PMC-HPDI32B

User's Manual

32Bit High Speed Digital

with Deep Transmit and Receive FIFOs (up to 128Kx32)

RS-485
PECL

General Standards Corporation
8302A Whitesburg Drive
Huntsville, AL 35802
Phone: (256) 880-8787
Fax: (256) 880-8788
URL: www.generalstandards.com
E-mail: techsupport@generalstandards.com

Rev 0

PREFACE

Revision History

1. Rev 0 - 02/21 - Original from PMC-HPDI32A Rev F

Additional copies of this manual or other literature may be obtained from:

General Standards Corporation
8302A Whitesburg Dr.
Huntsville, Alabama 35802
Tele: (256) 880-8787
FAX: (256) 880-8788
E-mail: support@generalstandards.com

The information in this document is subject to change without notice.

General Standards Corporation makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Although extensive editing and reviews are performed before release to ECO control, **General Standards Corporation** assumes no responsibility for any errors that may exist in this document. No commitment is made to update or keep current the information contained in this document.

General Standards Corporation does not assume any liability arising out of the application or use of any product or circuit described herein, nor is any license conveyed under any patent rights or any rights of others.

General Standards Corporation assumes no responsibility for any consequences resulting from omissions or errors in this manual or from the use of information contained herein.

General Standards Corporation reserves the right to make any changes, without notice, to this product to improve reliability, performance, function, or design.

RELATED PUBLICATIONS

The following manuals and specifications provide the necessary information for in depth understanding of the specialized parts used on this board.

PLX PCI 9080 Data Book

PLX Technology Inc.
390 Potrero Avenue
Sunnyvale, CA 4085
(408) 774-3735
http://www.plxtech.com/

EIA-422-A – Electrical Characteristics of Balanced Voltage Digital Interface Circuits
(EIA order number EIA-RS-422A)

EIA-485 – Standard for Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems
(EIA order number EIA-RS-485)

EIA Standards and Publications can be purchased from:

GLOBAL ENGINEERING DOCUMENTS
15 Inverness Way East
Englewood, CO 80112
Phone: (800) 854-7179
http://global.ihs.com/

IEEE P1386 - Standard Mechanic for a Common Mezzanine Card Family: CMC

IEEE P1386.1 - Standard Physical and Environmental Layers for PCI Mezzanine Cards: PMC
Sponsored by the Microprocessor & Microcomputer Standards Committee (MMSC) of the IEEE Computer Society

Copies of IEEE specifications available from:

Institute of Electrical and Electronics Engineers
Service Center
445 Hoes Lane
Piscataway, NJ 08855-1331 USA
http://www.ieee.org/

PCI Local Bus Specification Revision 2.1 June 1, 1995.

Copies of PCI specifications available from:

PCI Special Interest Group
NE 2575 Kathryn Street, #17
Hillsboro, OR 97124
http://www.pcisig.com/

TABLE of CONTENTS

CHAPTER 1: INTRODUCTION

1.0 FUNCTIONAL DESCRIPTION

The PMC-HPDI32B is intended to be a drop-in replacement for the General Standards PMC-HPDI32A board. The board has been redesigned due to obsolete parts on the PMC-HPDI32A. The PMC-HPDI32B can also be used with a PCI-to-PMC adapter as a replacement for the General Standards PCI-HPDI32A board. The PMC-HPDI32A and PCI-HPDI32A have been declared End Of Life products.

The PMC-HPDI32B Board is a high-speed 32-bit parallel digital interface card capable of transmitting or receiving data at up to 100 Mbytes per second (25 MHz Clock) over a RS-485 or PECL interface. Transmit and receive FIFOs of up to 128k words deep buffer transfer data between the PMC bus and the cable interface. This allows the HPDI32B to maintain maximum bursts on the cable interface (at least up to the depth of the FIFOs) independent of the PMC bus interface. The deep FIFOs buffer data between the cable interface and the PMC bus to maintain a sustained data throughput for real-time applications.

