREEDM-32 register write accesses during production test (PMCTEST set high). The contents of the test data bus (TDAT[15:0]) are clocked into the addressed register on the rising edge of TWRB. TWRB replaces RD[23] when PMCTEST is set high.

Pin Name	Type	Pin No.	Function
TDAT[0] TDAT[1] TDAT[2] TDAT[3] TDAT[4] TDAT[5] TDAT[6] TDAT[7] TDAT[8] TDAT[9] TDAT[10] TDAT[11] TDAT[12] TDAT[13] TDAT[14] TDAT[15]	I/O		The bi-directional test mode data bus (TDAT[15:0]) carries data read from or written to FREEDM-32 registers during production test. TDAT[15:0] replace TD[31:16] when PMCTEST is set high.

Table 5 – Power and Ground Signals (60)

Pin Name	Type	Pin No.	Function
VDD1 VDD2 VDD3 VDD4 VDD5 VDD6 VDD7 VDD8 VDD9 VDD10 VDD11 VDD12 VDD13 VDD14 VDD15 VDD16 VDD17 VDD18 VDD19 VDD20 VDD21 VDD22 VDD23 VDD24 VDD25 VDD26 VDD27 VDD28	Power	B2 B3 B18 B19 C2 C3 C18 C19 D7 D10 D14 G4 G17 K17 L4 P4 P17 U7 U11 U14 V2 V3 V18 V19 W2 W3 W1 8 W1 9	The DC power pins should be connected to a well decoupled +3.3 V DC supply.

Pin Name	Type	Pin No.	Function
VSS1 VSS2 VSS3 VSS4 VSS5 VSS6 VSS7 VSS8 VSS9 VSS10 VSS11 VSS12 VSS13 VSS14 VSS15 VSS16 VSS17 VSS18 VSS19 VSS20 VSS21 VSS22 VSS23 VSS24 VSS25 VSS26 VSS27 VSS28 VSS29 VSS30 VSS31 VSS32	Ground	A1 A2 A3 A9 A10 A13 A18 A19 A20 B1 B20 C1 C20 H1 J20 K20 L1 M1 N20 V1 V20 W1 W2 0 Y1 Y2 Y3 Y8 Y11 Y12 Y18 Y19 Y20	The DC ground pins should be connected to ground.

Notes on Pin Description:

1. All FREEDM-32 inputs and bi-directionals present minimum capacitive loading and operate at TTL compatible logic levels. PCI signals conform to the 3.3 or 5 Volt signaling environment depending on the setting of the EN5V input.

2. Most FREEDM-32 non-PCI digital outputs and bi-directionals have 4 mA drive capability, except the PCICLKO, RBCLK, TBCLK, RBD and PCIINTB outputs which have 6 mA drive capability and the TD[0], TD[1], and TD[2] outputs which have 8 mA drive capability.

3. All FREEDM-32 non-PCI digital outputs and bi-directionals are 5 V tolerant when tristated except those with 8 mA drive capability, i.e. TD[2:0]. (TD[2:0] are never tristated in normal operation – only under JTAG boundary scan control.)

4. Inputs TMS, TDI, TRSTB and EN5V are Schmitt triggered and have internal pull-up resistors.

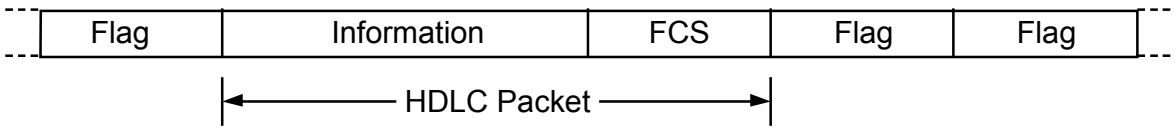
5. Inputs RD[31:0], RCLK[31:0], TCLK[31:0], SYSCLK, PCICLK, TBD, RSTB, GNTB, IDSEL, LOCKB, TCK and PMCTEST are Schmitt triggered.

9 FUNCTIONAL DESCRIPTION

9.1 High-Level Data Link Control Protocol

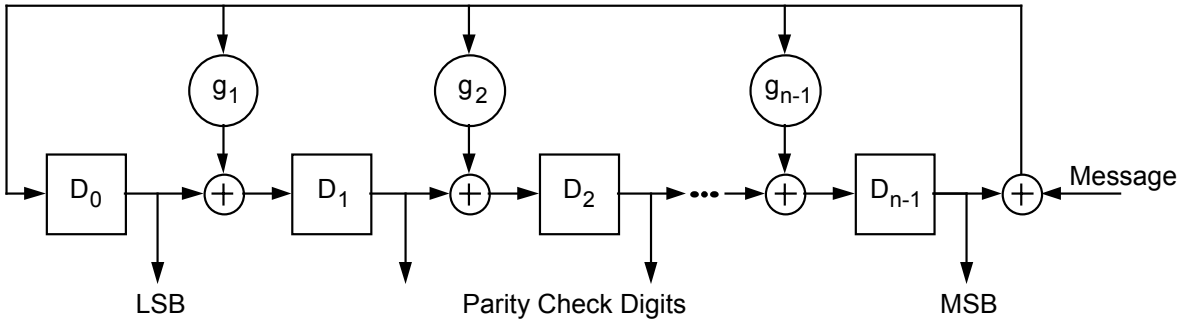
Figure 1 shows a diagram of the synchronous HDLC protocol supported by the FREEDM-32. The incoming stream is examined for flag bytes (01111110 bit pattern) which delineate the opening and closing of the HDLC packet. The packet is bit de-stuffed which discards a "0" bit which directly follows five contiguous "1" bits. The resulting HDLC packet size must be a multiple of an octet (8 bits) and within the expected minimum and maximum packet length limits. The minimum packet length is that of a packet containing two information bytes (address and control) and FCS bytes. For packets with CRC-CCITT as FCS, the minimum packet length is four bytes while those with CRC-32 as FCS, the minimum length is six bytes. An HDLC packet is aborted when seven contiguous "1" bits (with no inserted "0" bits) are received. At least one flag byte must exist between HDLC packets for delineation. Contiguous flag bytes, or all ones bytes between packets are used as an "inter-frame time fill". Adjacent flag bytes may share zeros.

Figure 1 – HDLC Frame



The CRC algorithm for the frame checking sequence (FCS) field is either a CRC-CCITT or CRC-32 function. Figure 2 shows a CRC encoder block diagram using the generating polynomial $g(X) = 1 + g_1X + g_2X^2 + \ldots + g_{n-1}X^{n-1} + X^n$. The CRC-CCITT FCS is two bytes in size and has a generating polynomial $g(X) = 1 + X^5 + X^{12} + X^{16}$. The CRC-32 FCS is four bytes in size and has a generating polynomial $g(X) = 1 + X + X^2 + X^4 + X^5 + X^7 + X^8 + X^{10} + X^{11} + X^{12} + X^{16} + X^{22} + X^{23} + X^{26} + X^{32}$. The first FCS bit received is the residue of the highest term.

Figure 2 – CRC Generator



9.2 Receive Channel Assigner

The Receive Channel Assigner block (RCAS) processes up to 32 serial links. Each link is independent and has its own associated clock. For each link, the RCAS performs a serial to parallel conversion to form data bytes. The data bytes are multiplexed, in byte serial format, for delivery to the Receive HDLC Processor / Partial Packet Buffer block (RHDL) at SYSCLK rate. In the event where multiple streams have accumulated a byte of data, multiplexing is performed on a fixed priority basis with link #0 having the highest priority and link #31 the lowest.

Links containing a T1 or an E1 stream may be channelised. Data at each time-slot may be independently assigned to a different channel. The RCAS performs a table lookup to associate the link and time-slot identity with a channel. T1 and E1 framing bits/bytes are identified by observing the gap in the link clock which is squelched during the framing bits/bytes. For unchannelised links, clock rates are limited to 52 MHz on link #0 to #2 and limited to 10 MHz for the remaining links. All data on each link belongs to one channel. For the case of two unchannelised links, the maximum link rate is 45 MHz for SYSCLK at 25 MHz and 52 MHz for SYSCLK at 33 MHz. For the case of more numerous unchannelised links or a mixture of channelise with unchannelised links, the total instantaneous link rate over all the links is limited to 64 MHz. The RCAS performs a table lookup using only the link number to determine the associated channel, as time-slots are non-existent in unchannelised links.

The RCAS provides diagnostic loopback that is selectable on a per channel basis. When a channel is in diagnostic loopback, stream data on the received links originally destined for that channel is ignored. Transmit data of that channel is substituted in its place.

9.2.1 Line Interface

There are 32 identical line interface blocks in the RCAS. Each line interface contains a bit counter, an 8-bit shift register and a holding register, that, together, perform serial to parallel conversion. Whenever the holding register is updated, a request for service is sent to the priority encoder block. When acknowledged by the priority encoder, the line interface would respond with the data residing in the holding register.

To support channelised links, each line interface block contains a time-slot counter and a clock activity monitor. The time-slot counter is incremented each time the holding register is updated. The clock activity monitor is a counter that increments at the system clock (SYSCLK) rate and is cleared by a rising edge of the receive clock (RCLK[n]). A framing bit (T1) or framing byte (E1) is detected when the counter reaches a programmable threshold. In which case, the bit and time-slot counters are initialised to indicate that the next bit is the most significant bit of the first time-slot. For unchannelised links, the time-slot counter and the clock activity monitor are held reset.

9.2.2 Priority Encoder

The priority encoder monitors the line interfaces for requests and synchronises them to the SYSCLK timing domain. Requests are serviced on a fixed priority scheme where highest to lowest priority is assigned from the line interface attached to RD[0] to that attached to RD[31]. Thus, simultaneous requests from RD[m] will be serviced ahead of RD[n], if m < n. When there are no pending requests, the priority encoder generates an idle cycle. In addition, once every fourth SYSCLK cycle, the priority encoder inserts a null cycle where no requests are serviced. This cycle is used by the channel assigner downstream for host microprocessor accesses to the provisioning RAMs.

9.2.3 Channel Assigner

The channel assigner block determines the channel number of the data byte currently being processed. The block contains a 1024 word channel provision RAM. The address of the RAM is constructed from concatenating the link number and the time-slot number of the current data byte. The fields of each RAM word include the channel number and a time-slot enable flag. The time-slot enable flag labels the current time-slot as belonging to the channel indicted by the channel number field.

9.2.4 Loopback Controller

The loopback controller block implements the channel based diagnostic loopback function. Every valid data byte belonging to a channel with diagnostic loopback enabled from the Transmit HDLC Processor / Partial Packet Buffer block (THDL) is written into a 64 word FIFO. The loopback controller monitors for an idle time-slot or a time-slot carrying a channel with diagnostic loopback enabled. If either conditions hold, the current data byte is replaced by data retrieved from the loopback data FIFO.

9.3 Receive HDLC Processor / Partial Packet Buffer

The Receive HDLC Processor / Partial Packet Buffer block (RHDL) processes up to 128 synchronous transmission HDLC data streams. Each channel can be individually configured to perform flag sequence detection, bit de-stuffing and CRC-CCITT or CRC-32 verification. The packet data is written into the partial packet buffer. At the end of a frame, packet status including CRC error, octet alignment error and maximum length violation are also loaded into the partial packet buffer. Alternatively, a channel can be provisioned as transparent, in which case, the HDLC data stream is passed to the partial packet buffer processor verbatim.

There is a natural precedence in the alarms detectable on a receive packet. Once a packet exceeds the programmable maximum packet length, no further processing is performed on it. Thus, octet alignment detection, FCS verification and abort recognition are squelched on packets with a maximum length violation. An abort indication squelches octet alignment detection, minimum packet length violations, and FCS verification. In addition, FCS verification is only performed on packets that do not have octet alignment errors, in order to allow the RHDL to perform CRC calculations on a byte-basis.

The partial packet buffer is an 8 Kbyte RAM that is divided into 16-byte blocks. Each block has an associated pointer which points to another block. A logical FIFO is created for each provisioned channel by programming the block pointers to form a circular linked list. A channel FIFO can be assigned a minimum of 3 blocks (48 bytes) and a maximum of 512 blocks (8 Kbytes). The depth of the channel FIFOs are monitored in a round-robin fashion. Requests are made to the Receive DMA Controller block (RMAC) to transfer, to the PCI host memory, data in channel FIFOs with depths exceeding their associated threshold.

9.3.1 HDLC Processor

The HDLC processor is a time-slice state machine which can process up to 128 independent channels. The state vector and provisioning information for each

channel is stored in a RAM. Whenever new channel data arrives, the appropriate state vector is read from the RAM, processed and written back to the RAM. The HDLC state-machine can be configured to perform flag delineation, bit de-stuffing, CRC verification and length monitoring. The resulting HDLC data and status information is passed to the partial packet buffer processor to be stored in the appropriate channel FIFO buffer.

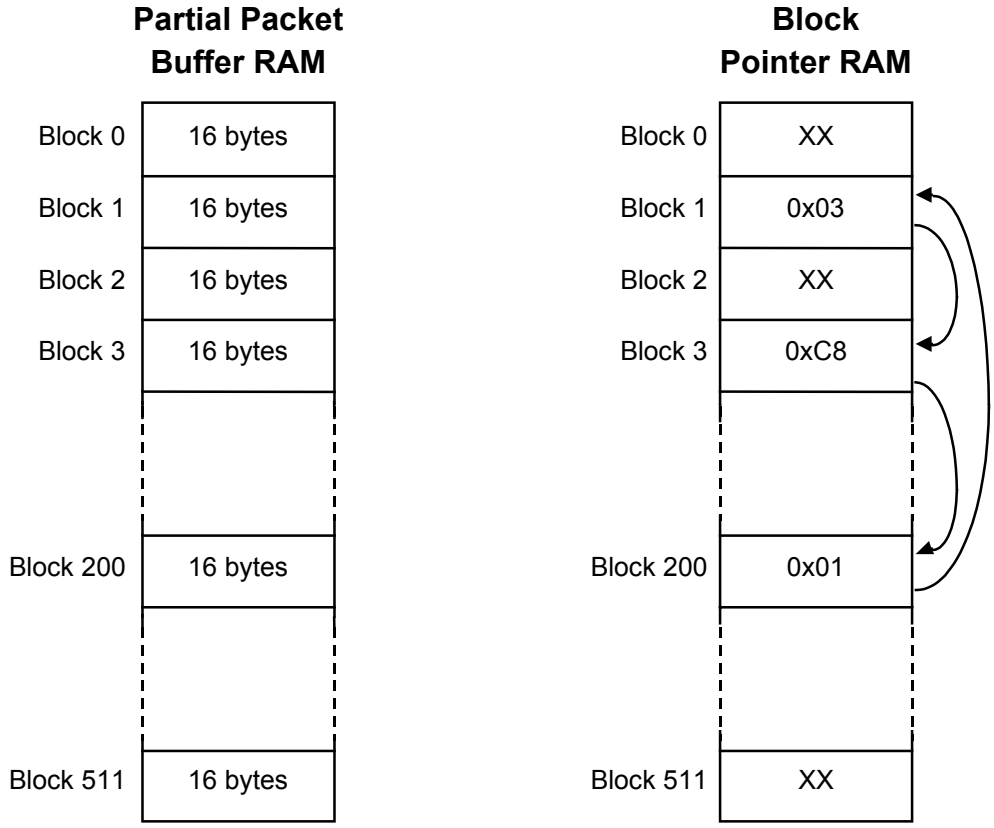
The configuration of the HDLC processor is accessed using indirect channel read and write operations. When an indirect operation is performed, the information is accessed from RAM during a null clock cycle generated by the upstream Receive Channel Assigner block (RCAS). Writing new provisioning data to a channel resets the channel's entire state vector.

9.3.2 Partial Packet Buffer Processor

The partial packet buffer processor controls the 8 Kbyte partial packet RAM which is divided into 16 byte blocks. A block pointer RAM is used to chain the partial packet blocks into circular channel FIFO buffers. Thus, non-contiguous sections of the RAM can be allocated in the partial packet buffer RAM to create a channel FIFO. System software is responsible for the assignment of blocks to individual channel FIFOs. Figure 3 shows an example of three blocks (blocks 1, 3, and 200) linked together to form a 48 byte channel FIFO.

The partial packet buffer processor is divided into three sections: writer, reader and roamer. The writer is a time-sliced state machine which writes the HDLC data and status information from the HDLC processor into a channel FIFO in the packet buffer RAM. The reader transfers channel FIFO data from the packet buffer RAM to the downstream Receive DMA Controller block (RMAC). The roamer is a time-sliced state machine which tracks channel FIFO buffer depths and signals the reader to service a particular channel. If a buffer over-run occurs, the writer ends the current packet from the HDLC processor in the channel FIFO with an over-run flag and ignores the rest of the packet.

Figure 3 – Partial Packet Buffer Structure



The FIFO algorithm of the partial packet buffer processor is based on a programmable per-channel transfer size. Instead of tracking the number of full blocks in a channel FIFO, the processor tracks the number of transactions. Whenever the partial packet writer fills a transfer-sized number of blocks or writes an end-of-packet flag to the channel FIFO, a transaction is created. Whenever the partial packet reader transmits a transfer-size number of blocks or an end-of-packet flag to the RMAC block, a transaction is deleted. Thus, small packets less than the transfer size will be naturally transferred to the RMAC block without having to precisely track the number of full blocks in the channel FIFO.

The partial packet roamer performs the transaction accounting for all channel FIFOs. The roamer increments the transaction count when the writer signals a new transaction and sets a per-channel flag to indicate a non-zero transaction count. The roamer searches the flags in a round-robin fashion to decide for which channel FIFO to request transfer by the RMAC block. The roamer informs the partial packet reader of the channel to process. The reader transfers the data to the RMAC until the channel transfer size is reached or an end of packet

is detected. The reader then informs the roamer that a transaction is consumed. The roamer updates its transaction count and clears the non-zero transaction count flag if required. The roamer then services the next channel with its transaction flag set high.

The writer and reader determine empty and full FIFO conditions using flags. Each block in the partial packet buffer has an associated flag. The writer sets the flag after the block is written and the reader clears the flag after the block is read. The flags are initialized (cleared) when the block pointers are written using indirect block writes. The writer declares a channel FIFO overrun whenever the writer tries to store data to a block with a set flag. In order to support optional removal of the FCS from the packet data, the writer does not declare a block as filled (set the block flag nor increment the transaction count) until the first double word of the next block in channel FIFO is filled. If the end of a packet resides in the first double word, the writer declares both blocks as full at the same time. When the reader finishes processing a transaction, it examines the first double word of the next block for the end-of-packet flag. If the first double word of the next block contains only FCS bytes, the reader would, optionally, process next transaction (end-of-packet) and consume the block, as it contains information not transferred to the RMAC block.

9.4 Receive DMA Controller

The Receive DMA Controller block (RMAC) is a DMA controller which stores received packet data in host computer memory. The RMAC is not directly connected to the host memory PCI bus. Memory accesses are serviced by a downstream PCI controller block (GPIC). The RMAC and the host exchange information using receive packet descriptors (RPDs). The descriptor contains the size and location of buffers in host memory and the packet status information associated with the data in each buffer. RPDs are transferred from the RMAC to the host and vice versa using descriptor reference queues. The RMAC maintains all the pointers for the operation of the queues. The RMAC provides two receive packet descriptor reference (RPDR) free queues to support small and large buffers. The RMAC acquires free buffers by reading RPDRs from the free queues. After a packet is received, the RMAC places the associated RPDR onto a RPDR ready queue. To minimise host bus accesses, the RMAC maintains a descriptor reference table to store current DMA information. This table contains separate DMA information entries for up to 128 receive channels.

9.4.1 Data Structures

For packet data, the RMAC communicates with the host using Receive Packet Descriptors (RPD), Receive Packet Descriptor References (RPDR), the Receive

Packet Descriptor Reference Ready (RPDRR) queue and the Receive Packet Descriptor Reference Small and Large Buffer Free (RPDRF) queues.

The RMAC copies packet data to data buffers in host memory. The RPD, RPDR, RPDRR queue, and Small and Large RPDRF queues are data structures which are used to transfer host memory data buffer information. All five data structures are manipulated by both the RMAC and the host computer. The RPD holds the data buffer size, data buffer address, and packet status information. The RPDR is a pointer which is used to index into a table of RPDs. The RPDRR queue and RPDRF queues allow the RMAC and the host to pass RPDRs back and forth. These data structures are described in more detail in the following sections.

Receive Packet Descriptor

The Receive Packet Descriptors (RPDs) pass buffer and packet information between the RMAC and the host. Both the RMAC and the host read and write information in the RPDs. The host writes RPD fields which describe the size and address of data buffers in host memory. The RMAC writes RPD fields which provide number of bytes used in each data buffer, RPD link information, and the status of the received packet. RPDs are stored in host memory in a Receive Packet Descriptor Table which is described in a later section. The Receive Packet Descriptor structure is shown in Figure 4.

Figure 4 – Receive Packet Descriptor

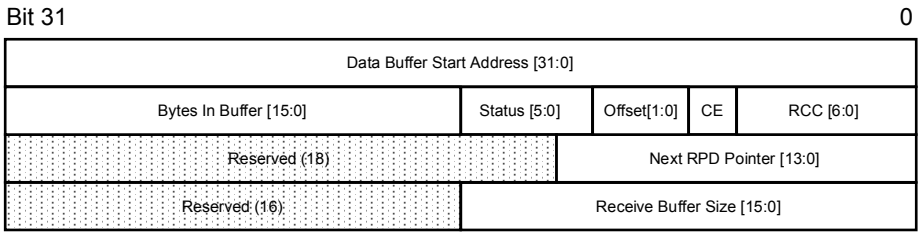


Table 6 – Receive Packet Descriptor Fields

Field	Description
Data Buffer Start Address[31:0]	The Data Buffer Start Address[31:0] bits point to the data buffer in host memory. This field is expected to be configured by the Host during initialisation. The Data Buffer Start Address field is valid in all RPDs.

Field	Description
RCC[6:0]	The Receive Channel Code (RCC[6:0]) bits are used by the RMAC to indicate which channel an RPD is associated with. For a linked list of RPDs, all the RPDs' RCC fields are valid, i.e. all contain the same channel value.
CE	The Chain End (CE) bit indicates the end of a linked list of RPDs. When CE is set to logic one, the current RPD is the last RPD of a linked list of RPDs. When CE is set to logic zero, the current RPD is not the last RPD of a linked list. The CE bit is valid for all RPDs written by the RMAC to the Receive Ready Queue. When a packet requires only one RPD, the CE bit is set to logic one. The CE bit is ignored for all RPDs read by the RMAC from the Receive Free Queues, each of which is assumed to point to only one buffer, i.e. not a chain.
Offset[1:0]	The Offset[1:0] bits indicate the byte offset of the data packet from the start of the buffer. If this value is non-zero, there will be 'dummy' (i.e. undefined) bytes at the start of the data buffer prior to the packet data proper. For a linked list of RPDs, only the first RPD's Offset field is valid. All other RPD Offset fields of the linked list are set to 0.
Status [5:0]	The Status[5:0] bits indicate the status of the received packet. Status[0] Rx buffer overrun Status[1] Packet exceeds max. allowed size Status[2] CRC error Status[3] Packet Length not an exact no. of bytes Status[4] HDLC abort detected Status[5] Unused (set to 0) For a linked list of RPDs, only the last RPD's Status field is valid. All other RPD Status fields of the linked list are invalid and should be ignored. When a packet requires only one RPD, the Status field is valid.

Field	Description
Bytes in Buffer [15:0]	The Bytes in Buffer[15:0] bits indicate the number of bytes actually used in the current RPD's data buffer to store packet data. The count excludes the 'dummy' bytes inserted as a result of a non-zero Offset field. A count greater than 32767 bytes indicates a packet that is shorter than the expected length of the FCS field. The Bytes in Buffer field is invalid when Status[0] or Status[4] is asserted .
Next RPD Pointer [13:0]	The Next RPD Pointer[13:0] bits store a RPDR which enables the RMAC to support linked lists of RPDs. This field, which is only valid when CE is equal to logic zero, contains the RPDR to the next RPD in a linked list. The RMAC links RPDs when more than one buffer is needed to store a packet. The Next RPD Pointer is not valid for the last RPD in a linked list (when CE=1). When a packet requires only one RPD, the Next RPD Pointer field is not valid.
Receive Buffer Size [15:0]	The Receive Buffer Size[15:0] bits indicate the size in bytes of the current RPD's data buffer. This field is expected to be configured by the Host during initialisation. The Receive Buffer Size must be a non-zero integer multiple of four and less than or equal to 32764. The Receive Buffer Size field is valid in all RPDs.

The Receive Buffer Size and Data Buffer Start Address fields are written only by the host. The RMAC reads these fields to determine where to store packet data. All other fields are written only by the RMAC.

Receive Packet Descriptor Table

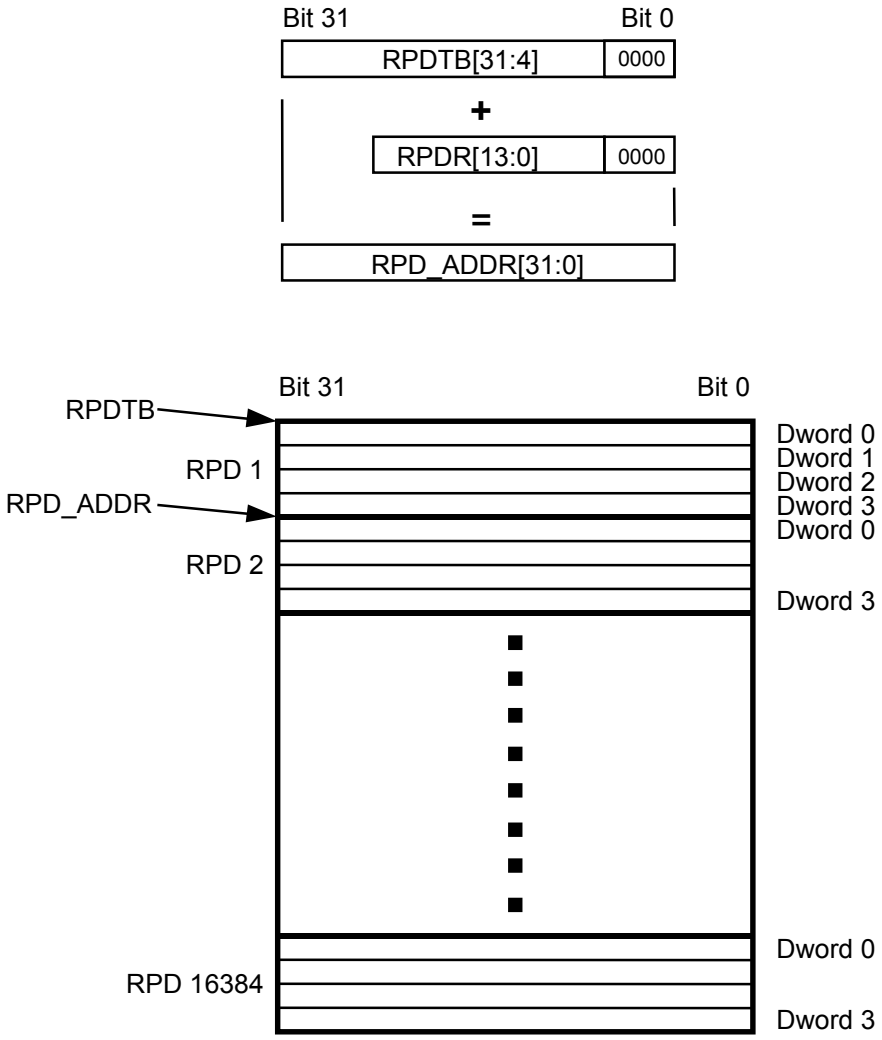
The Receive Packet Descriptor Table resides in host memory and stores all the RPDs. The RPD Table can contain a maximum of 16384 RPDs. The base of the RPD table is user programmable using the Rx Packet Descriptor Table Base (RPDTB) register. The table is indexed by a Receive Packet Descriptor Reference (RPDR) which is a 14-bit pointer defining the offset of a RPD from the table base. Thus, as shown in the following diagram, a RPD can be located by adding the RPDR to the Rx Packet Descriptor Table Base register.

Figure 5 – Receive Packet Descriptor Table

RPDTB[31:4] = Rx Packet Descriptor Table Base register

RPDR[13:0] = Receive Packet Desriptor Reference

RPD_ADDR[31:0] = Receive Packet Descriptor Address



The Receive Packet Descriptor Table resides in host memory. The Rx Packet Descriptor Table Base register resides in the RMAC; this register is initialised by the host. The RPDRs reside in host memory and are accessed using receive packet queues which are described in the next section.

Receive Packet Queues

Receive Packet Queues are used to transfer RPDRs between the host and the RMAC. There are three queues: a RPDR Large Buffer Free Queue (RPDRLFQ), a RPDR Small Buffer Free Queue (RPDRSFQ) and a RPDR Ready Queue (RPDRRQ). The free queues contain RPDRs referencing RPDs that define free buffers. The ready queue contains RPDRs referencing RPDs that define buffers ready for host processing. The RMAC pulls RPDRs from the free queues when it needs free data buffers. The RMAC places an RPDR onto the ready queue after it has filled the buffers with data from each complete packet. The host removes RPDRs from the ready queue to process the data buffers. The host places the RPDRs back onto the free queues after it finishes reading the data from the buffers.

When starting to process a packet, the RMAC uses a small buffer RPD to store the packet data. If the packet requires more than one buffer, the RMAC uses large buffer RPDs to store the remainder of the packet. The RMAC links together all the RPDs required to store the packet and returns the RPDR associated with the first RPD onto the ready queue.

All receive packet queues reside in host memory and are defined by the Rx Queue Base (RQB) register and index registers which reside in the RMAC. The Rx Queue Base is the base address for the receive packet queues. Each packet queue has four index registers which define the start and end of the queue and the read and write locations of the queue. Each index register is 16 bits in length and defines an offset from the Rx Queue Base. Thus, as shown in the Figure 6, the host address of a RPDR is calculated by adding the index register to the Rx Queue Base register. The host initialises the Rx Queue Base and all the index registers. When an entity (either the RMAC or the host) removes elements from a queue, the entity updates the read pointer for that queue. When an entity (either the RMAC or the host) places elements onto a queue, the entity updates the write pointer for that queue.

The read index for each queue points to the last valid RPDR read while the write index points to where the next RPDR can be written. The start index points to the first valid location within the queue; an RPDR can be written to this location. However, the end index points to a location that is beyond a queue; an RPDR can not be written to this location. Note however, the start index of one queue can be set to the end index of another queue. A queue is empty when the read index is one less than the write index; a queue is also empty if the read index is one less than the end index and the write index equals the start index. A queue is full when the read index is equal to the write index. Figure 6 shows the RPDR reference queues.

Figure 6 – RPDRF and RPDRR Queues

Receive Packet Descriptor (RPD) Reference Queues

Base Address:

RQB[31:2] = Rx Queue Base register

Index Registers:

Large Buffer Free Queue:

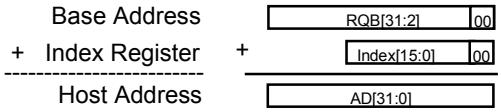
RPDRLFQS[15:0] = RPDR Large Free Queue Start register
RPDRLFQW[15:0] = RPDR Large Free Queue Write register
RPDRLFQR[15:0] = RPDR Large Free Queue Read register
RPDRLFQE[15:0] = RPDR Large Free Queue End register

Small Buffer Free Queue:

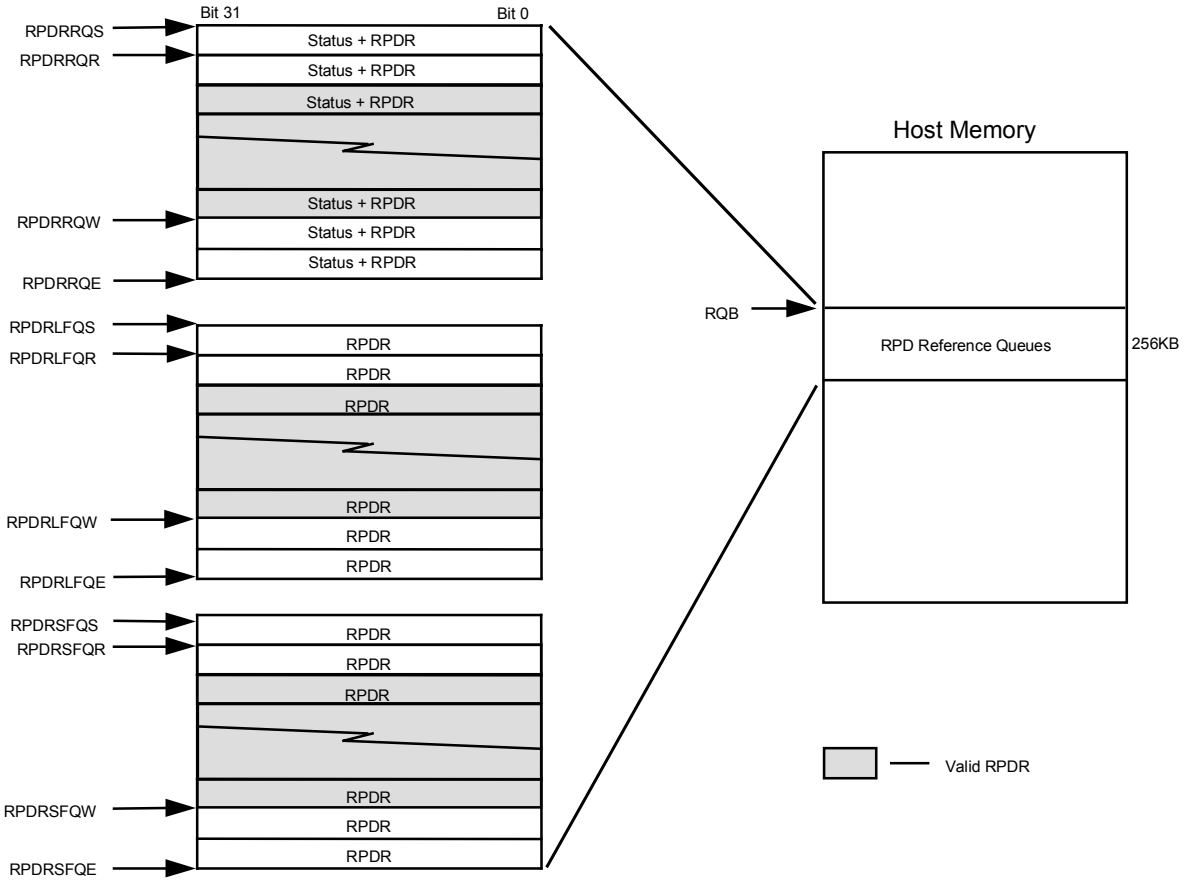
RPDRSFQS[15:0] = RPDR Small Free Queue Start register
RPDRSFQW[15:0] = RPDR Small Free Queue Write register
RPDRSFQR[15:0] = RPDR Small Free Queue Read register
RPDRSFQE[15:0] = RPDR Small Free Queue End register

Ready Queue:

RPDRRQS[15:0] = RPDR Ready Queue Start register
RPDRRQW[15:0] = RPDR Ready Queue Write register
RPDRRQR[15:0] = RPDR Ready Queue Read register
RPDRRQE[15:0] = RPDR Ready Queue End register



Rx Packet Descriptor Reference Queue Memory



Note that the maximum value to which an end pointer may be set is FFFF hex, resulting in a maximum offset from the queue base address of (4*(FFFF-1)) = 3FFF8 hex. An end pointer must not be set to 0 hex in an attempt to include offset 3FFFC hex in a queue.

As shown in Figure 6, the ready queue elements have a status field as well as an RPDR field. The RMAC fills in the status field to mark whether a packet was successfully received or not. The host reads the status field. The ready queue element is shown in Table 7 below along with the definition of the status bits.

If the RMAC requires a buffer of a particular size (i.e. small or large) and no RPDR is available in the corresponding free queue, a RPDR from the other free queue is substituted. The host may, therefore, force the RMAC to store received data in buffers of only one size by setting one of the free queues to zero length, i.e. by setting the start and end index registers of one of the queues to equal values. If the RMAC requires a buffer and neither free queue contains RPDRs, an RPQ_ERRI interrupt is generated.

Table 7 – RPDRR Queue Element

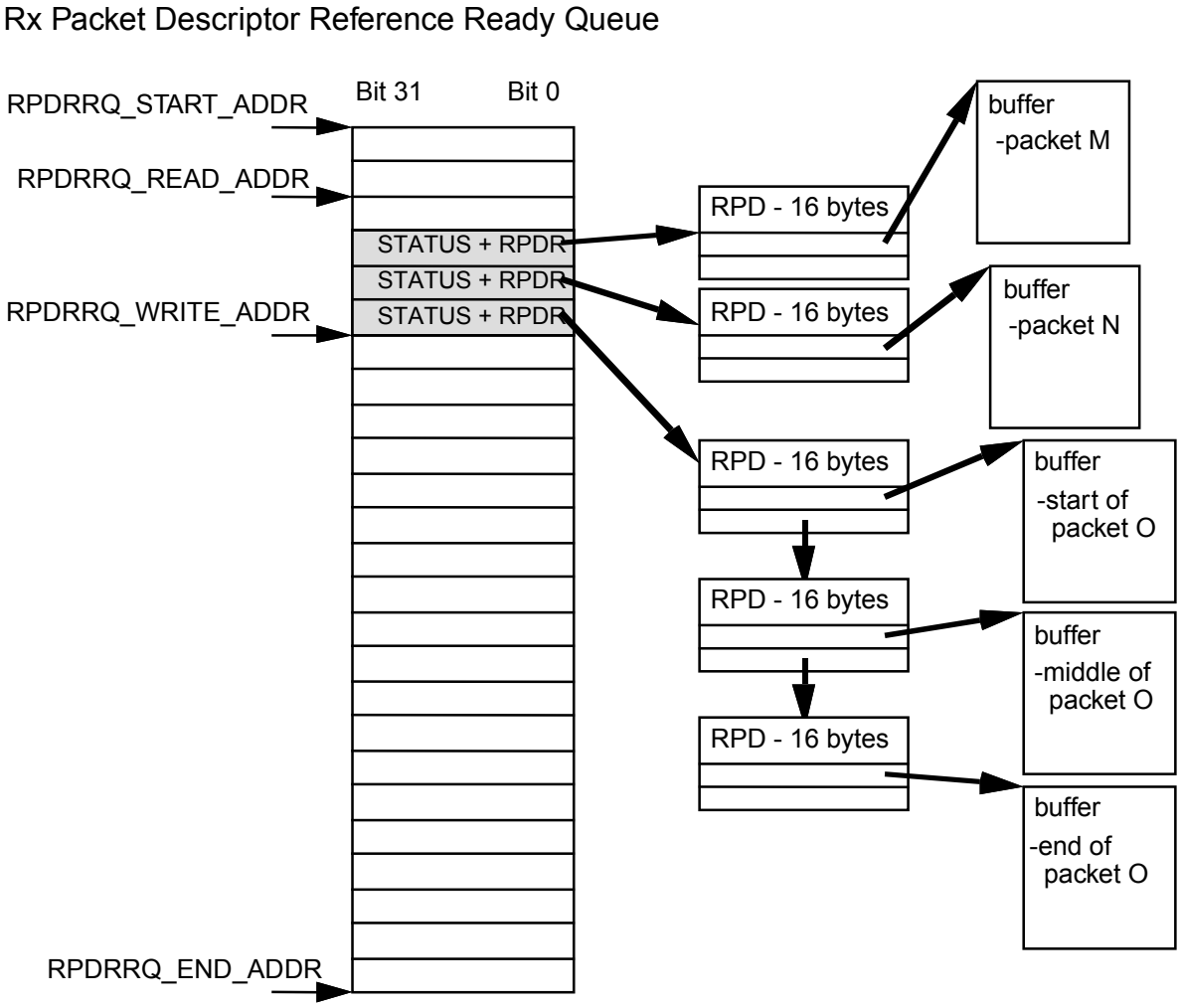
Bit 15	Bit 0
STATUS[1:0]	RPDR[13:0]

Field	Description
STATUS[1:0]	The encoding for the status field is as follows: 00 - Successful reception of packet. 01 - Unsuccessful reception of packet. 10 - Unprovisioned partial packet. 11 - Reserved.
RPDR[13:0]	The RPDR[13:0] field defines the offset of the first RPD in a linked chain of RPDs, each pointing to a buffer containing the received data.

As described previously, the RMAC links a large buffer RPD to a small buffer RPD if more than one buffer is needed for a packet. The RMAC links additional large buffer RPDs to the end of the chain as required until the entire packet is copied to host memory (provided that the host has not disabled use of both free queues by setting one of them to length zero). After storing the packet data, the RMAC places the STATUS+RPDR for the first RPD onto the ready queue. Only the RPDR associated with the first RPD is placed onto the ready queue. All other required RPDs are linked to the first RPD as shown in Figure 7.

Although a STATUS+RPDR only totals to 16 bits, each queue entry is a dword, i.e. 32 bits. When the RMAC block writes a STATUS+RPDR to the ready queue, it sets the third byte to 0 and the fourth (most significant) byte is unmodified.

Figure 7 – RPDRR Queue Operation



Receive Channel Descriptor Reference Table

On a per-channel basis, the RMAC caches information such as the current DMA information in a Receive Channel Descriptor Reference (RCDR) Table. The RMAC can process 128 channels and stores three dwords of information per channel. This information is cached internally in order to decrease the number of host bus accesses required to process each data packet. The structure of the RCDR table is shown in Figure 8.

Figure 8 – Receive Channel Descriptor Reference Table

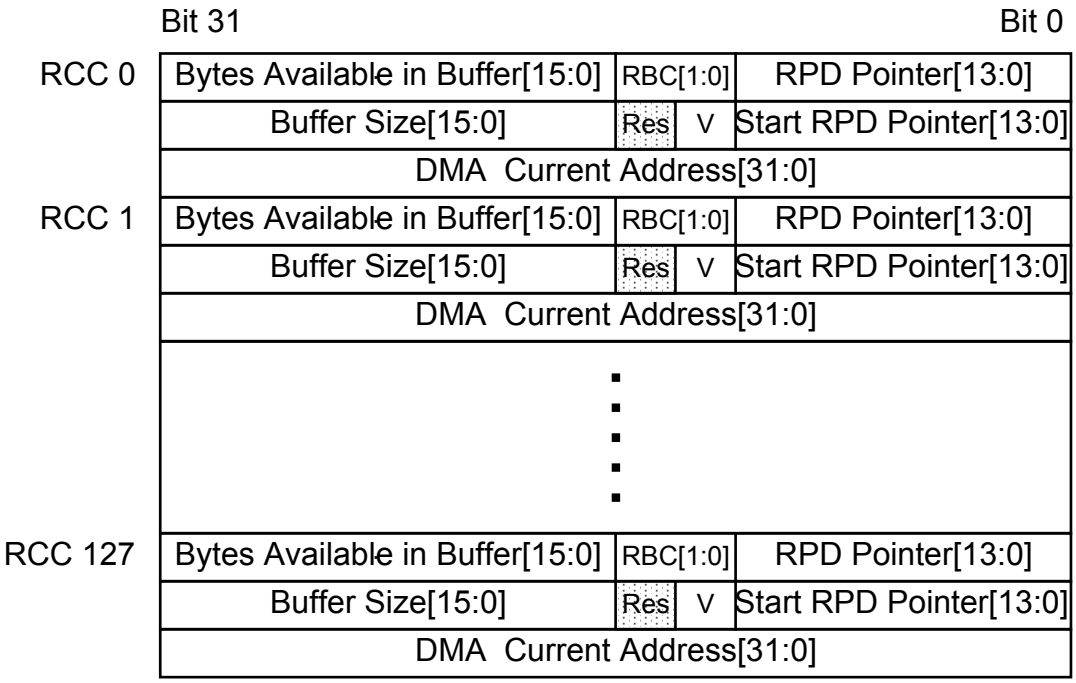


Table 8 – Receive Channel Descriptor Reference Table Fields

Field	Description
Bytes Available in Buffer[15:0]	This field is used to keep track of the number of bytes available in the current data buffer. The RMAC initialises the Bytes Available in Buffer to the Receive Buffer Size minus the offset at the head of the buffer. The field is decremented each time a byte is written into the buffer.
RBC[1:0]	This field is used to keep track of the number of buffers used when storing ‘raw’ (i.e. non packet delimited) data. The RMAC initialises the RBC field to the value of the RAWMAX[1:0] field in the RMAC Control Register. The field is decremented each time a buffer is filled with data. If the field reaches zero, the chain of RPDs is placed on the ready queue and a new chain started.
RPD Pointer[13:0]	This field contains the pointer to the current RPD.
Buffer Size[15:0]	This field contains the size in bytes of the buffer currently being written to.