The HPDI32B interface is a half-duplex interface (board is either transmitting or receiving data, not simultaneous). The HPDI32B is easily set up to transfer data by initializing a few local registers. Once the data link is established, the data is transferred to/from the user application by simply reading or writing the on-board FIFOs. The board has an advanced PCI interface engine, which provides for increased data throughput via DMA.

The cable interface provides a flexible interface suited to most high speed parallel applications. All data to/from the board is synchronous with the transmit clock supplied by the transmitter. A Frame Valid signal indicates data is present on the interface, and programmable Line Valid and Status Valid signals provide additional interface capability. The interface also provides for data throttling by the receiving device – the receiver can hold off data the transmitting device until ready to receive. The HPDI32B interface is further programmable to allow the user to disable most of these standard interface functions and use the interface signals as discrete IO, including external interrupts.

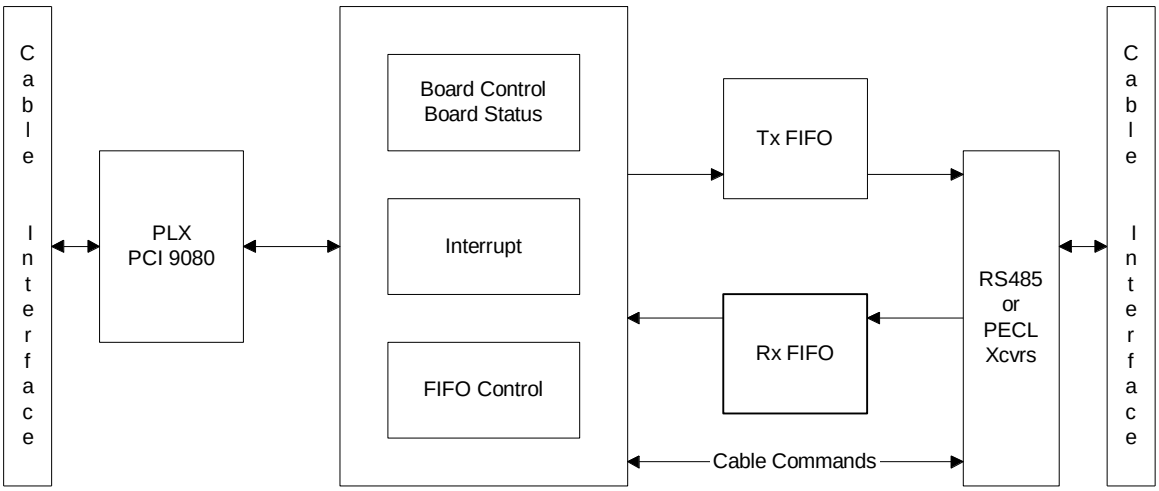


Figure 1.0: HPDI32B Block Diagram

1.1 CABLE INTERFACE

The cable interface consists of 32 bits of data, one clock, and 7 bi-directional signals. All cable interface signals are either differential RS485/RS422 or differential PECL, depending upon the version of the board. The clock and 7 bi-directional signals are used for controlling the data transfer and other pre-defined functions. Six of these signals may also be used for user IO or interrupt sources if the default function is not used.

Figure 1.1 shows the default HPDI32B interface control signals. All control signals and data are synchronous to the cable interface clock supplied by the transmitting device. The on-board transmit clock (20MHz standard) may be up to 25MHz. A Frame Valid signal indicates valid data is being transferred. Two optional programmable signals, Line Valid and Status Valid, can be used to further qualify the data. The Status Valid signal is used to flag a programmable number of words at the start of the cycle as status words. The Line Valid signal can be used to signify valid data within a frame. If the Line Valid signal is used, data is only recorded when the Line Valid signal is asserted. The standard cable interface also provides for a cable-throttling signal to pause data transmission. This signal is driven by the receiving device to indicate it is capable of receiving data.