Field	Description
V	This bit (Valid) indicates whether a packet is currently being received on the DMA channel. When the V bit is set to 1, the other fields in the RCDR table entry for the DMA channel contain valid information.
Start RPD Pointer[13:0]	This field contains the pointer to the first RPD for the packet being received.
DMA Current Address[31:0]	The DMA Current Address [31:0] bits holds the host address of the next dword in the current buffer. The RMAC increments this field on each access to the buffer.

9.4.2 DMA Transaction Controller

The DMA Transaction Controller coordinates the reception of data packets from the Receive Packet Interface and their subsequent storage in host memory. A packet may be received over a number of separate transactions, interleaved with transactions belonging to other DMA channels. As well as sending the received data to host memory, the DMA Transaction Controller initiates data transactions of its own for the purposes of maintaining the data structures (queues, descriptors, etc.) in host memory.

9.4.3 Write Data Pipeline/Mux

The Write Data Pipeline/Mux performs two functions. First, it pipelines receive data between the RHDL block and the GPIC block, inserting enough delay to enable the DMA Transaction Controller to generate appropriate control signals at the GPIC interface. Second, it provides a multiplexor to the data out lines on the GPIC interface, allowing the DMA Transaction Controller to output data relating to the transactions the controller itself initiates.

9.4.4 Descriptor Information Cache

The Descriptor Information Cache provides the storage for the Receive Channel Descriptor Reference (RCDR) Table described above (Figure 8).

9.4.5 Free Queue Cache

The Free Queue Cache block implements the 6 element RPDR Small Buffer Free Queue cache and the 6 element RPDR Large Buffer Free Queue cache. These caches are used to store free small buffer and large buffer RPDRs.

Caching RPDRs reduces the number of host bus accesses that the RMAC makes.

Each cache is managed independently. The elements of the cache are consumed one at a time as they are needed by the RMAC. The RPDR small buffer cache is reloaded when it is empty and the RMAC requires a new small buffer RPDR. The large buffer RPDR cache is reloaded when it is empty and the RMAC requires a new large buffer RPDR. When reloading either of the caches, the appropriate cache controller will read up to six new elements. The cache controller may read fewer than six elements if there are fewer than six new elements available, or the read pointer index is within six elements of the end of the free queue. If the read pointer is near the end of the free queue, the cache controller reads only to the end of the queue and does not start reading from the top of the queue until the next time a reload is required. To do so would require two host memory transactions and would be of no benefit.

9.5 PCI Controller

The General-Purpose Peripheral Component Interconnect Controller block (GPIC) provides a 32-bit Master and Target interface core which contains all the required control functions for Peripheral Component Interconnect (PCI) Bus Revision 2.1 interfacing. Communications between the PCI bus and other FREEDM-32 blocks can be made through either an internal asynchronous16-bit bus or through one of two synchronous FIFO interfaces. One of the FIFO interfaces is dedicated to servicing the Receive DMA Controller block (RMAC) and the other to the Transmit DMA Controller block (TMAC).

The GPIC supports a 32-bit PCI bus operating at up to 33 MHz and bridges between the timing domain of the DMA controllers (SYSCLK) and the timing domain of the PCI bus (PCICLK). By itself, the GPIC does not generate any PCI bus accesses. All transactions on the bus are initiated by another PCI bus master or by the core device. The GPIC transforms each access to and from the PCI bus to the intended target or initiator in the core device. Except for the configuration space registers and parity generating/checking, the GPIC performs no operations on the data.

The GPIC is made up of four sections: master state machine, a target state machine, internal microprocessor bus interface and error/bus controller. The target and master blocks operate independent of each other. The error/bus control block monitors the control signals from the target and master blocks to determine the state of the PCI I/O pads. This block also generates and/or checks parity for all data going to or coming from the PCI bus. The internal microprocessor bus interface block contains configuration and status registers together with the production test logic for the GPIC block.

9.5.1 Master Machine

The GPIC master machine translates requests from the RMAC and TMAC block interfaces into PCI bus transactions. The GPIC initiates four types of PCI cycles: memory read (burst or single), memory read multiple, memory read line and memory write (burst or single). The number of data transfers in any cycle is controlled by the DMA controllers. The maximum burst size is determined by the particular data path. A read cycle to the RMAC is restricted to a maximum burst size of 8 dwords and a write cycle is limited to a maximum of 32. The TMAC interface has a limit of 32 dwords on a read cycle and 8 on a write cycle.

In response to a DMA controller requesting a cycle, the GPIC must arbitrate for control of the PCI bus. Before asserting the PCI Request line, the GPIC first does an internal arbitration to determine the priority of service in the event that both the RMAC and TMAC are requesting service. The GPIC arbitrates between the four FIFOs based on either a RMAC priority or a round-robin scheme that is software selectable. It is possible for all four FIFOs (RMAC read, TMAC read, RMAC write, TMAC write) to request service simultaneously.

When an external PCI bus arbitrator issues a Grant in response to the Request from the GPIC, the master state machine monitors the PCI bus to insure that the previous master has completed its transaction and has released the bus before beginning the cycle. Once the GPIC has control of the bus, it will assert the FRAME signal and drive the bus with the address and command. The value for the address is provided by the selected DMA controller. After the initial data transfer, the GPIC tracks the address for all remaining transfers in the burst internally in case the GPIC is disconnected by the target and must retry the transaction.

The target of the GPIC master burst cycle has the option of stopping or disconnecting the burst at any point. In the event of a target disconnect the GPIC will terminate the present cycle and release the PCI bus. If the GPIC is asserting the REQUEST line at the time of the disconnect, it will remove the REQUEST for two PCI clock cycles then reassert it. When the PCI bus arbitrator returns the GRANT, the GPIC will restart the burst access at the next address and continue until the burst is completed or repeat the sequence if the target disconnects again.

During burst reads, the GPIC accepts the data without inserting any wait states. Data is written directly into the read FIFO where the RMAC or TMAC can remove it at its own rate. During burst writes, the GPIC will output the data without inserting any wait states, but may terminate the transaction early if the local master fails to fill the write FIFO with data before the GPIC requires it. (If a write transaction is terminated early due to data starvation, the GPIC will automatically

initiate a further transaction to write the remaining data when it becomes available.)

Normally, the GPIC will begin requesting the PCI bus for a write transaction shortly after data starts to be loaded into the write FIFO by the RMAC or TMAC. The RMAC, however, is not required to supply a transaction length when writing packet data and in addition, may insert pauses during the transfer. In the case of packet data writes by the RMAC, the GPIC will hold off requesting the PCI bus until the write FIFO has filled up with a number of dwords equal to a programmable threshold. If the FIFO empties without reaching the end of the transition, the GPIC will terminate the current transaction and restart a new transaction to transfer any remaining data when the RMAC signals an end of transaction. Beginning the PCI transaction before all the data is in the write FIFO allows the GPIC to reduce the impact of the bus latency on the core device.

Each master PCI cycle generated by the GPIC can be terminated in three ways: Completion, Timeout or Master Abort. The normal mode of operation of the GPIC is to terminate after transferring all the data from the master FIFO selected. As noted above this may involve multiple PCI accesses because of the inability of the target to accept the full burst or data starvation during writes. After the completion of the burst transfer the GPIC will release the bus unless another FIFO is requesting service, in which case if the GRANT is asserted the GPIC will insert one idle cycle on the bus and then start a new transfer.

The maximum duration of the a master burst cycle is controlled by the value set in the LATENCY TIMER register in the GPIC Configuration Register block. This value is set by the host on boot and is loaded into a counter in the GPIC master state at the start of each access. If the counter reaches zero and the GRANT signal has been removed the GPIC will release the bus regardless of whether it has completed the present burst cycle. This type of termination is referred to as a Master Time-out. In the case of a Master Time-out the GPIC will remove the REQUEST signal for two PCI clocks and then reassert it to complete the burst cycle.

If no target responds to the address placed on the bus by the GPIC after 4 PCI clocks the GPIC will terminate the cycle and flag the cycle in the PCI Command/ Status Configuration Register as a Master Abort. If the Stop on Error enable (SOE_E) bit is set in the GPIC Command Register, the GPIC will not process any more requests until the error condition is cleared. If the SOE_E is not set, the GPIC will discard the REQUEST and indicate to the local master that the cycle is complete. This action will result in any write data being lost and any read data being erroneous.

9.5.2 Master Local Bus Interface

The master local bus is a 32 bit data bus which connects the local master device to the GPIC. The GPIC contains two local master interface blocks, with one supporting the RMAC and the other the TMAC. Each local master interface has been optimised to support the traffic pattern generated by the RMAC or the TMAC and are not interchangeable.

The data path between the GPIC and local master device provides a mechanism to segregate the system timing domain of the core from the PCI bus. Transfers on each of the RMAC and TMAC interfaces are timed to its own system clock. The DMA controllers isolated from all aspects of the PCI bus protocol, and instead “sees” a simple synchronous protocol. Read or write cycles on the local master bus will initiate a request for service to the GPIC which will then transfer the data via the PCI bus.

The GPIC maximises data throughput between the PCI bus and the local device by paralleling local bus data transfers with PCI access latency. The GPIC allows either DMA controller to write data independent of each other and independent of PCI bus control. The GPIC temporarily buffers the data from each DMA controller while it is arbitrating for control of the PCI bus. After completion of a write transfer, the DMA controller is then released to perform other tasks. The GPIC can buffer only a single transaction from each DMA controller.

Read accesses on the local bus are optimised by allowing the DMA controllers access to the data from the PCI bus as soon as the first data becomes available. After the initial synchronisation and PCI bus latency data is transferred at the slower of PCI bus rate or the core logic SYSCLK rate. Once a read transaction is started, the DMA controller is held waiting for the ready signal while the GPIC is arbitrating for the PCI bus.

All data is passed between the GPIC and the DMA controllers in little Endian format and, in the default mode of operation, the GPIC expects all data on the PCI bus to also be in little Endian format. The GPIC provides a selection bit in the internal Control register which allows the Endian format of the PCI bus data to be changed. If enabled, the GPIC will swizzle all packet data on the PCI bus (but not descriptor references and the contents of descriptors). The swizzling is performed according to the “byte address invariance” rule, i.e. the only change to the data is the mirror-imaging of byte lanes.

The interface for the RMAC provides for byte addressability of write transactions whereas the interface for the TMAC provides for byte addressability of read transactions. Other transactions must be dword aligned. For byte-addressable transactions, the data transferred between the local device and the GPIC need not be dword aligned with the data as it is presented on the PCI bus. The GPIC

will perform any byte-realignment required. In order to complete a transfer involving byte re-alignment, the GPIC may need to add an extra burst cycle to the PCI transaction.

9.5.3 Target Machine

The GPIC target machine performs all the required functions of a stand alone PCI target device. The target block performs three main functions. The first is the target state machine which controls the protocol of PCI target accesses to the GPIC. The second function is to provide all PCI Configuration registers. Last, the target block provides a Target Interface to the CBI registers in the other FREEDM-32 blocks.

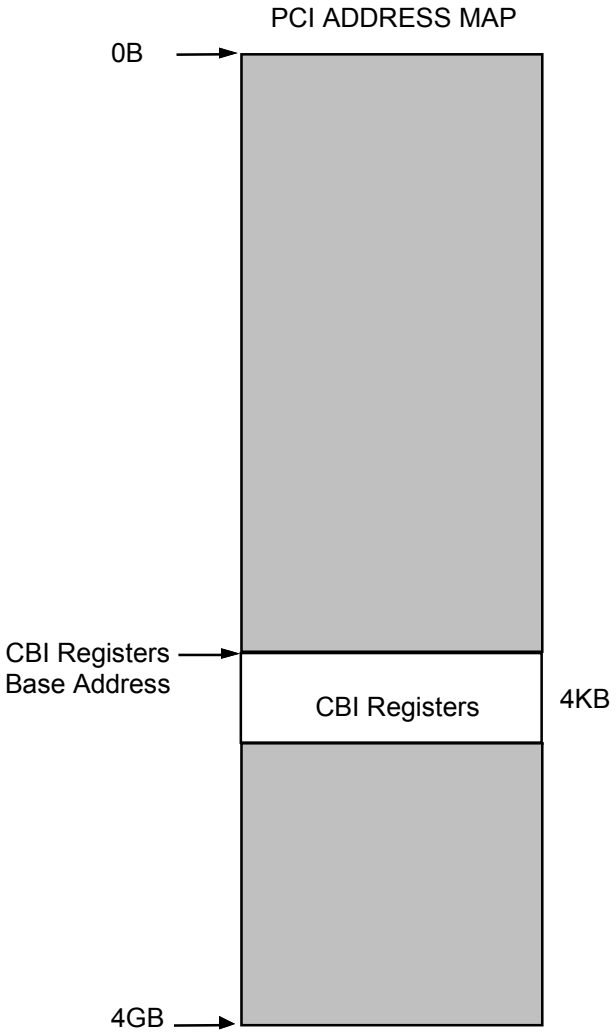
The GPIC tracks the PCI bus and decodes all addresses and commands placed on the bus to determine whether to respond to the access. The GPIC responds to the following types of PCI bus commands only: Configuration read and write, memory read and write, memory-read-multiple and memory-read-line which are aliased to memory read and memory-write-and-invalidate which is aliased to memory write. The GPIC will ignore any access that falls within the address range but has any other command type.

After accepting a target access as a medium speed device, the FREEDM inserts one wait state for a configuration read/write and five wait states for other command types before completing the transaction by asserting TRDYB.

Burst accesses to the GPIC are accepted provided they are of linear type. If a master makes a memory access to the GPIC with the lower two address bits set to any value but "00" (linear burst type) the GPIC ignores the cycle. Burst accesses of any length are accepted, but the FREEDM will disconnect if the master inserts any wait states during the transaction. The FREEDM will also disconnect on every read and write access to configuration space after transferring one Dword of data.

Figure 9 illustrates the GPIC address space.

Figure 9 – GPIC Address Map



The GPIC responds with medium timing to master accesses. (I.e. DEVSELB is asserted 2 PCICLK cycles after FRAMEB asserted) It inserts three wait states on reads to the internal CBI register space (four wait states for the 2nd and subsequent dwords of a burst read). The target machine will only terminate an access with a Retry if the target is locked and another master tries to access the GPIC. The GPIC will terminate any access to a non-burst area with a Disconnect and always with data transferred. The target does not support delayed transactions. The GPIC will perform a Target-Abort termination only in the case of an address parity error in an address that the GPIC claims.

9.5.4 CBI Bus Interface

The CBI bus interface provides access to the CBI address space of the FREEDM-32 blocks. The CBI address space is set by the associated BAR in the PCI Configuration registers.

Write transfers to the CBI space always write all 32 bits provided that at least one byte enable is asserted. A write command with all byte enables negated will be ignored. Read transfers always return the 32 bits regardless of the status of the byte enables, as long as at least one byte enable is asserted. A read command with all byte enables negated will be ignored.

9.5.5 Error / Bus Control

The Error/Bus Control block monitors signals from both the Target block and Master Block to determine the direction of the PCI bus pads and to generate or check parity. After reset, the GPIC sets all bi-directional PCI bus pads to inputs and monitors the bus for accesses. The Error/Bus control unit remains in this state unless either the Master requests the PCI bus or the Target responds to a PCI Master Access. The Error/Bus control unit decodes the state of each state machine to determine the direction of each PCI bus signal.

All PCI bus devices are required to check and generate even parity across AD[31:0] and C/BEB[3:0] signals. The GPIC generates parity on Master address and write data phases; the target generates parity on read data phases. The GPIC is required to check parity on all PCI bus phases even if it is not participating in the cycle. But, the GPIC will report parity errors only if the GPIC is involved in the PCI cycle or if the GPIC detects an address parity error or data parity is detected in a PCI special cycle. The GPIC updates the PCI Configuration Status register for all detected error conditions.

9.6 Transmit DMA Controller

The Transmit DMA Controller block (TMAC) is a DMA controller which retrieves packet data from host computer memory for transmission. The minimum packet data length is two bytes. The TMAC communicates with the host computer bus through the master interface connected to PCI Controller block (GPIC) which translates host bus specific signals from the host to the master interface format. The TMAC uses the master interface whenever it wishes to initiate a host bus read or write; in this case, the TMAC is the initiator and the host memory is the target.

The TMAC and the host exchange information using transmit descriptors (TDs). The descriptor contains the size and location of buffers in host memory and the

packet status information associated with the data in each buffer. TDs are transferred from the TMAC to the host and vice versa using descriptor reference queues. The TMAC maintains all the pointers for the operation of the queues. The TMAC acquires buffers with data ready for transmission by reading TDRs from a TDR ready queue. After a packet has been transmitted, the TMAC places the associated TDR onto a TDR free queue.

To minimise host bus accesses, the TMAC maintains a descriptor reference table to store current DMA information. This table contains separate DMA information entries for up to 128 transmit channels. The TMAC also performs per-channel sorting of packets received in the TDR ready queue to eliminate head-of-line blocking.

9.6.1 Data Structures

The TMAC communicates with the host using Transmit Descriptors (TD), Transmit Descriptor References (TDR), the Transmit Data Reference Ready (TDRR) queue and the Transmit Data Reference Free (TDRF) queue.

The TMAC reads packet data from data buffers in host memory. The TD, TDR, TDRR queue, and TDRF queue are data structures which are used to transfer host memory data buffer information. All four data structures are manipulated by both the TMAC and the host computer. The TD holds the data buffer size, data buffer address, and other packet information. The TDR is a pointer which is used to index into a table of TDs. The TDRR queue and TDRF queue allow the TMAC and the host to pass TDRs back and forth. These data structures are described in more detail in the following sections.

Transmit Descriptor

The Transmit Descriptors (TDs) pass buffer and packet information between the TMAC and the host. Both the TMAC and the host read and write information in the TDs. TDs are stored in host memory in a Transmit Descriptor Table. The Transmit Descriptor structure is shown in Figure 10.

Figure 10 – Transmit Descriptor

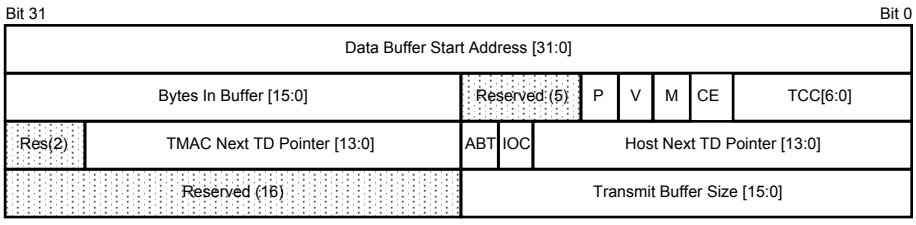


Table 9 – Transmit Descriptor Fields

Field	Description
Data Buffer Start Address [31:0]	The Data Buffer Start Address[31:0] bits point to the data buffer in host memory. The Data Buffer Start Address field is valid in all TDs
Bytes In Buffer [15:0]	The Bytes In Buffer[15:0] field is used by the host to indicate the total number of bytes to be transmitted in the current TD. Zero length buffers are illegal.
P	The Priority bit is set by the host to indicate the priority of the associated packet in a two level quality of service scheme. Packets with its P bit set high are queued in the high priority queue in the TMAC. Packets with the P bit set low are queued in the low priority queue. Packets in the low priority queue will not begin transmission until the high priority queue is empty.
V	The V bit is used to indicate that the TMAC Next TD Pointer field is valid. When set to logic 1, the TMAC Next TD Pointer[13:0] field is valid. When V is set to logic 0, the TMAC Next TD Pointer[13:0] field is invalid. The V bit is used by the host to reclaim data buffers in the event that data presented to the TMAC is returned to the host due to a channel becoming unprovisioned. The V bit is expected to be initialised to logic 0 by the host.
M	The More (M) bit is used by the host to support packets that require multiple TDs. If M is set to logic 1, the current TD is just one of several TDs for the current packet. If M is set to logic 0, this TD either describes the entire packet (in the single TD packet case) or describes the end of a packet (in the multiple TD packet case). Note: When M is set to logic 1, the only valid value for CE is logic 0.

Field	Description
CE	The Chain End (CE) bit is used by the host to indicate the end of a linked list of TDs presented to the TMAC. The linked list can contain one or more packets as delineated by the M bit (see above). When CE is set to logic 1, the current TD is the last TD of a linked list of TDs. When CE is set to logic 0, the current TD is not the last TD of a linked list. When the current TD is not the last of the linked list, the Host Next TD Pointer[13:0] field is valid, otherwise the field is not valid. Note: When CE is set to logic 1, the only valid value for M is logic 0. Note: When presenting raw (i.e. unpacketised) data for transmission, the host should code the M and CE bits as for a single packet chain, i.e. M=1, CE=0 for all TDs except the last in the chain and M=0, CE=1 for the last TD in the chain.
TCC[6:0]	The Transmit Channel Code (TCC[6:0]) field is used by the host to indicate with which channel a TD is associated. All TD in a chain must be associated with the same channel, i.e. have this field set to the same value.
TMAC Next TD Pointer [13:0]	The TMAC Next TD Pointer[13:0] bits are used to store TDRs which permits the TMAC to create linked lists of TDs passed to it via the TDRR queue. The TDs are linked with other TDs belonging to the same channel and same priority level. In the case that data presented to the TMAC is returned to the host due to a channel becoming unprovisioned, a TDR pointing to the start of the per-channel linked list of TDs is placed on the TDRF queue. It is the responsibility of the host to follow the TMAC and host links in order to recover all the buffers.
ABT	The Abort (ABT) bit is used by the host to abort the transmission of a packet. When ABT is set to logic 1, the packet will be aborted after all the data in the buffer has been transmitted. If ABT is set to logic 1 in the current TD, the M bit must be set low and the CE bit must be set to high..

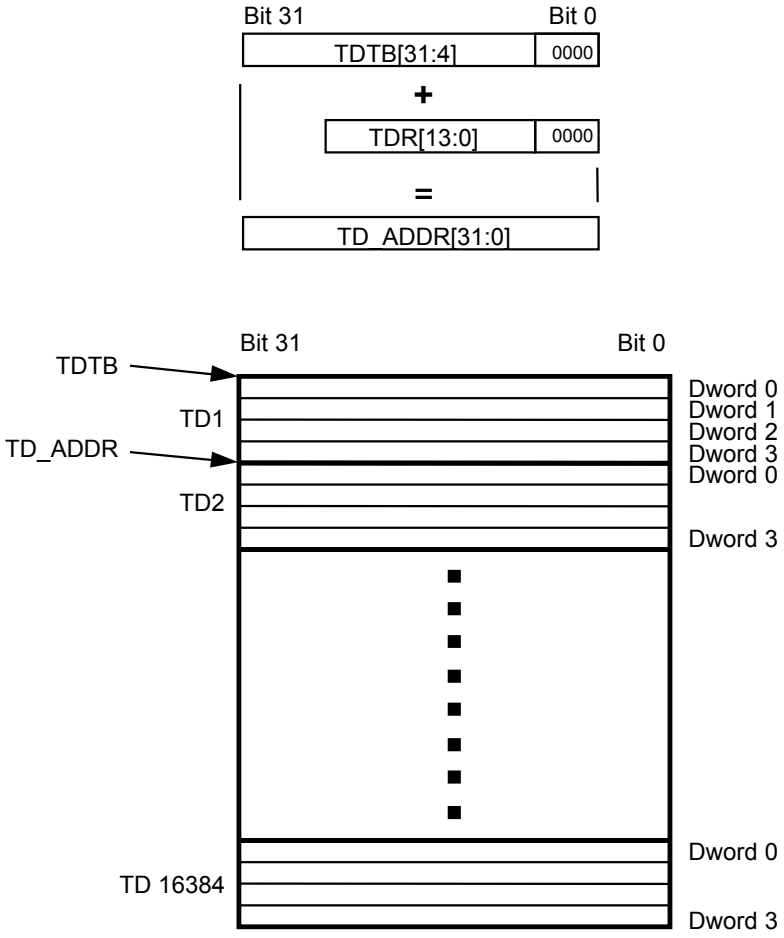
Field	Description
IOC	The Interrupt On Complete (IOC) bit is used by the host to instruct the TMAC to interrupt the host when the current TD's data buffer has been read. When IOC is logic 1, the TMAC asserts the IOCI interrupt when the data buffer has been read. Additionally, the Free Queue FIFO will be flushed. If IOC is logic zero, the TMAC will not generate an interrupt and the Free Queue FIFO will operate normally.
Host Next TD Pointer [13:0]	The Host Next TD Pointer[13:0] bits are used to store TDRs which permits the host to support linked lists of TDs. As described above, linked lists of TDs are terminated by setting the CE bit to logic 1. Linked lists of TDs are used by the host to pass multiple TD packets or multiple packets associated with the same channel and priority level to the TMAC.
Transmit Buffer Size [15:0]	The Transmit Buffer Size[15:0] field is used to indicate the size in bytes of the current TD's data buffer. (N.B. The TMAC does not make use of this field.)

Transmit Descriptor Table

The Transmit Descriptor Table, which resides in host memory, contains all of the Transmit Descriptors referenced by the TMAC. To access a TD, the TMAC takes a TDR from a TDRR queue or from the TCDR table and adds 16 times its value (because each TD is 16 bytes in size) to the Transmit Descriptor Table Base (TDTB) pointer to form the actual address of the TD in host memory. Each TD must reside in the Transmit Descriptor Table. The Transmit Descriptor Table can contain a maximum of 16384 TDs. The base of the Transmit Descriptor Table is user programmable using the TMAC Tx Descriptor Table Base register. Thus, as shown below, each TD can be located using a Transmit Descriptor Reference (TDR) combined with the TMAC Tx Descriptor Table Base register.

Figure 11 – Transmit Descriptor Table

TDTB[31:4] = Tx Descriptor Table Base register
TDR[13:0] = Transmit Descriptor Reference
TD_ADDR[31:0] = Transmit Descriptor Address



Transmit Queues

Pointers to the transmit descriptors (TDs) containing packet(s) ready for transmission are passed from the host to the TMAC using the Transmit Descriptor Reference Ready (TDRR) queue, which resides in host memory. Pointers to transmit descriptor structures whose buffers have been read by the TMAC are passed from the TMAC to the host using the Transmit Descriptor Reference Free (TDRF) queue, which also resides in host memory. The TMAC contains a Free Queue cache which can store up to six TDRs. If caching is

enabled, free TDRs are written into the TDRF queue six at a time, to reduce the number of host memory accesses. The Free Queue cache is also flushed to the TDRF queue if the Interrupt On Completion (IOC) bit is set in the TD, which sends the corresponding TDR directly to the TDRF queue.

The queues, shown in Figure 12 are defined by a common base pointer residing in the Transmit Queue Base register and eight offset pointers, four per queue. For each queue, two pointers define the start and the end of the queue, and two pointers keep track of the current read and write locations within the queue. The read pointer for each queue points to the offset of the last valid TDR read, and the write pointer points to the offset where next TDR can be written. The end of a queue is not a valid location for a TDR to be read or written. A queue is empty when the read pointer is one less than the write pointer or if the read pointer is one less than the end pointer and the write pointer equals the start pointer. A queue is full when the read pointer is equal to the write pointer. Each queue element is 32 bits in size, but only the 17 least significant bits are valid. The 17 least significant bits consist of a 14-bit TDR and three status bits. The status bits are used by the TMAC to inform the host of the success or failure of transmission (see Table 10). When the TMAC writes TDRs to the TDRF queue, it sets bits [23:17] of the queue element to 0. Once a TDR is placed on the TDRF queue, the FREEDM-32 will make no further accesses to the TD nor the associated buffer.

Note that the maximum value to which an end pointer may be set is FFFF hex, resulting in a maximum offset from the queue base address of (4*(FFFF-1)) = 3FFF8 hex. An end pointer must not be set to 0 hex in an attempt to include offset 3FFFC hex in a queue.

Figure 12 – TDRR and TDRF Queues

Transmit Descriptor Referance Queues

Base Address:

TQB[31:2] = Tx Queue Base register

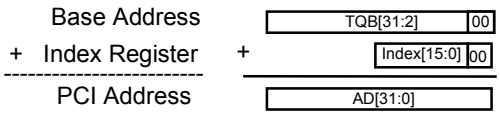
Index Registers:

Ready:

TDRRQS[15:0] = TDR Ready Queue Start register
TDRRQW[15:0] = TDR Ready Queue Write register
TDRRQR[15:0] = TDR Ready Queue Read register
TDRRQE[15:0] = TDR Ready Queue End register

Free:

TDRFQS[15:0] = TDR Free Queue Start register
TDRFQW[15:0] = TDR Free Queue Write register
TDRFQR[15:0] = TDR Free Queue Read register
TDRFQE[15:0] = TDR Free Queue End register



Tx Descriptor Reference Queue Memory Map

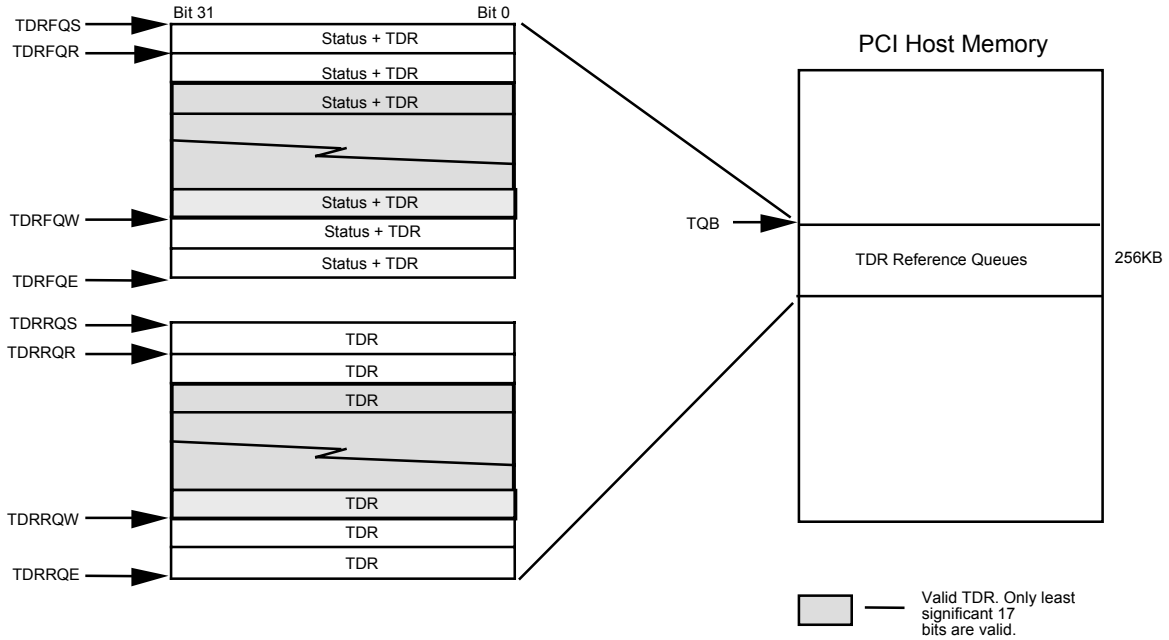


Table 10 – Transmit Descriptor Reference

Bit 16	Bit 0
STATUS[2:0]	TDR[13:0]

Field	Description
Status[2:0]	The TMAC fills in the Status field to indicate to the host the results of processing the TD. The encoding is: Status[1:0] Description 00 Last or only buffer of packet, buffer read. 01 Buffer of partial packet, buffer read. 10 Unprovisioned channel, buffer not read. 11 Malformed packet (e.g. Bytes In Buffer field set to 0), buffer not read. Status[2] Description 0 No underflow detected. 1 Underflow detected.
TDR[13:0]	The TDR[13:0] field contains the offset of the TD returned.

If a TDR is returned to the host with the status field set to “10” (unprovisioned channel), the TDR may point to a binary tree of TDs and buffers (as indicated by the CE and V bits in the TDs). It is the responsibility of the host to traverse the tree to reclaim all the buffers. If a TDR is returned to the host with the status field set to any other value, the TDR will only point to one TD and buffer regardless of the values of V and CE in that TD.

The underflow status bit (Status[2]) is normally attached to the TDR belonging to a packet experiencing underflow. For long packets spanning multiple buffers, underflow is reported only once at the first available TDR of that channel. All subsequent TDRs of that packet will be returned normally without the underflow status. In rare cases, due to internal buffering by the FREEDM-32, a packet may experience underflow at the very end of a packet, just as the TDR is being returned to the TDR free queue. The underflow status will then be reported in the first TDR of the immediate next packet of that channel. Because of the uncertainty with the reporting of underflows between the current verse the subsequent packet, the underflow status should only be used to gather

performance statistics on channels and not for initiating packet specific responses such as retransmission.

Transmit Channel Descriptor Reference Table

The TMAC maintains a Transmit Channel Descriptor Reference (TCDR) table in which is stored certain information relating to DMA activity on each channel together with TD pointers which are used by the TMAC to sort packet chains supplied by the host into per-channel linked lists (see below). The caching of DMA-related information reduces the number of host bus accesses required to process each data packet, while the sorting into per-channel linked lists eliminates head of line blocking. Each channel is provided with two entries in the TCDR table, one for high priority packets (Pri 1) and one for low priority packets (Pri 0). The structure of the TCDR table is shown in Figure 13 below.

Figure 13 – Transmit Channel Descriptor Reference Table

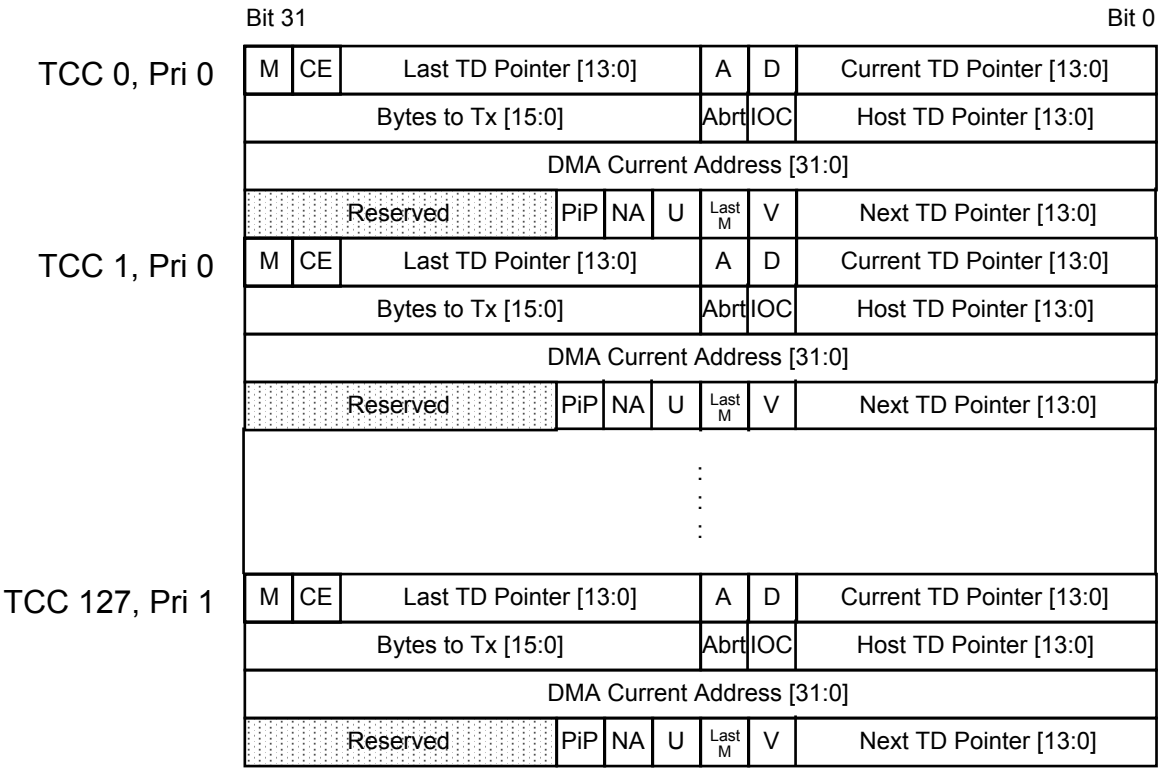


Table 11 – Transmit Channel Descriptor Reference Table Fields

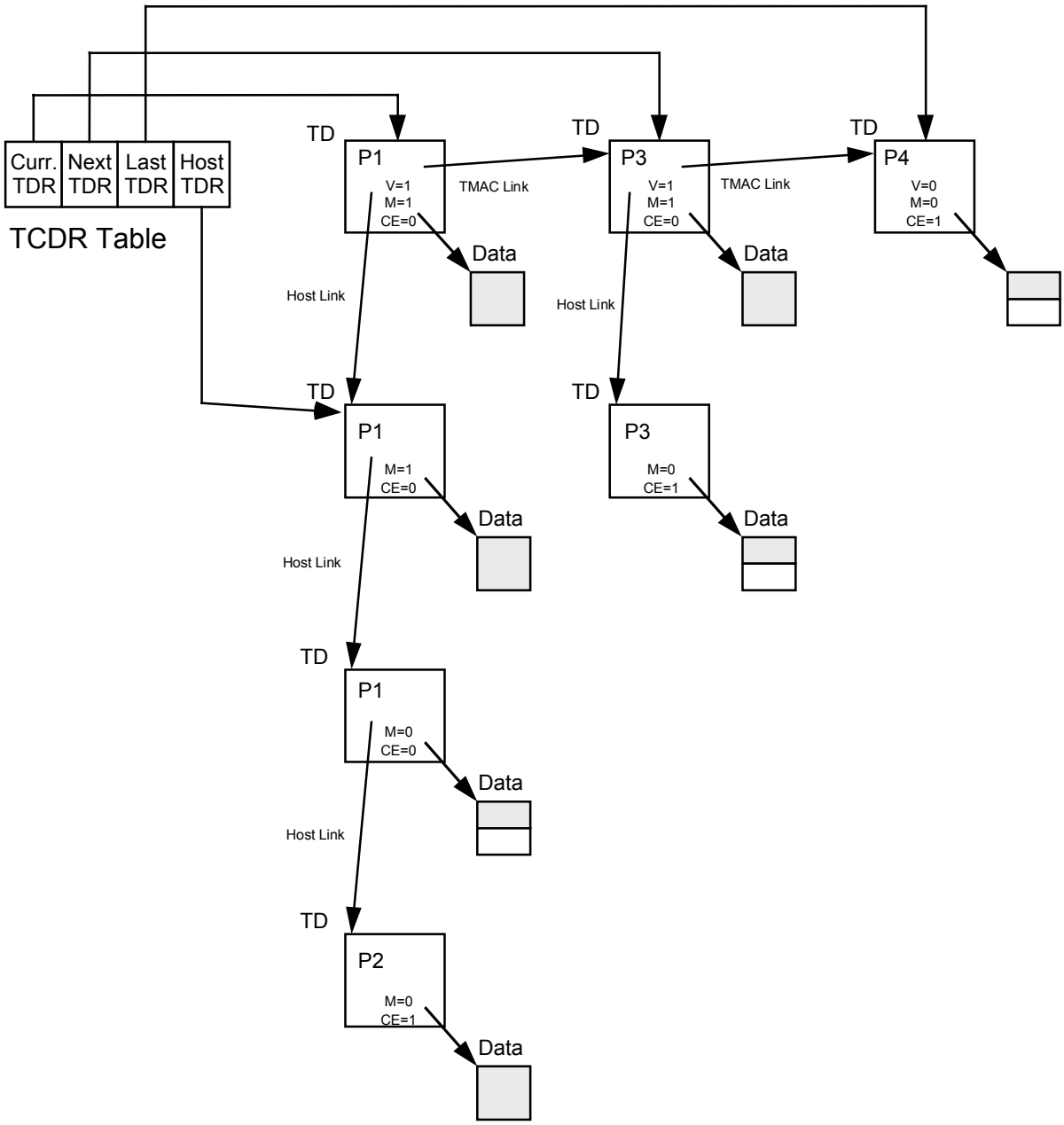
Field	Description
M	A copy of the M bit in the TD currently being read.
CE	A copy of the CE bit in the TD currently being read.
Last TD Pointer [13:0]	Offset to the head of the last host-linked chain of TDs to be read. (See Figure 14)
A	Indicates if this channel is active (i.e. provisioned). If the channel is active, the A bit is set to logic 1. If the channel is inactive, the A bit is set to logic 0.
D	Indicates whether the linked list of packets for this channel is empty or not. If the D bit is set to logic 1, the list is not empty and the current TD pointer field is valid (i.e., it points to a valid TD). If the D bit is set to logic 0, the list is empty and the current TD pointer field is invalid.
Current TD Pointer [13:0]	Offset to the TD currently being read.
Bytes To Tx[15:0]	The Bytes to Tx[15:0] bits are used to indicate the total number of bytes that remain to be read in the current buffer. Each access to the data buffer decrements this value. A value of zero in this field indicates the buffer has been completely read.
ABRT	A copy of the ABRT bit in the TD currently being read.
IOC	A copy of the IOC bit in the TD currently being read.
PiP	The Packet Transfer in Progress bit indicates that a packet is currently being transmitted on this channel at this priority level.
NA	Indicates that a ‘null abort’ is to be sent to the downstream block when it next requests data on this channel. The NA bit is set if a mal-formed TD is encountered while searching down a host chain.
U	Indicates that a underflow has occurred on this channel. This bit is set in response to an underflow indication for the downstream THDL block and is cleared when a TDR is written to the TDR Free Queue (or to the free queue cache).

Field	Description
Host TD Pointer [13:0]	A copy of the Host Next TD Pointer field of the TD currently being read, i.e. a pointer to the next TD in the chain currently being read. (See Figure 14)
DMA Current Address[31:0]	The DMA Current Address [31:0] bits hold the address of the next dword in the current buffer. This field is incremented on each access to the buffer.
V	Indicates if the linked list of packets for this channel contains more than one host-linked chain (See Figure 14). If the V bit is set to logic 1, the list contains more than one chain and the next and last TD pointer fields are valid. If the V bit is set to logic 0, the list is either empty or contains only one host-linked chain and the next and last TD pointer fields are invalid.
Next TD Pointer [13:0]	Offset to the head of the next host-linked chain of TDs to be read. (See Figure 14)

Transmit Descriptor Linking

As described above, the TCDR table contains pointers which the TMAC uses to construct linked lists of data packets to be transmitted. After the host places a new TDR in the TDR Ready queue, the TMAC retrieves the TDR and links it to the TD pointed at by the Last TD Pointer field. The TMAC may create up to 256 linked lists, viz. a high-priority list and a low-priority list for each DMA channel. Whenever a new data packet is requested by the downstream block, the TMAC picks a packet from the high-priority linked list unless it is empty, in which case, a packet from the low-priority linked list is used.

Figure 14 – TD Linking



The host links the TDs vertically while the TMAC links TDs horizontally. Figure 14 shows the TDs for packets P1 and P2 linked by the host before the TDR is placed on the TDRR queue, as are the TDs for packet P3. Packet P3 is linked to packet P1 by the TMAC, as is packet P4 linked to packet P3. The TMAC indicates valid horizontal links by setting the V bit to logic 1.

9.6.2 Task Priorities

The TMAC must perform a number of tasks concurrently in order to maintain a steady flow of data through the system. The main tasks of the TMAC are managing the Ready Queue (i.e. removing chains of data packets from the queue and attaching them to the appropriate per-channel linked list) and servicing requests for data from the Transmit Packet Interface. The priority of service for each of the tasks is fixed by the TMAC as follows:-

- Top priority is given to servicing 'expedited' read requests from the Transmit HDLC Processor / Partial Packet Buffer block (THDL).
- Second priority is given to removing chains of data packets from the TDRR queue and attaching them to the appropriate per-channel linked list.
- Third priority is given to servicing non-expedited read requests from the THDL.

9.6.3 DMA Transaction Controller

The DMA Transaction Controller coordinates the processing of requests from the THDL with the reading of data stored in host memory. The reading of a data packet may require a number of separate host memory transactions, interleaved with transactions of other DMA channels. As well as reading data from the Host Master Interface, the DMA Transaction Controller initiates read and write transactions to the PCI Controller block (GPIC) for the purposes of maintaining the data structures (queues, descriptors, etc.) in host memory.

9.6.4 Read Data Pipeline

The Read Data Pipeline inserts delay in the data stream between the GPIC interface and the THDL interface to enable the DMA Transaction Controller to generate appropriate control signals at the Transmit Packet Interface.

9.6.5 Descriptor Information Cache

The Descriptor Information Cache provides the storage for the Transmit Channel Descriptor Reference (TCDR) Table.

9.6.6 Free Queue Cache

The Free Queue Cache block implements the 6 element TDR Free Queue cache. Caching TDRs reduces the number of host bus accesses that the TMAC makes.

TDRs are written to the cache one at a time as they are released by the TMAC. The cache is then flushed to host memory when it becomes full, when a TD with the IOC bit set high is released or when a TD is released as the result of unprovisioning a channel. The cache controller may also flush the cache when it contains fewer than six elements or if the pointer index is within six elements of the end of the free queue. If the write pointer is near the end of the free queue, the cache controller writes only to the end of the queue and does not start writing from the top of the queue until the next time a flush is required. To do so would require two host memory transactions and would be of no benefit.