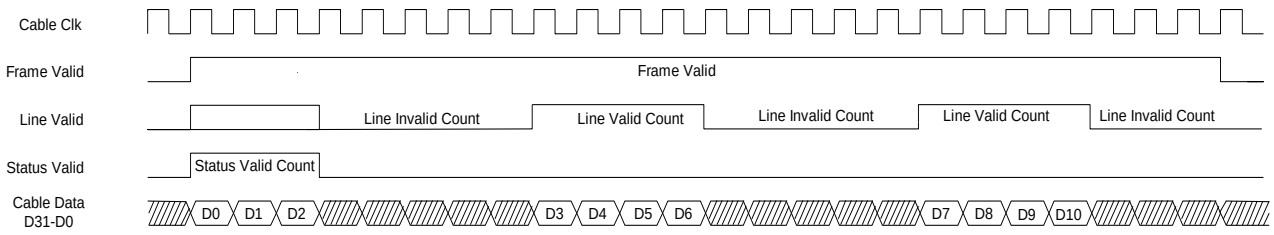


Figure 1.1: HPDI32B Cable Interface

1.1.1 CABLE INTERFACE SIGNALS

Data is transferred to and from the HPDI32B board via an 80 pin cable interface consisting of 40 differential signals – 1 Clock, 7 Command /Control signals, and 32 Data bits. The seven Command/Control signals provide many cable protocol options. The default Cable Command signals are:

Command 0 – Frame Valid. Provides an indication that data is currently being transferred on the cable. This signal is driven by the transmitter and must be asserted before data is recorded. Frame Valid may be disabled via the Board Control Register and used as discrete IO, although this is only useful for test purposes.

Command 1 – Line Valid. Programmable signal to allow for multiple lines or rows within a frame. If Line Valid and Line Invalid counters are used during transmit, the signal will be and negated for Line Invalid Count and asserted for Line Valid Count, alternating for the entire frame. The Line Invalid Count begins after the Status Valid Count. Board Control Register Bit 6 controls the Line Valid state during the Status Word Count. If Line Valid is enabled but the Line Valid counter is zero (default), Line Valid will be asserted for the entire Frame. Line Valid may be disabled via the Board Control Register and used as discrete IO – GPIO0.

Command 2 – Status Valid.	Programmable signal to allow for status words at the start of frame. If Status Valid counters are used during transmit, the signal will be asserted for Status Valid Count at the beginning of every frame, and negated for the remainder of the frame. Status Valid may be disabled used as discrete IO..
Command 3 – Rx Ready.	Provides a method for the receiving device to pause the data transfer. The receiving device drives this signal. This function is enabled by Board Control Bit D9. When the HPDI32B is in receive mode, this signal is negated when the Rx FIFO Almost Full flag is reached. Rx Ready may be disabled and used as discrete IO..
Command 4 – Tx Data Ready	Provides a signal to indicate data is present in the Tx FIFO. The transmitting device drives this signal Tx Data Ready may be disabled and used as discrete IO.
Command 5 – Tx Enabled	Provides a signal to indicate the HPDI32B is in transmit mode (Board Control Bit D4). If enabled, the HPDI32B card always drives this signal. Tx Enabled may be disabled and used as discrete IO.
Command 6 – Rx Enabled	Provides a signal to indicate the HPDI32B is in receive mode (Board Control Bit D5). If enabled, the HPDI32B card always drives this signal. Rx Enabled may be disabled and used as discrete IO.

Data is transferred using the Cable Clock. All transmit data and command/control signals are clocked on the rising edge of the TxCLK. The transmit clock on the HPDI32B is supplied by an on-board oscillator. This oscillator is socketed so the user can customize the clock interface speed. The board is shipped with a 20MHz oscillator standard, but may run up to 25MHz. To ensure maximum setup and hold times, all receive data and Command/Control signals are clocked on the falling edge of the RxClk. Figure 2.1 shows the data setup and hold times.