9.7 Transmit HDLC Controller / Partial Packet Buffer

The Transmit HDLC Controller / Partial Packet Buffer block (THDL) contains a partial packet buffer for PCI latency control and a transmit HDLC controller. Packet data retrieved from the PCI host memory by the Transmit DMA Controller block (TMAC) is stored in channel specific FIFOs residing in the partial packet buffer. When the amount of data in a FIFO reaches a programmable threshold, the HDLC controller is enabled to initiate transmission. The HDLC controller performs flag generation, bit stuffing and, optionally, frame check sequence (FCS) insertion. The FCS is software selectable to be CRC-CCITT or CRC-32. The minimum packet size, excluding FCS, is two bytes. A single byte payload is illegal. The HDLC controller delivers data to the Transmit Channel Assigner block (TCAS) on demand. A packet in progress is aborted if an under-run occurs. The THDL is programmable to operate in transparent mode where packet data retrieved from the PCI host is transmitted verbatim.

9.7.1 Transmit HDLC Processor

The HDLC processor is a time-slice state machine which can process up to 128 independent channels. The state vector and provisioning information for each channel is stored in a RAM. Whenever the TCAS requests data, the appropriate state vector is read from the RAM, processed and finally written back to the RAM. The HDLC state-machine can be configured to perform flag insertion, bit stuffing and CRC generation. The HDLC processor requests data from the partial packet processor whenever a request for channel data arrives. However, the HDLC processor does not start transmitting a packet until the entire packet is stored in the channel FIFO or until the FIFO free space is less than the software programmable limit. If a channel FIFO under-runs, the HDLC processor aborts the packet.

The configuration of the HDLC processor is accessed using indirect channel read and write operations. When an indirect operation is performed, the information is accessed from RAM during a null clock cycle inserted by the TCAS

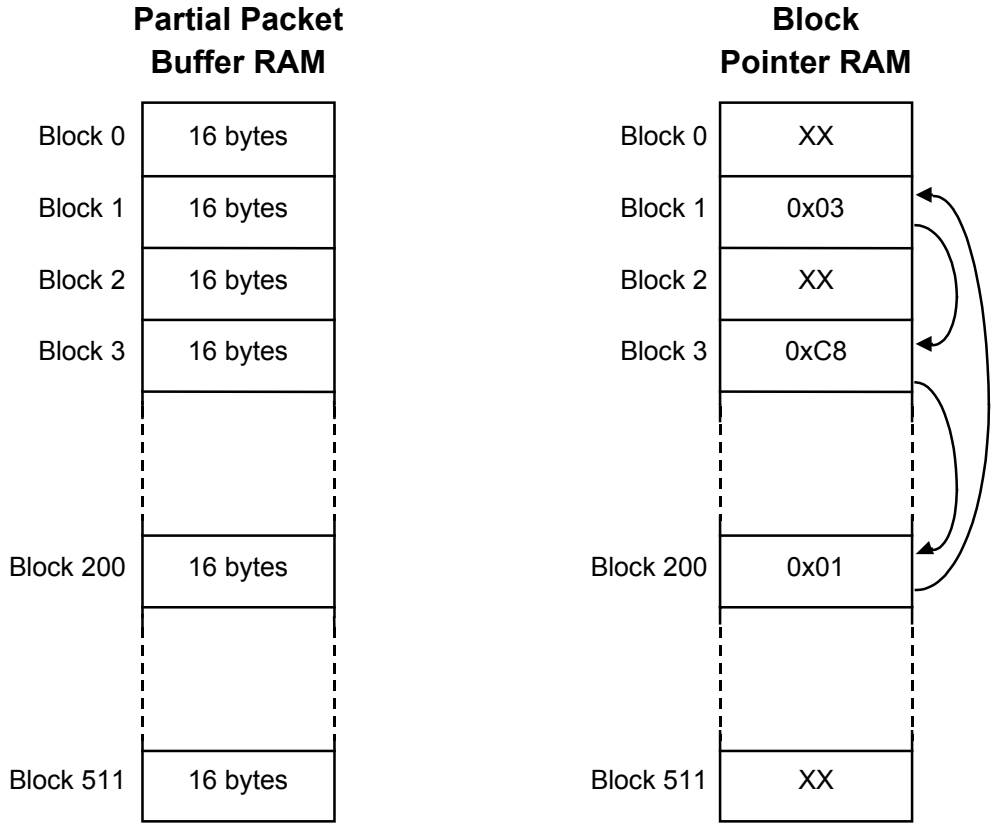
block. Writing new provisioning data to a channel resets the channel's entire state vector.

9.7.2 Transmit Partial Packet Buffer Processor

The partial packet buffer processor controls the 8 Kbyte partial packet RAM which is divided into 16 byte blocks. A block pointer RAM is used to chain the partial packet blocks into circular channel FIFO buffers. Thus, non-contiguous sections of RAM can be allocated in the partial packet buffer RAM to create a channel FIFOFigure 15 shows an example of three blocks (blocks 1, 3, and 200) linked together to form a 48 byte channel FIFO. The three pointer values would be written sequentially using indirect block write accesses. When a channel is provisioned with this FIFO, the state machine can be initialised to point to any one of the three blocks.

The partial packet buffer processor is divided into three sections: reader, writer and roamer. The roamer is a time-sliced state machine which tracks each channel's FIFO buffer free space and signals the writer to service a particular channel. The writer requests data from the TMAC block and transfers packet data from the TMAC to the associated channel FIFO. The reader is a time-sliced state machine which transfers the HDLC information from a channel FIFO to the HDLC processor when the HDLC processor requests it. If a buffer under-run occurs for a channel, the reader informs the HDLC processor and purges the rest of the packet.

Figure 15 – Partial Packet Buffer Structure



The writer and reader determine empty and full FIFO conditions using flags. Each block in the partial packet buffer has an associated flag. The writer sets the flag after the block is written and the reader clears the flag after the block is read. The flags are initialized (cleared) when the block pointers are written using indirect block writes. The reader declares a channel FIFO under-run whenever it tries to read data from a block without a set flag.

The FIFO algorithm of the partial packet buffer processor is based on per-channel software programmable transfer size and free space trigger level. Instead of tracking the number of full blocks in a channel FIFO, the processor tracks the number of empty blocks, called free space, as well as the number of end of packets stored in the FIFO. Recording the number of empty blocks instead of the number of full blocks reduces the amount of information the roamer must store in its state RAM.

The partial packet roamer records the FIFO free space and end-of-packet count for all channel FIFOs. When the reader signals that a block has been read, the roamer increments the FIFO free space and sets a per-channel request flag if

the free space is greater than the limit set by XFER[2:0]. The roamer also decrements the end-of-packet count when the reader signals that it has passed an end of a packet to the HDLC processor. If the HDLC is transmitting a packet and the FIFO free space is greater than the free space trigger level and there are no complete packets within the FIFO (end-of-packet count equal to zero), a per-channel expedite flag is set. The roamer searches the expedite flags in a round-robin fashion to decide which channel FIFO should make expedited data requests to the TMAC block. If no expedite flags are set, the roamer searches the request flags in a round-robin fashion to decide which channel FIFO should make regular data requests to the TMAC block. The roamer informs the partial packet writer of the channel FIFO to process, the FIFO free space and the type of request it should make. The writer sends a request for data to the TMAC block and writes the response data to the channel FIFO setting block full flags. The writer reports back to the roamer the number of blocks and end-of-packets transferred. The maximum amount of data transferred during one request is limited by a software programmable limit.

The configuration of the HDLC processor is accessed using indirect channel read and write operations as well as indirect block read and write operations. When an indirect operation is performed, the information is accessed from RAM during a null clock cycle identified by the TCAS block. Writing new provisioning data to a channel resets the entire state vector.

9.8 Transmit Channel Assigner

The Transmit Channel Assigner block (TCAS) processes up to 128 channels. Data for all channels is sourced from a single byte-serial stream from the Transmit HDLC Controller / Partial Packet Buffer block (THDL). The TCAS demultiplexes the data and assigns each byte to any one of 32 links. Each link is independent and has its own associated clock. For each high-speed link (TD[2:0]), the TCAS provides a six byte FIFO. For the remaining links (TD[31:3]), the TCAS provides a holding register. The TCAS also performs parallel to serial conversion to form a bit-serial stream. In the event where multiple links are in need of data, TCAS requests data from upstream blocks on a fixed priority basis with link TD[0] having the highest priority and link TD[31] the lowest.

Links containing a T1 or an E1 stream may be channelised. Data at each time-slot may be independently assigned to be sourced from a different channel. The link clock is only active during time-slots 1 to 24 of a T1 stream and is inactive during the frame bit. Similarly, the clock is only active during time-slots 1 to 31 of an E1 stream and is inactive during the FAS and NFAS framing bytes. The most significant bit of time-slot 1 of a channelised link is identified by noting the absence of the clock and its re-activation. With knowledge of the transmit link

and time-slot identity, the TCAS performs a table look-up to identify the channel from which a data byte is to be sourced.

Links may also be unchannelised. Then, all data bytes on that link belong to one channel. The TCAS performs a table look-up to identify the channel to which a data byte belongs using only the outgoing link identity, as no time-slots are associated with unchannelised links. Link clocks are no longer limited to T1 or E1 rates and may range up to 52 MHz for TCLK[2:0]. For TCLK[31:3] the maximum clock rate is 10 MHz. The link clock is only active during bit times containing data to be transmitted and inactive during bits that are to be ignored by the downstream devices, such as framing and overhead bits. For the case of two unchannelised links, the maximum link rate is 45 MHz for SYSCLK at 25 MHz and 52 MHz for SYSCLK at 33 MHz. For the case of more numerous unchannelised links or a mixture of channelised and unchannelised links, the total instantaneous link rate over all the links is limited to 64 MHz.

9.8.1 Line Interface

There are two types of line interfaces in the TCAS; high-speed and low-speed interfaces. Three identical high-speed interfaces are attached to the first three links, while 29 identical low-speed interfaces are attached to the remaining links. Each line interface contains a bit counter, an 8-bit shift register and a byte FIFO, that, together, perform parallel to serial conversion. For the high-speed interfaces the FIFO is six bytes deep. For the low-speed interfaces, the FIFO is a single byte holding register. Whenever the shift register is updated, a request for service is sent to the priority encoder block. The request will eventually be serviced by the THDL block and the data is written into the FIFO.

To support channelised links, each line interface block contains a time-slot counter and a clock activity monitor. The time-slot counter is incremented each time the shift register is updated. The clock activity monitor is a counter that increments at the system clock (SYSCLK) rate and is cleared by a rising edge of the transmit clock (TCLK[n]). A framing bit (T1) or framing byte (E1) is detected when the counter reaches a programmable threshold. At which point, the bit and time-slot counters are initialised to indicate the next bit sampled is the most significant bit of the first time-slot. For unchannelised links, the time-slot counter and the clock activity monitor are held reset.

9.8.2 Priority Encoder

The priority encoder monitors the line interfaces for requests and synchronises them to the SYSCLK timing domain. Requests are serviced on a fixed priority scheme where highest to lowest priority is assigned from line interface TD[0] to line interface TD[31]. Thus, simultaneous requests from line interface TD[m] will be serviced ahead of line interface TD[n], if m < n. The priority encoder selects

the request from the link with the highest priority for service. When there are no pending requests, the priority encoder generates an idle cycle. In addition, once every fourth SYSCLK cycle, the priority encoder inserts a null cycle where no requests are serviced. This cycle is used by the channel assigner downstream for CBI accesses to the channel provision RAM.

9.8.3 Channel Assigner

The channel assigner block determines the channel number of the request currently being processed. The block contains a 1024 word channel provision RAM. The address of the RAM is constructed from concatenating the link number and the time-slot number of the highest priority requester. The fields of each RAM word include the channel number and a time-slot enable flag. The time-slot enable flag labels the current time-slot as belonging to the channel indicted by the channel number field. For time-slots that are enabled, the channel assigner issues a request to the THDL block which responds with packet data within one byte period of the transmit stream.

9.9 Performance Monitor

The Performance Monitor block (PMON) contains four counters. The first two accumulate receive partial packet buffer FIFO overrun events and transmit partial packet buffer FIFO underflow events, respectively. The remaining two counters are software programmable to accumulate a variety of events, such as receive packet count, FCS error counts, etc. All counters saturate upon reaching maximum value. The accumulation logic consists of a counter and holding register pair. The counter is incremented when the associated event is detected. Writing to the FREEDM-32 Master Clock / BERT Activity Monitor and Accumulation Trigger register transfer the count to the corresponding holding register and clear the counter. The contents of the holding register is accessible via the PCI interface.

9.10 JTAG Test Access Port Interface

The JTAG Test Access Port block provides JTAG support for boundary scan. The standard JTAG EXTEST, SAMPLE, BYPASS, IDCODE and STCTEST instructions are supported. The FREEDM-32 identification code is 173640CD hexadecimal.

9.11 PCI Host Interface

The FREEDM-32 supports two different normal mode register types as defined below:

1. PCI Host Accessible registers (PA) - these registers can be accessed through the PCI Host interface.

2. PCI Configuration registers (PC) - these register can only be accessed through the PCI Host interface during a PCI configuration cycle.

The PCI registers are addressable on dword boundaries only. The PCI offset shown in the table below must be combined with a base address to form the PCI Interface address. The base address can be found in the FREEDM-32 Memory Base Address register in the PCI Configuration memory space.

Table 12 – Normal Mode PCI Host Accessible Register Memory Map

PCI Offset	Register
0x000	FREEDM-32 Master Reset
0x004	FREEDM-32 Master Interrupt Enable
0x008	FREEDM-32 Master Interrupt Status
0x00C	FREEDM-32 Master Clock / BERT Activity Monitor and Accumulation Trigger
0x010	FREEDM-32 Master Link Activity Monitor
0x014	FREEDM-32 Master Line Loopback #1
0x018	FREEDM-32 Master Line Loopback #2
0x01C	Reserved
0x020	FREEDM-32 Master BERT Control
0x024	FREEDM-32 Master Performance Monitor Control
0x028 - 0x03C	Reserved
0x040	GPIC Control
0x044 - 0x07C	GPIC Reserved
0x080 - 0x0FC	Reserved
0x100	RCAS Indirect Channel and Time-slot Select
0x104	RCAS Indirect Channel Data
0x108	RCAS Framing Bit Threshold
0x10C	RCAS Channel Disable
0x110 - 0x17C	RCAS Reserved
0x180	RCAS Link #0 Configuration

PCI Offset	Register
0x184 - 0x1FC	RCAS Link #1 to Link #31 Configuration
0x200	RHDL Indirect Channel Select
0x204	RHDL Indirect Channel Data Register #1
0x208	RHDL Indirect Channel Data Register #2
0x20C	RHDL Reserved
0x210	RHDL Indirect Block Select
0x214	RHDL Indirect Block Data Register
0x218	RHDL Reserved
0x21C	RHDL Reserved
0x220	RHDL Configuration
0x224	RHDL Maximum Packet Length
0x228 - 0x23C	RHDL Reserved
0x240 - 0x27C	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

PCI Offset	Register
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
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
0x380	THDL Indirect Channel Select
0x384	THDL Indirect Channel Data #1
0x388	THDL Indirect Channel Data #2
0x38C	THDL Indirect Channel Data #3
0x390	THDL Reserved

PCI Offset	Register
0x394	THDL Reserved
0x398	THDL Reserved
0x39C	THDL Reserved
0x3A0	THDL Indirect Block Select
0x3A4	THDL Indirect Block Data
0x3A8	THDL Reserved
0x3AC	THDL Reserved
0x3B0	THDL Configuration
0x3B4 - 0x3BC	THDL Reserved
0x3C0 - 0x3FF	Reserved
0x400	TCAS Indirect Channel and Time-slot Select
0x404	TCAS Indirect Channel Data
0x408	TCAS Framing Bit Threshold
0x40C	TCAS Idle Time-slot Fill Data and Config.
0x410	TCAS Channel Disable
0x414 - 0x47C	TCAS Reserved
0x480	TCAS Link #0 Configuration
0x484 - 0x4FC	TCAS Link #1 to Link #31 Configuration
0x500	PMON Status
0x504	PMON Receive FIFO Overflow Count
0x508	PMON Transmit FIFO Underflow Count
0x50C	PMON Configurable Count #1
0x510	PMON Configurable Count #2
0x514 - 0x51C	PMON Reserved
0x520 - 0x7FC	Reserved

The following PCI configuration registers are implemented by the PCI Interface. These registers can only be accessed when the PCI Interface is a target and a configuration cycle is in progress as indicated using the IDSEL input.

Table 13 – PCI Configuration Register Memory Map

PCI Offset	Register
0x00	Vendor Identification/Device Identification
0x04	Command/Status
0x08	Revision Identifier/Class Code
0x0C	Cache Line Size/Latency Timer/Header Type/BIST
0x10	CBI Memory Base Address Register
0x14 - 0x24	Unused Base Address Register
0x28	Reserved
0x2C	Reserved
0x30	Reserved
0x34	Reserved
0x38	Reserved
0x3C	Interrupt Line/Interrupt Pin/MIN_GNT/MAX_LAT

10 NORMAL MODE REGISTER DESCRIPTION

Normal mode registers are used to configure and monitor the operation of the FREEDM.

Notes on Normal Mode Register Bits:

1. Writing values into unused register bits has no effect. However, to ensure software compatibility with future, feature-enhanced versions of the product, unused register bits must 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.

2. Except where noted, all configuration bits that can be written into can also be read back. This allows the processor controlling the FREEDM-32 to determine the programming state of the block.

3. Writable normal mode register bits are cleared to logic zero upon reset unless otherwise noted.

4. Writing into read-only normal mode register bit locations does not affect FREEDM-32 operation unless otherwise noted.

5. Certain register bits are reserved. These bits are associated with megacell functions that are unused in this application. To ensure that the FREEDM-32 operates as intended, reserved register bits must only be written with their default values. Similarly, writing to reserved registers should be avoided.

10.1 PCI Host Accessible Registers

PCI host accessible registers can be accessed by the PCI host. For each register description below, the hexadecimal register number indicates the PCI offset from the base address in the FREEDM-32 CBI Register Base Address Register when accesses are made using the PCI Host Port.

Note

These registers are not byte addressable. Writing to any one of these registers modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to the register.

Register 0x000 : FREEDM-32 Master Reset

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	Reset	0
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3		Unused	X
Bit 2		Unused	X
Bit 1		Unused	X
Bit 0		Unused	X

This register provides software reset capability.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RESET:

The RESET bit allows the FREEDM-32 to be reset under software control. If the RESET bit is a logic one, the entire FREEDM-32 except the PCI Interface

is held in reset. This bit is not self-clearing. Therefore, a logic zero must be written to bring the FREEDM-32 out of reset. Holding the FREEDM-32 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.

Note

Unlike the hardware reset input (RSTB), RESET does not force the FREEDM-32's PCI pins tri-state. Transmit link data pins (TD[31:0]) are forced high. In addition, all registers except the GPIC PCI Configuration registers, are reset to their default values.

Register 0x004 : FREEDM-32 Master Interrupt Enable

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TFUDRE	0
Bit 14	R/W	IOCE	0
Bit 13	R/W	TDFQEE	0
Bit 12	R/W	TDQRDYE	0
Bit 11	R/W	TDQFE	0
Bit 10	R/W	RPDRQEE	0
Bit 9	R/W	RPDFQEE	0
Bit 8	R/W	RPQRDYE	0
Bit 7	R/W	RPQLFE	0
Bit 6	R/W	RPQSFE	0
Bit 5	R/W	RFOVRE	0
Bit 4	R/W	RPFEE	0
Bit 3	R/W	RABRTE	0
Bit 2	R/W	RFCSEE	0
Bit 1	R/W	PERRE	0
Bit 0	R/W	SERRE	0

This register provides interrupt enables for various events detected or initiated by the FREEDM.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

SERRE:

The system error interrupt enable bit (SERRE) enables PCI system error interrupts to the PCI host. When SERRE is set high, any address parity error, data parity error on Special Cycle commands, reception of a master abort or detection of a target abort will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when SERRE is set low. However, the SERRI bit remains valid when interrupts are disabled and may be polled to detect PCI system error events.

PERRE:

The parity error interrupt enable bit (PERRE) enables PCI parity error interrupts to the PCI host. When PERRE is set high, data parity errors detected by the FREEDM-32 or parity errors reported by a target will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when PERRE is set low. However, the PERRI bit remains valid when interrupts are disabled and may be polled to detect PCI parity error events.

RFCSEE:

The receive frame check sequence error interrupt enable bit (RFCSEE) enables receive FCS error interrupts to the PCI host. When RFCSEE is set high, a mismatch between the received FCS code and the computed CRC residue will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when RFCSEE is set low. However, the RFCSEI bit remains valid when interrupts are disabled and may be polled to detect receive FCS error events.

RABRTE:

The receive abort interrupt enable bit (RABRTE) enables receive HDLC abort interrupts to the PCI host. When RABRTE is set high, receipt of an abort code (at least 7 contiguous 1's) will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when RABRTE is set low. However, the RABRTI bit remains valid when interrupts are disabled and may be polled to detect receive abort events.

RPFEE:

The receive packet format error interrupt enable bit (RPFEE) enables receive packet format error interrupts to the PCI host. When RPFEE is set high, receipt of a packet that is longer than the maximum specified in the RHDL Maximum Packet Length register, of a packet that is shorter than 32 bits (CRC-CCITT) or 48 bits (CRC-32), or of a packet that is not octet aligned will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when RPFEE is set low. However, the RPFEI bit remains valid when

interrupts are disabled and may be polled to detect receive packet format error events.

RFOVRE:

The receive FIFO overrun error interrupt enable bit (RFOVRE) enables receive FIFO overrun error interrupts to the PCI host. When RFOVRE is set high, attempts to write data into the logical FIFO of a channel when it is already full will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when RFOVRE is set low. However, the RFOVRI bit remains valid when interrupts are disabled and may be polled to detect receive FIFO overrun events.

RPQSFE:

The receive packet descriptor small buffer free queue cache read interrupt enable bit (RPQSFE) enables receive packet descriptor small free queue cache read interrupts to the PCI host. When RPQSFE is set high, reading a programmable number of RPDR blocks from the RPDR Small Buffer Free Queue will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when RPQSFE is set low. However, the RPQSFI bit remains valid when interrupts are disabled and may be polled to detect RPDR small buffer free queue cache read events.

RPQLFE:

The receive packet descriptor large buffer free queue cache read interrupt enable bit (RPQLFE) enables receive packet descriptor large free queue cache read interrupts to the PCI host. When RPQLFE is set high, reading a programmable number of RPDR blocks from the RPDR Large Buffer Free Queue will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when RPQLFE is set low. However, the RPQLFI bit remains valid when interrupts are disabled and may be polled to detect RPDR large buffer free queue cache read events.

RPQRDYE:

The receive packet descriptor ready queue write interrupt enable bit (RPQRDYE) enables receive packet descriptor ready queue write interrupts to the PCI host. When RPQRDYE is set high, writing a programmable number of RPDRs to the RPDR Ready Queue will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when RPQRDYE is set low. However, the RPQRDYI bit remains valid when interrupts are disabled and may be polled to detect RPDR ready queue write events.

RPDFQEE:

The receive packet descriptor free queue error interrupt enable bit (RPDFQEE) enables receive packet descriptor free queue error interrupts to the PCI host. When RPDFQEE is set high, attempts to retrieve an RPDR when both the large buffer and small buffer free queues are empty will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when RPDFQEE is set low. However, the RPDFQEI bit remains valid when interrupts are disabled and may be polled to detect RPDR free queue empty error events.

RPDRQEE:

The receive packet descriptor ready queue error interrupt enable bit (RPDRQEE) enables receive packet descriptor ready queue error interrupts to the PCI host. When RPDRQEE is set high, attempts to write an RPDR when ready queue is ready full will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when RPDRQEE is set low. However, the RPDRQEI bit remains valid when interrupts are disabled and may be polled to detect RPDR ready queue full error events.

TDQFE:

The transmit packet descriptor free queue write interrupt enable bit (TDQFE) enables transmit packet descriptor free queue write interrupts to the PCI host. When TDQFE is set high, writing a programmable number of TDRs to the TDR Free Queue will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when TDQFE is set low. However, the TDQFI bit remains valid when interrupts are disabled and may be polled to detect TDR free queue write events.

TDQRDYE:

The transmit descriptor ready queue cache read interrupt enable bit (TDQRDYE) enables transmit descriptor ready queue cache read interrupts to the PCI host. When TDQRDYE is set high, reading a programmable number of TDRs from the TDR Ready Queue will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when TDQRDYE is set low. However, the TDQRDYI bit remains valid when interrupts are disabled and may be polled to detect TDR ready queue cache read events.

TDFQEE:

The transmit descriptor free queue error interrupt enable bit (TDFQEE) enables transmit descriptor free queue error interrupts to the PCI host. When TDFQEE is set high, attempting to write to the transmit free queue while the queue is full will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when TDFQEE is set low. However, the TDFQEI bit

remains valid when interrupts are disabled and may be polled to detect TD free queue error events.

IOCE:

The transmit interrupt on complete enable bit (IOCE) enables transmission complete interrupts to the PCI host. When IOCE is set high, complete transmission of a packet with the IOC bit in the TD set high will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when IOCE is set low. However, the IOCI bit remains valid when interrupts are disabled and may be polled to detect transmission of IOC tagged packets.

TFUDRE:

The transmit FIFO underflow error interrupt enable bit (TFUDRE) enables transmit FIFO underflow error interrupts to the PCI host. When TFUDRE is set high, attempts to read data from the logical FIFO when it is already empty will cause an interrupt to be generated on the PCIINTB output. Interrupts are masked when TFUDRE is set low. However, the TFUDRI bit remains valid when interrupts are disabled and may be polled to detect transmit FIFO underflow events.

Register 0x008 : FREEDM-32 Master Interrupt Status

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	TFUDRI	X
Bit 14	R	IOCI	X
Bit 13	R	TDFQEI	X
Bit 12	R	TDQRDYI	X
Bit 11	R	TDQFI	X
Bit 10	R	RPDRQEI	X
Bit 9	R	RPDFQEI	X
Bit 8	R	RPQRDYI	X
Bit 7	R	RPQLFI	X
Bit 6	R	RPQSFI	X
Bit 5	R	RFOVRI	X
Bit 4	R	RPFEI	X
Bit 3	R	RABRTI	X
Bit 2	R	RFCSEI	X
Bit 1	R	PERRI	X
Bit 0	R	SERRI	X

This register reports the interrupt status for various events detected or initiated by the FREEDM. Reading this registers acknowledges and clears the interrupts.

Note

This register is not byte addressable. Reading this register clears all the interrupt bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

SERRI:

The system error interrupt status bit (SERRI) reports PCI system error interrupts to the PCI host. SERRI is set high upon detection of any address parity error, data parity error on Special Cycle commands, reception of a master abort or detection of a target abort event. The SERRI bit remains valid when interrupts are disabled and may be polled to detect PCI system error events.

PERRI:

The parity error interrupt status bit (PERRI) reports PCI parity error interrupts to the PCI host. PERRI is set high when data parity errors are detected by the FREEDM-32 while acting as a master, and when parity errors are reported to the FREEDM-32 by a target via the PERRB input. The PERRI bit remains valid when interrupts are disabled and may be polled to detect PCI parity error events.

RFCSEI:

The receive frame check sequence error interrupt status bit (RFCSEI) reports receive FCS error interrupts to the PCI host. RFCSEI is set high, when a mismatch between the received FCS code and the computed CRC residue is detected. RFCSEI remains valid when interrupts are disabled and may be polled to detect receive FCS error events.

RABRTI:

The receive abort interrupt status bit (RABRTI) reports receive HDLC abort interrupts to the PCI host. RABRTI is set high upon receipt of an abort code (at least 7 contiguous 1's). RABRTI remains valid when interrupts are disabled and may be polled to detect receive abort events.

RPFEI:

The receive packet format error interrupt status bit (RPFEI) reports receive packet format error interrupts to the PCI host. RPFEI is set high upon receipt of a packet that is longer than the maximum programmed length, of a packet that is shorter than 32 bits (CRC-CCITT) or 48 bits (CRC-32), or of a packet that is not octet aligned. RPFEI remains valid when interrupts are disabled and may be polled to detect receive packet format error events.

RFOVRI:

The receive FIFO overrun error interrupt status bit (RFOVRI) reports receive FIFO overrun error interrupts to the PCI host. RFOVRI is set high on attempts to write data into the logical FIFO of a channel when it is already full. RFOVRI remains valid when interrupts are disabled and may be polled to detect receive FIFO overrun events.

RPQSFI:

The receive packet descriptor small buffer free queue cache read interrupt status bit (RPQSFI) reports receive packet descriptor small free queue cache read interrupts to the PCI host. RPQSFI is set high when the programmable number of RPDR blocks is read from the RPDR Small Buffer Free Queue. RPQSFI remains valid when interrupts are disabled and may be polled to detect RPDR small buffer free queue cache read events.

RPQLFI:

The receive packet descriptor large buffer free queue cache read interrupt status bit (RPQLFI) reports receive packet descriptor large free queue cache read interrupts to the PCI host. RPQLFI is set high when the programmable number of RPDR blocks is read from the RPDR Large Buffer Free Queue. RPQLFI remains valid when interrupts are disabled and may be polled to detect RPDR large buffer free queue cache read events.

RPQRDYI:

The receive packet descriptor ready queue write interrupt status bit (RPQRDYI) reports receive packet descriptor ready queue write interrupts to the PCI host. RPQRDYI is set high when the programmable number of RPDRs is written to the RPDR Ready Queue. RPQRDYI remains valid when interrupts are disabled and may be polled to detect RPDR ready queue write events.

RPDFQEI:

The receive packet descriptor free queue error interrupt status bit (RPDFQEI) reports receive packet descriptor free queue error interrupts to the PCI host. RPDFQEI is set high upon attempts to retrieve an RPDR when both the large buffer and small buffer free queues are empty. RPDFQEI remains valid when interrupts are disabled and may be polled to detect RPDR free queue empty error events.

RPDRQEI:

The receive packet descriptor ready queue error interrupt status bit (RPDRQEI) reports receive packet descriptor ready queue error interrupts to the PCI host. RPDRQEI is set high upon attempts to write an RPDR when ready queue is ready full. RPDRQEI remains valid when interrupts are disabled and may be polled to detect RPDR ready queue full error events.

TDQFI:

The transmit packet descriptor free queue write interrupt status bit (TDQFI) reports transmit packet descriptor free queue write interrupts to the PCI host. TDQFI is set high when the programmable number of TDRs is written to the

TDR Free Queue. TDQFI remains valid when interrupts are disabled and may be polled to detect TDR free queue write events.

TDQRDYI:

The transmit descriptor ready queue cache read interrupt status bit (TDQRDYI) reports transmit descriptor ready queue cache read interrupts to the PCI host. TDQRDYI is set high when the programmable number of TDRs is read from the TDR Ready Queue. TDQRDYI remains valid when interrupts are disabled and may be polled to detect TDR ready queue cache read events.

TDFQEI:

The transmit descriptor free queue error interrupt status bit (TDFQEI) reports transmit descriptor free queue error interrupts to the PCI host. TDFQEI is set high when an attempt to write to the transmit free queue fail due to the queue being already full. TDFQEI bit remains valid when interrupts are disabled and may be polled to detect TD free queue error events.

IOCI:

The transmit interrupt on complete status bit (IOCI) reports transmission complete interrupts to the PCI host. IOCI is set high, when a packet with the IOC bit in the TD set high is completely transmitted. IOCI remains valid when interrupts are disabled and may be polled to detect transmission of IOC tagged packets.

TFUDRI:

The transmit FIFO underflow error interrupt status bit (TFUDRI) reports transmit FIFO underflow error interrupts to the PCI host. TFUDRI is set high upon attempts to read data from the logical FIFO when it is already empty. TFUDRI remains valid when interrupts are disabled and may be polled to detect transmit FIFO underflow events.

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

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3		Unused	X
Bit 2		Unused	X
Bit 1	R	TBDA	X
Bit 0	R	SYSCLKA	X

This register provides activity monitoring on FREEDM-32 clock inputs and BERT port input. When a monitored input makes a low to high transition, the corresponding register bit is set high. The bit will remain high until this register is read, at which point, all the bits in this register are cleared. A lack of transitions is indicated by the corresponding register bit reading low. This register should be read periodically to detect for stuck at conditions.

Writing to this register delimits the accumulation intervals in the PMON accumulation registers. Counts accumulated in those registers are transferred to holding registers where they can be read. The counters themselves are then

cleared to begin accumulating events for a new accumulation interval. The bits in this register are not affected by write accesses.

Note

This register is not byte addressable. Reading this register clears all the activity bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

SYSCLKA:

The system clock active bit (SYSCLKA) monitors for low to high transitions on the SYSCLK input. SYSCLKA is set high on a rising edge of SYSCLK, and is set low when this register is read.

TBDA:

The transmit BERT data active bit (TBDA) monitors for low to high transitions on the TBD input. TBDA is set high on a rising edge of TDB, and is set low when this register is read.

Register 0x010 : FREEDM-32 Master Link Activity Monitor

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	TLGA[7]	X
Bit 14	R	TLGA[6]	X
Bit 13	R	TLGA[5]	X
Bit 12	R	TLGA[4]	X
Bit 11	R	TLGA[3]	X
Bit 10	R	TLGA[2]	X
Bit 9	R	TLGA[1]	X
Bit 8	R	TLGA[0]	X
Bit 7	R	RLGA[7]	X
Bit 6	R	RLGA[6]	X
Bit 5	R	RLGA[5]	X
Bit 4	R	RLGA[4]	X
Bit 3	R	RLGA[3]	X
Bit 2	R	RLGA[2]	X
Bit 1	R	RLGA[1]	X
Bit 0	R	RLGA[0]	X

This register provides activity monitoring on FREEDM-32 receive and transmit link inputs. When a monitored input makes a low to high transition, the corresponding register bit is set high. The bit will remain high until this register is read, at which point, all the bits in this register are cleared. A lack of transitions is indicated by the corresponding register bit reading low. This register should be read at periodically to detect for stuck at conditions.

Note

This register is not byte addressable. Reading this register clears all the activity bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not

implemented. However, when all four byte enables are negated, no access is made to this register.

RLGA[0]:

The receive link group #0 active bit (RLGA[0]) monitors for transitions on the RD[3:0] and RCLK[3:0] inputs. RLGA[0] is set high when each of RD[3:0] has been sampled low and sampled high by rising edges of the corresponding RCLK[3:0] inputs, and is set low when this register is read.

RLGA[1]:

The receive link group #1 active bit (RLGA[1]) monitors for transitions on the RD[7:4] and RCLK[7:4] inputs. RLGA[1] is set high when each of RD[7:4] has been sampled low and sampled high by rising edges of the corresponding RCLK[7:4] inputs, and is set low when this register is read.

RLGA[2]:

The receive link group #2 active bit (RLGA[2]) monitors for transitions on the RD[11:8] and RCLK[11:8] inputs. RLGA[2] is set high when each of RD[11:8] has been sampled low and sampled high by rising edges of the corresponding RCLK[11:8] inputs, and is set low when this register is read.

RLGA[3]:

The receive link group #3 active bit (RLGA[3]) monitors for transitions on the RD[15:12] and RCLK[15:12] inputs. RLGA[3] is set high when each of RD[15:12] has been sampled low and sampled high by rising edges of the corresponding RCLK[15:12] inputs, and is set low when this register is read.

RLGA[4]:

The receive link group #4 active bit (RLGA[4]) monitors for transitions on the RD[19:16] and RCLK[19:16] inputs. RLGA[4] is set high when each of RD[19:16] has been sampled low and sampled high by rising edges of the corresponding RCLK[19:16] inputs, and is set low when this register is read.

RLGA[5]:

The receive link group #5 active bit (RLGA[5]) monitors for transitions on the RD[23:20] and RCLK[23:20] inputs. RLGA[5] is set high when each of RD[23:20] has been sampled low and sampled high by rising edges of the corresponding RCLK[23:20] inputs, and is set low when this register is read.

RLGA[6]:

The receive link group #6 active bit (RLGA[6]) monitors for transitions on the RD[27:24] and RCLK[27:24] inputs. RLGA[6] is set high when each of

RD[27:24] has been sampled low and sampled high by rising edges of the corresponding RCLK[27:24] inputs, and is set low when this register is read.

RLGA[7]:

The receive link group #7 active bit (RLGA[7]) monitors for transitions on the RD[31:28] and RCLK[31:28] inputs. RLGA[7] is set high when each of RD[31:28] has been sampled low and sampled high by rising edges of the corresponding RCLK[31:28] inputs, and is set low when this register is read.

TLGA[0]:

The transmit link group #0 active bit (TLGA[0]) monitors for low to high transitions on the TCLK[3:0] inputs. TLGA[0] is set high when rising edges have been observed on all the signals on the TCLK[3:0] inputs, and is set low when this register is read.

TLGA[1]:

The transmit link group #1 active bit (TLGA[1]) monitors for low to high transitions on the TCLK[7:4] inputs. TLGA[1] is set high when rising edges have been observed on all the signals on the TCLK[7:4] inputs, and is set low when this register is read.

TLGA[2]:

The transmit link group #2 active bit (TLGA[2]) monitors for low to high transitions on the TCLK[11:8] inputs. TLGA[2] is set high when rising edges have been observed on all the signals on the TCLK[11:8] inputs, and is set low when this register is read.

TLGA[3]:

The transmit link group #3 active bit (TLGA[3]) monitors for low to high transitions on the TCLK[15:12] inputs. TLGA[3] is set high when rising edges have been observed on all the signals on the TCLK[15:12] inputs, and is set low when this register is read.

TLGA[4]:

The transmit link group #4 active bit (TLGA[4]) monitors for low to high transitions on the TCLK[19:16] inputs. TLGA[4] is set high when rising edges have been observed on all the signals on the TCLK[19:16] inputs, and is set low when this register is read.

TLGA[5]:

The transmit link group #5 active bit (TLGA[5]) monitors for low to high transitions on the TCLK[23:20] inputs. TLGA[5] is set high when rising edges

have been observed on all the signals on the TCLK[23:20] inputs, and is set low when this register is read.

TLGA[6]:

The transmit link group #6 active bit (TLGA[6]) monitors for low to high transitions on the TCLK[27:24] inputs. TLGA[6] is set high when rising edges have been observed on all the signals on the TCLK[27:24] inputs, and is set low when this register is read.

TLGA[7]:

The transmit link group #7 active bit (TLGA[7]) monitors for low to high transitions on the TCLK[31:28] inputs. TLGA[7] is set high when rising edges have been observed on all the signals on the TCLK[31:28] inputs, and is set low when this register is read.

Register 0x014 : FREEDM-32 Master Line Loopback #1

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	LLBEN[15]	0
Bit 14	R/W	LLBEN[14]	0
Bit 13	R/W	LLBEN[13]	0
Bit 12	R/W	LLBEN[12]	0
Bit 11	R/W	LLBEN[11]	0
Bit 10	R/W	LLBEN[10]	0
Bit 9	R/W	LLBEN[9]	0
Bit 8	R/W	LLBEN[8]	0
Bit 7	R/W	LLBEN[7]	0
Bit 6	R/W	LLBEN[6]	0
Bit 5	R/W	LLBEN[5]	0
Bit 4	R/W	LLBEN[4]	0
Bit 3	R/W	LLBEN[3]	0
Bit 2	R/W	LLBEN[2]	0
Bit 1	R/W	LLBEN[1]	0
Bit 0	R/W	LLBEN[0]	0

This register controls line loopback for links #0 to #15.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

LLBEN[15:0]:

The line loopback enable bits (LLBEN[15:0]) controls line loopback for links #15 to #0. When LLBEN[n] is set high, the data on RD[n] is passed

verbatim to TD[n] which is then updated on the falling edge of RCLK[n]. TCLK[n] is ignored. When LLBEN[n] is set low, TD[n] is processed normally.

Register 0x018 : FREEDM-32 Master Line Loopback #2

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	LLBEN[31]	0
Bit 14	R/W	LLBEN[30]	0
Bit 13	R/W	LLBEN[29]	0
Bit 12	R/W	LLBEN[28]	0
Bit 11	R/W	LLBEN[27]	0
Bit 10	R/W	LLBEN[26]	0
Bit 9	R/W	LLBEN[25]	0
Bit 8	R/W	LLBEN[24]	0
Bit 7	R/W	LLBEN[23]	0
Bit 6	R/W	LLBEN[22]	0
Bit 5	R/W	LLBEN[21]	0
Bit 4	R/W	LLBEN[20]	0
Bit 3	R/W	LLBEN[19]	0
Bit 2	R/W	LLBEN[18]	0
Bit 1	R/W	LLBEN[17]	0
Bit 0	R/W	LLBEN[16]	0

This register controls line loopback for links #16 to #31.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

LLBEN[31:16]:

The line loopback enable bits (LLBEN[31:16]) controls line loopback for links #31 to #16. When LLBEN[n] is set high, the data on RD[n] is passed

verbatim to TD[n] which is then updated on the falling edge of RCLK[n]. TCLK[n] is ignored. When LLBEN[n] is set low, TD[n] is processed normally.

Register 0x020 : FREEDM-32 Master BERT Control

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TBEN	0
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12	R/W	TBSEL[4]	0
Bit 11	R/W	TBSEL[3]	0
Bit 10	R/W	TBSEL[2]	0
Bit 9	R/W	TBSEL[1]	0
Bit 8	R/W	TBSEL[0]	0
Bit 7	R/W	RBEN	0
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4	R/W	RBSEL[4]	0
Bit 3	R/W	RBSEL[3]	0
Bit 2	R/W	RBSEL[2]	0
Bit 1	R/W	RBSEL[1]	0
Bit 0	R/W	RBSEL[0]	0

This register controls the bit error rate testing of the receive and transmit links.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RBSEL[4:0]:

The receive bit error rates testing link select bits (RBSEL[4:0]) controls the source of data on the RBD and RBCLK outputs when receive bit error rate

testing is enabled (RBEN set high). RBSEL[4:0] is a binary number that selects a receive link (RD[31:0]/RCLK[31:0]) to be the source of data for RBD and RBCLK outputs. RBSEL[4:0] is ignored when RBEN is set low.

RBEN:

The receive bit error rates testing link enable bit (RBEN) controls the receive bit error rate testing port. When RBEN is set high, RBSEL[4:0] is a binary number that selects a receive link (RD[31:0]/RCLK[31:0]) to be the source of data for RBD and RBCLK outputs. When RBEN is set low, RBD and RBCLK are held tri-stated.

TBSEL[4:0]:

The transmit bit error rates testing link select bits (TBSEL[4:0]) controls the over-writing of transmit data on TD[31:0] by data on TBD when transmit bit error rate testing is enabled (TBEN set high) and the selected link is not in line loopback (LLBEN[n] set low). TBSEL[4:0] is a binary number that selects a transmit link (TD[31:0]/TCLK[31:0]) to carry the data on TBD. The TBCLK output is a buffered version of the selected one of TCLK[31:0]. TBSEL[4:0] is ignored when TBEN is set low.

TBEN:

The transmit bit error rates testing link enable bit (TBEN) controls the transmit bit error rate testing port. When TBEN is set high and the associated link in not in line loopback (LLBEN set low), TBSEL[4:0] is a binary number that selects a transmit link data (TD[31:0]) to be the carry the data on TBD and selects a transmit link clock (TCLK[31:0]) as the source of TBCLK. When TBEN is set low, all transmit links are processed normally and TBCLK is held tri-stated.

Register 0x024 : FREEDM-32 Master Performance Monitor Control

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14	R/W	TP2EN	0
Bit 13	R/W	TABRT2EN	0
Bit 12	R/W	RP2EN	0
Bit 11	R/W	RLENE2EN	0
Bit 10	R/W	RABRT2EN	0
Bit 9	R/W	RFCSE2EN	0
Bit 8	R/W	RSPE2EN	0
Bit 7		Unused	X
Bit 6	R/W	TP1EN	0
Bit 5	R/W	TABRT1EN	0
Bit 4	R/W	RP1EN	0
Bit 3	R/W	RLENE1EN	0
Bit 2	R/W	RABRT1EN	0
Bit 1	R/W	RFCSE1EN	0
Bit 0	R/W	RSPE1EN	0

This register configures the events that are accumulated in the two configurable performance monitor counters in the PMON block.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RSPE1EN:

The receive small packet error accumulate enable bit (RSPE1EN) enables counting of minimum packet size violation events. When RSPE1EN is set high, receipt of a packet that is shorter than 32 bits (CRC-CCITT, Unspecified CRC or no CRC) or 48 bits (CRC-32) will cause the PMON Configurable Accumulator #1 register to increment. Small packet errors are ignored when RSPE1EN is set low.