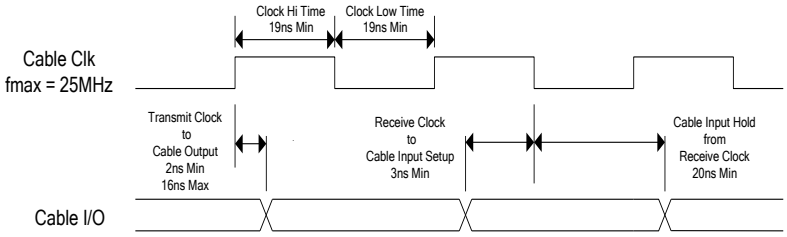


Figure 1.2: Cable Interface Timing

1.2 FIFOs

The FIFOs on the PMC-HPDI32B are used for buffering the transmit or receive data. This allows the data on cable interface to run independent of the PCI interface. There are two sets of FIFOs on the board; a set of four FIFOs for transmit data, and a second set of four for receive data. Each set consists of 32 bits of data and 4 status flags. The receive FIFOs are loaded by the cable receive control logic and read by either the CPU or the DMA. The transmit FIFOs are loaded by either the CPU or the DMA and read by the cable transmit control logic. Four status flags accompany each set the FIFOs: Empty, Almost Empty, Almost Full, Full. The Almost Empty and the Almost Full status flags can be programmed via software to assert most desired levels within the FIFOs. These programmable flags can be used for DMA control, throttling cable data, or to indicate when a desired amount of data has been received.

CHAPTER 2: PROGRAMMING

2.0 INITIALIZATION

Several functions on the HPDI32B board will generally be unchanged in a given application. These include interrupt setup, FIFO Almost Flag Values, and General Purpose IO direction setup. Therefore, initializing these functions board will generally need to be done only once by the software. However, if a Board Reset is performed, all registers will return to their default values. Software must reinitialize the board following a Board Reset.

2.1 RESETS

There are three bits in this Board Control Register that are used as resets to the local logic functions. These bits perform a reset when the software writes a '1'. The software does not need to clear the bits following a Reset since the reset bits are self-clearing

Board Reset	Board Control Register D0 – Setting this bit will reset the local logic, reset (clear) the FIFOs and program the default Almost Flag Values, and place all registers into a known state.
Tx FIFO Reset	Board Control Register D1 – Setting this bit will clear the Tx FIFOs and program the current Almost Flag Values.
Rx FIFO Reset	Board Control Register D2 – Setting this bit will clear the Rx FIFOs and program the current Almost Flag Values.

2.2 FIFOs

The FIFOs on the PMC-HPDI32B are used for buffering the transmit or receive data. This allows the data on cable interface to run independent of the PCI interface, ensuring data will be transferred on the cable regardless of software overhead to the board (assuming the average receive throughput can be maintained). The board allows for different FIFO depths to be installed from 8k to 128k (32k standard). This allows the user to customize the board for specific real-time applications. The FIFO depth must be specified when ordering – consult factory for further information.

There are two sets of FIFOs on the board – one for transmit and one for receive. Each set has a width of 32 bits of data and provides four FIFO status flags. These flags indicate Empty, Full, Almost Empty, and Almost Full. Two of these flags, Almost Empty and Almost Full are user programmable. These programmable flags are used for Demand Mode DMA (see Section 3.3), or may be used for cable throttling (pause transmitter when Rx FIFO Almost Full) or to provide the user with a trigger at a specific FIFO level. The Almost Flags of the FIFOs are programmed with the Almost Register values during a FIFO Reset (See Resets, Section 3.1).

NOTE: The Almost Empty Flag value represents the number of words from empty. The Almost Full Flag value represents the number of words from Full –1 (not the number of words from Empty.)

The board also provides FIFO Size registers for both transmit and receive FIFOs. These registers may be useful if writing software (especially drivers) which needs to support multiple FIFO depth. The board also tracks the number of words currently in each FIFO. This count may be useful when accessing the FIFOs to prevent underruns or overflows.