RFCSE1EN:

The receive frame check sequence error accumulate enable bit (RFCSE1EN) enables counting of receive FCS error events. When RFCSE1EN is set high, a mismatch between the received FCS code and the computed CRC residue will cause the PMON Configurable Accumulator #1 register to increment. Receive frame check sequence errors are ignored when RFCSE1EN is set low.

RABRT1EN:

The receive abort accumulate enable bit (RABRT1EN) enables counting of receive HDLC abort events. When RABRT1EN is set high, receipt of an abort code (at least 7 contiguous 1's) will cause the PMON Configurable Accumulator #1 register to increment. Receive aborts are ignored when RABRT1EN is set low.

RLENE1EN:

The receive packet length error accumulate enable bit (RLENE1EN) enables counting of receive packet length error events. When RLENE1EN is set high, receipt of a packet that is longer than the programmable maximum or of a packet that in not octet aligned will cause the PMON Configurable Accumulator #1 register to increment. Receive packet length errors are ignored when RLENE1EN is set low.

RP1EN:

The receive packet enable bit (RP1EN) enables counting of receive error-free packets. When RP1EN is set high, receipt of an error-free packet will cause the PMON Configurable Accumulator #1 register to increment. Receive error-free packets are ignored when RP1EN is set low.

TABRT1EN:

The transmit abort accumulate enable bit (TABRT1EN) enables counting of transmit HDLC abort events. When TABRT1EN is set high, insertion of an abort in the outgoing stream will cause the PMON Configurable Accumulator #1 register to increment. Transmit aborts are ignored when TABRT1EN is set low.

TP1EN:

The transmit packet enable bit (TP1EN) enables counting of transmit error-free packets. When TP1EN is set high, transmission of an error-free packet will cause the PMON Configurable Accumulator #1 register to increment. Transmit error-free packets are ignored when TP1EN is set low.

RSPE2EN:

The receive small packet error accumulate enable bit (RSPE2EN) enables counting of minimum packet size violation events. When RSPE2EN is set high, receipt of a packet that is shorter than 32 bits (CRC-CCITT, Unspecified CRC or no CRC) or 48 bits (CRC-32) will cause the PMON Configurable Accumulator #2 register to increment. Small packet errors are ignored when RSPE2EN is set low.

RFCSE2EN:

The receive frame check sequence error accumulate enable bit (RFCSE2EN) enables counting of receive FCS error events. When RFCSE2EN is set high, a mismatch between the received FCS code and the computed CRC residue will cause the PMON Configurable Accumulator #2 register to increment. Receive frame check sequence errors are ignored when RFCSE2EN is set low.

RABRT2EN:

The receive abort accumulate enable bit (RABRT2EN) enables counting of receive HDLC abort events. When RABRT2EN is set high, receipt of an abort code (at least 7 contiguous 2's) will cause the PMON Configurable Accumulator #2 register to increment. Receive aborts are ignored when RABRT2EN is set low.

RLENE2EN:

The receive packet length error accumulate enable bit (RLENE2EN) enables counting of receive packet length error events. When RLENE2EN is set high, receipt of a packet that is longer than the programmable maximum or of a packet that in not octet aligned will cause the PMON Configurable Accumulator #2 register to increment. Receive packet length errors are ignored when RLENE2EN is set low.

RP2EN:

The receive packet enable bit (RP2EN) enables counting of receive error-free packets. When RP2EN is set high, receipt of an error-free packet will cause the PMON Configurable Accumulator #2 register to increment. Receive error-free packets are ignored when RP2EN is set low.

TABRT2EN:

The transmit abort accumulate enable bit (TABRT2EN) enables counting of transmit HDLC abort events. When TABRT2EN is set high, insertion of an abort in the outgoing stream will cause the PMON Configurable Accumulator #2 register to increment. Transmit aborts are ignored when TABRT2EN is set low.

TP2EN:

The transmit packet enable bit (TP2EN) enables counting of transmit error-free packets. When TP2EN is set high, transmission of an error-free packet will cause the PMON Configurable Accumulator #2 register to increment. Transmit error-free packets are ignored when TP2EN is set low.

Register 0x040 : GPIC Control

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12	R/W	RPWTH[4]	0
Bit 11	R/W	RPWTH[3]	0
Bit 10	R/W	RPWTH[2]	0
Bit 9	R/W	RPWTH[1]	0
Bit 8	R/W	RPWTH[0]	0
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3	R/W	PONS_E	0
Bit 2	R/W	SOE_E	0
Bit 1	R/W	LENDIAN	1
Bit 0	R/W	Reserved	0

This register configures the operation of the GPIC.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

Reserved:

The Reserved bit must be set low for correct operation of the FREEDM.

LENDIAN:

The Little Endian mode bit (LENDIAN) selects between Big Endian or Little Endian format when reading packet data from and writing packet data to PCI host memory. When LENDIAN is set low, Big Endian format is selected. When LENDIAN is set high, Little Endian format is selected. Descriptor references and the contents of descriptors are always transferred in Little Endian Format. Please refer below for each format's byte ordering.

Table 14 – Big Endian Format

	00	Bit 31 24	23 16	15 8	7 Bit 0
DWORD Address	04	BYTE 0	BYTE 1	BYTE 2	BYTE 3
		BYTE 4	BYTE 5	BYTE 6	BYTE 7
		•	•	•	•
		•	•	•	•
		•	•	•	•
	n-4	BYTE n-4	BYTE n-3	BYTE n-2	BYTE n-1

Table 15 – Little Endian Format

	00	Bit 31 24	23 16	15 8	7 Bit 0
DWORD Address	04	BYTE 3	BYTE 2	BYTE 1	BYTE 0
		BYTE 7	BYTE 6	BYTE 5	BYTE 4
		•	•	•	•
		•	•	•	•
		•	•	•	•
	n-4	BYTE n-1	BYTE n-2	BYTE n-3	BYTE n-4

SOE_E:

The stop on error enable (SOE_E) bit determines the action the PCI controller will take when a system or parity error occurs. When set high the PCI controller will disconnect the PCI REQB signal from the PCI bus. This prevents the GPIC from the becoming a master device on the PCI bus in event of one of the following bits in the PCI Configuration Command/Status register being set: DPR, RTABT, MABT and SERR. When the SOE_E bit is set low the PCI controller will continue to allow master transactions on the PCI bus. Setting this bit low after an error has occurred or clearing the appropriate bit the PCI Configuration Command/Status register will reactivate the PCI REQB signal and allow the GPIC to resume servicing the local

masters. In the event of a system or parity error it is recommended that the core device be reset unless the cause of the fault can be determined.

PONS_E:

The Report PERR on SERR enable (PONS_E) bit controls the source of system errors. When set high all parity errors will be signaled to the host via the SERRB output signal.

RPWTH[4:0]:

The Receive Packet Write Threshold bits (RPWTH[4:0]) controls the amount of data in the write FIFO before the GPIC begins arbitrating for the bus. The GPIC will begin requesting access to the PCI bus when the number of dwords of packet data loaded by the RMAC reaches the threshold specified by RPWTH[4:0].

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

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	BUSY	X
Bit 14	R/W	RWB	0
Bit 13		Unused	X
Bit 12	R/W	LINK[4]	0
Bit 11	R/W	LINK[3]	0
Bit 10	R/W	LINK[2]	0
Bit 9	R/W	LINK[1]	0
Bit 8	R/W	LINK[0]	0
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4	R/W	TSLOT[4]	0
Bit 3	R/W	TSLOT[3]	0
Bit 2	R/W	TSLOT[2]	0
Bit 1	R/W	TSLOT[1]	0
Bit 0	R/W	TSLOT[0]	0

This register provides the receive link and time-slot number used to access the channel provision RAM. Writing to this register triggers an indirect register access.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

TSLOT[4:0]:

The indirect time-slot number bits (TSLOT[4:0]) indicate the time-slot to be configured or interrogated in the indirect access. For a channelised T1 link, time-slots 1 to 24 are valid. For a channelised E1 link, time-slots 1 to 31 are valid. For unchannelised links, only time-slot 0 is valid.

LINK[4:0]:

The indirect link number bits (LINK[4:0]) select amongst the 32 receive links to be configured or interrogated in the indirect access.

RWB:

The indirect access control bit (RWB) selects between a configure (write) or interrogate (read) access to the channel provision RAM. The address to the channel provision RAM is constructed by concatenating the TSLOT[4:0] and LINK[4:0] bits. Writing a logic zero to RWB triggers an indirect write operation. Data to be written is taken from the PROV, the CDLBEN and the CHAN[6:0] bits of the RCAS Indirect Channel Data register. Writing a logic one to RWB triggers an indirect read operation. Addressing of the RAM is the same as in an indirect write operation. The data read can be found in the PROV, the CDLBEN and the CHAN[6:0] bits of the RCAS Indirect Channel Data register.

BUSY:

The indirect access status bit (BUSY) reports the progress of an indirect access. BUSY is set high when this register is written to trigger an indirect access, and will stay high until the access is complete. At which point, BUSY will be set low. This register should be polled to determine when data from an indirect read operation is available in the RCAS Indirect Channel Data register or to determine when a new indirect write operation may commence.

Register 0x104 : RCAS Indirect Channel Data

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9	R/W	CDLBEN	0
Bit 8	R/W	PROV	0
Bit 7		Unused	X
Bit 6	R/W	CHAN[6]	0
Bit 5	R/W	CHAN[5]	0
Bit 4	R/W	CHAN[4]	0
Bit 3	R/W	CHAN[3]	0
Bit 2	R/W	CHAN[2]	0
Bit 1	R/W	CHAN[1]	0
Bit 0	R/W	CHAN[0]	0

This register contains the data read from the channel provision RAM after an indirect read operation or the data to be inserted into the channel provision RAM in an indirect write operation.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

CHAN[6:0]:

The indirect data bits (CHAN[6:0]) report the channel number read from the channel provision RAM after an indirect read operation has completed. Channel number to be written to the channel provision RAM in an indirect write operation must be set up in this register before triggering the write. CHAN[6:0] reflects the value written until the completion of a subsequent indirect read operation.

PROV:

The indirect provision enable bit (PROV) reports the channel provision enable flag read from the channel provision RAM after an indirect read operation has completed. The provision enable flag to be written to the channel provision RAM, in an indirect write operation, must be set up in this register before triggering the write. When PROV is set high, the current receive data byte is processed as part of the channel as indicated by CHAN[6:0]. When PROV is set low, the current time-slot does not belong to any channel and the receive data byte ignored. PROV reflects the value written until the completion of a subsequent indirect read operation.

CDLBEN:

The indirect channel based diagnostic loopback enable bit (CDLBEN) reports the loopback enable flag read from channel provision RAM after an indirect read operation has complete. The loopback enable flag to be written to the channel provision RAM, in an indirect write operation, must be set up in this register before triggering the write. When CDLBEN is set high, the current receive data byte is to be over-written by data retrieved from the loopback FIFO of the channel as indicated by CHAN[6:0]. When CDLBEN is set low, the current receive data byte is processed normally. CDLBEN reflects the value written until the completion of a subsequent indirect read operation.

Register 0x108 : RCAS Framing Bit Threshold

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6	R/W	FTHRES[6]	0
Bit 5	R/W	FTHRES[5]	1
Bit 4	R/W	FTHRES[4]	1
Bit 3	R/W	FTHRES[3]	1
Bit 2	R/W	FTHRES[2]	1
Bit 1	R/W	FTHRES[1]	1
Bit 0	R/W	FTHRES[0]	1

This register contains the threshold used by the clock activity monitor to detect for framing bits/bytes.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

FTHRES[6:0]:

The framing bit threshold bits (FTHRES[6:0]) contains the threshold used by the clock activity monitor to detect for the presence of framing bits. A counter in the clock activity monitor of each receive link increments at each SYSCLK and is cleared, when the BSYNC bit of that link is set low, by each rising edge of the corresponding RCLK[n]. When the BSYNC bit of that link is set high, the counter is cleared at every fourth rising edge of the corresponding RCLK[n]. When the counter exceeds the threshold given by FTHRES[6:0], a framing bit/byte has been detected. FTHRES[6:0] should be set as a function of the SYSCLK period and the expected gapping width of RCLK[n] during data bits and during framing bits/bytes. Legal range of FTHRESH[6:0] is 'b0000001 to 'b1111110.

Note: For operation with T1 links and SYSCLK = 33 MHz, FTHRESH[6:0] should be set to ‘b0011111’. The default value of this register is inconsistent with that of the TCAS Framing Bit Threshold register 0x408.

Register 0x10C : RCAS Channel Disable

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	CHDIS	0
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6	R/W	DCHAN[6]	0
Bit 5	R/W	DCHAN[5]	0
Bit 4	R/W	DCHAN[4]	0
Bit 3	R/W	DCHAN[3]	0
Bit 2	R/W	DCHAN[2]	0
Bit 1	R/W	DCHAN[1]	0
Bit 0	R/W	DCHAN[0]	0

This register controls the disabling of one specific channel to allow orderly provisioning of timeslots associated with that channel.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

DCHAN[6:0]:

The disable channel number bits (DCHAN[6:0]) selects the channel to be disabled. When CHDIS is set high, the channel specified by DCHAN[6:0] is disabled. Data in timeslots associated with the specified channel is ignored. When CHDIS is set low, the channel specified by DCHAN[6:0] operates normally.

CHDIS:

The channel disable bit (CHDIS) controls the disabling of the channels specified by DCHAN[6:0]. When CHDIS is set high, the channel selected by DCHAN[6:0] is disabled. Data in timeslots associated with the specified channel is ignored. When CHDIS is set low, the channel specified by DCHAN[6:0] operates normally.

Register 0x180 : RCAS Link #0 Configuration

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3		Unused	X
Bit 2	R/W	BSYNC	0
Bit 1	R/W	E1	0
Bit 0	R/W	CEN	0

This register configures operational modes of receive link #0 (RD[0] / RCLK[0]).

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

CEN:

The channelise enable bit (CEN) configures receive link #0 for channelised operation. RCLK[0] is held quiescent during the T1 framing bit and during the

E1 framing bytes. The data bit on RD[0] clocked in by the first rising edge of RCLK[0] after an extended quiescent period is considered to be the most significant bit of time-slot 1. When CEN is set low, receive link #0 is unchannelised. The E1 register bit is ignored. RCLK[0] is gapped during non-data bytes. The choice between treating all data bits as a contiguous stream with arbitrary byte alignment or byte aligned to gaps in RCLK[0] is controlled by the BSYNC bit.

E1:

The E1 frame structure select bit (E1) configures receive link #0 for channelised E1 operation when CEN is set high. RCLK[0] is held quiescent during the FAS and NFAS framing bytes. The data bit on RD[0] associated with the first rising edge of RCLK[0] after an extended quiescent period is considered to be the most significant bit of time-slot 1. Link data is present at time-slots 1 to 31. When E1 is set low and CEN is set high, receive link #0 is configured for channelised T1 operation. RCLK[0] is held quiescent during the framing bit. The data bit on RD[0] associated with the first rising edge of RCLK[0] after an extended quiescent period is considered to be the most significant bit of time-slot 1. Link data is present at time-slots 1 to 24. E1 is ignored when CEN is set low.

BSYNC:

The byte synchronisation enable bit (BSYNC) controls the interpretation of gaps in RCLK[0] when link #0 is in unchannelised mode (CEN set low). When BSYNC is set high, the data bit on RD[0] clocked in by the first rising edge of RCLK[0] after an extended quiescent period is considered to be the most significant bit of a data byte. When BSYNC is set low, gaps in RCLK[0] carry no special significance. BSYNC is ignore when CEN is set high.

Register 0x184 - 0x188 : RCAS Link #1 to #2 Configuration

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3		Unused	X
Bit 2	R/W	BSYNC	0
Bit 1	R/W	E1	0
Bit 0	R/W	CEN	0

This register set configures operational modes of receive link #1 to link #2 (RD[n] / RCLK[n]; where $1 \leq n \leq 2$).

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

CEN:

The channelise enable bit (CEN) configures the corresponding receive link for channelised operation. RCLK[n] is held quiescent during the T1 framing bit and during the E1 framing bytes. The data bit on RD[n] clocked in by the first rising edge of RCLK[n] after an extended quiescent period is considered to be the most significant bit of time-slot 1. When CEN is set low, receive link #n is unchannelised. The E1 register bit is ignored. RCLK[n] is gapped during non-data bytes. The choice between treating all data bits as a contiguous stream with arbitrary byte alignment or byte aligned to gaps in RCLK[n] is controlled by the BSYNC bit.

E1:

The E1 frame structure select bit (E1) configures the corresponding receive link for channelised E1 operation when CEN is set high. RCLK[n] is held quiescent during the FAS and NFAS framing bytes. The data bit on RD[n] associated with the first rising edge of RCLK[n] after an extended quiescent period is considered to be the most significant bit of time-slot 1. Link data is present at time-slots 1 to 31. When E1 is set low and CEN is set high, the corresponding receive link is configured for channelised T1 operation. RCLK[n] is held quiescent during the framing bit. The data bit on RD[n] associated with the first rising edge of RCLK[n] after an extended quiescent period is considered to be the most significant bit of time-slot 1. Link data is present at time-slots 1 to 24. E1 is ignored when CEN is set low.

BSYNC:

The byte synchronisation enable bit (BSYNC) controls the interpretation of gaps in RCLK[n] when the corresponding link is in unchannelised mode (CEN set low). When BSYNC is set high, the data bit on RD[n] clocked in by the first rising edge of RCLK[n] after an extended quiescent period is considered to be the most significant bit of a data byte. When BSYNC is set low, gaps in RCLK[n] carry no special significance. BSYNC is ignore when CEN is set high.

Register 0x18C : RCAS Link #3 Configuration

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3		Unused	X
Bit 2		Unused	X
Bit 1	R/W	E1	0
Bit 0	R/W	CEN	0

This register set configures operational modes of receive link #3 (RD[3] / RCLK[3]).

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

CEN:

The channelise enable bit (CEN) configures receive link #3 for channelised operation. RCLK[3] is held quiescent during the T1 framing bit and the E1 framing bytes. The data bit on RD[3] clocked in by the first rising edge of RCLK[3] after an extended quiescent period is considered to be the most significant bit of time-slot 1. When CEN is set low, the corresponding receive link is unchannelised. The E1 register bit is ignored. RCLK[3] is gapped during non-data bytes. All data bits are treated as a contiguous stream with arbitrary byte alignment.

E1:

The E1 frame structure select bit (E1) configures receive link #3 for channelised E1 operation when CEN is set high. RCLK[3] is held quiescent during the FAS and NFAS framing bytes. The data bit on RD[3] clocked in by the first rising edge of RCLK[3] after an extended quiescent period is considered to be the most significant bit of time-slot 1. Link data is present at time-slots 1 to 31. When E1 is set low and CEN is set high, receive link #3 is configured for channelised T1 operation. RCLK[3] is held quiescent during the framing bit. The data bit on RD[3] clocked in by the first rising edge of RCLK[3] after an extended quiescent period is considered to be the most significant bit of time-slot 1. Link data is present at time-slots 1 to 24. E1 is ignored when CEN is set low.

Register 0x190-0x1FC : RCAS Link #4 to Link #31 Configuration

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3		Unused	X
Bit 2		Unused	X
Bit 1	R/W	E1	0
Bit 0	R/W	CEN	0

This register set configures operational modes of receive link #4 to link # 31 (RD[n] / RCLK[n]; where $4 \leq n \leq 31$).

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

CEN:

The channelise enable bit (CEN) configures the corresponding receive link for channelised operation. RCLK[n] is held quiescent during the T1 framing bit and the E1 framing bytes. The data bit on RD[n] clocked in by the first rising edge of RCLK[n] after an extended quiescent period is considered to be the most significant bit of time-slot 1. When CEN is set low, the corresponding receive link is unchannelised. The E1 register bit is ignored. RCLK[n] is gapped during non-data bytes. All data bits are treated as a contiguous stream with arbitrary byte alignment.

E1:

The E1 frame structure select bit (E1) configures the corresponding receive link for channelised E1 operation when CEN is set high. RCLK[n] is held quiescent during the FAS and NFAS framing bytes. The data bit on RD[n] clocked in by the first rising edge of RCLK[n] after an extended quiescent period is considered to be the most significant bit of time-slot 1. Link data is present at time-slots 1 to 31. When E1 is set low and CEN is set high, the corresponding receive link is configured for channelised T1 operation. RCLK[n] is held quiescent during the framing bit. The data bit on RD[n] clocked in by the first rising edge of RCLK[n] after an extended quiescent period is considered to be the most significant bit of time-slot 1. Link data is present at time-slots 1 to 24. E1 is ignored when CEN is set low.

Register 0x200 : RHDL Indirect Channel Select

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	BUSY	X
Bit 14	R/W	CRWB	0
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6	R/W	CHAN[6]	0
Bit 5	R/W	CHAN[5]	0
Bit 4	R/W	CHAN[4]	0
Bit 3	R/W	CHAN[3]	0
Bit 2	R/W	CHAN[2]	0
Bit 1	R/W	CHAN[1]	0
Bit 0	R/W	CHAN[0]	0

This register provides the channel number used to access the receive channel provision RAM. Writing to this register triggers an indirect channel register access.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

CHAN[6:0]:

The indirect channel number bits (CHAN[6:0]) indicate the receive channel to be configured or interrogated in the indirect access.

CRWB:

The channel indirect access control bit (CRWB) selects between a configure (write) or interrogate (read) access to the receive channel provision RAM. Writing a logic zero to CRWB triggers an indirect write operation. Data to be written is taken from the Indirect Channel Data registers. Writing a logic one to CRWB triggers an indirect read operation. The data read can be found in the Indirect Channel Data registers.

BUSY:

The indirect access status bit (BUSY) reports the progress of an indirect access. BUSY is set high when this register is written to trigger an indirect access, and will stay high until the access is complete. At which point, BUSY will be set low. This register should be polled to determine when data from an indirect read operation is available in the RHDL Indirect Channel Data #1 and #2 registers or to determine when a new indirect write operation may commence.

Register 0x204 : RHDL Indirect Channel Data Register #1

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	PROV	0
Bit 14	R/W	CRC[1]	0
Bit 13	R/W	CRC[0]	0
Bit 12	R/W	STRIP	0
Bit 11	R/W	DELIN	0
Bit 10	R	TAVAIL	X
Bit 9		Unused	X
Bit 8	W	FPTR[8]	X
Bit 7	W	FPTR[7]	X
Bit 6	W	FPTR[6]	X
Bit 5	W	FPTR[5]	X
Bit 4	W	FPTR[4]	X
Bit 3	W	FPTR[3]	X
Bit 2	W	FPTR[2]	X
Bit 1	W	FPTR[1]	X
Bit 0	W	FPTR[0]	X

This register contains data read from the channel provision RAM after an indirect read operation or data to be inserted into the channel provision RAM in an indirect write operation.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

FPTR[8:0]:

The indirect FIFO block pointer (FPTR[8:0]) identifies one of the blocks of the circular linked list in the partial packet buffer used in the logical FIFO of the current channel. The FIFO pointer to be written to the channel provision RAM, in an indirect write operation, must be set up in this register before triggering the write. The FIFO pointer value can be any one of the blocks provisioned to form the circular buffer.

TAVAIL:

The indirect transaction available bit (TAVAIL) reports the fill level of the partial packet buffer used in the logical FIFO of the current channel. TAVAIL is set high when the FIFO of the current channel contains sufficient data, as controlled by XFER[2:0], to request a DMA transfer to the host memory. TAVAIL is set low when the amount of receive data is too small to require a transfer to host memory. TAVAIL is update by an indirect channel read operation.

DELIN:

The indirect delineate enable bit (DELIN) configures the HDLC processor to perform flag sequence delineation and bit de-stuffing on the incoming data stream. The delineate enable bit to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. When DELIN is set high, flag sequence delineation and bit de-stuffing is performed on the incoming data stream. When DELIN is set low, the HDLC processor does not perform any processing (flag sequence delineation, bit de-stuffing nor CRC verification) on the incoming stream. DELIN reflects the value written until the completion of a subsequent indirect channel read operation.

STRIP:

The indirect frame check sequence discard bit (STRIP) configures the HDLC processor to remove the CRC from the incoming frame when writing the data to the channel FIFO. The FCS discard bit to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. When STRIP is set high and CRC[1:0] is not equal to "00", the received CRC value is not written to the FIFO. When STRIP is set low, the received CRC value is written to the FIFO. The bytes in buffer field of the RPD correctly reflect the presence/absence of CRC bytes in the buffer. The value of STRIP is ignored when DELIN is low. STRIP reflects the value written until the completion of a subsequent indirect channel read operation.

CRC[1:0]:

The CRC algorithm bits (CRC[1:0]) configures the HDLC processor to perform CRC verification on the incoming data stream. The value of CRC[1:0] to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. CRC[1:0] is ignored when DELIN is low. CRC[1:0] reflects the value written until the completion of a subsequent indirect channel read operation.

Table 16 – CRC[1:0] Settings

CRC[1]	CRC[0]	Operation
0	0	No Verification
0	1	CRC-CCITT
1	0	CRC-32
1	1	Reserved

PROV:

The indirect provision enable bit (PROV) reports the channel provision enable flag read from the channel provision RAM after an indirect channel read operation has completed. The provision enable flag to be written to the channel provision RAM, in an indirect write operation, must be set up in this register before triggering the write. When PROV is set high, the HDLC processor will process data on the channel specified by CHAN[6:0]. When PROV is set low, the HDLC processor will ignore data on the channel specified by CHAN[6:0]. PROV reflects the value written until the completion of a subsequent indirect channel read operation.

Register 0x208 : RHDL Indirect Channel Data Register #2

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	7BIT	0
Bit 14	R/W	PRIORITY	0
Bit 13	R/W	INVERT	0
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9	R/W	OFFSET[1]	0
Bit 8	R/W	OFFSET[0]	0
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused[	X
Bit 3		Unused	X
Bit 2	R/W	XFER[2]	X
Bit 1	R/W	XFER[1]	0
Bit 0	R/W	XFER[0]	0

This register contains data read from the channel provision RAM after an indirect read operation or data to be inserted into the channel provision RAM in an indirect write operation.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

XFER[2:0]:

The indirect channel transfer size (XFER[2:0]) configures the amount of data transferred in each transaction. The channel transfer size to be written to the channel provision RAM, in an indirect write operation, must be set up in this register before triggering the write. When the channel FIFO depth reaches the depth specified by XFER[2:0] or when an end-of-packet exists in the FIFO, a request will be made to the RMAC to initiate a PCI write access to transfer the data to the PCI host. Channel transfer size is measured in 16 byte blocks. The amount of data transferred and the depth threshold are specified by given setting is:

XFER[2:0] + 1 blocks = 16 * (XFER[2:0] + 1) bytes

XFER[2:0] should be set such that the number of blocks transferred is at least two fewer than the total allocated to the associated channel. XFER[2:0] reflects the value written until the completion of a subsequent indirect channel read operation.

OFFSET[1:0]:

The packet byte offset (OFFSET[1:0]) configures the partial packet processor to insert invalid bytes at the beginning of a packet stored in the channel FIFO. The value of OFFSET[1:0] to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. The number of bytes inserted before the beginning of a HDLC packet is defined by the binary value of OFFSET[1:0]. OFFSET[1:0] reflects the value written until the completion of a subsequent indirect channel read operation.

INVERT:

The HDLC data inversion bit (INVERT) configures the HDLC processor to logically invert the incoming HDLC stream from the RCAS before processing it. The value of INVERT to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. When INVERT is set to one, the HDLC stream is logically inverted before processing. When INVERT is set to zero, the HDLC stream is not inverted before processing. INVERT reflects the value written until the completion of a subsequent indirect channel read operation.

PRIORITY:

The channel FIFO priority bit (PRIORITY) informs the partial packet processor that the channel has precedence over other channels when being serviced by the RMAC block for transfer to the PCI host. The value of PRIORITY to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write.

Channel FIFOs with PRIORITY set to one are serviced by the RMAC before channel FIFOs with PRIORITY set to zero. Channels with an HDLC data rate to FIFO size ratio that is significantly higher than other channels should have PRIORITY set to one. PRIORITY reflects the value written until the completion of a subsequent indirect channel read operation.

7BIT:

The 7BIT enable bit (7BIT) configures the HDLC processor to ignore the least significant bit of each octet in the corresponding link RD[n]. The value of 7BIT to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. When 7BIT is set high, the least significant bit (last bit of each octet received), is ignored. When 7BIT is set low, the entire receive data stream is processed. 7BIT reflects the value written until the completion of a subsequent indirect channel read operation.

Register 0x210 : RHDL Indirect Block Select

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	BUSY	X
Bit 14	R/W	BRWB	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8	R/W	BLOCK[8]	X
Bit 7	R/W	BLOCK[7]	X
Bit 6	R/W	BLOCK[6]	X
Bit 5	R/W	BLOCK[5]	X
Bit 4	R/W	BLOCK[4]	X
Bit 3	R/W	BLOCK[3]	X
Bit 2	R/W	BLOCK[2]	X
Bit 1	R/W	BLOCK[1]	X
Bit 0	R/W	BLOCK[0]	X

This register provides the block number used to access the block pointer RAM. Writing to this register triggers an indirect block register access.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

BLOCK[8:0]:

The indirect block number (BLOCK[8:0]) indicate the block to be configured or interrogated in the indirect access.

BRWB:

The block indirect access control bit (BRWB) selects between a configure (write) or interrogate (read) access to the block pointer RAM. Writing a logic zero to BRWB triggers an indirect block write operation. Data to be written is taken from the Indirect Block Data register. Writing a logic one to BRWB triggers an indirect block read operation. The data read can be found in the Indirect Block Data register.

BUSY:

The indirect access status bit (BUSY) reports the progress of an indirect access. BUSY is set high when this register is written to trigger an indirect access, and will stay high until the access is complete. At which point, BUSY will be set low. This register should be polled to determine when data from an indirect read operation is available in the RHDL Indirect Block Data register or to determine when a new indirect write operation may commence.

Register 0x214 : RHDL Indirect Block Data

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8	R/W	BPTR[8]	0
Bit 7	R/W	BPTR[7]	0
Bit 6	R/W	BPTR[6]	0
Bit 5	R/W	BPTR[5]	0
Bit 4	R/W	BPTR[4]	0
Bit 3	R/W	BPTR[3]	0
Bit 2	R/W	BPTR[2]	0
Bit 1	R/W	BPTR[1]	0
Bit 0	R/W	BPTR[0]	0

This register contains data read from the block pointer RAM after an indirect block read operation or data to be inserted into the block pointer RAM in an indirect block write operation.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

BPTR[8:0]:

The indirect block pointer (BPTR[8:0]) configures the block pointer of the block specified by the Indirect Block Select register. The block pointer to be written to the block pointer RAM, in an indirect write operation, must be set up in this register before triggering the write. The block pointer value is the block number of the next block in the linked list. A circular list of blocks must be formed in order to use the block list as a receive channel FIFO buffer. BPTR[8:0] reflects the value written until the completion of a subsequent indirect block read operation.

When provisioning a channel FIFO, all blocks pointers must be re-written to properly initialize the FIFO.

Register 0x220 : RHDL Configuration

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9	R/W	LENCHK	0
Bit 8	R/W	TSTD	0
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3		Unused	X
Bit 2	R/W	Reserved[2]	1
Bit 1	R/W	Reserved[1]	1
Bit 0	R/W	Reserved[0]	1

This register configures all provisioned receive channels.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

Reserved[2:0]:

The reserved bits (Reserved[2:0]) must be set to all ones for correct operation of the FREEDM-32 device.

TSTD:

The telecom standard bit (TSTD) controls the bit ordering of the HDLC data transferred to the PCI host. When TSTD is set low, the least significant bit of the each byte on the PCI bus (AD[0], AD[8], AD[16] and AD[24]) is the first HDLC bit received and the most significant bit of each byte (AD[7], AD[15], AD[23] and AD[31]) is the last HDLC bit received (datacom standard). When TSTD is set high, AD[0], AD[8], AD[16] and AD[24] are the last HDLC bit received and AD[7], AD[15], AD[23] and AD[31] are the first HDLC bit received (telecom standard).

LENCHK:

The packet length error check bit (LENCHK) controls the checking of receive packets that are longer than the maximum programmed length. When LENCHK is set high, receive packets are aborted and the remainder of the frame discarded when the packet exceeds the maximum packet length given by MAX[15:0]. When LENCHK is set low, receive packets are not checked for maximum size and MAX[15:0] must be set to 'hFFFF.

Register 0x224 : RHDL Maximum Packet Length

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	MAX[15]	1
Bit 14	R/W	MAX[14]	1
Bit 13	R/W	MAX[13]	1
Bit 12	R/W	MAX[12]	1
Bit 11	R/W	MAX[11]	1
Bit 10	R/W	MAX[10]	1
Bit 9	R/W	MAX[9]	1
Bit 8	R/W	MAX[8]	1
Bit 7	R/W	MAX[7]	1
Bit 6	R/W	MAX[6]	1
Bit 5	R/W	MAX[5]	1
Bit 4	R/W	MAX[4]	1
Bit 3	R/W	MAX[3]	1
Bit 2	R/W	MAX[2]	1
Bit 1	R/W	MAX[1]	1
Bit 0	R/W	MAX[0]	1

This register configures the maximum legal HDLC packet byte length.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

MAX[15:0]:

The maximum HDLC packet length (MAX[15:0]) configures the FREEDM-32 to reject HDLC packets longer than a maximum size when LENCHK is set

high. Receive packets with total length, including address, control, information and FCS fields, greater than MAX[15:0] bytes are aborted. When LENCHK is set low, aborts are not generated regardless of packet length and MAX[15:0] must be set to 'hFFFF.

Register 0x280 : RMAC Control

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11	R/W	RPQ_SFN[1]	0
Bit 10	R/W	RPQ_SFN[0]	0
Bit 9	R/W	RPQ_LFN[1]	0
Bit 8	R/W	RPQ_LFN[0]	0
Bit 7	R/W	RPQ_RDYN[2]	0
Bit 6	R/W	RPQ_RDYN[1]	0
Bit 5	R/W	RPQ_RDYN[0]	0
Bit 4	R/W	RAWMAX[1]	1
Bit 3	R/W	RAWMAX[0]	1
Bit 2	R/W	SCACHE	1
Bit 1	R/W	LCACHE	1
Bit 0	R/W	ENABLE	0

This register configures the RMAC block.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

ENABLE:

The ENABLE bit determines whether or not the RMAC accepts data from the RHDL block and sends it to host memory. When set to 1, these tasks are enabled. When set to 0, they are disabled.

LCACHE:

The large buffer cache enable bit (LCACHE) enables caching of Large Buffer RPDRs. When LCACHE is set high, RPDRs are fetched from the RPDR Large Buffer Free Queue in groups of up to six. When LCACHE is set low, RPDRs are fetched one at a time.

SCACHE:

The small buffer cache enable bit (SCACHE) enables caching of Small Buffer RPDRs. When SCACHE is set high, RPDRs are fetched from the RPDR Small Buffer Free Queue in groups of up to six. When SCACHE is set low, RPDRs are fetched one at a time.

RAWMAX[1:0]:

The RAWMAX[1:0] field determines how 'raw' (i.e. non packet delimited) data is written to host memory. Raw data is written to buffers in host memory in the same manner as packet delimited data. Whenever RAWMAX[1:0] + 1 buffers have been filled, the resulting buffer chain is placed in the ready queue.

RPQ_RDYN[2:0]:

The RPQ_RDYN[2:0] field sets the number of receive packet descriptor references (RPDRs) that must be placed onto the RPDR ready queue before the RPDR ready interrupt (RPQRDYI) is asserted, as follows:

Table 17 – RPQ_RDYN[2:0] settings

RPQ_RDYN[2:0]	No of RPDRs
000	1
001	4
010	6
011	8
100	16
101	32
110	Reserved

RPQ_RDYN[2:0]	No of RPDRs
111	Reserved

RPQ_LFN[1:0]:

The RPQ_LFN[1:0] field sets the number of times that a block of RPDRs are read from the Large Buffer Free Queue to the RMACs internal cache before the RPDR Large Buffer Free Queue interrupt (RPQLFI) is asserted, as follows:

Table 18 – RPQ_LFN[1:0] Settings

RPQ_LFN[1:0]	No of Reads
00	1
01	4
10	8
11	Reserved

RPQ_SFN[1:0]:

The RPQ_SFN[1:0] field sets the number of times that a block of RPDRs are read from the Small Buffer Free Queue to the RMACs internal cache before the RPDR Small Buffer Free Queue interrupt (RPQSFI) is asserted, as follows:

Table 19 – RPQ_SFN[1:0] Settings

RPQ_SFN[1:0]	No of Reads
00	1
01	4
10	8
11	Reserved

Register 0x284 : RMAC Indirect Channel Provisioning

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	BUSY	X
Bit 14	R/W	RWB	0
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7	R/W	PROV	1
Bit 6	R/W	CHAN[6]	0
Bit 5	R/W	CHAN[5]	0
Bit 4	R/W	CHAN[4]	0
Bit 3	R/W	CHAN[3]	0
Bit 2	R/W	CHAN[2]	0
Bit 1	R/W	CHAN[1]	0
Bit 0	R/W	CHAN[0]	0

The Channel Provisioning Register is used to temporarily unprovision channels, and also to query the provision status of channels. Channel is permanently provisioned and can only be unprovisioned transiently. When a channel is unprovisioned, a partially received packet, if any, will be flushed and marked as unprovisioned in the RPDRR queue status field. The channel then returns to being provisioned automatically.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not

implemented. However, when all four byte enables are negated, no access is made to this register.

CHAN[6:0]:

The indirect data bits (CHAN[6:0]) report the channel number read from the RMAC internal memory after an indirect read operation has completed. Channel number to be written to the RMAC internal memory in an indirect write operation must be set up in this register before triggering the write. CHAN[6:0] reflects the value written until the completion of a subsequent indirect read operation.

PROV:

The indirect provision enable bit (PROV) reports the channel provision enable flag read from the RMAC internal memory after an indirect read operation has completed. The provision enable flag to be written to the RMAC internal memory, in an indirect write operation, must be set up in this register before triggering the write. When PROV is set high, the channel as indicated by CHAN[6:0] is provisioned. When PROV is set low, the channel indicated by CHAN[6:0] is unprovisioned temporarily. Any partially received packets are flushed and the status in the RPDRR queue is marked unprovisioned. The channel then returns to being provisioned and PROV will report a logic high at the next indirect read operation. PROV reflects the value written until the completion of a subsequent indirect read operation.

RWB:

The Read/Write Bar (RWB) bit selects between a provisioning/unprovisioning operation (write) or a query operation (read). Writing a logic 0 to RWB triggers the provisioning or unprovisioning of a channel as specified by CHAN[6:0] and PROV. Writing a logic 1 to RWB triggers a query of the channel specified by CHAN[6:0].

BUSY:

The indirect access status bit (BUSY) reports the progress of an indirect access. BUSY is set high when this register is written to trigger an indirect access, and will stay high until the access is complete. At which point, BUSY will be set low. This register should be polled to determine when data from an indirect read operation is available or to determine when a new indirect write operation may commence.

Register 0x288 : RMAC Packet Descriptor Table Base LSW

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDTB[15]	0
Bit 14	R/W	RPDTB[14]	0
Bit 13	R/W	RPDTB[13]	0
Bit 12	R/W	RPDTB[12]	0
Bit 11	R/W	RPDTB[11]	0
Bit 10	R/W	RPDTB[10]	0
Bit 9	R/W	RPDTB[9]	0
Bit 8	R/W	RPDTB[8]	0
Bit 7	R/W	RPDTB[7]	0
Bit 6	R/W	RPDTB[6]	0
Bit 5	R/W	RPDTB[5]	0
Bit 4	R/W	RPDTB[4]	0
Bit 3	R/W	RPDTB[3]	0
Bit 2	R/W	RPDTB[2]	0
Bit 1	R/W	RPDTB[1]	0
Bit 0	R/W	RPDTB[0]	0

This register provides the less significant word of the Receive Descriptor Table Base address. The contents of this register is held in a holding register until a write access to the companion RMAC Receive Descriptor Table Base MSW register.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

Register 0x28C : RMAC Packet Descriptor Table Base MSW

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDTB[31]	0
Bit 14	R/W	RPDTB[30]	0
Bit 13	R/W	RPDTB[29]	0
Bit 12	R/W	RPDTB[28]	0
Bit 11	R/W	RPDTB[27]	0
Bit 10	R/W	RPDTB[26]	0
Bit 9	R/W	RPDTB[25]	0
Bit 8	R/W	RPDTB[24]	0
Bit 7	R/W	RPDTB[23]	0
Bit 6	R/W	RPDTB[22]	0
Bit 5	R/W	RPDTB[21]	0
Bit 4	R/W	RPDTB[20]	0
Bit 3	R/W	RPDTB[19]	0
Bit 2	R/W	RPDTB[18]	0
Bit 1	R/W	RPDTB[17]	0
Bit 0	R/W	RPDTB[16]	0

This register provides the more significant word of the Receive Descriptor Table Base address. The contents of the companion RMAC Receive Descriptor Table Base LSW register is held in a holding register until a write access to this register, at which point, the base address of the receive packet descriptor table is updated.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDTB[31:0]:

The receive packet descriptor table base bits (RPDTB[31:0]) provides the base address of the Receive Packet Descriptor Table in PCI host memory. This register is initialised by the host. To calculate the physical address of a RPD, the 14 bit RPD offset must be added to bits 31 to 4 of the Receive Packet Descriptor Table Base (RPDTB[31:4]).

The table must be on a 16 byte boundary and thus the least significant four bits must be written to logic zero.

Register 0x290 : RMAC Queue Base LSW

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RQB[15]	0
Bit 14	R/W	RQB[14]	0
Bit 13	R/W	RQB[13]	0
Bit 12	R/W	RQB[12]	0
Bit 11	R/W	RQB[11]	0
Bit 10	R/W	RQB[10]	0
Bit 9	R/W	RQB[9]	0
Bit 8	R/W	RQB[8]	0
Bit 7	R/W	RQB[7]	0
Bit 6	R/W	RQB[6]	0
Bit 5	R/W	RQB[5]	0
Bit 4	R/W	RQB[4]	0
Bit 3	R/W	RQB[3]	0
Bit 2	R/W	RQB[2]	0
Bit 1	R/W	RQB[1]	0
Bit 0	R/W	RQB[0]	0

This register provides the less significant word of the Receive Queue Base address. The contents of this register is held in a holding register until a write access to the companion RMAC Receive Queue Base MSW register.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

Register 0x294 : RMAC Queue Base MSW

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RQB[31]	0
Bit 14	R/W	RQB[30]	0
Bit 13	R/W	RQB[29]	0
Bit 12	R/W	RQB[28]	0
Bit 11	R/W	RQB[27]	0
Bit 10	R/W	RQB[26]	0
Bit 9	R/W	RQB[25]	0
Bit 8	R/W	RQB[24]	0
Bit 7	R/W	RQB[23]	0
Bit 6	R/W	RQB[22]	0
Bit 5	R/W	RQB[21]	0
Bit 4	R/W	RQB[20]	0
Bit 3	R/W	RQB[19]	0
Bit 2	R/W	RQB[18]	0
Bit 1	R/W	RQB[17]	0
Bit 0	R/W	RQB[16]	0

This register provides the more significant word of the Receive Queue Base address. The contents of the companion RMAC Receive Queue Base LSW register is held in a holding register until a write access to this register, at which point, the base address of the receive queue is updated.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RQB[31:0]:

The receive queue base bits (RQB[31:0]) provides the base address of the Large Buffer RPDR Free, Small Buffer RPDR Free and RPDR Ready queues in PCI host memory. This register is initialised by the host. To calculate the physical address of a particular receive queue element, the RQB bits are added with the appropriate queue start, end, read or write index registers to form the physical address.

The base address must be dword aligned and thus the least significant two bits must be written to logic zero.

Register 0x298 : RMAC Packet Descriptor Reference Large Buffer Free Queue Start

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDRLFQS[15]	0
Bit 14	R/W	RPDRLFQS[14]	0
Bit 13	R/W	RPDRLFQS[13]	0
Bit 12	R/W	RPDRLFQS[12]	0
Bit 11	R/W	RPDRLFQS[11]	0
Bit 10	R/W	RPDRLFQS[10]	0
Bit 9	R/W	RPDRLFQS[9]	0
Bit 8	R/W	RPDRLFQS[8]	0
Bit 7	R/W	RPDRLFQS[7]	0
Bit 6	R/W	RPDRLFQS[6]	0
Bit 5	R/W	RPDRLFQS[5]	0
Bit 4	R/W	RPDRLFQS[4]	0
Bit 3	R/W	RPDRLFQS[3]	0
Bit 2	R/W	RPDRLFQS[2]	0
Bit 1	R/W	RPDRLFQS[1]	0
Bit 0	R/W	RPDRLFQS[0]	0

This register provides the Packet Descriptor Reference Large Buffer Free Queue start address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDRLFQS[15:0]:

The receive packet descriptor reference (RPDR) large buffer free queue start bits (RPDRLFQS[15:0]) define bits 17 to 2 of the Receive Packet Descriptor Reference Large Buffer Free Queue start address. This register is initialised by the host. The physical start address of the RPDRLF queue is the sum of RPDRLFQS[15:0] left shifted by 2 bits with the RQB[31:0] bits in the RMAC Receive Queue Base register.

Register 0x29C : RMAC Packet Descriptor Reference Large Buffer Free Queue Write

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDRLFQW[15]	0
Bit 14	R/W	RPDRLFQW[14]	0
Bit 13	R/W	RPDRLFQW[13]	0
Bit 12	R/W	RPDRLFQW[12]	0
Bit 11	R/W	RPDRLFQW[11]	0
Bit 10	R/W	RPDRLFQW[10]	0
Bit 9	R/W	RPDRLFQW[9]	0
Bit 8	R/W	RPDRLFQW[8]	0
Bit 7	R/W	RPDRLFQW[7]	0
Bit 6	R/W	RPDRLFQW[6]	0
Bit 5	R/W	RPDRLFQW[5]	0
Bit 4	R/W	RPDRLFQW[4]	0
Bit 3	R/W	RPDRLFQW[3]	0
Bit 2	R/W	RPDRLFQW[2]	0
Bit 1	R/W	RPDRLFQW[1]	0
Bit 0	R/W	RPDRLFQW[0]	0

This register provides the Packet Descriptor Reference Large Buffer Free Queue write address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDRLFQW[15:0]:

The receive packet descriptor reference (RPDR) large buffer free queue write bits (RPDRLFQW[15:0]) define bits 17 to 2 of the Receive Packet Descriptor Reference Large Buffer Free Queue write pointer. This register is initialised by the host. The physical write address in the RPDRLF queue is the sum of RPDRLFQW[15:0] left shifted by 2 bits with the RQB[31:0] bits in the RMAC Receive Queue Base register.

Register 0x2A0 : RMAC Packet Descriptor Reference Large Buffer Free Queue Read

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDRLFQR[15]	0
Bit 14	R/W	RPDRLFQR[14]	0
Bit 13	R/W	RPDRLFQR[13]	0
Bit 12	R/W	RPDRLFQR[12]	0
Bit 11	R/W	RPDRLFQR[11]	0
Bit 10	R/W	RPDRLFQR[10]	0
Bit 9	R/W	RPDRLFQR[9]	0
Bit 8	R/W	RPDRLFQR[8]	0
Bit 7	R/W	RPDRLFQR[7]	0
Bit 6	R/W	RPDRLFQR[6]	0
Bit 5	R/W	RPDRLFQR[5]	0
Bit 4	R/W	RPDRLFQR[4]	0
Bit 3	R/W	RPDRLFQR[3]	0
Bit 2	R/W	RPDRLFQR[2]	0
Bit 1	R/W	RPDRLFQR[1]	0
Bit 0	R/W	RPDRLFQR[0]	0

This register provides the Packet Descriptor Reference Large Buffer Free Queue read address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDRLFQR[15:0]:

The receive packet descriptor reference (RPDR) large buffer free queue read bits (RPDRLFQR[15:0]) define bits 17 to 2 of the Receive Packet Descriptor Reference Large Buffer Free Queue read pointer. This register is initialised by the host. The physical read address in the RPDRLF queue is the sum of RPDRLFQR[15:0] left shifted by 2 bits with the RQB[31:0] bits in the RMAC Receive Queue Base register.

Register 0x2A4 : RMAC Packet Descriptor Reference Large Buffer Free Queue End

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDRLFQE[15]	0
Bit 14	R/W	RPDRLFQE[14]	0
Bit 13	R/W	RPDRLFQE[13]	0
Bit 12	R/W	RPDRLFQE[12]	0
Bit 11	R/W	RPDRLFQE[11]	0
Bit 10	R/W	RPDRLFQE[10]	0
Bit 9	R/W	RPDRLFQE[9]	0
Bit 8	R/W	RPDRLFQE[8]	0
Bit 7	R/W	RPDRLFQE[7]	0
Bit 6	R/W	RPDRLFQE[6]	0
Bit 5	R/W	RPDRLFQE[5]	0
Bit 4	R/W	RPDRLFQE[4]	0
Bit 3	R/W	RPDRLFQE[3]	0
Bit 2	R/W	RPDRLFQE[2]	0
Bit 1	R/W	RPDRLFQE[1]	0
Bit 0	R/W	RPDRLFQE[0]	0

This register provides the Packet Descriptor Reference Large Buffer Free Queue end address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDRLFQE[15:0]:

The receive packet descriptor reference (RPDR) large buffer free queue end bits (RPDRLFQE[15:0]) define bits 17 to 2 of the Receive Packet Descriptor Reference Large Buffer Free Queue end address. This register is initialised by the host. The physical end address in the RPDRLF queue is the sum of RPDRLFQE[15:0] left shifted by 2 bits with the RQB[31:0] bits in the RMAC Receive Queue Base register.

Register 0x2A8 : RMAC Packet Descriptor Reference Small Buffer Free Queue Start

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDRSFQS[15]	0
Bit 14	R/W	RPDRSFQS[14]	0
Bit 13	R/W	RPDRSFQS[13]	0
Bit 12	R/W	RPDRSFQS[12]	0
Bit 11	R/W	RPDRSFQS[11]	0
Bit 10	R/W	RPDRSFQS[10]	0
Bit 9	R/W	RPDRSFQS[9]	0
Bit 8	R/W	RPDRSFQS[8]	0
Bit 7	R/W	RPDRSFQS[7]	0
Bit 6	R/W	RPDRSFQS[6]	0
Bit 5	R/W	RPDRSFQS[5]	0
Bit 4	R/W	RPDRSFQS[4]	0
Bit 3	R/W	RPDRSFQS[3]	0
Bit 2	R/W	RPDRSFQS[2]	0
Bit 1	R/W	RPDRSFQS[1]	0
Bit 0	R/W	RPDRSFQS[0]	0

This register provides the Packet Descriptor Reference Small Buffer Free Queue start address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDRSFQS[15:0]:

The receive packet descriptor reference (RPDR) small buffer free queue start bits (RPDRSFQS[15:0]) define bits 17 to 2 of the Receive Packet Descriptor Reference Small Buffer Free Queue start address. This register is initialised by the host. The physical start address of the RPDRSF queue is the sum of RPDRSFQS[15:0] left shifted by 2 bits with the RQB[31:0] bits in the RMAC Receive Queue Base register.

Register 0x2AC : RMAC Packet Descriptor Reference Small Buffer Free Queue Write

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDRSFQW[15]	0
Bit 14	R/W	RPDRSFQW[14]	0
Bit 13	R/W	RPDRSFQW[13]	0
Bit 12	R/W	RPDRSFQW[12]	0
Bit 11	R/W	RPDRSFQW[11]	0
Bit 10	R/W	RPDRSFQW[10]	0
Bit 9	R/W	RPDRSFQW[9]	0
Bit 8	R/W	RPDRSFQW[8]	0
Bit 7	R/W	RPDRSFQW[7]	0
Bit 6	R/W	RPDRSFQW[6]	0
Bit 5	R/W	RPDRSFQW[5]	0
Bit 4	R/W	RPDRSFQW[4]	0
Bit 3	R/W	RPDRSFQW[3]	0
Bit 2	R/W	RPDRSFQW[2]	0
Bit 1	R/W	RPDRSFQW[1]	0
Bit 0	R/W	RPDRSFQW[0]	0

This register provides the Packet Descriptor Reference Small Buffer Free Queue write address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDRSFQW[15:0]:

The receive packet descriptor reference (RPDR) small buffer free queue write bits (RPDRSFQW[15:0]) define bits 17 to 2 of the Receive Packet Descriptor Reference Small Buffer Free Queue write pointer. This register is initialised by the host. The physical write address in the RPDRSF queue is the sum of RPDRSFQW[15:0] left shifted by 2 bits with the RQB[31:0] bits in the RMAC Receive Queue Base register.

Register 0x2B0 : RMAC Packet Descriptor Reference Small Buffer Free Queue Read

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDRSFQR[15]	0
Bit 14	R/W	RPDRSFQR[14]	0
Bit 13	R/W	RPDRSFQR[13]	0
Bit 12	R/W	RPDRSFQR[12]	0
Bit 11	R/W	RPDRSFQR[11]	0
Bit 10	R/W	RPDRSFQR[10]	0
Bit 9	R/W	RPDRSFQR[9]	0
Bit 8	R/W	RPDRSFQR[8]	0
Bit 7	R/W	RPDRSFQR[7]	0
Bit 6	R/W	RPDRSFQR[6]	0
Bit 5	R/W	RPDRSFQR[5]	0
Bit 4	R/W	RPDRSFQR[4]	0
Bit 3	R/W	RPDRSFQR[3]	0
Bit 2	R/W	RPDRSFQR[2]	0
Bit 1	R/W	RPDRSFQR[1]	0
Bit 0	R/W	RPDRSFQR[0]	0

This register provides the Packet Descriptor Reference Small Buffer Free Queue read address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDRSFQR[15:0]:

The receive packet descriptor reference (RPDR) small buffer free queue read bits (RPDRSFQR[15:0]) define bits 17 to 2 of the Receive Packet Descriptor Reference Small Buffer Free Queue read pointer. This register is initialised by the host. The physical read address in the RPDRSF queue is the sum of RPDRSFQR[15:0] left shifted by 2 bits with the RQB[31:0] bits in the RMAC Receive Queue Base register.

Register 0x2B4 : RMAC Packet Descriptor Reference Small Buffer Free Queue End

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDRSFQE[15]	0
Bit 14	R/W	RPDRSFQE[14]	0
Bit 13	R/W	RPDRSFQE[13]	0
Bit 12	R/W	RPDRSFQE[12]	0
Bit 11	R/W	RPDRSFQE[11]	0
Bit 10	R/W	RPDRSFQE[10]	0
Bit 9	R/W	RPDRSFQE[9]	0
Bit 8	R/W	RPDRSFQE[8]	0
Bit 7	R/W	RPDRSFQE[7]	0
Bit 6	R/W	RPDRSFQE[6]	0
Bit 5	R/W	RPDRSFQE[5]	0
Bit 4	R/W	RPDRSFQE[4]	0
Bit 3	R/W	RPDRSFQE[3]	0
Bit 2	R/W	RPDRSFQE[2]	0
Bit 1	R/W	RPDRSFQE[1]	0
Bit 0	R/W	RPDRSFQE[0]	0

This register provides the Packet Descriptor Reference Small Buffer Free Queue end address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDRSFQE[15:0]:

The receive packet descriptor reference (RPDR) small buffer free queue end bits (RPDRSFQE[15:0]) define bits 17 to 2 of the Receive Packet Descriptor Reference Small Buffer Free Queue end address. This register is initialised by the host. The physical end address in the RPDRSF queue is the sum of RPDRSFQE[15:0] left shifted by 2 bits with the RQB[31:0] bits in the RMAC Receive Queue Base register.

Register 0x2B8 : RMAC Packet Descriptor Reference Ready Queue Start

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDRRQS[15]	0
Bit 14	R/W	RPDRRQS[14]	0
Bit 13	R/W	RPDRRQS[13]	0
Bit 12	R/W	RPDRRQS[12]	0
Bit 11	R/W	RPDRRQS[11]	0
Bit 10	R/W	RPDRRQS[10]	0
Bit 9	R/W	RPDRRQS[9]	0
Bit 8	R/W	RPDRRQS[8]	0
Bit 7	R/W	RPDRRQS[7]	0
Bit 6	R/W	RPDRRQS[6]	0
Bit 5	R/W	RPDRRQS[5]	0
Bit 4	R/W	RPDRRQS[4]	0
Bit 3	R/W	RPDRRQS[3]	0
Bit 2	R/W	RPDRRQS[2]	0
Bit 1	R/W	RPDRRQS[1]	0
Bit 0	R/W	RPDRRQS[0]	0

This register provides the Packet Descriptor Reference Ready Queue start address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDRRQS[15:0]:

The receive packet descriptor reference (RPDR) ready queue start bits (RPDRRQS[15:0]) define bits 17 to 2 of the Receive Packet Descriptor Reference Ready Queue start address. This register is initialised by the host. The physical start address of the RPDRR queue is the sum of RPDRRQS[15:0] left shifted by 2 bits with the RQB[31:0] bits in the RMAC Receive Queue Base register.

Register 0x2BC : RMAC Packet Descriptor Reference Ready Queue Write

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDRRQW[15]	0
Bit 14	R/W	RPDRRQW[14]	0
Bit 13	R/W	RPDRRQW[13]	0
Bit 12	R/W	RPDRRQW[12]	0
Bit 11	R/W	RPDRRQW[11]	0
Bit 10	R/W	RPDRRQW[10]	0
Bit 9	R/W	RPDRRQW[9]	0
Bit 8	R/W	RPDRRQW[8]	0
Bit 7	R/W	RPDRRQW[7]	0
Bit 6	R/W	RPDRRQW[6]	0
Bit 5	R/W	RPDRRQW[5]	0
Bit 4	R/W	RPDRRQW[4]	0
Bit 3	R/W	RPDRRQW[3]	0
Bit 2	R/W	RPDRRQW[2]	0
Bit 1	R/W	RPDRRQW[1]	0
Bit 0	R/W	RPDRRQW[0]	0

This register provides the Packet Descriptor Reference Ready Queue write address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDRRQW[15:0]:

The receive packet descriptor reference (RPDR) ready queue write bits (RPDRRQW[15:0]) define bits 17 to 2 of the Receive Packet Descriptor Reference Ready Queue write pointer. This register is initialised by the host. The physical write address in the RPDRR queue is the sum of RPDRRQW[15:0] left shifted by 2 bits with the RQB[31:0] bits in the RMAC Receive Queue Base register.

Register 0x2C0 : RMAC Packet Descriptor Reference Ready Queue Read

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDRRQR[15]	0
Bit 14	R/W	RPDRRQR[14]	0
Bit 13	R/W	RPDRRQR[13]	0
Bit 12	R/W	RPDRRQR[12]	0
Bit 11	R/W	RPDRRQR[11]	0
Bit 10	R/W	RPDRRQR[10]	0
Bit 9	R/W	RPDRRQR[9]	0
Bit 8	R/W	RPDRRQR[8]	0
Bit 7	R/W	RPDRRQR[7]	0
Bit 6	R/W	RPDRRQR[6]	0
Bit 5	R/W	RPDRRQR[5]	0
Bit 4	R/W	RPDRRQR[4]	0
Bit 3	R/W	RPDRRQR[3]	0
Bit 2	R/W	RPDRRQR[2]	0
Bit 1	R/W	RPDRRQR[1]	0
Bit 0	R/W	RPDRRQR[0]	0

This register provides the Packet Descriptor Reference Ready Queue read address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDRRQR[15:0]:

The receive packet descriptor reference (RPDR) ready queue read bits (RPDRRQR[15:0]) define bits 17 to 2 of the Receive Packet Descriptor Reference Ready Queue read pointer. This register is initialised by the host. The physical read address in the RPDRR queue is the sum of RPDRRQR[15:0] left shifted by 2 bits with the RQB[31:0] bits in the RMAC Receive Queue Base register.

Register 0x2C4 : RMAC Packet Descriptor Reference Ready Queue End

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	RPDRRQE[15]	0
Bit 14	R/W	RPDRRQE[14]	0
Bit 13	R/W	RPDRRQE[13]	0
Bit 12	R/W	RPDRRQE[12]	0
Bit 11	R/W	RPDRRQE[11]	0
Bit 10	R/W	RPDRRQE[10]	0
Bit 9	R/W	RPDRRQE[9]	0
Bit 8	R/W	RPDRRQE[8]	0
Bit 7	R/W	RPDRRQE[7]	0
Bit 6	R/W	RPDRRQE[6]	0
Bit 5	R/W	RPDRRQE[5]	0
Bit 4	R/W	RPDRRQE[4]	0
Bit 3	R/W	RPDRRQE[3]	0
Bit 2	R/W	RPDRRQE[2]	0
Bit 1	R/W	RPDRRQE[1]	0
Bit 0	R/W	RPDRRQE[0]	0

This register provides the Packet Descriptor Reference Ready Queue end address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

RPDRRQE[15:0]:

The receive packet descriptor reference (RPDR) ready queue end bits (RPDRRQE[15:0]) define bits 17 to 2 of the Receive Packet Descriptor Reference Ready Queue end address. This register is initialised by the host. The physical end address in the RPDRR queue is the sum of RPDRRQE[15:0] left shifted by 2 bits with the RQB[31:0] bits in the RMAC Receive Queue Base register.

Register 0x300 : TMAC Control

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6	R/W	TDQ_FRN[1]	0
Bit 5	R/W	TDQ_FRN[0]	0
Bit 4	R/W	TDQ_RDYN[2]	0
Bit 3	R/W	TDQ_RDYN[1]	0
Bit 2	R/W	TDQ_RDYN[0]	0
Bit 1	R/W	CACHE	1
Bit 0	R/W	ENABLE	0

This register provides control of the TMAC block.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

ENABLE:

The transmit DMA controller enable bit (ENABLE) enables the TMAC to accept TDRs from the TDR Ready Queue and reads packet data from host

memory. When ENABLE is set high, the TMAC is enabled. When ENABLE is set low, the TDR Ready Queue is ignored. Once all linked lists of TDs built up by the TMAC have been exhausted, no more data will be transmitted on the TD[31:0] links.

CACHE:

The transmit descriptor reference cache enable bit (CACHE) controls the frequency at which TDRs are written to the TDR Free Queue. When CACHE is set high, freed TDRs are cache and then written up to six at a time. When CACHE is set low, freed TDRs are written one at a time.

TDQ_RDYN[2:0]:

The TDQ_RDYN[2:0] field sets the number of transmit descriptor references (TDRs) that must be read from the TDR Ready Queue before the TDR Ready interrupt (TDQRDYI) is asserted, as follows:

Table 20 – TDQ_RDYN[2:0] Settings

TDQ_RDYN[2:0]	No of TDRs
000	1
001	4
010	6
011	8
100	16
101	32
110	Reserved
111	Reserved

TDQ_FRN[1:0]:

The TDQ_FRN[1:0] field sets the number of times that a block of TDRs are written to the TDR Free Queue from the TMACs internal cache before the TDR Free Queue Interrupt (TDQFI) is asserted, as follows:

Table 21 – TDQ_FRN[1:0] Settings

TDQ_FRN[1:0]	No of Reads
00	1
01	4

TDQ_FRN[1:0]	No of Reads
10	8
11	Reserved

Register 0x304 : TMAC Indirect Channel Provisioning

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	BUSY	X
Bit 14	R/W	RWB	0
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7	R/W	PROV	0
Bit 6	R/W	CHAN[6]	0
Bit 5	R/W	CHAN[5]	0
Bit 4	R/W	CHAN[4]	0
Bit 3	R/W	CHAN[3]	0
Bit 2	R/W	CHAN[2]	0
Bit 1	R/W	CHAN[1]	0
Bit 0	R/W	CHAN[0]	0

The Channel Provisioning Register is used to provision and unprovision channels, and also to query the provision status of channels. When a channel is provisioned, chains of packet data for that channel will be accepted by the TMAC and placed on the channel's linked list of packets to be transmitted. When a channel is unprovisioned, chains of packet data for that channel will be rejected by the TMAC and returned to the TDR Free Queue with the status bits in the queue element set to indicate the rejection.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not

implemented. However, when all four byte enables are negated, no access is made to this register.

CHAN[6:0]:

The indirect data bits (CHAN[6:0]) report the channel number read from the TMAC internal memory after an indirect read operation has completed. Channel number to be written to the TMAC internal memory in an indirect write operation must be set up in this register before triggering the write. CHAN[6:0] reflects the value written until the completion of a subsequent indirect read operation.

PROV:

The indirect provision enable bit (PROV) reports the channel provision enable flag read from the TMAC internal memory after an indirect read operation has completed. The provision enable flag to be written to the TMAC internal memory, in an indirect write operation, must be set up in this register before triggering the write. When PROV is set high, the channel as indicated by CHAN[6:0] is provisioned. When PROV is set low, the channel indicated by CHAN[6:0] is unprovisioned. PROV reflects the value written until the completion of a subsequent indirect read operation.

RWB:

The Read/Write Bar (RWB) bit selects between a provisioning/unprovisioning operation (write) or a query operation (read). Writing a logic 0 to RWB triggers the provisioning or unprovisioning of a channel as specified by CHAN[6:0] and PROV. Writing a logic 1 to RWB triggers a query of the channel specified by CHAN[6:0].

BUSY:

The indirect access status bit (BUSY) reports the progress of an indirect access. BUSY is set high when this register is written to trigger an indirect access, and will stay high until the access is complete. At which point, BUSY will be set low. This register should be polled to determine when data from an indirect read operation is available or to determine when a new indirect write operation may commence.

Register 0x308 : TMAC Descriptor Table Base LSW

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TDTB[15]	0
Bit 14	R/W	TDTB[14]	0
Bit 13	R/W	TDTB[13]	0
Bit 12	R/W	TDTB[12]	0
Bit 11	R/W	TDTB[11]	0
Bit 10	R/W	TDTB[10]	0
Bit 9	R/W	TDTB[9]	0
Bit 8	R/W	TDTB[8]	0
Bit 7	R/W	TDTB[7]	0
Bit 6	R/W	TDTB[6]	0
Bit 5	R/W	TDTB[5]	0
Bit 4	R/W	TDTB[4]	0
Bit 3	R/W	TDTB[3]	0
Bit 2	R/W	TDTB[2]	0
Bit 1	R/W	TDTB[1]	0
Bit 0	R/W	TDTB[0]	0

This register provides the less significant word of the Transmit Descriptor Table Base address. The contents of this register is held in a holding register until a write access to the companion TMAC Transmit Descriptor Table Base MSW register.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

Register 0x30C : TMAC Descriptor Table Base MSW

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TDTB[31]	0
Bit 14	R/W	TDTB[30]	0
Bit 13	R/W	TDTB[29]	0
Bit 12	R/W	TDTB[28]	0
Bit 11	R/W	TDTB[27]	0
Bit 10	R/W	TDTB[26]	0
Bit 9	R/W	TDTB[25]	0
Bit 8	R/W	TDTB[24]	0
Bit 7	R/W	TDTB[23]	0
Bit 6	R/W	TDTB[22]	0
Bit 5	R/W	TDTB[21]	0
Bit 4	R/W	TDTB[20]	0
Bit 3	R/W	TDTB[19]	0
Bit 2	R/W	TDTB[18]	0
Bit 1	R/W	TDTB[17]	0
Bit 0	R/W	TDTB[16]	0

This register provides the more significant word of the Transmit Descriptor Table Base address. The contents of the companion TMAC Transmit Descriptor Table Base LSW register is held in a holding register until a write access to this register, at which point, the base address of the transmit descriptor table is updated.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

TDTB[31:0]:

The transmit descriptor table base bits (TDTB[31:0]) provides the base address of the Transmit Descriptor Table in PCI host memory. This register is initialised by the host. To calculate the physical address of a TD, the 14 bit TD offset must be added to bits 31 to 4 of the Transmit Descriptor Table Base (TDTB[31:4]).

The table must be on a 16 byte boundary and thus the least significant four bits must be written to logic zero.

Register 0x310 : TMAC Queue Base LSW

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TQB[15]	0
Bit 14	R/W	TQB[14]	0
Bit 13	R/W	TQB[13]	0
Bit 12	R/W	TQB[12]	0
Bit 11	R/W	TQB[11]	0
Bit 10	R/W	TQB[10]	0
Bit 9	R/W	TQB[9]	0
Bit 8	R/W	TQB[8]	0
Bit 7	R/W	TQB[7]	0
Bit 6	R/W	TQB[6]	0
Bit 5	R/W	TQB[5]	0
Bit 4	R/W	TQB[4]	0
Bit 3	R/W	TQB[3]	0
Bit 2	R/W	TQB[2]	0
Bit 1	R/W	TQB[1]	0
Bit 0	R/W	TQB[0]	0

This register provides the less significant word of the Transmit Queue Base address. The contents of this register is held in a holding register until a write access to the companion TMAC Transmit Queue Base MSW register.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

Register 0x314 : TMAC Queue Base MSW

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TQB[31]	0
Bit 14	R/W	TQB[30]	0
Bit 13	R/W	TQB[29]	0
Bit 12	R/W	TQB[28]	0
Bit 11	R/W	TQB[27]	0
Bit 10	R/W	TQB[26]	0
Bit 9	R/W	TQB[25]	0
Bit 8	R/W	TQB[24]	0
Bit 7	R/W	TQB[23]	0
Bit 6	R/W	TQB[22]	0
Bit 5	R/W	TQB[21]	0
Bit 4	R/W	TQB[20]	0
Bit 3	R/W	TQB[19]	0
Bit 2	R/W	TQB[18]	0
Bit 1	R/W	TQB[17]	0
Bit 0	R/W	TQB[16]	0

This register provides the more significant word of the Transmit Queue Base address. The contents of the companion TMAC Transmit Descriptor Table Base LSW register is held in a holding register until a write access to this register, at which point, the base address of the transmit queue is updated.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

TQB[31:0]:

The transmit queue base bits (TQB[31:0]) provides the base address of the Transmit Descriptor Reference Free and Transmit Descriptor Reference Ready queue in PCI host memory. This register is initialised by the host. To calculate the physical address of a particular transmit queue element, the TQB bits are added with the appropriate queue start, end, read or write index registers to form the physical address.

The base address must be dword aligned and thus the least significant two bits must be written to logic zero.

Register 0x318 : TMAC Descriptor Reference Free Queue Start

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TDRFQS[15]	0
Bit 14	R/W	TDRFQS[14]	0
Bit 13	R/W	TDRFQS[13]	0
Bit 12	R/W	TDRFQS[12]	0
Bit 11	R/W	TDRFQS[11]	0
Bit 10	R/W	TDRFQS[10]	0
Bit 9	R/W	TDRFQS[9]	0
Bit 8	R/W	TDRFQS[8]	0
Bit 7	R/W	TDRFQS[7]	0
Bit 6	R/W	TDRFQS[6]	0
Bit 5	R/W	TDRFQS[5]	0
Bit 4	R/W	TDRFQS[4]	0
Bit 3	R/W	TDRFQS[3]	0
Bit 2	R/W	TDRFQS[2]	0
Bit 1	R/W	TDRFQS[1]	0
Bit 0	R/W	TDRFQS[0]	0

This register provides the Transmit Descriptor Reference Free Queue start address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

TDRFQS[15:0]:

The transmit packet descriptor reference (TDR) free queue start bits (TDRFQS[15:0]) define bits 17 to 2 of the Transmit Packet Descriptor Reference Free Queue start address. This register is initialised by the host. The physical start address of the TDRF queue is the sum of TDRFQS[15:0] left shifted by 2 bits with the TQB[31:0] bits in the TMAC Transmit Queue Base register.

Register 0x31C TMAC Descriptor Reference Free Queue Write

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TDRFQW[15]	0
Bit 14	R/W	TDRFQW[14]	0
Bit 13	R/W	TDRFQW[13]	0
Bit 12	R/W	TDRFQW[12]	0
Bit 11	R/W	TDRFQW[11]	0
Bit 10	R/W	TDRFQW[10]	0
Bit 9	R/W	TDRFQW[9]	0
Bit 8	R/W	TDRFQW[8]	0
Bit 7	R/W	TDRFQW[7]	0
Bit 6	R/W	TDRFQW[6]	0
Bit 5	R/W	TDRFQW[5]	0
Bit 4	R/W	TDRFQW[4]	0
Bit 3	R/W	TDRFQW[3]	0
Bit 2	R/W	TDRFQW[2]	0
Bit 1	R/W	TDRFQW[1]	0
Bit 0	R/W	TDRFQW[0]	0

This register provides the Transmit Descriptor Reference Free Queue write address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

TDRFQW[15:0]:

The transmit packet descriptor reference (TPDR) free queue write bits (TDRFQW[15:0]) define bits 17 to 2 of the Transmit Packet Descriptor Reference Free Queue write pointer. This register is initialised by the host. The physical write address in the TDRF queue is the sum of TDRFQW[15:0] left shifted by 2 bits with the TQB[31:0] bits in the TMAC Transmit Queue Base register.

Register 0x320 : TMAC Descriptor Reference Free Queue Read

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TDRFQR[15]	0
Bit 14	R/W	TDRFQR[14]	0
Bit 13	R/W	TDRFQR[13]	0
Bit 12	R/W	TDRFQR[12]	0
Bit 11	R/W	TDRFQR[11]	0
Bit 10	R/W	TDRFQR[10]	0
Bit 9	R/W	TDRFQR[9]	0
Bit 8	R/W	TDRFQR[8]	0
Bit 7	R/W	TDRFQR[7]	0
Bit 6	R/W	TDRFQR[6]	0
Bit 5	R/W	TDRFQR[5]	0
Bit 4	R/W	TDRFQR[4]	0
Bit 3	R/W	TDRFQR[3]	0
Bit 2	R/W	TDRFQR[2]	0
Bit 1	R/W	TDRFQR[1]	0
Bit 0	R/W	TDRFQR[0]	0

This register provides the Transmit Descriptor Reference Free Queue read address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

TDRFQR[15:0]:

The transmit packet descriptor reference (TPDR) free queue read bits (TDRFQR[15:0]) define bits 17 to 2 of the Transmit Packet Descriptor Reference Free Queue read pointer. This register is initialised by the host. The physical read address in the TDRF queue is the sum of TDRFQR[15:0] left shifted by 2 bits with the TQB[31:0] bits in the TMAC Transmit Queue Base register.

Register 0x324 : TMAC Descriptor Reference Free Queue End

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TDRFQE[15]	0
Bit 14	R/W	TDRFQE[14]	0
Bit 13	R/W	TDRFQE[13]	0
Bit 12	R/W	TDRFQE[12]	0
Bit 11	R/W	TDRFQE[11]	0
Bit 10	R/W	TDRFQE[10]	0
Bit 9	R/W	TDRFQE[9]	0
Bit 8	R/W	TDRFQE[8]	0
Bit 7	R/W	TDRFQE[7]	0
Bit 6	R/W	TDRFQE[6]	0
Bit 5	R/W	TDRFQE[5]	0
Bit 4	R/W	TDRFQE[4]	0
Bit 3	R/W	TDRFQE[3]	0
Bit 2	R/W	TDRFQE[2]	0
Bit 1	R/W	TDRFQE[1]	0
Bit 0	R/W	TDRFQE[0]	0

This register provides the Transmit Descriptor Reference Free Queue end address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

TDRFQE[15:0]:

The transmit packet descriptor reference (TDR) free queue end bits (TDRFQE[15:0]) define bits 17 to 2 of the Transmit Packet Descriptor Reference Free Queue end address. This register is initialised by the host. The physical end address of the TDRF queue is the sum of TDRFQE[15:0] left shifted by 2 bits with the TQB[31:0] bits in the TMAC Transmit Queue Base register.

Register 0x328 :TMAC Descriptor Reference Ready Queue Start

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TDRRQS[15]	0
Bit 14	R/W	TDRRQS[14]	0
Bit 13	R/W	TDRRQS[13]	0
Bit 12	R/W	TDRRQS[12]	0
Bit 11	R/W	TDRRQS[11]	0
Bit 10	R/W	TDRRQS[10]	0
Bit 9	R/W	TDRRQS[9]	0
Bit 8	R/W	TDRRQS[8]	0
Bit 7	R/W	TDRRQS[7]	0
Bit 6	R/W	TDRRQS[6]	0
Bit 5	R/W	TDRRQS[5]	0
Bit 4	R/W	TDRRQS[4]	0
Bit 3	R/W	TDRRQS[3]	0
Bit 2	R/W	TDRRQS[2]	0
Bit 1	R/W	TDRRQS[1]	0
Bit 0	R/W	TDRRQS[0]	0

This register provides the Transmit Descriptor Reference Ready Queue start address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

TDRRQS[15:0]:

The transmit packet descriptor reference (TDR) ready queue start bits (TDRRQS[15:0]) define bits 17 to 2 of the Transmit Packet Descriptor Reference Ready Queue start address. This register is initialised by the host. The physical start address of the TDRF queue is the sum of TDRRQS[15:0] left shifted by 2 bits with the TQB[31:0] bits in the TMAC Transmit Queue Base register.

Register 0x32C : TMAC Descriptor Reference Ready Queue Write

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TDRRQW[15]	0
Bit 14	R/W	TDRRQW[14]	0
Bit 13	R/W	TDRRQW[13]	0
Bit 12	R/W	TDRRQW[12]	0
Bit 11	R/W	TDRRQW[11]	0
Bit 10	R/W	TDRRQW[10]	0
Bit 9	R/W	TDRRQW[9]	0
Bit 8	R/W	TDRRQW[8]	0
Bit 7	R/W	TDRRQW[7]	0
Bit 6	R/W	TDRRQW[6]	0
Bit 5	R/W	TDRRQW[5]	0
Bit 4	R/W	TDRRQW[4]	0
Bit 3	R/W	TDRRQW[3]	0
Bit 2	R/W	TDRRQW[2]	0
Bit 1	R/W	TDRRQW[1]	0
Bit 0	R/W	TDRRQW[0]	0

This register provides the Transmit Descriptor Reference Ready Queue write address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

TDRRQW[15:0]:

The transmit packet descriptor reference (TPDR) ready queue write bits (TDRRQW[15:0]) define bits 17 to 2 of the Transmit Packet Descriptor Reference Ready Queue write pointer. This register is initialised by the host. The physical write address in the TDRF queue is the sum of TDRRQW[15:0] left shifted by 2 bits with the TQB[31:0] bits in the TMAC Transmit Queue Base register.

Register 0x330 : TMAC Descriptor Reference Ready Queue Read

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TDRRQR[15]	0
Bit 14	R/W	TDRRQR[14]	0
Bit 13	R/W	TDRRQR[13]	0
Bit 12	R/W	TDRRQR[12]	0
Bit 11	R/W	TDRRQR[11]	0
Bit 10	R/W	TDRRQR[10]	0
Bit 9	R/W	TDRRQR[9]	0
Bit 8	R/W	TDRRQR[8]	0
Bit 7	R/W	TDRRQR[7]	0
Bit 6	R/W	TDRRQR[6]	0
Bit 5	R/W	TDRRQR[5]	0
Bit 4	R/W	TDRRQR[4]	0
Bit 3	R/W	TDRRQR[3]	0
Bit 2	R/W	TDRRQR[2]	0
Bit 1	R/W	TDRRQR[1]	0
Bit 0	R/W	TDRRQR[0]	0

This register provides the Transmit Descriptor Reference Ready Queue read address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

TDRRQR[15:0]:

The transmit packet descriptor reference (TPDR) ready queue read bits (TDRRQR[15:0]) define bits 17 to 2 of the Transmit Packet Descriptor Reference Ready Queue read pointer. This register is initialised by the host. The physical read address in the TDRF queue is the sum of TDRRQR[15:0] left shifted by 2 bits with the TQB[31:0] bits in the TMAC Transmit Queue Base register.

Register 0x334 : TMAC Descriptor Reference Ready Queue End

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TDRRQE[15]	0
Bit 14	R/W	TDRRQE[14]	0
Bit 13	R/W	TDRRQE[13]	0
Bit 12	R/W	TDRRQE[12]	0
Bit 11	R/W	TDRRQE[11]	0
Bit 10	R/W	TDRRQE[10]	0
Bit 9	R/W	TDRRQE[9]	0
Bit 8	R/W	TDRRQE[8]	0
Bit 7	R/W	TDRRQE[7]	0
Bit 6	R/W	TDRRQE[6]	0
Bit 5	R/W	TDRRQE[5]	0
Bit 4	R/W	TDRRQE[4]	0
Bit 3	R/W	TDRRQE[3]	0
Bit 2	R/W	TDRRQE[2]	0
Bit 1	R/W	TDRRQE[1]	0
Bit 0	R/W	TDRRQE[0]	0

This register provides the Transmit Descriptor Reference Ready Queue end address.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

TDRRQE[15:0]:

The transmit packet descriptor reference (TDR) ready queue end bits (TDRRQE[15:0]) define bits 17 to 2 of the Transmit Packet Descriptor Reference Ready Queue end address. This register is initialised by the host. The physical end address of the TDRF queue is the sum of TDRRQE[15:0] left shifted by 2 bits with the TQB[31:0] bits in the TMAC Transmit Queue Base register.

Register 0x380 : THDL Indirect Channel Select

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	BUSY	X
Bit 14	R/W	CRWB	0
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6	R/W	CHAN[6]	0
Bit 5	R/W	CHAN[5]	0
Bit 4	R/W	CHAN[4]	0
Bit 3	R/W	CHAN[3]	0
Bit 2	R/W	CHAN[2]	0
Bit 1	R/W	CHAN[1]	0
Bit 0	R/W	CHAN[0]	0

This register provides the channel number used to access the transmit channel provision RAM. Writing to this register triggers an indirect channel register access.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

CHAN[6:0]:

The indirect channel number bits (CHAN[6:0]) indicate the channel to be configured or interrogated in the indirect access.

CRWB:

The channel indirect access control bit (CRWB) selects between a configure (write) or interrogate (read) access to the channel provision RAM. Writing a logic zero to CRWB triggers an indirect write operation. Data to be written is taken from the Indirect Channel Data registers. Writing a logic one to CRWB triggers an indirect read operation. The data read can be found in the Indirect Channel Data registers.

BUSY:

The indirect access status bit (BUSY) reports the progress of an indirect access. BUSY is set high when this register is written to trigger an indirect access, and will stay high until the access is complete. At which point, BUSY will be set low. This register should be polled to determine when data from an indirect read operation is available in the THDL Indirect Channel Data #1, #2 and #3 registers or to determine when a new indirect write operation may commence.

Register 0x384 : THDL Indirect Channel Data #1

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	PROV	0
Bit 14	R/W	CRC[1]	0
Bit 13	R/W	CRC[0]	0
Bit 12	R/W	IDLE	0
Bit 11	R/W	DELIN	0
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8	W	FPTR[8]	0
Bit 7	W	FPTR[7]	0
Bit 6	W	FPTR[6]	0
Bit 5	W	FPTR[5]	0
Bit 4	W	FPTR[4]	0
Bit 3	W	FPTR[3]	0
Bit 2	W	FPTR[2]	0
Bit 1	W	FPTR[1]	0
Bit 0	W	FPTR[0]	0

This register contains data read from the channel provision RAM after an indirect channel read operation or data to be inserted into the channel provision RAM in an indirect channel write operation.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

FPTR[8:0]:

The indirect FIFO block pointer (FPTR[8:0]) informs the partial packet buffer processor the circular linked list of blocks to use for a FIFO for the channel. The FIFO pointer to be written to the channel provision RAM, in an indirect write operation, must be set up in this register before triggering the write. The FIFO pointer value can be any one of the block numbers provisioned, by indirect block write operations, to form the circular buffer.

DELIN:

The indirect delineate enable bit (DELIN) configures the HDLC processor to perform flag sequence insertion and bit stuffing on the outgoing data stream. The delineate enable bit to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. When DELIN is set high, flag sequence insertion, bit stuffing and ,optionally, CRC generation is performed on the outgoing HDLC data stream. When DELIN is set low, the HDLC processor does not perform any processing (flag sequence insertion, bit stuffing nor CRC generation) on the outgoing stream. DELIN reflects the value written until the completion of a subsequent indirect channel read operation.

IDLE:

The interframe time fill bit (IDLE) configures the HDLC processor to use flag bytes or HDLC idle as the interframe time fill between HDLC packets. The value of IDLE to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. When IDLE is set low, the HDLC processor uses flag bytes as the interframe time fill. When IDLE is set high, the HDLC processor uses HDLC idle (all one's bit with no bit-stuffing pattern is transmitted) as the interframe time fill. IDLE reflects the value written until the completion of a subsequent indirect channel read operation.

CRC[1:0]:

The CRC algorithm (CRC[1:0]) configures the HDLC processor to perform CRC generation on the outgoing HDLC data stream. The value of CRC[1:0] to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. CRC[1:0] is ignored when DELIN is low. CRC[1:0] reflects the value written until the completion of a subsequent indirect channel read operation.

Table 22 – CRC[1:0] Settings

CRC[1]	CRC[0]	Operation
0	0	No CRC

CRC[1]	CRC[0]	Operation
0	1	CRC-CCITT
1	0	CRC-32
1	1	Reserved

PROV:

The indirect provision enable bit (PROV) reports the channel provision enable flag read from the channel provision RAM after an indirect channel read operation has completed. The provision enable flag to be written to the channel provision RAM, in an indirect write operation, must be set up in this register before triggering the write. When PROV is set high, the HDLC processor will service requests for data from the TCAS block. When PROV is set low, the HDLC processor will ignore requests from the TCAS block. PROV reflects the value written until the completion of a subsequent indirect channel read operation.

Register 0x388 : THDL Indirect Channel Data #2

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	7BIT	0
Bit 14	R/W	PRIORITYB	0
Bit 13	R/W	INVERT	0
Bit 12	R/W	DFCS	0
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8	W	FLEN[8]	0
Bit 7	W	FLEN[7]	0
Bit 6	W	FLEN[6]	0
Bit 5	W	FLEN[5]	0
Bit 4	W	FLEN[4]	0
Bit 3	W	FLEN[3]	0
Bit 2	W	FLEN[2]	0
Bit 1	W	FLEN[1]	0
Bit 0	W	FLEN[0]	0

This register contains data to be inserted into the channel provision RAM in an indirect write operation.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

FLEN[8:0]:

The indirect FIFO length (FLEN[8:0]) is the number of blocks, less one, that is provisioned to the circular channel FIFO specified by the FPTR[8:0] block pointer. The FIFO length to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write.

DFCS:

The diagnose frame check sequence bit (DFCS) controls the inversion of the FCS field inserted into the transmit packet. The value of DFCS to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. When DFCS is set to one, the FCS field in the outgoing HDLC stream is logically inverted allowing diagnosis of downstream FCS verification logic. The outgoing FCS field is not inverted when DFCS is set to zero. DFCS reflects the value written until the completion of a subsequent indirect channel read operation.

INVERT:

The HDLC data inversion bit (INVERT) configures the HDLC processor to logically invert the outgoing HDLC stream. The value of INVERT to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. When INVERT is set to one, the outgoing HDLC stream is logically inverted. The outgoing HDLC stream is not inverted when INVERT is set to zero. INVERT reflects the value written until the completion of a subsequent indirect channel read operation.

PRIORITYB:

The active low channel FIFO expedite enable bit (PRIORITYB) informs the partial packet processor of the priority of the channel relative to other channels when requesting data from the DMA port. The value of PRIORITYB to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. Channel FIFOs with PRIORITYB set to one are inhibited from making expedited requests for data to the TMAC. When PRIORITYB is set to zero, both normal and expedited requests can be made to the TMAC. Channels with HDLC data rate to FIFO size ratio that is significantly lower than other channels should have PRIORITYB set to one. PRIORITYB reflects the value written until the completion of a subsequent indirect channel read operation.

7BIT:

The least significant stuff enable bit (7BIT) configures the HDLC processor to stuff the least significant bit of each octet in the corresponding transmit link (TD[n]). The value of 7BIT to be written to the channel provision RAM, in an

indirect channel write operation, must be set up in this register before triggering the write. When 7BIT is set high, 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. When 7BIT is set low, the entire octet contains valid data and BIT8 is ignored. 7BIT reflects the value written until the completion of a subsequent indirect channel read operation.

Register 0x38C : THDL Indirect Channel Data #3

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	TRANS	0
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11	R/W	LEVEL[3]	0
Bit 10	R/W	LEVEL[2]	0
Bit 9	R/W	LEVEL[1]	0
Bit 8	R/W	LEVEL[0]	0
Bit 7	R/W	FLAG[2]	0
Bit 6	R/W	FLAG[1]	0
Bit 5	R/W	FLAG[0]	0
Bit 4		Unused[	X
Bit 3		Unused	X
Bit 2	R/W	XFER[2]	0
Bit 1	R/W	XFER[1]	0
Bit 0	R/W	XFER[0]	0

This register contains data read from the channel provision RAM after an indirect read operation or data to be inserted into the channel provision RAM in an indirect write operation.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

XFER[2:0]:

The indirect channel transfer size (XFER[2:0]) specifies the amount of data the partial packet processor requests from the TMAC block. The channel transfer size to be written to the channel provision RAM, in an indirect write operation, must be set up in this register before triggering the write. When the channel FIFO free space reaches or exceeds the limit specified by XFER[2:0], the partial packet processor will make a request for data to the TMAC to retrieve the XFER[2:0] + 1 blocks of data. FIFO free space and transfer size are measured in the number of 16-byte blocks. XFER[2:0] reflects the value written until the completion of a subsequent indirect channel read operation.