2.3 INTERRUPTS

The HPDI32B contains both PLX 9080 PCI Interface interrupt sources (DMA done is the most useful), and the local HPDI32B firmware has 16 potential interrupt sources. All local interrupts are passed through the PLX chip and will be requested on PCI INTA. In order for this board to generate interrupts to the PCI bus, they must be enabled in the PLX chip. The device driver typically handles enabling board interrupts, so the average user should not be concerned with accessing the PLX registers.

The 16 local interrupt sources allow for interrupts on Frame Start, Frame End, six cable inputs, and all FIFO flags. All interrupt sources are can be individually enabled, allowing the user to enable some interrupts and leave others disabled. The Interrupt Control Register (ICR) allows interrupts to be enabled individually..

Each interrupt source is individually programmable as Rising Edge, Falling Edge, Level High, or Level Low triggering. All interrupts default to edge triggered to prevent a potential interrupt lockup seen with level interrupts (the level interrupt may keep re-interrupting indefinitely). Since interrupts are latched, edge triggered interrupts are more user friendly. The interrupts default at reset to the most common configuration, but this configuration may not suit all users. For example, one may wish to interrupt when the Tx FIFO is almost full to stop writing data to the FIFO, while another may wish to interrupt when it is not almost full to know when to resume writing to the FIFO. The HPDI32B provides the user the flexibility of interrupt configuration.

Interrupts are latched in the Interrupt Status Register (ISR). Since all interrupts are multiplexed onto a single interrupt request, the ISR provides the means of determining the unique interrupt source. The user should clear the interrupt by writing the specific bit back to the ISR as part of the interrupt service routine.

2.4 DMA

Although the 33MHz PMC/PCI bus is capable of burst transfers up to 132Mbytes per sec, actual sustained throughput on the PCI bus will be much lower. This is due to many factors such as bus overhead, operating system overhead, application overhead, and possibly data storage overhead such as hard disk drive accesses. Since sustained PCI data rates will be typically slower than the maximum cable interface rate, DMA on the PCI bus is supported to make the PCI data transfers as fast as possible.

There are 3 methods the software application can move data to and from the HPDI32B board: Programmed IO (PIO) , Non-Demand DMA, and Demand Mode DMA. The two DMA modes are only supported to the on-board data FIFOs (DMA accesses to local registers are not supported).

2.4.1 PROGRAMMED IO (PIO) MODE

In PIO Mode (Programmed IO - No DMA), the user accesses the FIFOs through is single register reads and writes to the board. This is the slowest data transfer mode. In PIO mode, the user must check to make sure the Tx FIFO is not full prior to writing the Tx FIFOs, and check to ensure the Rx FIFO contains data before reading. The device driver should normally handle checking of this status prior to reading/writing the FIFO. See specific driver information for further details.

2.4.2 NON-DEMAND MODE DMA

In Non-Demand DMA mode, the user specifies a DMA transfer size and initiates the transfer. Since there is no checking for data validity (no check to determine if Tx FIFO Full or Rx FIFO empty), the user must ensure the FIFOs can handle the entire transfer prior to transfer initiation. If the Tx FIFO becomes full or Rx FIFO Empty during a Non-Demand mode DMA transfer, the extra data will be discarded and no error indication is received – the DMA appears to have completed normally. Therefore, the user should always check the FIFO count and never attempt to transfer more data than is present in the FIFO. When performed correctly, Non-Demand DMA is the preferred DMA method.

2.4.3 DEMAND DMA

Demand Mode DMA is similar to Non-Demand DMA, except the data will automatically be transferred based on FIFO availability. After the user specifies a DMA transfer size and initiates the transfer, the data transfer if the Tx FIFO is not Full (transmit) or the Rx FIFO is not empty (receive).

When transmit Demand DMA is initiated, data will be transferred to the Tx FIFO. If the Tx FIFO becomes full, the logic will pause requesting transmit data.. As data is transmitted out of the FIFO, the request will again be asserted to re-fill the FIFO. This will resume until the transfer completes.