To prevent lockup, the channel transfer size (XFER[2:0]) 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.

The case of a single block transfer size is a special. When BURSTEN is set high and XFER[2:0] = 'b000, the transfer size is variable. The THDL will request the TMAC to transfer as much data as there is free space in the FIFO, up to a maximum set by BURST[2:0].

FLAG[2:0]:

The flag insertion control (FLAG[2:0]) configures the minimum number of flags or bytes of idle bits the HDLC processor inserts between HDLC packets. The value of FLAG[2:0] to be written to the channel provision RAM, in an indirect channel write operation, must be set up in this register before triggering the write. The minimum number of flags or bytes of idle (8 bits of 1's) inserted between HDLC packets is shown in the table below. FLAG[2:0] reflects the value written until the completion of a subsequent indirect channel read operation.

Table 23 – FLAG[2:0] Settings

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

FLAG[2:0]	Minimum Number of Flag/Idle Bytes
101	32 flags / 30 idle bytes
110	64 flags / 62 idle bytes
111	128 flags / 126 idle bytes

LEVEL[3:0]:

The indirect channel FIFO trigger level (LEVEL[3:0]), in concert with the TRANS bit, configure the various channel FIFO free space levels which trigger the HDLC processor to start transmission of a HDLC packet as well as trigger the partial packet buffer to make DMA request for data as shown in the following table. The channel FIFO trigger level to be written to the channel provision RAM, in an indirect write operation, must be set up in this register before triggering the write. LEVEL[3:0] reflects the value written until the completion of a subsequent indirect channel read operation.

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 the level specified in the Expedite 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 expedite requests to the TMAC to retrieve XFER[2:0] + 1 blocks of data.

To prevent lockup, the channel transfer size (XFER[2:0]) 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.

TRANS:

The indirect transmission start bit (TRANS), in concert with the LEVEL[3:0] bits, configure the various channel FIFO free space levels which trigger the HDLC processor to start transmission of a HDLC packet as well as trigger the partial packet buffer to make DMA request for data as shown in the following table. The transmission start mode to be written to the channel provision RAM, in an indirect write operation, must be set up in this register before triggering the write. TRANS reflects the value written until the completion of a subsequent indirect channel read operation.

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 greater than the level specified in the Expedite 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 TMAC to retrieve XFER[2:0] + 1 blocks of data.

To prevent lockup, the channel transfer size (XFER[2:0]) 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.

Table 24 – Level[3:0]/TRANS Settings

LEVEL[3:0]	Expedite 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)

LEVEL[3:0]	Expedite Trigger Level	Start Transmission Level (TRANS=0)	Start Transmission Level (TRANS=1)
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)
1100	128 Blocks (2 Kbytes free)	96 Blocks (1.5 Kbytes free)	64 Blocks (1 Kbytes free)
1101	192 Blocks (3 Kbytes free)	128 Blocks (2 Kbytes free)	96 Blocks (1.5 Kbytes free)
1110	256 Blocks (4 Kbytes free)	192 Blocks (3 Kbytes free)	128 Blocks (2 Kbytes free)
1111	384 Blocks (6 Kbytes free)	256 Blocks (4 Kbytes free)	192 Blocks (3 Kbytes free)

Register 0x3A0 : THDL Indirect Block Select

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	BUSY	X
Bit 14	R/W	BRWB	0
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8	R/W	BLOCK[8]	0
Bit 7	R/W	BLOCK[7]	0
Bit 6	R/W	BLOCK[6]	0
Bit 5	R/W	BLOCK[5]	0
Bit 4	R/W	BLOCK[4]	0
Bit 3	R/W	BLOCK[3]	0
Bit 2	R/W	BLOCK[2]	0
Bit 1	R/W	BLOCK[1]	0
Bit 0	R/W	BLOCK[0]	0

This register provides the block number used to access the block pointer RAM. Writing to this register triggers an indirect block register access.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

BLOCK[8:0]:

The indirect block number (BLOCK[8:0]) indicate the block to be configured or interrogated in the indirect access.

BRWB:

The block indirect access control bit (BRWB) selects between a configure (write) or interrogate (read) access to the block pointer RAM. Writing a logic zero to BRWB triggers an indirect block write operation. Data to be written is taken from the Indirect Block Data register. Writing a logic one to BRWB triggers an indirect block read operation. The data read can be found in the Indirect Block Data register.

BUSY:

The indirect access status bit (BUSY) reports the progress of an indirect access. BUSY is set high when this register is written to trigger an indirect access, and will stay high until the access is complete. At which point, BUSY will be set low. This register should be polled to determine when data from an indirect read operation is available in the THDL Indirect Block Data register or to determine when a new indirect write operation may commence.

Register 0x3A4 : THDL Indirect Block Data

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	Reserved	0
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8	R/W	BPTR[8]	0
Bit 7	R/W	BPTR[7]	0
Bit 6	R/W	BPTR[6]	0
Bit 5	R/W	BPTR[5]	0
Bit 4	R/W	BPTR[4]	0
Bit 3	R/W	BPTR[3]	0
Bit 2	R/W	BPTR[2]	0
Bit 1	R/W	BPTR[1]	0
Bit 0	R/W	BPTR[0]	0

This register contains data read from the transmit block pointer RAM after an indirect block read operation or data to be inserted into the transmit block pointer RAM in an indirect block write operation.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

BPTR[8:0]:

The indirect block pointer (BPTR[8:0]) configures the block pointer of the block specified by the Indirect Block Select register. The block pointer to be written to the transmit block pointer RAM, in an indirect write operation, must be set up in this register before triggering the write. The block pointer value is the block number of the next block in the linked list. A circular list of blocks must be formed in order to use the block list as a channel FIFO buffer. FPTR[8:0] reflects the value written until the completion of a subsequent indirect block read operation.

When provisioning a channel FIFO, all blocks pointers must be re-written to properly initialize the FIFO.

Reserved:

The reserved bit (Reserved) must be set low for correct operation of the FREEDM-32 device.

Register 0x3B0 : THDL Configuration

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9	R/W	BIT8	0
Bit 8	R/W	TSTD	0
Bit 7	R/W	BURSTEN	0
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3		Unused	X
Bit 2	R/W	BURST[2]	0
Bit 1	R/W	BURST[1]	0
Bit 0	R/W	BURST[0]	0

This register configures all provisioned channels.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

BURST[2:0]:

The DMA burst length bits (BURST[2:0]) configure the maximum amount of transmit data that can be requested in a single DMA transaction for channels

whose channel transfer size is set to one block (XFER[2:0] = 'b000). BURST[2:0] has no effect when BURSTEN is set low, nor on channels configured with other transfer sizes. BURST[2:0] defines the maximum number of 16 byte blocks, less one, that is transferred in each DMA transaction. Thus, the minimum number of blocks is one (16 bytes) and the maximum is eight (128 bytes).

BURSTEN:

The burst length enable bit (BURSTEN) controls the use of BURST[2:0] in determining the amount of data requested in a single DMA transaction for channels whose channel transfer size is set to one block (XFER[2:0] = 'b000). BURSTEN has no effect on channels configured with other transfer sizes. When BURSTEN is set high, the maximum size of DMA transfer is limited by BURST[2:0]. The transmit HDLC processor may combine several channel transfer size amounts into a single transaction. When BURSTEN is set low, the amount of data in a DMA transfer is limited to one block.

TSTD:

The telecom standard bit (TSTD) controls the bit ordering of the HDLC data transferred from the PCI host. When TSTD is set low, the least significant bit of the each byte on the PCI bus (AD[0], AD[8], AD[16] and AD[24]) is the first HDLC bit transmitted and the most significant bit of each byte (AD[7], AD[15], AD[23] and AD[31]) is the last HDLC bit transmitted (datacom standard). When TSTD is set high, AD[0], AD[8], AD[16] and AD[24] are the last HDLC bit transmitted and AD[7], AD[15], AD[23] and AD[31] are the first HDLC bit transmitted (telecom standard).

BIT8:

The least significant stuff control bit (BIT8) carries the value placed in the least significant bit of each octet when the HDLC processor is configured (7BIT set high) to stuff the least significant bit of each octet in the corresponding transmit link (TD[n]). When BIT8 is set high, the least significant bit (last bit of each octet transmitted) is forced high. When BIT8 is set low, the least significant bit is forced low. BIT8 is ignored when 7BIT is set low.

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

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	BUSY	X
Bit 14	R/W	RWB	0
Bit 13		Unused	X
Bit 12	R/W	LINK[4]	0
Bit 11	R/W	LINK[3]	0
Bit 10	R/W	LINK[2]	0
Bit 9	R/W	LINK[1]	0
Bit 8	R/W	LINK[0]	0
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4	R/W	TSLOT[4]	0
Bit 3	R/W	TSLOT[3]	0
Bit 2	R/W	TSLOT[2]	0
Bit 1	R/W	TSLOT[1]	0
Bit 0	R/W	TSLOT[0]	0

This register provides the link number and time-slot number used to access the transmit channel provision RAM. Writing to this register triggers an indirect register access and transfers the contents of the Indirect Channel Data register to an internal holding register.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

TSLOT[4:0]:

The indirect time-slot number bits (TSLOT[4:0]) indicate the time-slot to be configured or interrogated in the indirect access. For a channelised T1 link, time-slots 1 to 24 are valid. For a channelised E1 link, time-slots 1 to 31 are valid. For unchannelised links, only time-slot 0 is valid.

LINK[4:0]:

The indirect link number bits (LINK[4:0]) select amongst the 32 transmit links to be configured or interrogated in the indirect access.

RWB:

The indirect access control bit (RWB) selects between a configure (write) or interrogate (read) access to the transmit channel provision RAM. The address to the transmit channel provision RAM is constructed by concatenating the TSLOT[4:0] and LINK[4:0] bits. Writing a logic zero to RWB triggers an indirect write operation. Data to be written is taken from the PROV and the CHAN[6:0] bits of the Indirect Data register. Writing a logic one to RWB triggers an indirect read operation. Addressing of the RAM is the same as in an indirect write operation. The data read can be found in the PROV and the CHAN[6:0] bits of the Indirect Channel Data register.

BUSY:

The indirect access status bit (BUSY) reports the progress of an indirect access. BUSY is set high when this register is written to trigger an indirect access, and will stay high until the access is complete. At which point, BUSY will be set low. This register should be polled to determine when data from an indirect read operation is available in the TCAS Indirect Channel Data register or to determine when a new indirect write operation may commence.

Register 0x404 : TCAS Indirect Channel Data

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8	R/W	PROV	0
Bit 7		Unused	X
Bit 6	R/W	CHAN[6]	0
Bit 5	R/W	CHAN[5]	0
Bit 4	R/W	CHAN[4]	0
Bit 3	R/W	CHAN[3]	0
Bit 2	R/W	CHAN[2]	0
Bit 1	R/W	CHAN[1]	0
Bit 0	R/W	CHAN[0]	0

This register contains the data read from the transmit channel provision RAM after an indirect read operation or the data to be inserted into the transmit channel provision RAM in an indirect write operation.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

CHAN[6:0]:

The indirect data bits (CHAN[6:0]) report the channel number read from the transmit channel provision RAM after an indirect read operation has completed. Channel number to be written to the transmit channel provision RAM in an indirect write operation must be set up in this register before triggering the write. CHAN[6:0] reflects the value written until the completion of a subsequent indirect read operation.

PROV:

The indirect provision enable bit (PROV) reports the channel provision enable flag read from transmit channel provision RAM after an indirect read operation has completed. The provision enable flag to be written to the transmit channel provision RAM in an indirect write operation must be set up in this register before triggering the write. When PROV is set high, the current time-slot is assigned to the channel as indicated by CHAN[6:0]. When PROV is set low, the time-slot does not belong to any channel. The transmit link data is set to the contents of the Idle Time-slot Fill Data register. PROV reflects the value written until the completion of a subsequent indirect read operation.

Register 0x408 : TCAS Framing Bit Threshold

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6	R/W	FTHRES[6]	0
Bit 5	R/W	FTHRES[5]	0
Bit 4	R/W	FTHRES[4]	1
Bit 3	R/W	FTHRES[3]	1
Bit 2	R/W	FTHRES[2]	1
Bit 1	R/W	FTHRES[1]	1
Bit 0	R/W	FTHRES[0]	1

This register contains the threshold used by the clock activity monitors to detect for framing bits/bytes.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

FTHRES[6:0]:

The framing bit threshold bits (FTHRES[6:0]) contains the threshold used by the clock activity monitor to detect for the presence of framing bits. A counter in the clock activity monitor of each receive link increments at each SYSCLK and is cleared, when the BSYNC bit of that link is set low, by each rising edge of the corresponding TCLK[n]. When the BSYNC bit of that link is set high, the counter is cleared at every fourth rising edge of the corresponding TCLK[n]. When the counter exceeds the threshold given by FTHRES[6:0], a framing bit/byte has been detected. FTHRES[6:0] should be set as a function of the SYSCLK period and the expected gapping width of TCLK[n] during data bits and during framing bits/bytes. Legal range of FTHRES[6:0] is 'b0000001 to 'b1111110.

Register 0x40C : TCAS Idle Time-slot Fill Data

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7	R/W	FDATA[7]	1
Bit 6	R/W	FDATA[6]	1
Bit 5	R/W	FDATA[5]	1
Bit 4	R/W	FDATA[4]	1
Bit 3	R/W	FDATA[3]	1
Bit 2	R/W	FDATA[2]	1
Bit 1	R/W	FDATA[1]	1
Bit 0	R/W	FDATA[0]	1

This register contains the data to be written to disabled time-slots of a channelised link.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

FDATA[7:0]:

The fill data bits (FDATA[7:0]) are transmitted during disabled (PROV set low) time-slots of channelised links.

Register 0x410 : TCAS Channel Disable

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R/W	CHDIS	0
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6	R/W	DCHAN[6]	0
Bit 5	R/W	DCHAN[5]	0
Bit 4	R/W	DCHAN[4]	0
Bit 3	R/W	DCHAN[3]	0
Bit 2	R/W	DCHAN[2]	0
Bit 1	R/W	DCHAN[1]	0
Bit 0	R/W	DCHAN[0]	0

This register controls the disabling of one specific channel to allow orderly provisioning of timeslots.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

DCHAN[6:0]:

The disable channel number bits (DCHAN[6:0]) selects the channel to be disabled. When CHDIS is set high, the channel specified by DCHAN[6:0] is disabled. Data in timeslots associated with the specified channel is set to FDATA[7:0] in the Idle Time-slot Fill Data register. When CHDIS is set low, the channel specified by DCHAN[6:0] operates normally.

CHDIS:

The channel disable bit (CHDIS) controls the disabling of the channels specified by DCHAN[6:0]. When CHDIS is set high, the channel selected by DCHAN[6:0] is disabled. Data in timeslots associated with the specified channel is set to FDATA[7:0] in the Idle Time-slot Fill Data register. When CHDIS is set low, the channel specified by DCHAN[6:0] operates normally.

Register 0x480 : TCAS Link #0 Configuration

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3		Unused	X
Bit 2	R/W	BSYNC	0
Bit 1	R/W	E1	0
Bit 0	R/W	CEN	0

This register configures operational modes of transmit link #0 (TD[0] / TCLK[0]).

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

CEN:

The channelise enable bit (CEN) configures transmit link #0 for channelised operation. TCLK[0] is held quiescent during the T1 framing bit or the E1

framing byte. Thus, on the first rising edge of TCLK[0] after the extended quiescent period, a downstream block can sample the m.s.b. of time-slot 1. When CEN is set low, link #0 is unchannelised and the E1 register bit is ignored. TCLK[0] is gapped during non-data bytes. The choice between treating all data bits as a contiguous stream with arbitrary byte alignment or byte aligned to gaps in TCLK[0] is controlled by the BSYNC bit.

E1:

The E1 frame structure select bit (E1) configures link #0 for channelised E1 operation when CEN is set high. TCLK[0] is held quiescent during the FAS and NFAS framing bytes. The most significant bit of time-slot 1 is placed on TD[0] on the last falling edge of TCLK[0] ahead of the extended quiescent period. Link data is present at time-slots 1 to 31. When E1 is set low and CEN is set high, link #0 is configured for channelised T1 operation. TCLK[0] is held quiescent during the framing bit. The m.s.b. of time-slot 1 is placed on TD[0] on the last falling edge of TCLK[0] ahead of the extended quiescent period. Link data is present at time-slots 1 to 24. E1 is ignored when CEN is set low.

BSYNC:

The byte synchronisation enable bit (BSYNC) controls the interpretation of gaps in TCLK[0] when link #0 is in unchannelised mode (CEN set low). When BSYNC is set high, the data bit on TD[0] clocked in by a downstream device on the first rising edge of TCLK[0] after an extended quiescent period is considered to be the most significant bit of a data byte. When BSYNC is set quiescent , gaps in TCLK[0] carry no special significance. BSYNC is ignored when CEN is set high.

Register 0x484-0x488 : TCAS Link #1 to Link #2 Configuration

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3		Unused	X
Bit 2	R/W	BSYNC	0
Bit 1	R/W	E1	0
Bit 0	R/W	CEN	0

This register set configures operational modes of transmit link #1 to link # 2 (TD[n] / TCLK[n]; where $1 \leq n \leq 2$).

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

CEN:

The channelise enable bit (CEN) configures the corresponding transmit link for channelised operation. TCLK[n] is held quiescent during the T1 framing bit or the E1 framing byte. Thus, on the first rising edge of TCLK[n] after the extended quiescent period, a downstream block can sample the m.s.b. of time-slot 1. When CEN is set low, the corresponding link is unchannelised and the E1 register bit is ignored. TCLK[n] is gapped during non-data bytes. The choice between treating all data bits as a contiguous stream with arbitrary byte alignment or byte aligned to gaps in TCLK[n] is controlled by the BSYNC bit.

E1:

The E1 frame structure select bit (E1) configures the corresponding link for channelised E1 operation when CEN is set high. TCLK[n] is held quiescent during the FAS and NFAS framing bytes. The most significant bit of time-slot 1 is placed on TD[n] on the last falling edge of TCLK[n] ahead of the extended quiescent period. Link data is present at time-slots 1 to 31. When E1 is set low and CEN is set high, the corresponding link is configured for channelised T1 operation. TCLK[n] is held quiescent during the framing bit. The m.s.b. of time-slot 1 is placed on TD[n] on the last falling edge of TCLK[n] ahead of the extended quiescent period. Link data is present at time-slots 1 to 24. E1 is ignored when CEN is set low.

BSYNC:

The byte synchronisation enable bit (BSYNC) controls the interpretation of gaps in TCLK[n] when the corresponding link is in unchannelised mode (CEN set low). When BSYNC is set high, the data bit on TD[n] clocked in by a downstream device on the first rising edge of TCLK[n] after an extended quiescent period is considered to be the most significant bit of a data byte. When BSYNC is set low, gaps in TCLK[n] carry no special significance. BSYNC is ignored when CEN is set high.

Register 0x48C : TCAS Link #3 Configuration

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3		Unused	X
Bit 2		Unused	0
Bit 1	R/W	E1	0
Bit 0	R/W	CEN	0

This register set configures operational modes of transmit link #3.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

CEN:

The channelise enable bit (CEN) configures transmit link #3 for channelised operation. TCLK[3] is held quiescent during the T1 framing bit or the E1

framing byte. Thus, on the first rising edge of TCLK[3] after the extended quiescent period, a downstream device can sample the m.s.b. of time-slot 1. When CEN is set low, link #3 is unchannelised and the E1 register bit is ignored. TCLK[3] is gapped during non-data bytes. All data bits are treated as a contiguous stream with arbitrary byte alignment.

E1:

The E1 frame structure select bit (E1) configures link #3 for channelised E1 operation when CEN is set high. TCLK[3] is held quiescent during the FAS and NFAS framing bytes. The most significant bit of time-slot 1 is placed on TD[3] on the last falling edge of TCLK[3] ahead of the extended quiescent period. Link data is present at time-slots 1 to 31. When E1 is set low and CEN is set high, link #3 is configured for channelised T1 operation. TCLK[3] is held quiescent during the framing bit. The m.s.b. of time-slot 1 is placed on TD[3] on the last falling edge of TCLK[3] ahead of the extended quiescent period. Link data is present at time-slots 1 to 24. E1 is ignored when CEN is set low.

Register 0x490-0x4FC : TCAS Link #4 to Link #31 Configuration

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5		Unused	X
Bit 4		Unused	X
Bit 3		Unused	X
Bit 2		Unused	0
Bit 1	R/W	E1	0
Bit 0	R/W	CEN	0

This register set configures operational modes of transmit link #4 to link # 31 (TD[n] / TCLK[n]; where $4 \leq n \leq 31$).

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

CEN:

The channelise enable bit (CEN) configures the corresponding transmit link for channelised operation. TCLK[n] is held quiescent during the T1 framing bit or the E1 framing byte. Thus, on the first rising edge of TCLK[n] after the extended quiescent period, a downstream device can sample the m.s.b. of time-slot 1. When CEN is set low, the corresponding link is unchannelised and the E1 register bit is ignored. TCLK[n] is gapped during non-data bytes. All data bits are treated as a contiguous stream with arbitrary byte alignment.

E1:

The E1 frame structure select bit (E1) configures the corresponding link for channelised E1 operation when CEN is set high. TCLK[n] is held quiescent during the FAS and NFAS framing bytes. The most significant bit of time-slot 1 is placed on TD[n] on the last falling edge of TCLK[n] ahead of the extended quiescent period. Link data is present at time-slots 1 to 31. When E1 is set low and CEN is set high, the corresponding link is configured for channelised T1 operation. TCLK[n] is held quiescent during the framing bit. The m.s.b. of time-slot 1 is placed on TD[n] on the last falling edge of TCLK[n] ahead of the extended low period. Link data is present at time-slots 1 to 24. E1 is ignored when CEN is set low.

Register 0x500 : PMON Status

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15		Unused	X
Bit 14		Unused	X
Bit 13		Unused	X
Bit 12		Unused	X
Bit 11		Unused	X
Bit 10		Unused	X
Bit 9		Unused	X
Bit 8		Unused	X
Bit 7		Unused	X
Bit 6		Unused	X
Bit 5	R	C2DET	X
Bit 4	R	C1DET	X
Bit 3	R	UFDET	X
Bit 2	R	OFDET	X
Bit 1		Unused	X
Bit 0		Unused	X

This register contains status information indicating whether a non-zero count has been latched in the count registers.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

OFDET:

The overflow detect bit (OFDET) indicates the status of the PMON Receive FIFO Overflow Count register. OFDET is set high when overflow events have occurred during the latest PMON accumulation interval. OFDET is set low if no overflow events are detected.

UFDET:

The underflow detect bit (UFDET) indicates the status of the PMON Transmit FIFO Underflow Count register. UFDET is set high when underflow events have occurred during the latest PMON accumulation interval. UFDET is set low if no underflow events are detected.

C1DET:

The configurable event #1 detect bit (C1DET) indicates the status of the PMON Configurable Count #1 register. C1DET is set high when selected events have occurred during the latest PMON accumulation interval. C1DET is set low if no selected events are detected.

C2DET:

The configurable event #2 detect bit (C2DET) indicates the status of the PMON Configurable Count #2 register. C2DET is set high when selected events have occurred during the latest PMON accumulation interval. C2DET is set low if no selected events are detected.

Register 0x504 : PMON Receive FIFO Overflow Count

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	OF[15]	X
Bit 14	R	OF[14]	X
Bit 13	R	OF[13]	X
Bit 12	R	OF[12]	X
Bit 11	R	OF[11]	X
Bit 10	R	OF[10]	X
Bit 9	R	OF[9]	X
Bit 8	R	OF[8]	X
Bit 7	R	OF[7]	X
Bit 6	R	OF[6]	X
Bit 5	R	OF[5]	X
Bit 4	R	OF[4]	X
Bit 3	R	OF[3]	X
Bit 2	R	OF[2]	X
Bit 1	R	OF[1]	X
Bit 0	R	OF[0]	X

This register reports the number of receive FIFO overflow events in the previous accumulation interval.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

OF[15:0]:

The OF[15:0] bits reports the number of receive FIFO overflow events that have been detected since the last time this register was polled. This register is polled by writing to the FREEDM-32 Master Clock / BERT Activity Monitor and Accumulation Trigger register. The write access transfers the internally accumulated error count to the FIFO overflow register and simultaneously resets the internal counter to begin a new cycle of error accumulation.

Register 0x508 : PMON Receive FIFO Underflow Count

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	UF[15]	X
Bit 14	R	UF[14]	X
Bit 13	R	UF[13]	X
Bit 12	R	UF[12]	X
Bit 11	R	UF[11]	X
Bit 10	R	UF[10]	X
Bit 9	R	UF[9]	X
Bit 8	R	UF[8]	X
Bit 7	R	UF[7]	X
Bit 6	R	UF[6]	X
Bit 5	R	UF[5]	X
Bit 4	R	UF[4]	X
Bit 3	R	UF[3]	X
Bit 2	R	UF[2]	X
Bit 1	R	UF[1]	X
Bit 0	R	UF[0]	X

This register reports the number of transmit FIFO underflow events in the previous accumulation interval.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

UF[15:0]:

The UF[15:0] bits reports the number of transmit FIFO underflow events that have been detected since the last time this register was polled. This register is polled by writing to the FREEDM-32 Master Clock / BERT Activity Monitor and Accumulation Trigger register. The write access transfers the internally accumulated error count to the FIFO underflow register and simultaneously resets the internal counter to begin a new cycle of error accumulation.

Register 0x50C : PMON Configurable Count #1

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	C1[15]	X
Bit 14	R	C1[14]	X
Bit 13	R	C1[13]	X
Bit 12	R	C1[12]	X
Bit 11	R	C1[11]	X
Bit 10	R	C1[10]	X
Bit 9	R	C1[9]	X
Bit 8	R	C1[8]	X
Bit 7	R	C1[7]	X
Bit 6	R	C1[6]	X
Bit 5	R	C1[5]	X
Bit 4	R	C1[4]	X
Bit 3	R	C1[3]	X
Bit 2	R	C1[2]	X
Bit 1	R	C1[1]	X
Bit 0	R	C1[0]	X

This register reports the number events, selected by the FREEDM-32 Master Performance Monitor Control register, that occurred in the previous accumulation interval.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

C1[15:0]:

The C1[15:0] bits reports the number of selected events that have been detected since the last time this register was polled. This register is polled by writing to the FREEDM-32 Master Clock / BERT Activity Monitor and Accumulation Trigger register. The write access transfers the internally accumulated error count to the configurable count #1 register and simultaneously resets the internal counter to begin a new cycle of event accumulation.

Register 0x510 : PMON Configurable Count #2

Bit	Type	Function	Default
Bit 31 to Bit 16		Unused	XXXXH
Bit 15	R	C2[15]	X
Bit 14	R	C2[14]	X
Bit 13	R	C2[13]	X
Bit 12	R	C2[12]	X
Bit 11	R	C2[11]	X
Bit 10	R	C2[10]	X
Bit 9	R	C2[9]	X
Bit 8	R	C2[8]	X
Bit 7	R	C2[7]	X
Bit 6	R	C2[6]	X
Bit 5	R	C2[5]	X
Bit 4	R	C2[4]	X
Bit 3	R	C2[3]	X
Bit 2	R	C2[2]	X
Bit 1	R	C2[1]	X
Bit 0	R	C2[0]	X

This register reports the number events, selected by the FREEDM-32 Master Performance Monitor Control register, that occurred in the previous accumulation interval.

Note

This register is not byte addressable. Writing to this register modifies all the bits in the register. Byte selection using byte enable signals (CBEB[3:0]) are not implemented. However, when all four byte enables are negated, no access is made to this register.

C2[15:0]:

The C2[15:0] bits reports the number of selected events that have been detected since the last time this register was polled. This register is polled by writing to the FREEDM-32 Master Clock / BERT Activity Monitor and Accumulation Trigger register. The write access transfers the internally accumulated error count to the configurable count #2 register and simultaneously resets the internal counter to begin a new cycle of event accumulation.

11 PCI CONFIGURATION REGISTER DESCRIPTION

PCI configuration registers are implemented by the PCI Interface. These registers can only be accessed when the PCI Interface is a target and a configuration cycle is in progress as indicated using the IDSEL input.

Notes on PCI Configuration Register Bits:

1. Writing values into unused register bits has no effect. However, to ensure software compatibility with future, feature-enhanced versions of the product, unused register bits must 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.

2. Except where noted, all configuration bits that can be written into can also be read back. This allows the processor controlling the FREEDM-32 to determine the programming state of the block.

3. Writable PCI configuration register bits are cleared to logic zero upon reset unless otherwise noted.

4. Writing into read-only PCI configuration register bit locations does not affect FREEDM-32 operation unless otherwise noted.

5. Certain register bits are reserved. These bits are associated with megacell functions that are unused in this application. To ensure that the FREEDM-32 operates as intended, reserved register bits must only be written with their default values. Similarly, writing to reserved registers should be avoided.

11.1 PCI Configuration Registers

PCI configuration registers can only be accessed by the PCI host. For each register description below, the hexadecimal register number indicates the PCI offset.

Register 0x00 : Vendor Identification/Device Identification

Bit	Type	Function	Default
Bit 31 to Bit 16	R	DEVID[15:0]	7364H
Bit 15 to Bit 0	R	VNDRID[15:0]	11F8H

VNDRID[15:0]:

The VNDRID[15:0] bits identifies the manufacturer of the device. Valid vendor identifiers are allocated by the PCI SIG.

DEVID[15:0]:

The DEVID[15:0] bits define the particular device. Valid device identifiers will be specified by PMC-Sierra. The default value of DEVID[15:0] is that of the FREEDM-32 device.

Register 0x04 : Command/Status

Bit	Type	Function	Default
Bit 31	R/W	PERR	0
Bit 30	R/W	SERR	0
Bit 29	R/W	MABT	0
Bit 28	R/W	RTABT	0
Bit 27	R/W	TABT	0
Bit 26	R	DVSLT[1]	0
Bit 25	R	DVSLT[0]	1
Bit 24	R/W	DPR	0
Bit 23	R	FBTBE	1
Bit 22 to Bit 16	R	Reserved	00H
Bit 15 to Bit 10	R	Reserved	00H
Bit 9	R	FBTBEN	0
Bit 8	R/W	SERREN	0
Bit 7	R	ADSTP	0
Bit 6	R/W	PERREN	0
Bit 5	R	VGASNP	0
Bit 4	R	MWAI	0
Bit 3	R	SPCEN	0
Bit 2	R/W	MSTREN	0
Bit 1	R/W	MCNTRL	0
Bit 0	R	IOCNTRL	0

The lower 16 bits of this register make up the Command register which provides basic control over the GPIC's ability to respond to PCI accesses. When a 0 is written to all bits in the command register, the GPIC is logically disconnected from the PCI bus for all accesses except configuration accesses. The upper 16-

bits is used to record status information for PCI bus related events. Reads to the status portion of this register behave normally. Writes are slightly different in that bits can be reset, but not set. A bit is reset whenever the register is written, and the data in the corresponding bit location is a 1.

IOCNTRL:

When IOCNTRL is set to zero, the GPIC will not respond to PCI bus I/O accesses.

MCNTRL:

When MCNTRL is set to one, the GPIC will respond to PCI bus memory accesses. Clearing MCNTRL disables memory accesses.

MSTREN:

When MSTREN is set to one, the GPIC can act as a Master. Clearing MSTREN disables the GPIC from becoming a Master.

SPCEN:

The GPIC does not decode PCI special cycles. The SPCEN bit is forced low.

MWAI:

The GPIC does not generate memory-write-and-invalidate commands. The MWAI bit is forced low.

VGASNP:

The GPIC is not a VGA device. The VGASNP bit is forced low.

PERREN:

When the PERREN bit is set to one, the GPIC can report parity errors. Clearing the PERREN bit causes the GPIC to ignore parity errors.

ADSTP:

The GPIC does not perform address and data stepping. The ADSTP bit is forced low.

SERREN:

When the SERREN bit is set high, the GPIC can drive the SERRB line. Clearing the SERREN bit disables the SERRB line. SERREN and PERREN must be set to report an address parity error.

FBTBEN:

As a master, the GPIC does not generate fast back-to-back cycles to different devices. This bit is forced low.

The upper 16-bits make up the PCI Status field. The status field tracks the status of PCI bus related events. Reads to this register behave normally. Writes are slightly different in that bits can be reset, but not set. A bit is reset whenever the register is written, and the data in the corresponding bit location is a one.

FBTBE:

The FBTBE bit is hardwired to one to indicate the GPIC supports fast back-to-back transactions with other targets.

DPR:

The Data Parity Reported (DPR) bit is set high if the GPIC is an initiator and asserts or detects a parity error on the PERRB signal while the PERREN bit is set in the Command register. The DPR bit is cleared by the PCI Host.

DVSLT[1:0]:

The Device Select Timing (DEVSLT) bits specify the allowable timings for the assertion of DEVSELB by the GPIC as a target. These are encoded as 00B for fast, 01B for medium, 10B for slow and 11B is not used. The GPIC allows for medium timing.

TABT:

The Target Abort (TABT) bit is set high by the GPIC when as a target, it terminates a transaction with a target abort. The TABT bit is cleared by the PCI Host.

RTABT:

The Received Target Abort (RTABT) bit is set high by the GPIC when as an initiator, its transaction is terminated by a target abort. The RTABT bit is cleared by the PCI Host.

MABT:

The Master Abort (MABT) bit is set high by the GPIC when as an initiator, its transaction is terminated by a master abort and a special cycle was not in progress. The MABT bit is cleared by the PCI Host.

SERR:

The System Error (SERR) bit is set high whenever the GPIC asserts the SERRB output. The SERR bit is cleared by the PCI Host.

PERR:

The Parity Error (PERR) bit is set high whenever the GPIC detects a parity error, even if parity error handling is disabled by clearing PERREN in the Command register. The PERR bit is cleared by the PCI Host.

Register 0x08 : Revision Identifier/Class Code

Bit	Type	Function	Default
Bit 31 to Bit 24	R	CCODE[23:16]	02H
Bit 23 to Bit 16	R	CCODE[15:8]	80H
Bit 15 to Bit 8	R	CCODE[7:0]	00H
Bit 7 to Bit 0	R	REVID[7:0]	01H

REVID[7:0]:

The Revision Identifier (REVID[7:0]) bits specify a device specific revision identifier and are chosen by PMC-Sierra.

CCODE[23:0]:

The class code (CCODE[23:0]) bits are divided into three groupings: CCODE[23:16] define the base class of the device, CCODE[15:8] define the sub-class of the device and CCODE[7:0] specify a register specific programming interface.

Note:

Base Class Code:	02H	Network Controller
Sub-Class Code:	80H	Other Controllers
Register Class Code:	00H	None defined.

Register 0x0C : Cache Line Size/Latency Timer/Header Type

Bit	Type	Function	Default
Bit 31 to Bit 24	R	Reserved	00H
Bit 23	R	MLTFNC	0
Bit 22 to Bit 16	R	HDTYPE[6:0]	00H
Bit 15 to Bit 8	R/W	LT[7:0]	00H
Bit 7 to Bit 0	R/W	CLSIZE	00H

CLSIZE[7:0]:

The Cache Line Size (CLSIZE[7:0]) bits specify the size of the system cacheline in units of dwords. The GPIC uses this value to determine the type of read command to issue in a Master Read transfer. If the transfer size is equal to one, the GPIC will issue a Memory Read command. If the transfer size is equal to or less than the CLSIZE, the GPIC will issue a Memory Read Line command. For transfers larger than CLSIZE, the GPIC issues a Memory Read Multiple command.

LT[7:0]:

The Latency Timer (LT[7:0]) bits specify in units of the PCI clock, the value of the Latency Timer for the GPIC. At reset the value is zero.

HDTYPE[6:0]:

The Header Type (HDTYPE[7:0]) bits specify the layout of the base address registers. Only the 00H encoding is supported.

MLTFNC:

The Multi-Function (MLTFNC) bit specifies if the GPIC supports multiple PCI functions. If this bit is set low, the device only supports one function and if the bit is set high, the device supports multi-functions. The MLTFNC bit is set low to indicate the GPIC only supports one PCI function.

Register 0x10 : CBI Memory Base Address Register

Bit	Type	Function	Default
Bit 31 to Bit 12	R/W	BSAD[27:8]	00000H
Bit 11 to Bit 4	R	BSAD[7:0]	00H
Bit 3	R	PRFTCH	0
Bit 2	R	TYPE[1]	0
Bit 1	R	TYPE[0]	0
Bit 0	R	MSI	0

The GPIC supports memory mapping only. At boot-up the internal registers space is mapped to memory space. The device driver can disable memory space through the PCI Configuration Command register.

MSI:

MSI is forced low to indicate that the internal registers map into memory space.

TYPE[1:0]:

The TYPE field indicates where the internal registers can be mapped. The encoding 00B indicates the registers may be located anywhere in the 32 bit address space, 01B indicates that the registers must be mapped below 1 Meg in memory space, 10B indicates the base register is 64 bits and the encoding 11B is reserved.

The TYPE field is set to 00B to indicate that the CBI registers can be mapped anywhere in the 32 bit address space.

PRFTCH:

The Prefetchable (PRFTCH) bit is set if there are no side effects on reads and data is returned on all the lanes regardless of the byte enables. Otherwise the bit is cleared. TSBs contain registers, such as interrupt status registers, in which bits are cleared on a read. If the PCI Host is caching data there is a possibility an interrupt status could be lost if data is prefetched, but the cache is flushed and the data is not used. The PRFTCH bit is forced low to indicate that prefetching of data is not supported for internal registers.

BSAD[27:0]:

The Base Address (BSAD[27:0]) bits defines the size and location of the memory space required for the CBI registers. The BSAD[27:0] bits correspond to the most significant 28 bits of the PCI address space.

The size of the address space required can be determined by writing all ones to Base Address register and then reading from it. By scanning the returned value from the least significant bit upwards, the size of the required address space can be determined. The binary weighted value of the first one bit found (after the configuration bits) indicates the required amount of space. The BSAD[7:0] bits are forced low to indicate that the CBI registers require 4K bytes of memory space.

After determining the memory requirements of the CBI registers, the PCI Host can map them to its desired location by modifying the BSAD[27:8] bits in the Base Address register.

Register 0x3C : Interrupt Line / Interrupt Pin / MIN_GNT / MAX_LAT

Bit	Type	Function	Default
Bit 31 to Bit 24	R	MAXLAT[7:0]	0FH
Bit 23 to Bit 16	R	MINGNT[7:0]	05H
Bit 15 to Bit 8	R	INTPIN[7:0]	01H
Bit 7 to Bit 0	R/W	INTLNE[7:0]	00H

INTLNE[7:0]:

The Interrupt Line (INTLNE[7:0]) field is used to indicate interrupt line routing information. The values in this register are system specific and set by the PCI Host.

INTPIN[7:0]:

The Interrupt Pin (INTPIN[7:0]) field is used to specify the interrupt pin the GPIC uses. Since the GPIC will use INTAB on the PCI bus, the value in this register is set to one.

MINGNT[7:0]:

The Minimum Grant (MINGNT[7:0]) field specifies how long of a burst period the bus master needs (in increments of 250 nsec).

MAXLAT[7:0]:

The Maximum Latency (MAXLAT[7:0]) field specifies how often a bus master needs access to the PCI bus (in increments of 250 nsec).

12 TEST FEATURES DESCRIPTION

The FREEDM-32 also supports a standard IEEE 1149.1 five signal JTAG boundary scan test port for use in board testing. All device inputs may be read and all device outputs may be forced via the JTAG test port.

12.1 Test Mode Registers

Test mode registers are used to apply test vectors during production testing of the FREEDM. Production testing is enabled by asserting the PMCTEST pin. During production tests, FREEDM-32 registers are selected by the TA[11:0] pins. The address of a register on TA[11:0] is identical to the PCI offset of that register when production testing is disabled (PMCTEST low). Read accesses are enabled by asserting TRDB low while write accesses are enabled by asserting TWRB low. Test mode register data is conveyed on the TDAT[15:0] pins. Test mode registers (as opposed to normal mode registers) are selected when TA[11]/TRS is set high.

Table 25 – Test Mode Register Memory Map

Address TA[10:0]	Register
0x000 - 0x7FC	Normal Mode Registers
0x800 - 0x83C	Reserved
0x840 - 0x87C	GPIC Test Registers
0x880 - 0x8FC	Reserved
0x900 - 0x9FC	RCAS Test Registers
0xA00 - 0xA3C	RHDL Test Registers
0xA40 - 0xA7C	Reserved
0xA80 - 0xAFC	RMAC Test Registers
0xB00 - 0xB7C	TMAC Test Registers
0xB80 - 0xBBC	THDL Test Registers
0xBC0 - 0xBFF	Reserved
0xC00 - 0xCFC	TCAS Test Registers
0xD00 - 0xD1C	PMON Test Registers
0xD20 - 0xFFF	Reserved

Notes on Test Mode Register Bits:

1. Writing values into unused register bits has no effect. However, to ensure software compatibility with future, feature-enhanced versions of the product, unused register bits must 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.

2. Writable test mode register bits are not initialized upon reset unless otherwise noted.

12.2 JTAG Test Port

The FREEDM-32 JTAG Test Access Port (TAP) allows access to the TAP controller and the 4 TAP registers: instruction, bypass, device identification and boundary scan. Using the TAP, device input logic levels can be read, device outputs can be forced, the device can be identified and the device scan path can

be bypassed. For more details on the JTAG port, please refer to the Operations section.