When receive Demand DMA is initiated, the DMA is requested whenever data is in the Rx FIFO. If the Rx FIFO becomes empty, the DMA will be paused until more data is available in the Rx FIFO. This will continue until the transfer completes.

While Demand Mode eliminates some overhead required for Non-Demand Mode (checking FIFO counts and calculating space is available in FIFO before initiating a DMA transfer), it has the potential to cause problems if the Demand Mode DMA is not completed normally. In this case, the DMA will need to be aborted/reset to prevent locking up and data may be lost. For this reason, Non-Demand DMA is the preferred method of DMA.

2.4 GENERAL PURPOSE IO (GPIO)

Since most users will not require all of the default Cable Command functions, six of the cable command signals can be changed from their default function and used as discrete IO. This allows users to control cable outputs, read cable inputs, or receive cable interrupts via simple software control. The General Purpose IO bits are controlled from the Board Control register. If a bit is set as a discrete output, the Board Control register also defines the output value. If a bit is set as an input, the Cable Command state may be read through the Board Status register. To set a Cable Command as an interrupt source, set the signal as a discrete input, and setup the interrupt source via the interrupt registers.

The default configuration for the Cable Command bits is the factory defined cable functions. The user will need to re-initialize General Purpose IO setup following a board reset. If a Cable Command signal is changed from its default function, all on-board logic associated with that function is disabled.

2.5 LOOPBACK TESTING

Even though the HPDI32B interface is half-duplex (i.e. the interface signals are defined as transmit mode or receive mode), the HPDI32B is capable of receiving the data as it is being transmitted. This allows for a single board loopback testing.

To prevent erroneous data from being received, the transmit interface should be enabled before the receiver is enabled. Then the receiver may be enabled, and transmitting of data may begin.

CHAPTER 3: LOCAL SPACE REGISTERS

3.0 LOCAL REGISTERS

The Local Space registers control the transmission and reception of data to and from the board.

Offset Address	Size	Access*	Register Name	Value after Reset
0x00	D32	RO	Firmware Revision	0xC20302XX
0x04	D32	RW	Board Control	0x00000000
0x08	D32	RC	Board Status	0x00X0CCXX
0x0C	D32	RW	Tx Programmable Almost	0x0010000F
0x10	D32	RW	Rx Programmable Almost	0x0010000F
0x14	D32	RO	Features	0x000005FF
0x18	D32	RW	Rx/Tx FIFOs	0x00000000
0x1C	D32	RW	Tx Status Length Count	0x00000000
0x20	D32	RW	Tx Line Valid Length Count	0x00000000
0x24	D32	RW	Tx Line Invalid Length Count	0x00000000
0x28	D32	RO	Rx Status Counter	0x00000000
0x2C	D32	RO	Rx Line Counter	0x00000000
0x30	D32	RW	Interrupt Control	0x00000000
0x34	D32	RC	Interrupt Status	0x00000000
0x38	D32	RW	Tx Clock Divider	0x00000000
0x3C			Reserved	
0x40	D32	RO	Tx FIFO Size	0x000XXX00
0x44	D32	RO	Rx FIFO Size	0x000XXX00
0x48	D32	RO	Tx FIFO Words	0x00000000
0x4C	D32	RO	Rx FIFO Words	0x00000000
0x50	D32	RW	Interrupt Edge/Level	0x0000FFFF
0x54	D32	RW	Interrupt Hi/Lo	0x0000CFFD
0x58-0x5C			Reserved	
0x60	D32	RW	Test (Unused)	0x00000000
0x64	D32	RW	Tx Clock	0x00000000

* RO - read only
RW - read/write capability
RC - read clear (a write clears the specified bits)

Table 3.1 Local Register Map

3.1 FIRMWARE REVISION REGISTER (Offset 0x00000000)

Information Register to show current firmware version and general board information.