Table 26 – Instruction Register

Length - 3 bits

Instructions	Selected Register	Instruction Code IR[2:0]
EXTEST	Boundary Scan	000
IDCODE	Identification	001
SAMPLE	Boundary Scan	010
BYPASS	Bypass	011
BYPASS	Bypass	100
STCTEST	Boundary Scan	101
BYPASS	Bypass	110
BYPASS	Bypass	111

12.2.1 Identification Register

Length - 32 bits

Version number - 1H

Part Number - 7364H

Manufacturer's identification code - 0CDH

Device identification - 173640CDH

12.2.2 Boundary Scan Register

The boundary scan register is made up of 268 boundary scan cells, divided into input observation (in_cell), output (out_cell), and bi-directional (io_cell) cells. These cells are detailed in the pages which follow. The first 32 cells form the ID code register, and carry the code 173640CDH. The cells are arranged as follows:

Table 27 – Boundary Scan Chain

Pin/ Enable	Register Bit	Cell Type	Device I.D.
RD[0]	267	IN_CELL	0
RCLK[0]	266	IN_CELL	0
RD[1]	265	IN_CELL	0
RCLK[1]	264	IN_CELL	1
RD[2]	263	IN_CELL	0
RCLK[2]	262	IN_CELL	1
RD[3]	261	IN_CELL	1
RCLK[3]	260	IN_CELL	1
RD[4]	259	IN_CELL	0
RCLK[4]	258	IN_CELL	0
RD[5]	257	IN_CELL	1
RCLK[5]	256	IN_CELL	1
RD[6]	255	IN_CELL	0
RCLK[6]	254	IN_CELL	1
RD[7]	253	IN_CELL	1
RCLK[7]	252	IN_CELL	0
RD[8]	251	IN_CELL	0
RCLK[8]	250	IN_CELL	1
RD[9]	249	IN_CELL	0
RCLK[9]	248	IN_CELL	0
RD[10]	247	IN_CELL	0
RCLK[10]	246	IN_CELL	0
RD[11]	245	IN_CELL	0
RCLK[11]	244	IN_CELL	0
RD[12]	243	IN_CELL	1
RCLK[12]	242	IN_CELL	1
RD[13]	241	IN_CELL	0
RCLK[13]	240	IN_CELL	0

Pin/ Enable	Register Bit	Cell Type	Device I.D.
RD[14]	239	IN_CELL	1
RCLK[14]	238	IN_CELL	1
RD[15]	237	IN_CELL	0
RCLK[15]	236	IN_CELL	1
RD[16]	235	IN_CELL	-
RCLK[16]	234	IN_CELL	-
RD[17]	233	IN_CELL	-
RCLK[17]	232	IN_CELL	-
RD[18]	231	IN_CELL	-
RCLK[18]	230	IN_CELL	-
RD[19]	229	IN_CELL	-
RCLK[19]	228	IN_CELL	-
RD[20]	227	IN_CELL	-
RCLK[20]	226	IN_CELL	-
RD[21]	225	IN_CELL	-
RCLK[21]	224	IN_CELL	-
RD[22]	223	IN_CELL	-
RCLK[22]	222	IN_CELL	-
RD[23]	221	IN_CELL	-
RCLK[23]	220	IN_CELL	-
RD[24]	219	IN_CELL	-
RCLK[24]	218	IN_CELL	-
RD[25]	217	IN_CELL	-
RCLK[25]	216	IN_CELL	-
RD[26]	215	IN_CELL	-
RCLK[26]	214	IN_CELL	-
RD[27]	213	IN_CELL	-
RCLK[27]	212	IN_CELL	-
RD[28]	211	IN_CELL	-

Pin/ Enable	Register Bit	Cell Type	Device I.D.
RCLK[28]	210	IN_CELL	-
RD[29]	209	IN_CELL	-
RCLK[29]	208	IN_CELL	-
PCICLKO	207	OUT_CELL	-
PCICLKO_OEN	206	OUT_CELL	-
PCICLK	205	IN_CELL	-
GNTB	204	IN_CELL	-
REQB[0]	203	OUT_CELL	-
REQB_OEN[0]	202	OUT_CELL	-
AD[31]	201	IO_CELL	-
AD_OEN[31]	200	OUT_CELL	-
AD[30]	199	IO_CELL	-
AD_OEN[30]	198	OUT_CELL	-
AD[29]	197	IO_CELL	-
AD_OEN[29]	196	OUT_CELL	-
AD[28]	195	IO_CELL	-
AD_OEN[28]	194	OUT_CELL	-
AD[27]	193	IO_CELL	-
AD_OEN[27]	192	OUT_CELL	-
AD[26]	191	IO_CELL	-
AD_OEN[26]	190	OUT_CELL	-
AD[25]	189	IO_CELL	-
AD_OEN[25]	188	OUT_CELL	-
AD[24]	187	IO_CELL	-
AD_OEN[24]	186	OUT_CELL	-
CBEB[3]	185	IO_CELL	-
CBEB_OEN[3]	184	OUT_CELL	-
IDSEL	183	IN_CELL	-
AD[23]	182	IO_CELL	-

Pin/ Enable	Register Bit	Cell Type	Device I.D.
AD_OEN[23]	181	OUT_CELL	-
AD[22]	180	IO_CELL	-
AD_OEN[22]	179	OUT_CELL	-
AD[21]	178	IO_CELL	-
AD_OEN[21]	177	OUT_CELL	-
AD[20]	176	IO_CELL	-
AD_OEN[20]	175	OUT_CELL	-
AD[19]	174	IO_CELL	-
AD_OEN[19]	173	OUT_CELL	-
AD[18]	172	IO_CELL	-
AD_OEN[18]	171	OUT_CELL	-
AD[17]	170	IO_CELL	-
AD_OEN[17]	169	OUT_CELL	-
AD[16]	168	IO_CELL	-
AD_OEN[16]	167	OUT_CELL	-
CBEB[2]	166	IO_CELL	-
CBEB[2]_OEN	165	OUT_CELL	-
FREMEB	164	IO_CELL	-
FRAMEB_OEN	163	OUT_CELL	-
IRDYB	162	IO_CELL	-
IRDY_OEN	161	OUT_CELL	-
TRDYB	160	IO_CELL	-
TRDYB_OEN	159	OUT_CELL	-
DEVSELB	158	IO_CELL	-
DEVSELB_OEN	157	OUT_CELL	-
STOPB	156	IO_CELL	-
STOPB_OEN	155	OUT_CELL	-
LOCKB	154	IN_CELL	-
PERRB	153	IO_CELL	-

Pin/ Enable	Register Bit	Cell Type	Device I.D.
PERRB_OEN	152	OUT_CELL	-
SERRB	151	IO_CELL	-
SERRB_OEN	150	OUT_CELL	-
PAR	149	IO_CELL	-
PAR_OEN	148	OUT_CELL	-
CBEB[1]	147	IO_CELL	-
CBEB_OEN[1]	146	OUT_CELL	-
AD[15]	145	IO_CELL	-
AD_OEN[15]	144	OUT_CELL	-
AD[14]	143	IO_CELL	-
AD_OEN[14]	142	OUT_CELL	-
AD[13]	141	IO_CELL	-
AD_OEN[13]	140	OUT_CELL	-
AD[12]	139	IO_CELL	-
AD_OEN[12]	138	OUT_CELL	-
AD[11]	137	IO_CELL	-
AD_OEN[11]	136	OUT_CELL	-
AD[10]	135	IO_CELL	-
AD_OEN[10]	134	OUT_CELL	-
AD[9]	133	IO_CELL	-
AD_OEN[9]	132	OUT_CELL	-
AD[8]	131	IO_CELL	-
AD_OEN[8]	130	OUT_CELL	-
CBEB[0]	129	IO_CELL	-
CBEB_OEN[0]	128	OUT_CELL	-
AD[7]	127	IO_CELL	-
AD_OEN[7]	126	OUT_CELL	-
AD[6]	125	IO_CELL	-
AD_OEN[6]	124	OUT_CELL	-

Pin/ Enable	Register Bit	Cell Type	Device I.D.
AD[5]	123	IO_CELL	-
AD_OEN[5]	122	OUT_CELL	-
AD[4]	121	IO_CELL	-
AD_OEN[4]	120	OUT_CELL	-
AD[3]	119	IO_CELL	-
AD_OEN[3]	118	OUT_CELL	-
AD[2]	117	IO_CELL	-
AD_OEN[2]	116	OUT_CELL	-
AD[1]	115	IO_CELL	-
AD_OEN[1]	114	OUT_CELL	-
AD[0]	113	IO_CELL	-
AD_OEN[0]	112	OUT_CELL	-
RD[30]	111	IN_CELL	-
RCLK[30]	110	IN_CELL	-
RD[31]	109	IN_CELL	-
RCLK[31]	108	IN_CELL	-
PCIINTB	107	OUT_CELL	
PCIINTB_OEN	106	OUT_CELL	-
PMCTEST	105	IN_CELL	-
RSTB	104	IN_CELL	-
TBCLK	103	OUT_CELL	-
TBCLK_OEN	102	OUT_CELL	-
TBD	101	IN_CELL	-
TCLK[31]	100	IN_CELL	-
TD[31]	99	OUT_CELL	-
TD_OEN[31]	98	OUT_CELL	-
TCLK[30]	97	IN_CELL	-
TD[30]	96	OUT_CELL	-
TD_OEN[30]	95	OUT_CELL	-

Pin/ Enable	Register Bit	Cell Type	Device I.D.
TCLK[29]	94	IN_CELL	-
TD[29]	93	OUT_CELL	-
TD_OEN[29]	92	OUT_CELL	-
TCLK[28]	91	IN_CELL	-
TD[28]	90	OUT_CELL	-
TD_OEN[28]	89	OUT_CELL	-
TCLK[27]	88	IN_CELL	-
TD[27]	87	OUT_CELL	-
TD_OEN[27]	86	OUT_CELL	-
TCLK[26]	85	IN_CELL	-
TD[26]	84	OUT_CELL	-
TD_OEN[26]	83	OUT_CELL	-
TCLK[25]	82	IN_CELL	-
TD[25]	81	OUT_CELL	-
TD_OEN[25]	80	OUT_CELL	-
TCLK[24]	79	IN_CELL	-
TD[24]	78	OUT_CELL	-
TD_OEN[24]	77	OUT_CELL	-
TCLK[23]	76	IN_CELL	-
TD[23]	75	OUT_CELL	-
TD_OEN[23]	74	OUT_CELL	-
TCLK[22]	73	IN_CELL	-
TD[22]	72	OUT_CELL	-
TD_OEN[22]	71	OUT_CELL	-
TCLK[21]	70	IN_CELL	-
TD[21]	69	OUT_CELL	-
TD_OEN[21]	68	OUT_CELL	-
TCLK[20]	67	IN_CELL	-
TD[20]	66	OUT_CELL	-

Pin/ Enable	Register Bit	Cell Type	Device I.D.
TD_OEN[20]	65	OUT_CELL	-
TCLK[19]	64	IN_CELL	-
TD[19]	63	OUT_CELL	-
TD_OEN[19]	62	OUT_CELL	-
TCLK[18]	61	IN_CELL	-
TD[18]	60	OUT_CELL	-
TD_OEN[18]	59	OUT_CELL	-
TCLK[17]	58	IN_CELL	-
TD[17]	57	OUT_CELL	-
TD_OEN[17]	56	OUT_CELL	-
TCLK[16]	55	IN_CELL	-
TD[16]	54	OUT_CELL	-
TD_OEN[16]	53	OUT_CELL	-
TCLK[15]	52	IN_CELL	-
TD[15]	51	OUT_CELL	-
TD_OEN[15]	50	OUT_CELL	-
TCLK[14]	49	IN_CELL	-
TD[14]	48	OUT_CELL	-
TD_OEN[14]	47	OUT_CELL	-
TCLK[13]	46	IN_CELL	-
TD[13]	45	OUT_CELL	-
TD_OEN[13]	44	OUT_CELL	-
TCLK[12]	43	IN_CELL	-
TD[12]	42	OUT_CELL	-
TD_OEN[12]	41	OUT_CELL	-
TCLK[11]	40	IN_CELL	-
TD[11]	39	OUT_CELL	-
TD_OEN[11]	38	OUT_CELL	-
TCLK[10]	37	IN_CELL	-

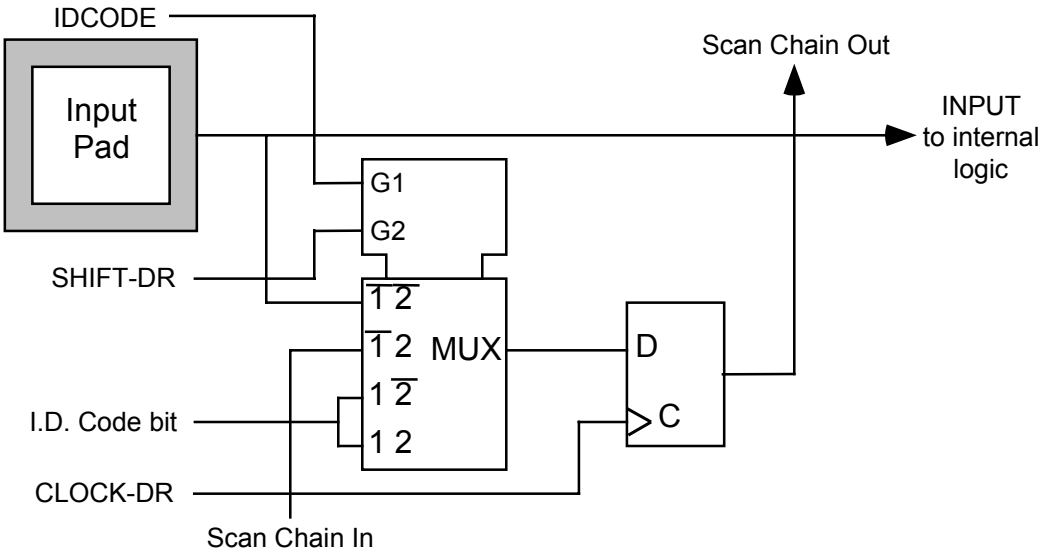
Pin/ Enable	Register Bit	Cell Type	Device I.D.
TD[10]	36	OUT_CELL	-
TD_OEN[10]	35	OUT_CELL	-
TCLK[9]	34	IN_CELL	-
TD[9]	33	OUT_CELL	-
TD_OEN[9]	32	OUT_CELL	-
TCLK[8]	31	IN_CELL	-
TD[8]	30	OUT_CELL	-
TD_OEN[8]	29	OUT_CELL	-
TCLK[7]	28	IN_CELL	-
TD[7]	27	OUT_CELL	-
TD_OEN[7]	26	OUT_CELL	-
TCLK[6]	25	IN_CELL	-
TD[6]	24	OUT_CELL	-
TD_OEN[6]	23	OUT_CELL	-
TCLK[5]	22	IN_CELL	-
TD[5]	21	OUT_CELL	-
TD_OEN[5]	20	OUT_CELL	-
TCLK[4]	19	IN_CELL	-
TD[4]	18	OUT_CELL	-
TD_OEN[4]	17	OUT_CELL	-
TCLK[3]	16	IN_CELL	-
TD[3]	15	OUT_CELL	-
TD_OEN[3]	14	OUT_CELL	-
TCLK[2]	13	IN_CELL	-
TD[2]	12	OUT_CELL	-
TD_OEN[2]	11	OUT_CELL	-
TCLK[1]	10	IN_CELL	-
TD[1]	9	OUT_CELL	-
TD_OEN[1]	8	OUT_CELL	-

Pin/ Enable	Register Bit	Cell Type	Device I.D.
TCLK[0]	7	IN_CELL	-
TD[0]	6	OUT_CELL	-
TD_OEN[0]	5	OUT_CELL	-
TDO		TAP Output	-
TDI		TAP Input	-
TCK		TAP Clock	-
TMS		TAP Input	-
TRSTB		TAP Input	-
SYSCLK	4	IN_CELL	-
RBD	3	OUT_CELL	-
RBD_OEN	2	OUT_CELL	-
RBCLK	1	OUT_CELL	-
RBCLK_OEN	0	OUT_CELL	-

Notes:

1. RD[0] is the first bit of the scan chain (closest to TDI).
2. Enable cell pinname_OEN, tristates pin pinname when set high.

Figure 16 – Input Observation Cell (IN_CELL)



In this diagram and those that follow, CLOCK-DR is equal to TCK when the current controller state is SHIFT-DR or CAPTURE-DR, and unchanging otherwise. The multiplexor in the center of the diagram selects one of four inputs, depending on the status of select lines G1 and G2. The ID Code bit is as listed in the table above.

Figure 17 – Output Cell (OUT_CELL)

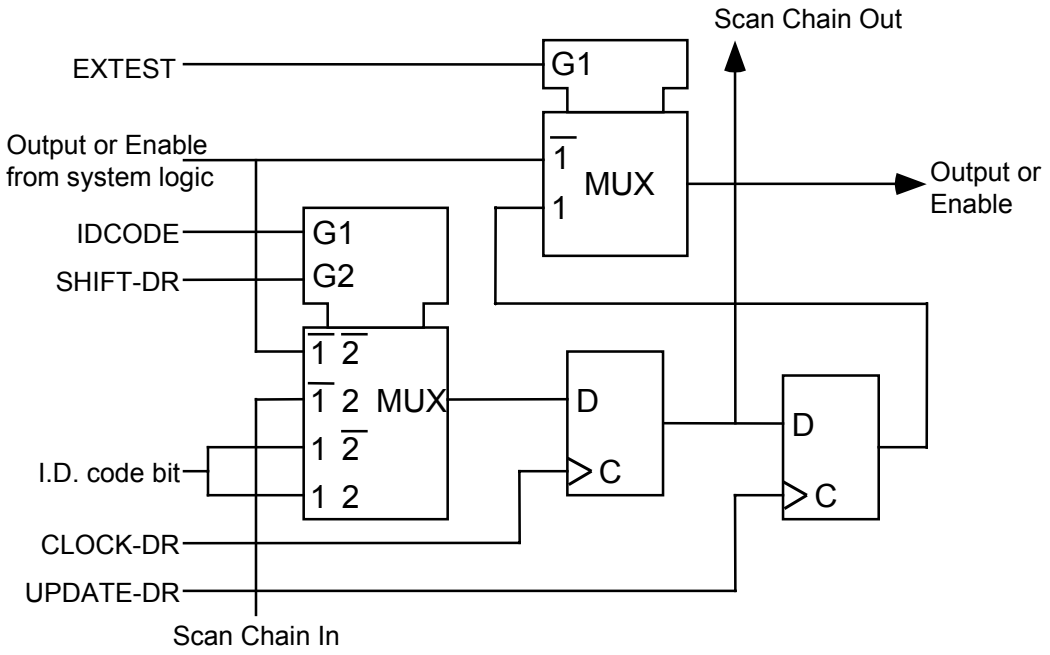


Figure 18 – Bi-directional Cell (IO_CELL)

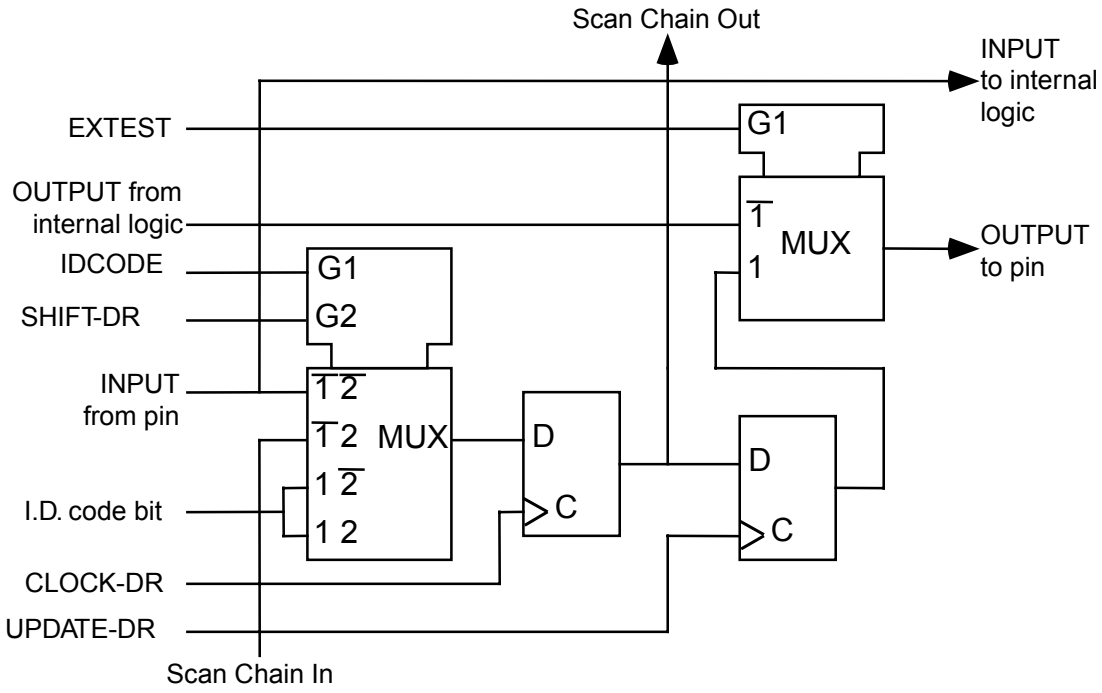
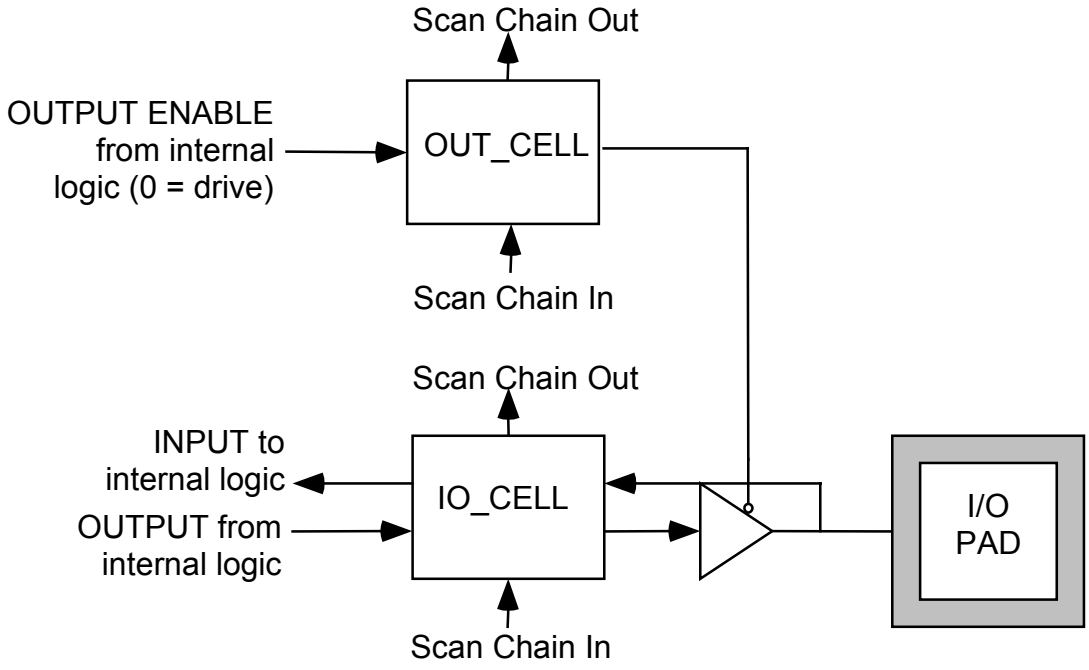


Figure 19 – Layout of Output Enable and Bi-directional Cells



13 OPERATIONS

This section presents connection details to the PM6344 EQUAD and PM4388 TOCTL devices, and operating details for the JTAG boundary scan feature.

13.1 EQUAD Connections

The required connections between the PM6344 EQUAD and the FREEDM-32 are shown in the following table:

Table 28 – FREEDM–EQUAD Connections

FREEDM Pin	Direction	EQUAD Pin
RCLK[n]	←	RDLCLK/RDLEOM[m]
RD[n]	←	RDLSIG/RDLINT[m]
n.c.	←	RFP[m]
n.c.	←	RCLKO[m]
TCLK[n]	←	TDLCLK/TDLURD[m]
TD[n]	→	TDLSIG/TDLINT[m]

The RFRACE1 and TFRACE1 bits in the EQUAD's Datalink Options Register should be set to '1'. In addition, the EQUAD's Channel Select Registers should be programmed such that timeslot 0 is disabled and timeslots 1 to 31 enabled.

13.2 TOCTL Connections

The required connections between the PM4388 TOCTAL and the FREEDM-32 are shown in the following table:

Table 29 – FREEDM–TOCTAL Connections

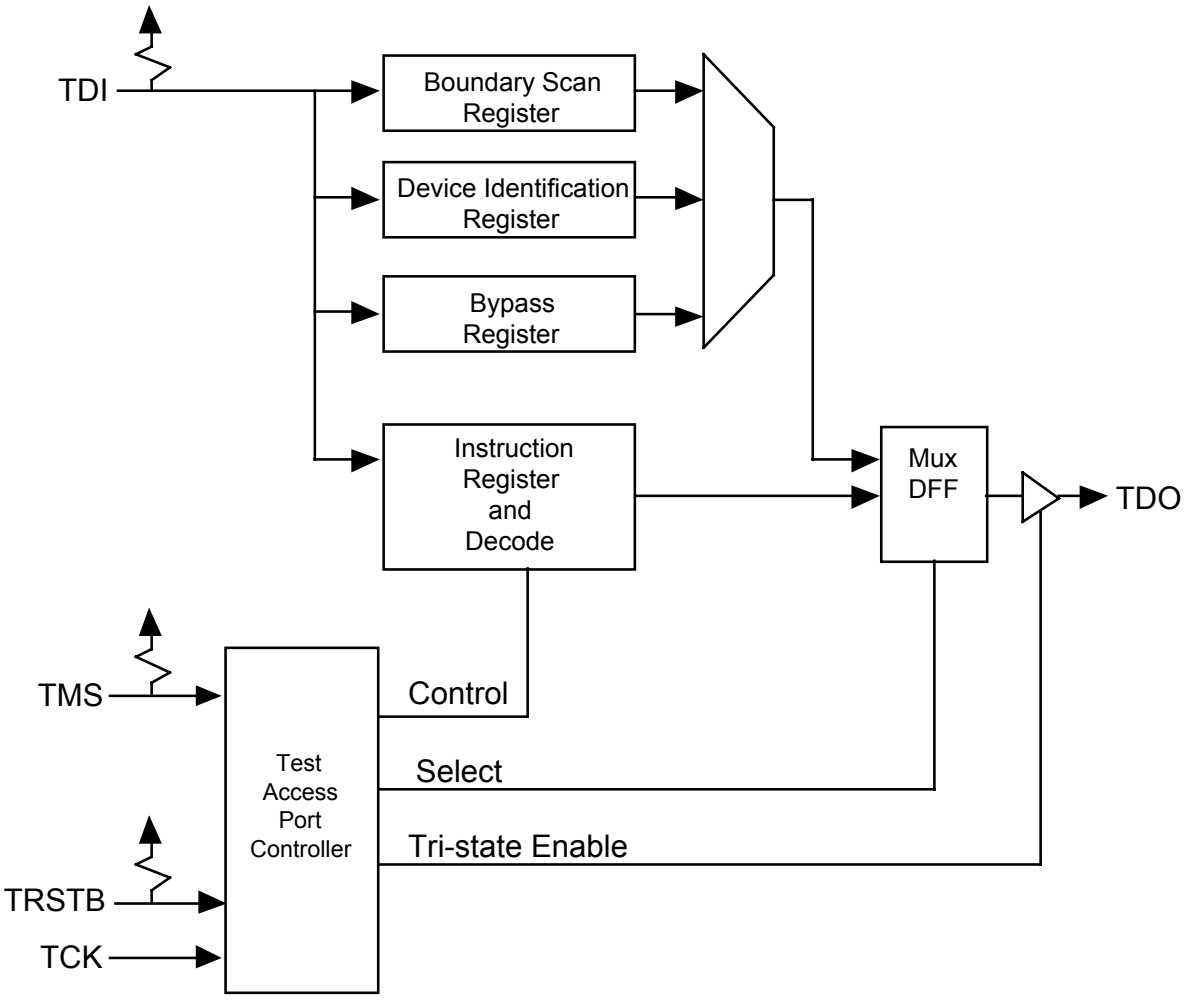
FREEDM Pin	Direction	TOCTAL Pin
RCLK[n]	←	ICLK/ISIG[m]
RD[n]	←	ID[m]
n.c.	←	IFP[m]
TCLK[n]	←	EFP/RCLK/ESIG[m]
TD[n]	→	ED[m]

All 8 framers in the TOCTAL should be programmed to operate in “Clock Master: NxDS0” mode in both the ingress and egress direction.

13.3 JTAG Support

The FREEDM-32 supports the IEEE Boundary Scan Specification as described in the IEEE 1149.1 standards. The Test Access Port (TAP) consists of the five standard pins, TRSTB, TCK, TMS, TDI and TDO used to control the TAP controller and the boundary scan registers. The TRSTB input is the active low reset signal used to reset the TAP controller. TCK is the test clock used to sample data on input, TDI and to output data on output, TDO. The TMS input is used to direct the TAP controller through its states. The basic boundary scan architecture is shown below.

Figure 20 – Boundary Scan Architecture



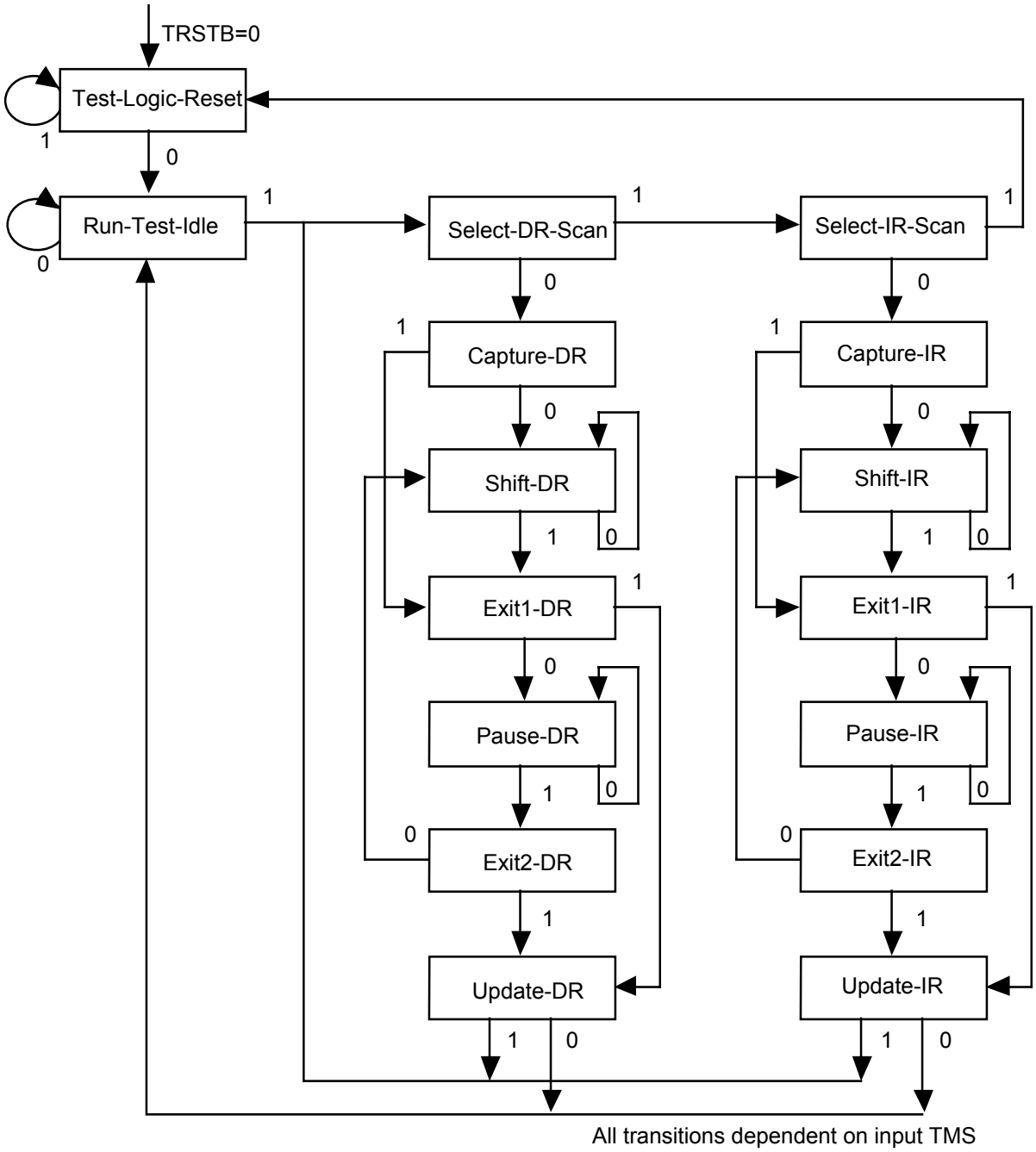
The boundary scan architecture consists of a TAP controller, an instruction register with instruction decode, a bypass register, a device identification register and a boundary scan register. The TAP controller interprets the TMS input and generates control signals to load the instruction and data registers. The instruction register with instruction decode block is used to select the test to be executed and/or the register to be accessed. The bypass register offers a single bit delay from primary input, TDI to primary output , TDO. The device identification register contains the device identification code.

The boundary scan register allows testing of board inter-connectivity. The boundary scan register consists of a shift register placed in series with device inputs and outputs. Using the boundary scan register, all digital inputs can be sampled and shifted out on primary output TDO. In addition, patterns can be shifted in on primary input, TDI and forced onto all digital outputs.

TAP Controller

The TAP controller is a synchronous finite state machine clocked by the rising edge of primary input, TCK. All state transitions are controlled using primary input, TMS. The finite state machine is described below.

Figure 21 – TAP Controller Finite State Machine



Test-Logic-Reset

The test logic reset state is used to disable the TAP logic when the device is in normal mode operation. The state is entered asynchronously by asserting input, TRSTB. The state is entered synchronously regardless of the current TAP controller state by forcing input, TMS high for 5 TCK clock cycles. While in this state, the instruction register is set to the IDCODE instruction.

Run-Test-Idle

The run test/idle state is used to execute tests.

Capture-DR

The capture data register state is used to load parallel data into the test data registers selected by the current instruction. If the selected register does not allow parallel loads or no loading is required by the current instruction, the test register maintains its value. Loading occurs on the rising edge of TCK.

Shift-DR

The shift data register state is used to shift the selected test data registers by one stage. Shifting is from MSB to LSB and occurs on the rising edge of TCK.

Update-DR

The update data register state is used to load a test register's parallel output latch. In general, the output latches are used to control the device. For example, for the EXTEST instruction, the boundary scan test register's parallel output latches are used to control the device's outputs. The parallel output latches are updated on the falling edge of TCK.

Capture-IR

The capture instruction register state is used to load the instruction register with a fixed instruction. The load occurs on the rising edge of TCK.

Shift-IR

The shift instruction register state is used to shift both the instruction register and the selected test data registers by one stage. Shifting is from MSB to LSB and occurs on the rising edge of TCK.

Update-IR

The update instruction register state is used to load a new instruction into the instruction register. The new instruction must be scanned in using the Shift-IR state. The load occurs on the falling edge of TCK.

The Pause-DR and Pause-IR states are provided to allow shifting through the test data and/or instruction registers to be momentarily paused.

Boundary Scan Instructions

The following is a description of the standard instructions. Each instruction selects a serial test data register path between input, TDI and output, TDO.

BYPASS

The bypass instruction shifts data from input, TDI to output, TDO with one TCK clock period delay. The instruction is used to bypass the device.

EXTEST

The external test instruction allows testing of the interconnection to other devices. When the current instruction is the EXTEST instruction, the boundary scan register is place between input, TDI and output, TDO. Primary device inputs can be sampled by loading the boundary scan register using the Capture-DR state. The sampled values can then be viewed by shifting the boundary scan register using the Shift-DR state. Primary device outputs can be controlled by loading patterns shifted in through input TDI into the boundary scan register using the Update-DR state.

SAMPLE

The sample instruction samples all the device inputs and outputs. For this instruction, the boundary scan register is placed between TDI and TDO. Primary device inputs and outputs can be sampled by loading the boundary scan register using the Capture-DR state. The sampled values can then be viewed by shifting the boundary scan register using the Shift-DR state.

IDCODE

The identification instruction is used to connect the identification register between TDI and TDO. The device's identification code can then be shifted out using the Shift-DR state.

STCTEST

The single transport chain instruction is used to test out the TAP controller and the boundary scan register during production test. When this instruction is the current instruction, the boundary scan register is connected between TDI and TDO. During the Capture-DR state, the device identification code is loaded into the boundary scan register. The code can then be shifted out output, TDO using the Shift-DR state.

INTEST

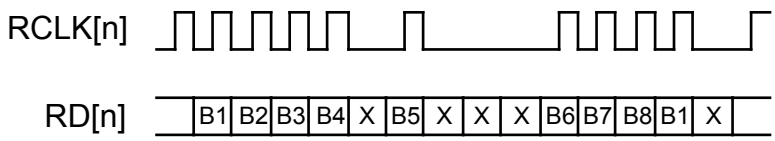
The internal test instruction is used to exercise the device's internal core logic. When this instruction is the current instruction, the boundary scan register is connected between TDI and TDO. During the Update-DR state, patterns shifted in on input, TDI are used to drive primary inputs. During the Capture-DR state, primary outputs are sampled and loaded into the boundary scan register.

14 FUNCTIONAL TIMING

14.1 Receive Link Input Timing

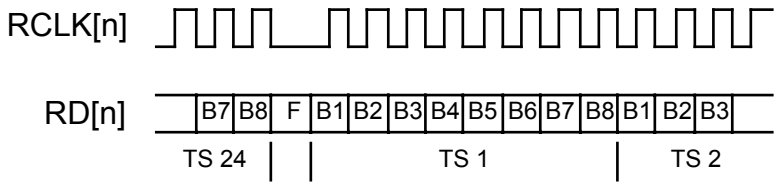
The timing relationship of the receive clock (RCLK[n]) and data (RD[n]) signals of an unchannelised link is shown in Figure 22. The receive data is viewed as a contiguous serial stream. There is no concept of time-slots in an unchannelised link. Every eight bits are grouped together into a byte with arbitrary alignment. The first bit received (B1 in Figure 22) is deemed the most significant bit of an octet. The last bit received (B8) is deemed the least significant bit. Bits that are to be processed by the FREEDM-32 are clocked in on the rising edge of RCLK[n]. Bits that should be ignored (X in Figure 22) are squelched by holding RCLK[n] quiescent. In Figure 22, the quiescent period is shown to be a low level on RCLK[n]. A high level, effected by extending the high phase of the previous valid bit, is also acceptable. Selection of bits for processing is arbitrary and is not subject to any byte alignment nor frame boundary considerations.

Figure 22 – Unchannelised Receive Link Timing



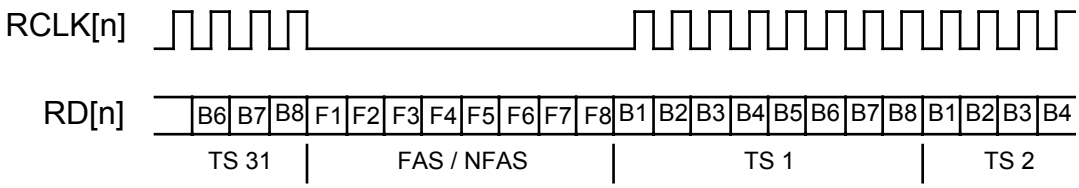
The timing relationship of the receive clock (RCLK[n]) and data (RD[n]) signals of a channelised T1 link is shown in Figure 23. The receive data stream is a T1 frame with a single framing bit (F in Figure 23) followed by octet bound time-slots 1 to 24. RCLK[n] is held quiescent during the framing bit. The RD[n] data bit (B1 of TS1) clocked in by the first rising edge of RCLK[n] after the framing bit is the most significant bit of time-slot 1. The RD[n] bit (B8 of TS24) clocked in by the last rising edge of RCLK[n] before the framing bit is the least significant bit of time-slot 24. In Figure 23, the quiescent period is shown to be a low level on RCLK[n]. A high level, effected by extending the high phase of bit B8 of time-slot TS24, is equally acceptable. In channelised T1 mode, RCLK[n] can only be gapped during the framing bit. It must be active continuously at 1.544 MHz during all time-slot bits. Time-slots can be ignored by setting the PROV bit in the corresponding word of the receive channel provision RAM in the RCAS block to low.

Figure 23 – Channelised T1 Receive Link Timing



The timing relationship of the receive clock (RCLK[n]) and data (RD[n]) signals of a channelised E1 link is shown in Figure 24. The receive data stream is an E1 frame with a singe framing byte (F1 to F8 in Figure 24) followed by octet bound time-slots 1 to 31. RCLK[n] is held quiescent during the framing byte. The RD[n] data bit (B1 of TS1) clocked in by the first rising edge of RCLK[n] after the framing byte is the most significant bit of time-slot 1. The RD[n] bit (B8 of TS31) clocked in by the last rising edge of RLCLK[n] before the framing byte is the least significant bit of time-slot 31. In Figure 24, the quiescent period is shown to be a low level on RCLK[n]. A high level, effected by extending the high phase of bit B8 of time-slot TS31, is equally acceptable. In channelised E1 mode, RCLK[n] can only be gapped during the framing byte. It must be active continuously at 2.048 MHz during all time-slot bits. Time-slots can be ignored by setting the PROV bit in the corresponding word of the receive channel provision RAM in the RCAS block to low.

Figure 24 – Channelised E1 Receive Link Timing

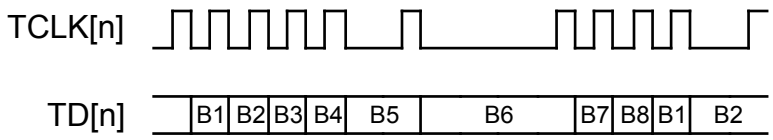


14.2 Transmit Link Output Timing

The timing relationship of the transmit clock (TCLK[n]) and data (TD[n]) signals of a unchannelised link is shown in Figure 25. The transmit data is viewed as a contiguous serial stream. There is no concept of time-slots in an unchannelised link. Every eight bits are grouped together into a byte with arbitrary byte alignment. Octet data is transmitted from most significant bit (B1 in Figure 25) and ending with the least significant bit (B8 in Figure 25). Bits are updated on the falling edge of TCLK[n]. A transmit link may be stalled by holding the corresponding TCLK[n] quiescent. In Figure 25, bits B5 and B2 are shown to be stalled for one cycle while bit B6 is shown to be stalled for three cycles. In Figure

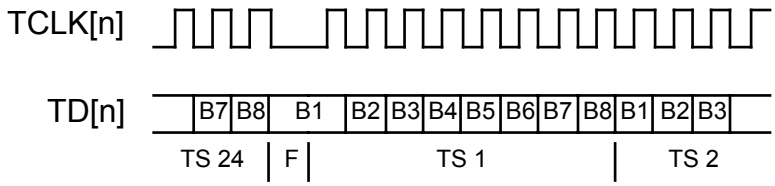
25, the quiescent period is shown to be a low level on TCLK[n]. A high level, effected by extending the high phase of the previous valid bit, is also acceptable. Gapping of TCLK[n] can occur arbitrarily without regard to byte nor frame boundaries.

Figure 25 – Unchannelised Transmit Link Timing



The timing relationship of the transmit clock (TCLK[n]) and data (TD[n]) signals of a channelised T1 link is shown in Figure 26. The transmit data stream is a T1 frame with a single framing bit (F in Figure 26) followed by octet bound time-slots 1 to 24. TCLK[n] is held quiescent during the framing bit. The most significant bit of each time-slot is transmitted first (B1 in Figure 26). The least significant bit of each time-slot is transmitted last (B8 in Figure 26). The TD[n] bit (B8 of TS24) before the framing bit is the least significant bit of time-slot 24. In Figure 26, the quiescent period is shown to be a low level on TCLK[n]. A high level, effected by extending the high phase of bit B8 of time-slot TS24, is equally acceptable. In channelised T1 mode, TCLK[n] can only be gapped during the framing bit. It must be active continuously at 1.544MHz during all time-slot bits. 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 TCAS block set low) transmit the contents of the Idle Fill Time-slot Data register.

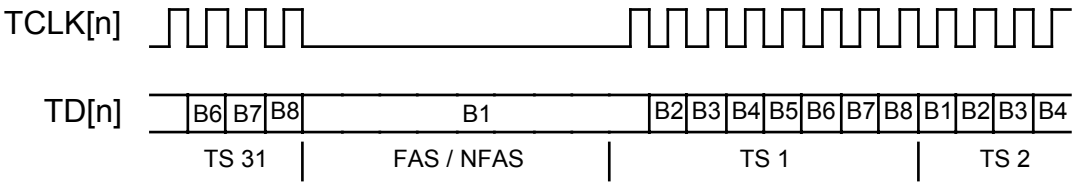
Figure 26 – Channelised T1 Transmit Link Timing



The timing relationship of the transmit clock (TCLK[n]) and data (TD[n]) signals of a channelised E1 link is shown in Figure 27. The transmit data stream is an E1 frame with a singe framing byte (FAS/NFAS in Figure 27) followed by octet bound time-slots 1 to 31. TCLK[n] is held quiescent during the framing byte. The most significant bit of each time-slot is transmitted first (B1 in Figure 27). The least significant bit of each time-slot is transmitted last (B8 in Figure 27). The TD[n] bit (B8 of TS31) before the framing byte is the least significant bit of time-slot 31. In Figure 27, the quiescent period is shown to be a low level on

TCLK[n]. A high level, effected by extending the high phase of bit B8 of time-slot 31, is equally acceptable. In channelised E1 mode, TCLK[n] can only be gapped during the framing byte. It must be active continuously at 2.048 MHz during all time-slot bits. 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 TCAS block set low) transmit the contents of the Idle Time-slot Fill Data register.

Figure 27 – Channelised E1 Transmit Link Timing



14.3 PCI Interface

A PCI burst read cycle is shown In Figure 28. The cycle is valid for target and initiator accesses. The target is responsible for incrementing the address during the data burst. The 'T' symbol stands for a turn around cycle. A turn around cycle is required on all signals which can be driven by more than one agent.

During Clock 1, the initiator drives FRAMEB to indicate the start of a cycle. It also drives the address onto the AD[31:0] bus and drives the C/BEB[3:0] lines with the read command. In the example below, the command would indicate a burst read. The IRDYB, TRDYB and DEVSELB signals are in turnaround mode (i.e. no agent is driving the signals for this clock cycle). This cycle on the PCI bus is called the address phase.

During Clock 2, the initiator ceases to drive the AD[31:0] bus in order that the target can drive it in the next cycle. The initiator also drives the C/BEB[3:0] lines with the byte enables for the read data. IRDYB is driven active by the initiator to indicate it is ready to accept the data transfer. All subsequent cycles on the PCI bus are called data phases.

During Clock 3, the target claims the transaction by driving DEVSELB active. It also places the first data word onto the AD[31:0] bus and drives TRDYB to indicate to the initiator that the data is valid.

During Clock 4, the initiator latches in the first data word. The target negates TRDYB to indicate to the initiator that it is not ready to transfer another data word.

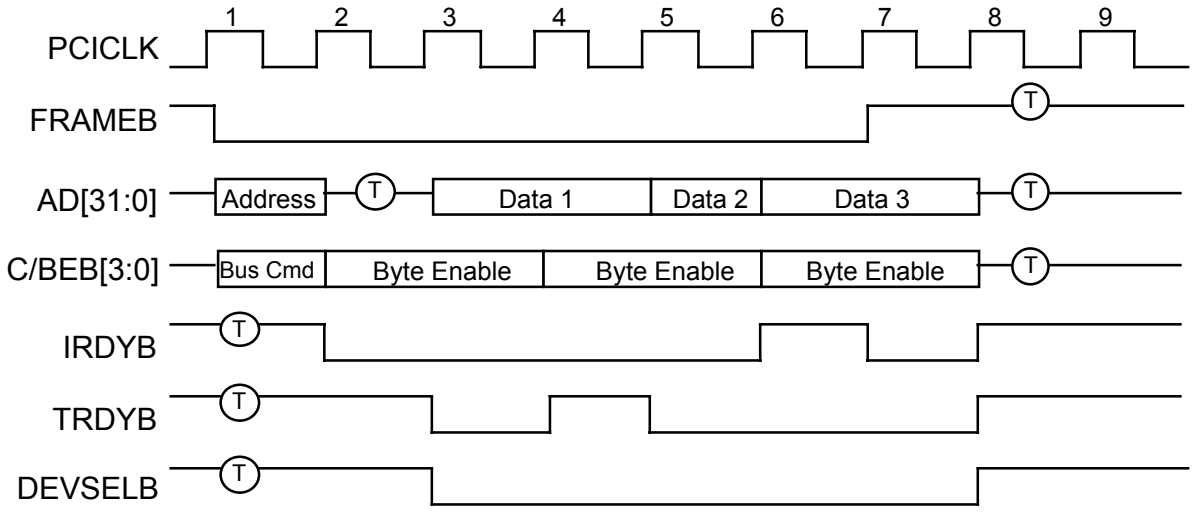
During Clock 5, the target places the second data word onto the AD[31:0] bus and drives TRDYB to indicate to the initiator that the data is valid.

During Clock 6, the initiator latches the second data word and negates IRDYB to indicate to the target that it is not ready for the next transfer. The target shall drive the third data word until the initiator accepts it.

During Clock 7, the initiator asserts IRDYB to indicate to the target it is ready for the third data word. It also negates FRAMEB since this shall be the last transfer.

During Clock 8, the initiator latches in the last word and negates IRDYB. The target, having seen FRAMEB negated in the last clock cycle, negates TRDYB and DEVSELB. All of the above signals shall be driven to their inactive state in this clock cycle, except for FRAMEB which shall be tri-stated. The target shall stop driving the AD[31:0] bus and the initiator shall stop driving the C/BEB[3:0] bus; this shall be the turnaround cycle for these signals.

Figure 28 – PCI Read Cycle



A PCI burst write transaction is shown in Figure 29. The cycle is valid for target and initiator accesses. The target is responsible for incrementing the address for the duration of the data burst. The 'T' symbol stands for a turn around cycle. A turn around cycle is required on all signals which can be driven by more than one agent.

During clock 1, the initiator drives FRAMEB to indicate the start of a cycle. It also drives the address onto the AD[31:0] bus and drives the C/BEB[3:0] lines with the write command (in the above example the command would indicate a burst write). The IRDYB, TRDYB and DEVSELB signals are in turnaround mode (no

agent is driving the signals for this clock cycle). This cycle on the PCI bus is called the address phase.

During clock 2, the initiator ceases to drive the address onto the AD[31:0] bus and starts driving the first data word. The initiator also drives the C/BEB[3:0] lines with the byte enables for the write data. IRDYB is driven active by the initiator to indicate it is ready to accept the data transfer. The target claims the transaction by driving DEVSELB active and drives TRDYB to indicate to the initiator that it is ready to accept the data. All subsequent cycles on the PCI bus are called data phases.

During clock 3, the target latches in the first data word. The initiator starts to drive the next data word onto the AD[31:0] lines.

During clock 4, the target latches in the second data word. Both the initiator and the target indicate that they are not ready to transfer any more data by negating the ready lines.

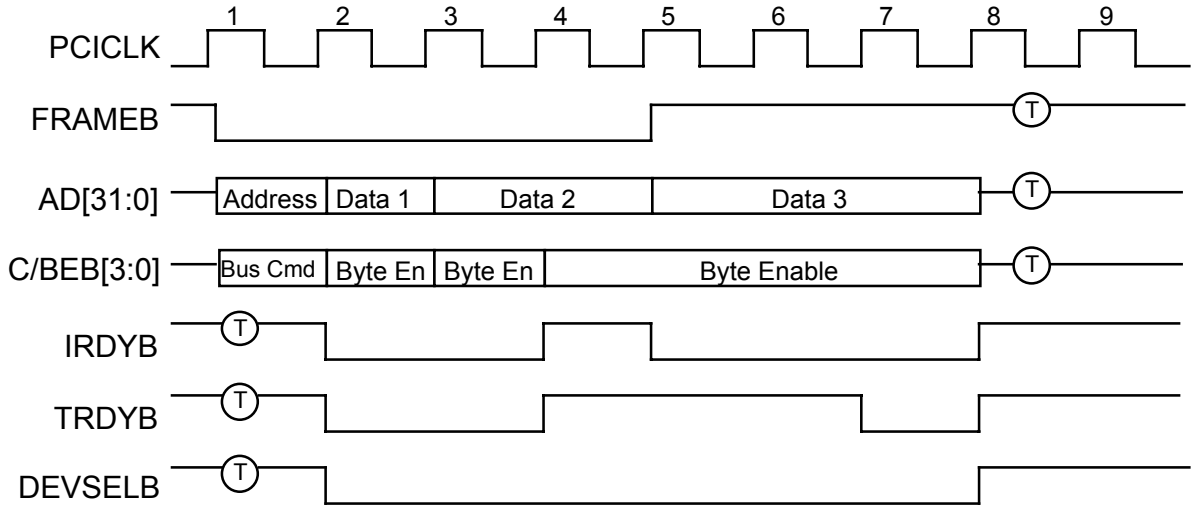
During clock 5, the initiator is ready to transfer the next data word so it drives the AD[31:0] lines with the third data word and asserts IRDYB. The initiator negates FRAMEB since this is the last data phase of this cycle. The target is still not ready so a wait state shall be added.

During clock 6, the target is still not ready so another wait state is added.

During clock 7, the target asserts TRDYB to indicate that it is ready to complete the transfer.

During clock 8, the target latches in the last word and negates TRDYB and DEVSELB, having seen FRAMEB negated previously. The initiator negates IRDYB. All of the above signals shall be driven to their inactive state in this clock cycle except for FRAMEB which shall be tri-stated.

Figure 29 – PCI Write Cycle



The PCI Target Disconnect (Figure 30) illustrates the case when the target wants to prematurely terminate the current cycle. Note, when the FREEDM-32 is the target, it never prematurely terminates the current cycle.

A target can terminate the current cycle by asserting the STOPB signal to the initiator. Whether data is transferred or not depends on the state of the ready signals at the time that the target disconnects. If the FREEDM-32 is the initiator and the target terminates the current access, the FREEDM-32 will retry the access after two PCI bus cycles.

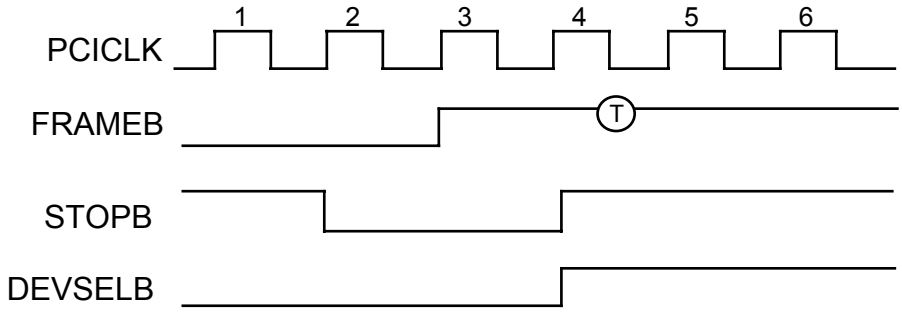
During clock 1, an access is in progress.

During clock 2, the target indicates that it wishes to disconnect by asserting STOPB. Data may be transferred depending on the state of the ready lines.

During clock 3, the initiator negates FRAMEB to signal the end of the cycle.

During clock 4, the target negates STOPB and DEVSELB in response to the FRAMEB signal being negated.

Figure 30 – PCI Target Disconnect



The PCI Target Abort Diagram (Figure 31) illustrates the case when the target wants to abort the current cycle. Note, when the FREEDM-32 is the target, it never aborts the current cycle. A target abort is an indication of a serious error and no data is transferred.

A target can terminate the current cycle by asserting STOPB and negating DEVSELB. If the FREEDM-32 is the initiator and the target aborts the current access, the abort condition is reported to the PCI Host.

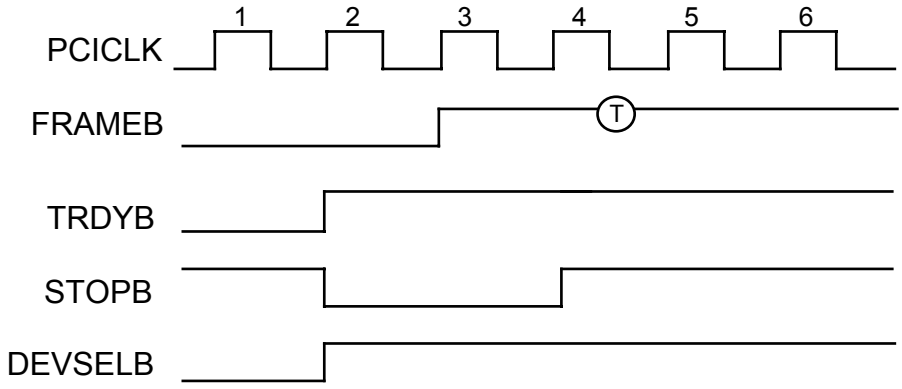
During clock 1, a cycle is in progress.

During clock 2, the target negates DEVSELB and TRDYB and asserts STOPB to indicate an abort condition to the initiator.

During clock 3, the initiator negates FRAMEB in response to the abort request.

During clock 4, the target negates STOPB signal in response to the FRAMEB signal being negated.

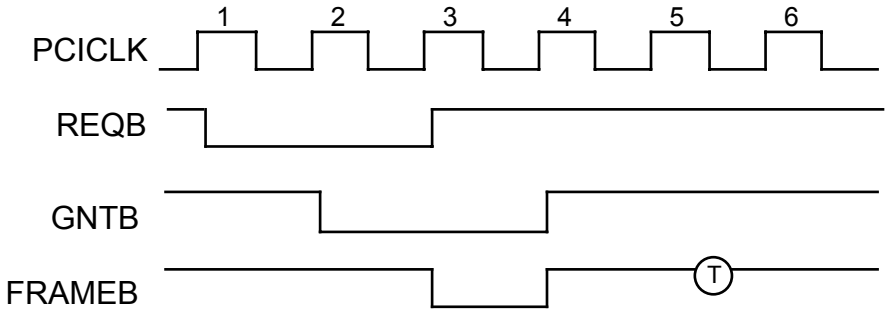
Figure 31 – PCI Target Abort



The PCI Bus Request Cycle Diagram (Figure 32) illustrates the case when the initiator is requesting the bus from the bus arbiter.

When the FREEDM-32 is the initiator, it requests the PCI bus by asserting its REQB output to the central arbiter. The arbiter grants the bus to the FREEDM-32 by asserting the GNTB line. The FREEDM-32 will wait till both the FRAMEB and IRDYB lines are idle before starting its access on the PCI bus. The arbiter can remove the GNTB signal at any time, but the FREEDM-32 will complete the current transfer before relinquishing the bus.

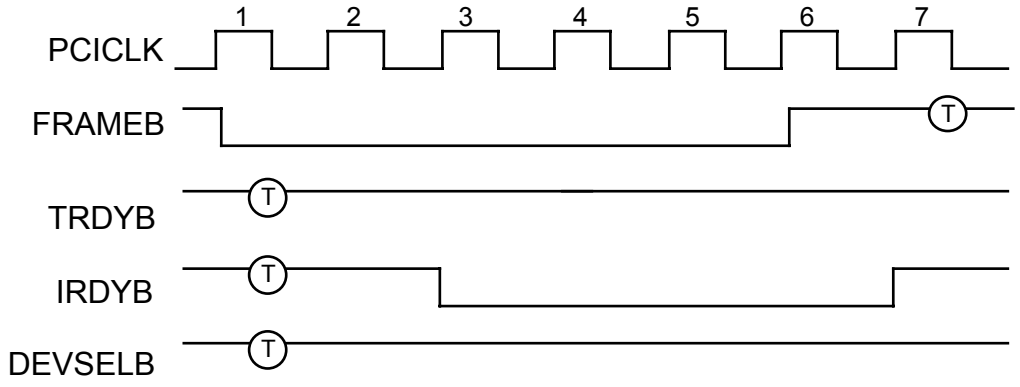
Figure 32 – PCI Bus Request Cycle



The PCI Initiator Abort Termination Diagram (Figure 33) illustrates the case when the initiator aborts a transaction on the PCI bus.

An initiator may terminate a cycle if no target claims it within five clock cycles. A target may not have responded because it was incapable of dealing with the request or a bad address was generated by the initiator. IRDYB must be valid one clock after FRAMEB is deasserted as in a normal cycle. When the FREEDM-32 is the initiator and aborts the transaction, it reports the error condition to the PCI Host.

Figure 33 – PCI Initiator Abort Termination



The PCI Exclusive Lock Cycle Diagram (Figure 34) illustrates the case when the current initiator locks the PCI bus. The FREEDM-32 will never initiate an lock, but will behave appropriately when acting as a target.

During clock 1, the present initiator has gained access of the LOCKB signal and the PCI bus. The first cycle of a locked access must be a read cycle. The initiator asserts FRAMEB and drives the address of the target on the AD[31:0] lines.

During clock 2, the present initiator asserts LOCKB to indicate to the target that a locked cycle is in progress.

During clock 3, the target samples the asserted LOCKB signal and marks itself locked. The data cycle has to complete in order for the lock to be maintained. If for some reason the cycle was aborted, the initiator must negate LOCKB.

During clock 4, the data transfer completes and the target is locked.

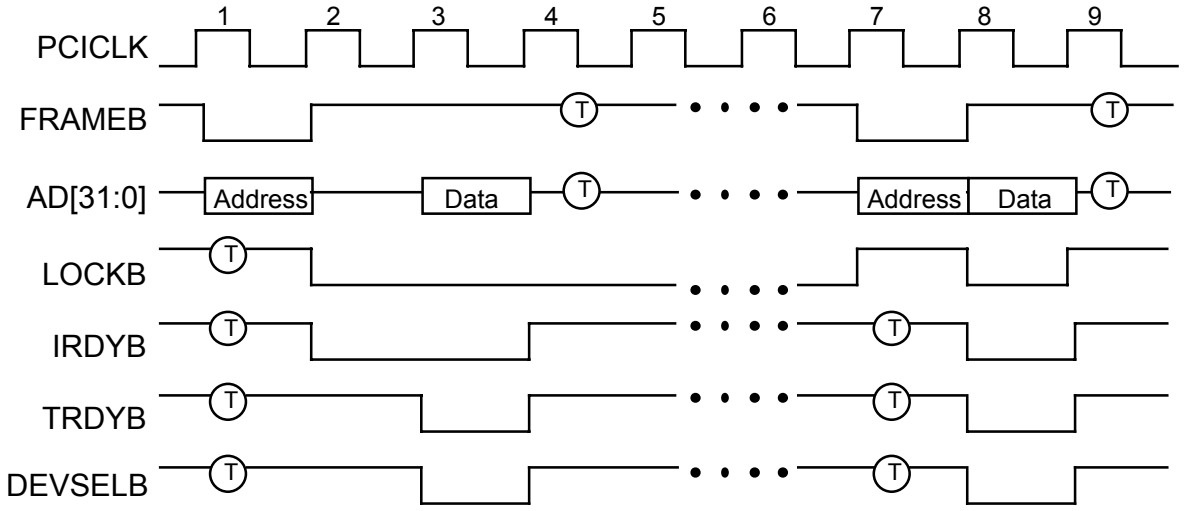
During clock 6, another initiator may use the PCI bus but it cannot use the LOCKB signal. If the other initiator attempts to access the locked target that it did not lock, the target would reject the access.

During clock 7, the same initiator that locked this target accesses the target. The initiator asserts FRAMEB and negates LOCKB to re-establish the lock.

During clock 8, the target samples LOCKB deasserted and locks itself.

During clock 9, the initiator does not want to continue the lock so it negates LOCKB. The target samples LOCKB and FRAMEB deasserted it removes its lock.

Figure 34 – PCI Exclusive Lock Cycle



Fast back-to-back transactions are used by an initiator to conduct two consecutive transactions on the PCI bus without the required idle cycle between them. This can only occur if there is a guarantee that there will be no contention between the initiator or targets involved in the two transactions. In the first case, an initiator may perform fast back-to-back transactions if the first transaction is a write and the second transaction is to the same target. All targets must be able to decode the above transaction. In the second case, all of the targets on the PCI bus support fast back-to-back transactions, as indicated in the PCI Status configuration register. The FREEDM-32 only supports the first type of fast back-to-back transactions and is shown in Figure 35.

During clock 1, the initiator drives FRAMEB to indicate the start of a cycle. It also drives the address onto the AD[31:0] bus and drives the C/BEB[3:0] lines with the write command. In this example the command would indicate a single write. The IRDYB, TRDYB and DEVSELB signals are in turnaround mode and are not being driven for this clock cycle. This cycle on the PCI bus is called the address phase.

During clock 2, the initiator ceases to drive the address onto the AD[31:0] bus and starts driving the data element. The initiator also drives the C/BEB[3:0] lines with the byte enables for the write data. IRDYB is driven active by the initiator to indicate that the data is valid.

The initiator negates FRAMEB since this the last data phase of this cycle. The target claims the transaction by driving DEVSELB active and drives TRDYB to indicate to the initiator that it is ready to accept the data.

During clock 3, the target latches in the data element and negates TRDYB and DEVSELB, having seen FRAMEB negated previously. The initiator negates IRDYB and drives FRAMEB to start the next cycle. It also drives the address onto the AD[31:0] bus and drives the C/BEB[3:0] lines with the write command. In this example the command would indicate a burst write.

During clock 4, the initiator ceases to drive the address onto the AD[31:0] bus and starts driving the first data element. The initiator also drives the C/BEB[3:0] lines with the byte enables for the write data. IRDYB is driven active by the initiator to indicate that the data is valid. The target claims the transaction by driving DEVSELB active and drives TRDYB to indicate to the initiator that it is ready to accept the data.

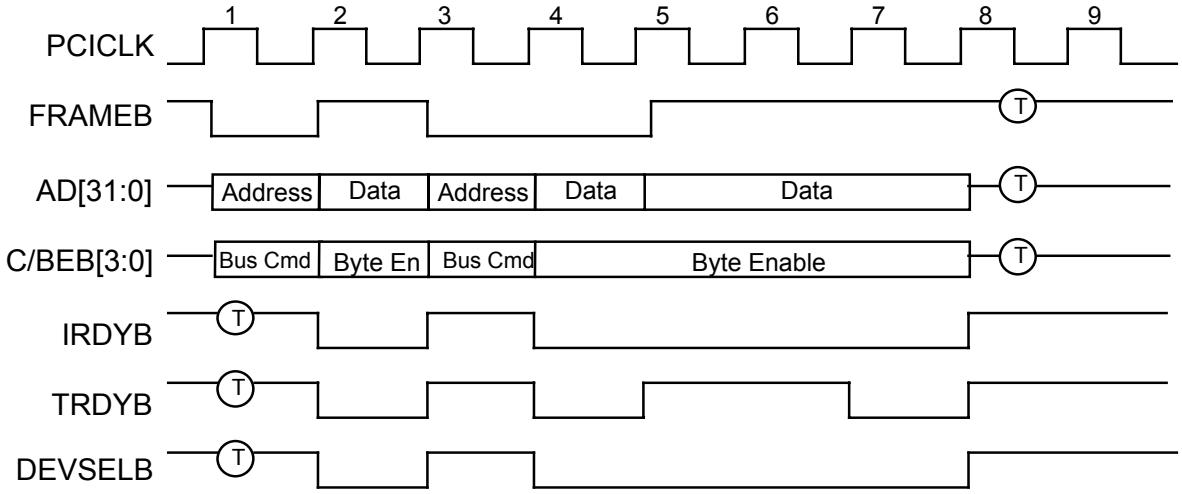
During clock 5, the initiator is ready to transfer the next data element so it drives the AD[31:0] lines with the second data element. The initiator negates FRAMEB since this is the last data phase of this cycle. The target accepts the first data element and negates TRDYB to indicate its is not ready for the next data element.

During clock 6, the target is still not ready so a wait state shall be added.

During clock 7, the target asserts TRDYB to indicate that it is ready to complete the transfer.

During clock 8, the target latches in the last element and negates TRDYB and DEVSELB, having seen FRAMEB negated previously. The initiator negates IRDYB. All of the above signals shall be driven to their inactive state in this clock cycle, except for FRAMEB which shall be tri-stated. The target stops driving the AD[31:0] bus and the initiator stops driving the C/BEB[3:0] bus. This shall be the turnaround cycle for these signals.

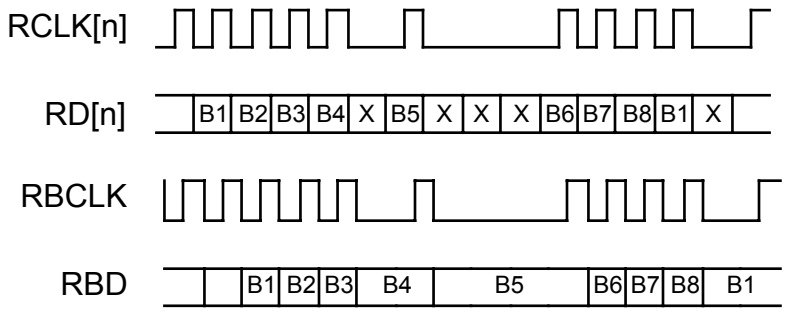
Figure 35 – PCI Fast Back to Back



14.4 BERT Interface

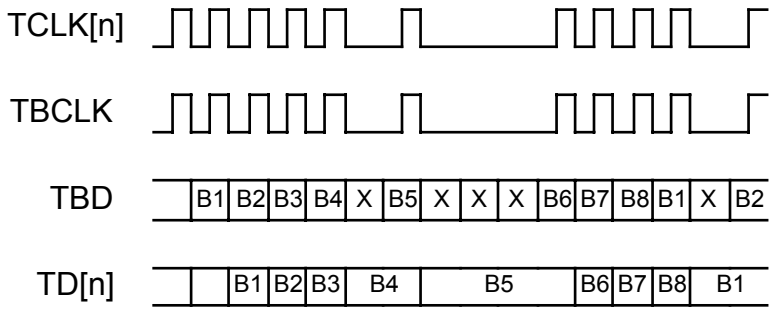
The timing relationship between the receive link clock and data (RCLK[n] / RD[n]) and the receive BERT port signals (RBCLK / RBD) is shown in Figure 36. The selected RCLK[n] is placed on RBCLK after an asynchronous delay. The selected receive link data (RD[n]) is sampled on the rising edge of the associated RCLK[n] and transferred to RBD on the falling edge of RBCLK.

Figure 36 – Receive BERT Port Timing



The timing relationship between the transmit link clock and data (TCLK[n] / TD[n]) and the transmit BERT port signals (TBCLK / TBD) is shown in Figure 37. TCLK[n] is shown to have an arbitrary gapping. When TCLK[n] is quiescent, TBD is ignored (X in Figure 37). The selected TCLK[n] is buffered and placed on TBCLK. The transmit BERT data (TBD) is sampled on the rising edge of the TBCLK and transferred to the selected TD[n] on the falling edge of TCLK[n].

Figure 37 – Transmit BERT Port Timing



15 ABSOLUTE MAXIMUM RATINGS

Maximum rating are the worst case limits that the device can withstand without sustaining permanent damage. They are not indicative of normal operating conditions.

Table 30 – FREEDM-32 Absolute Maximum Ratings

Case Temperature under Bias	-40°C to +85°C
Storage Temperature	-40°C to +125°C
Supply Voltage (V_{DD})	-0.3V to +4.6V
Bias Voltage (V_{BIAS})	(V_{DD} - 0.3) to +5.5V
Voltage on Any Pin	-0.3V to V_{BIAS} + 0.3V
Static Discharge Voltage	±1000 V
Latch-Up Current	±100 mA
DC Input Current	±20 mA
Lead Temperature	+230°C
Absolute Maximum Junction Temperature	+150°C

16 D.C. CHARACTERISTICS

(T_A = -40°C to +85°C, V_{DD} = 3.3 V ±10%)

Table 31 – FREEDM-32 D.C. Characteristics

Symbol	Parameter	Min	Typ	Max	Units	Conditions
V_{DD}	Power Supply	3.0	3.3	3.6	Volts	Note 5.
V_{BIAS}	5V Bias Supply	V_{DD}	5.0	5.5	Volts	Note 5.
V_{IL}	PCI Input Low Voltage	-0.2		0.8	Volts	Guaranteed Input LOW Voltage for PCI inputs.
V_{IH}	PCI Input High Voltage	2.0		V_{BIAS}	Volts	Guaranteed Input HIGH Voltage for PCI inputs.
V_{OL}	Output or Bi-directional Low Voltage		0.26	0.4	Volts	I_{OL} = -4 mA for all outputs except TBCLK, RBCLK, RBD, PCICLKO, PCIINTB, where I_{OL} = -6 mA and TD[2:0], where I_{OL} = -8 mA. Notes 3, 5.
V_{OH}	Output or Bi-directional High Voltage	2.4	2.52		Volts	I_{OH} = 4 mA for all outputs except TBCLK, RBCLK, RBD, PCICLKO, PCIINTB, where I_{OH} = 6 mA and TD[2:0], where I_{OH} = 8 mA. Notes 3, 5.
V_{T+}	Schmitt Triggered Input High Voltage	2.0		V_{BIAS}	Volts	
V_{T-}	Schmitt Triggered Input Low Voltage	-0.2		0.8	Volts	
I_{ILPU}	Input Low Current	10	45	100	µA	V_{IL} = GND, Notes 1, 3, 5.
I_{IHPU}	Input High Current	-10	0	+10	µA	V_{IH} = V_{DD}, Notes 1, 3
I_{IL}	Input Low Current	-10	0	+10	µA	V_{IL} = GND, Notes 2, 3
I_{IH}	Input High Current	-10	0	+10	µA	V_{IH} = V_{DD}, Notes 2, 3

Symbol	Parameter	Min	Typ	Max	Units	Conditions
C_{IN}	Input Capacitance		5		pF	Excludes package. Package typically 2 pF. Note 5.
C_{OUT}	Output Capacitance		5		pF	All pins. Excludes package. Package typically 2 pF. Note 5.
C_{IO}	Bi-directional Capacitance		5		pF	All pins. Excludes package. Package typically 2 pF. Note 5.
L_{PIN}	Pin Inductance		2		nH	All pins. Note 5.
I_{DDOP}	Operating Current.		285	360.2	mA	V_{DD} = 3.6V, Outputs Unloaded. Aggregate link clock rate = 64 MHz

Notes on D.C. Characteristics:

1. Input pin or bi-directional pin with internal pull-up resistor.
2. Input pin or bi-directional pin without internal pull-up resistor
3. Negative currents flow into the device (sinking), positive currents flow out of the device (sourcing).
4. Input pin or bi-directional pin with internal pull-down resistor.
5. Typical values are given as a design aid. The product is not tested to the typical values given in the data sheet.

17 FREEDM-32 TIMING CHARACTERISTICS

(T_A = -40°C to +85°C, V_{DD} = 3.3 V ±10%, V_{BIAS} = 5.0V ±10%)

Table 32 – FREEDM-32 Link Input (Figure 38, Figure 39)

Symbol	Description	Min	Max	Units
	RCLK[31:0] Frequency (See Note 3)	1.542	1.546	MHz
	RCLK[31:0] Frequency (See Note 4)	2.046	2.05	MHz
	RCLK[2:0] Frequency (See Note 5)		52	MHz
	RCLK[31:3] Frequency (See Note 5)		10	MHz
	RCLK[31:0] Duty Cycle	40	60	%
	SYSCLK Frequency	25	33	MHz
	SYSCLK Duty Cycle	40	60	%
tS_{RD}	RD[31:3] Set-Up Time	5		ns
tH_{RD}	RD[31:3] Hold Time	5		ns
tS_{RD}	RD[2:0] Set-Up Time	2		ns
tH_{RD}	RD[2:0] Hold Time	2		ns
tS_{TBD}	TBD Set-Up Time (See Note 6)	15		ns
tH_{TBD}	TBD Hold Time	0		ns

Notes on Input Timing:

1. When a set-up time is specified between an input and a clock, the set-up time is the time in nanoseconds from the 1.4 Volt point of the input to the 1.4 Volt point of the clock.

2. When a hold time is specified between an input and a clock, the hold time is the time in nanoseconds from the 1.4 Volt point of the clock to the 1.4 Volt point of the input.

3. Applicable only to channelised T1 links and measured between framing bits.

4. Applicable only to channelised E1 links and measured between framing bytes.

5. Applicable only to unchannelised links of any format and measured between any two RCLK rising edges.

6. TBD set-up time is measured with a 20 pF load on TBCLK. The set-up time increases by 1 ns for each 10 pF of extra load on TBCLK.

Figure 38 – Receive Link Input Timing

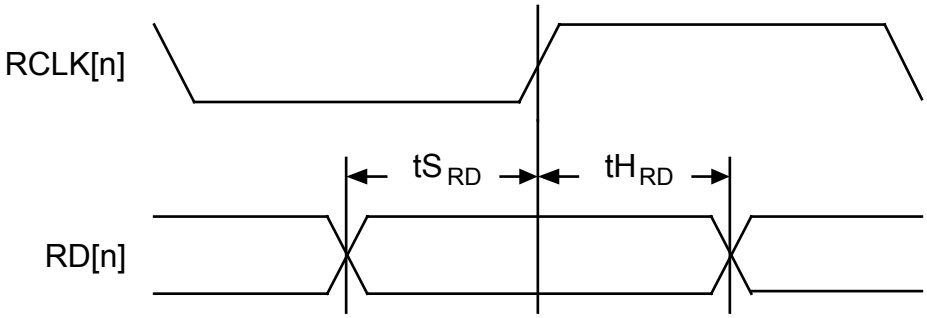


Figure 39 – BERT Input Timing

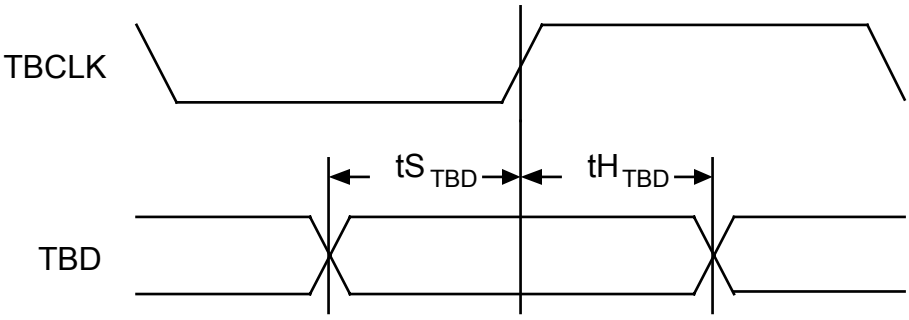


Table 33 – FREEDM-32 Link Output (Figure 40, Figure 41)

Symbol	Description	Min	Max	Units
	TCLK[31:0] Frequency (See Note 4)	1.542	1.546	MHz
	TCLK[31:0] Frequency (See Note 5)	2.046	2.05	MHz
	TCLK[2:0] Frequency (See Note 6)		52	MHz
	TCLK[31:3] Frequency (See Note 6)		10	MHz
	TCLK[31:0] Duty Cycle	40	60	%
tP_{TD}	TCLK[2:0] Low to TD[2:0] Valid	3	15	ns
tP_{TD}	TCLK[31:3] Low to TD[31:3] Valid	5	27	ns
tP_{RBD}	RBCLK Low to RBD Valid	-5	5	ns

Notes on Output Timing:

1. Output propagation delay time is the time in nanoseconds from the 1.4 Volt point of the reference signal to the 1.4 Volt point of the output.

2. Maximum and minimum output propagation delays are measured with a 50 pF load on all the outputs, except for PCI Bus outputs and TD[2:0] outputs. For PCI Bus outputs, maximum output propagation delays are measured with a 50 pF load while minimum output propagation delays are measured with a 0 pF load. For TD[2:0] outputs, propagation delays are measured with a 20 pF load. Maximum propagation delay for TD[2:0] increases by 1.0 ns for each 10 pF of extra load.

3. Output propagation delays of signal outputs that are specified in relation to a reference output are measured with a 50 pF load on both the signal output and the reference output.

4. Applicable only to channelised T1 links and measured between framing bits.

5. Applicable only to channelised E1 links and measured between framing bytes.

6. Applicable only to unchannelised links of any format and measured between any two TCLK rising edges.

7. Output tri-state delay is the time in nanoseconds from the 1.4 Volt point of the reference signal to the point where the total current delivered through the output is less than or equal to the leakage current.

Figure 40 – Transmit Link Output Timing

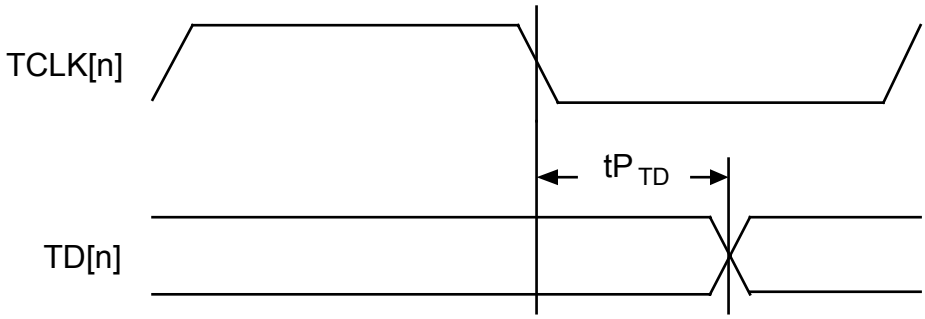


Figure 41 – BERT Output Timing

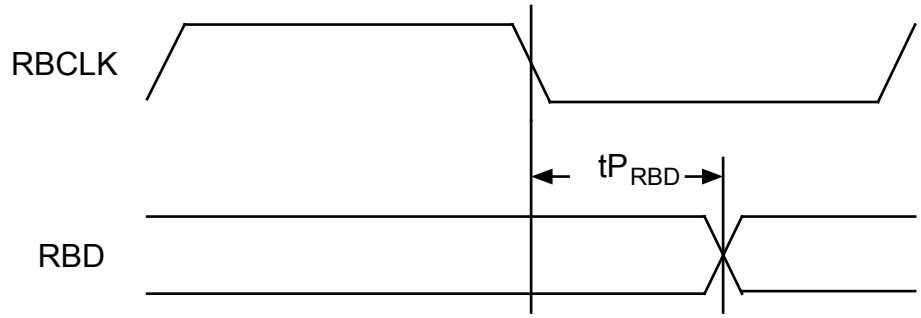


Table 34 – PCI Interface (Figure 42)

Symbol	Description	Min	Max	Units
	PCICLK Frequency		33	MHz
	PCICLK Duty Cycle	40	60	%
tS_{PCI}	All PCI Input and Bi-directional Set-up time to PCICLK	7		ns
tH_{PCI}	All PCI Output and Bi-directional Hold time to PCICLK	0		ns
tP_{PCI}	PCICLK to all PCI Outputs Valid	2	11	ns
tZ_{PCI}	All PCI Output PCICLK to Tri-state		28	ns
tZN_{PCI}	All PCI Output Tri-state from PCICLK to active	2		ns

Figure 42 – PCI Interface Timing

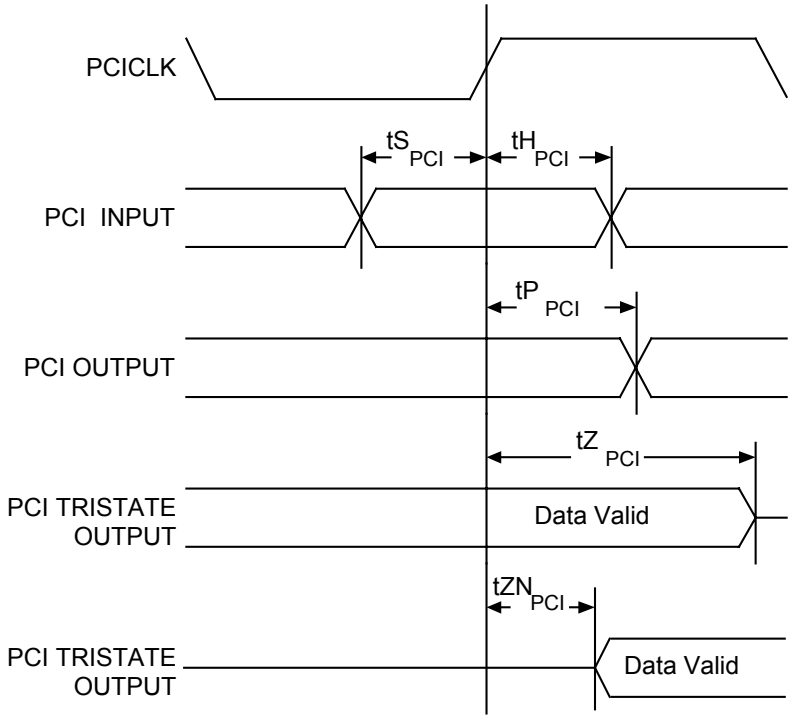
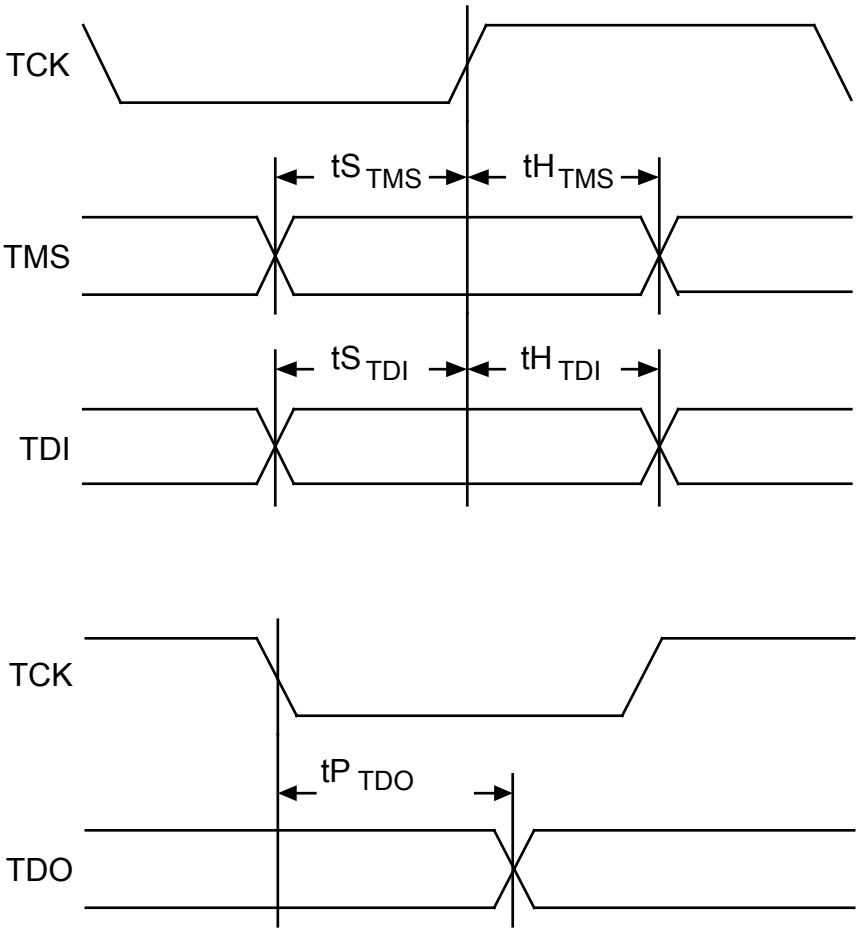


Table 35 – JTAG Port Interface (Figure 43)

Symbol	Description	Min	Max	Units
	TCK Frequency		1	MHz
	TCK Duty Cycle	40	60	%
tS_{TMS}	TMS Set-up time to TCK	50		ns
tH_{TMS}	TMS Hold time to TCK	50		ns
tS_{TDI}	TDI Set-up time to TCK	50		ns
tH_{TDI}	TDI Hold time to TCK	50		ns
tP_{TDO}	TCK Low to TDO Valid	2	50	ns

Figure 43 – JTAG Port Interface Timing



18 ORDERING AND THERMAL INFORMATION

Table 36 – FREEDM-32 Ordering Information

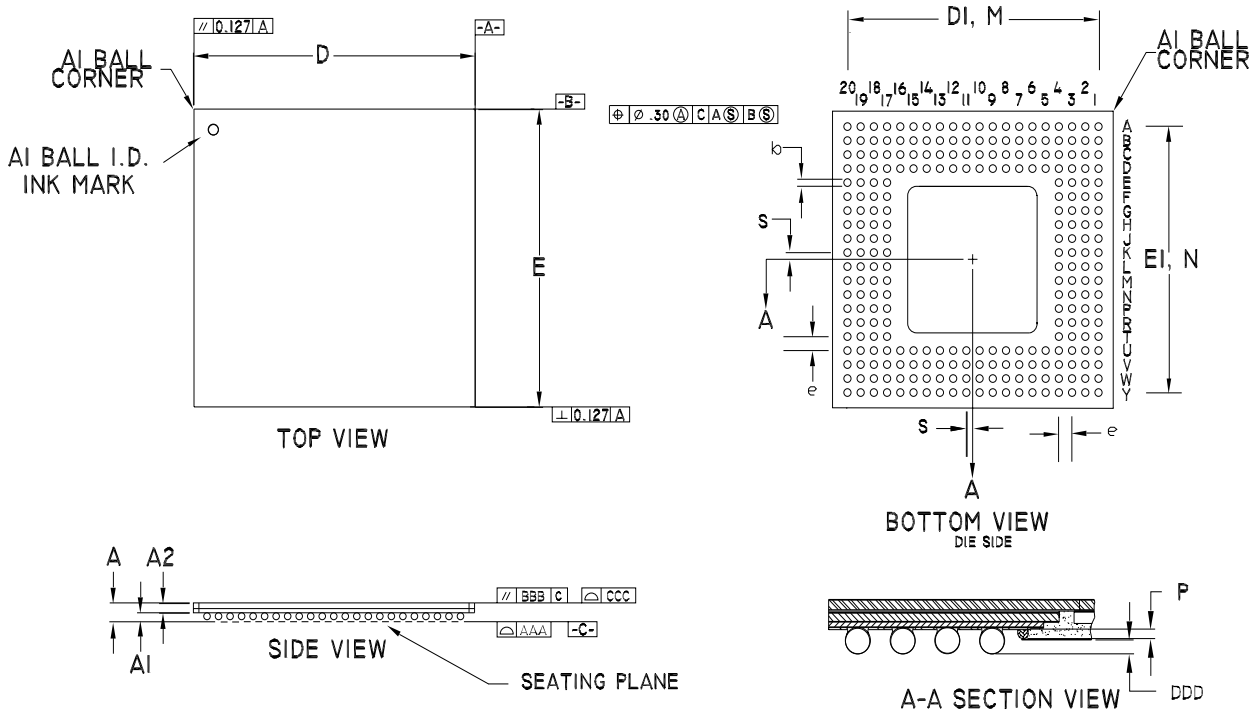
PART NO.	DESCRIPTION
PM7364-BI	256 Enhanced Ball Grid Array (SBGA)

Table 37 – FREEDM-32 Thermal Information

PART NO.	CASE TEMPERATURE	Theta Ja	Theta Jc
PM7364-BI	-40°C to +85°C	24 °C/W	2 °C/W

19 MECHANICAL INFORMATION

Figure 44 – 256 Pin Enhanced Ball Grid Array (SBGA)



NOTES: 1) ALL DIMENSIONS IN MILLIMETER.
2) DIMENSION AAA DENOTES COPLANARITY
3) DIMENSION BBB DENOTES PARALLEL
4) DIMENSION CCC DENOTES FLATNESS

PACKAGE TYPE: 256 PIN THERMAL BALL GRID ARRAY																
BODY SIZE: 27 X 27 X 1.45 MM																
DIM.	A	A1	A2	D	D1	E	E1	M,N	e	b	AAA	BBB	CCC	DDD	P	S
MIN.	1.41	0.56	0.85	26.90	24.03	26.90	24.03			0.60				0.15	0.20	
NOM.	1.54	0.63	0.91	27.00	24.13	27.00	24.13	20X20	1.27	0.75				0.33	0.30	
MAX.	1.67	0.70	0.97	27.10	24.23	27.10	24.23			0.90	0.15	0.15	0.20	0.50	0.35	0.635

NOTES

CONTACTING PMC-SIERRA, INC.

PMC-Sierra, Inc.
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
Web Site: http://www.pmc-sierra.com

PMC-1960758 (R6) ref PMC-1960113 (R10) Issue date: August 2001