D31	**1** = Features Register Present
D30	**1** = FW ID complies with this standard
D29	Unused
D28	1=64bit PCI bus **0**=32bit PCI bus
D27:D24	Form Factor 0 = Reserved 1 = PCI **2** = PMC 3 = cPCI 4 = PC104P
D23:D16	HW Rev of development board **3** = Rev C
D15:D8	Board Version **0x02** = HPDI32A/B Standard
D7:D0	FW Revision

3.2 BOARD CONTROL REGISTER (Offset 0x00000004)

Controls miscellaneous functions including Resets, Transmit/Receive Enables, and GPIOs.

D0	Board Reset Reset the on-board logic and the FIFOs (Automatic clear). The Rx/Tx FIFOs are also reset and reprogrammed with the default values following a board reset. To allow FIFO programming to occur, the user should wait about 10usec following Board Reset before the FIFOs are accessed.
D1	Tx FIFO Reset Reset the Tx FIFOs (Automatic clear). Also programs the Tx Almost Empty and Tx Almost Full flags from Tx Almost Programming Register after Reset(See Section 3.3).
D2	Rx FIFO Reset Reset the Rx FIFOs. (Automatic clear). Also programs the Rx Almost Empty and Rx Almost Full flags from Rx Almost Programming Register after Reset(See Section 3.4).
D3	Reserved
D4	Transmit Enable Enable data transmission. When Transmit Enabled, drive Cable Data, TxClk, Frame Valid, Line Valid (if enabled), Status Valid (if enabled), and Tx Data Ready (if enabled).
D5	Receive Enable Enable data reception into the Rx FIFOs from the Cable. When Receive Enabled, drive Rx Ready (if enabled).
D6	Demand DMA Direction Set the demand mode DMA direction . 1 = Transmit DMA, 0 = Rx DMA. This bit is not used in half-duplex operation, but is added for future full-duplex operation.
D7	Line Valid Hi On Status Valid Hi Assert the Line Valid signal while Status Valid is asserted. This bit is unused if Line Valid is not used or Status Valid counter is not used.
D8	Start Transmit Start data transmission. Automatically resets when Tx FIFO is empty (no data to send).

D9 Cable Throttle Enable
Allow Rx Ready input (Tx Cable Command D3) to throttle the data transmission. If enabled (and Tx enabled), data will be transmitted while Rx Ready is asserted.

D10 TxDMA DMA
Set DMA Channel 0 to Tx FIFO for Demand Mode DMA.

D11 DMA Single Cycle Disable
Disable single word transfers during Demand Mode DMA.

D15:D12 Reserved.

D30:D16 GPIO Control.

D24/D16 Cable Command D0 Control

D16=0 / D24=0	Function = Frame Valid
D16=0 / D24=1	Function = Discrete Input
D16=1 / D24=0	Function = Discrete Output Lo
D16=1 / D24=1	Function = Discrete Output Hi

D25/D17 Cable Command D1 Setup Control

D17=0 / D25=0	Function = Line Valid
D17=0 / D25=1	Function = Discrete Input
D17=1 / D25=0	Function = Discrete Output Lo
D17=1 / D25=1	Function = Discrete Output Hi

D26/D18 Cable Command D2 Setup Control

D18=0 / D26=0	Function = Status Valid
D18=0 / D26=1	Function = Discrete Input
D18=1 / D26=0	Function = Discrete Output Lo
D18=1 / D26=1	Function = Discrete Output Hi

D27/D19 Cable Command D3 Setup Control

D19=0 / D27=0	Function = Rx Ready
D19=0 / D27=1	Function = Discrete Input
D19=1 / D27=0	Function = Discrete Output Lo
D19=1 / D27=1	Function = Discrete Output Hi

D28/D20 Cable Command D4 Setup Control

D20=0 / D28=0	Function = Tx Data Ready
D20=0 / D28=1	Function = Discrete Input
D20=1 / D28=0	Function = Discrete Output Lo
D20=1 / D28=1	Function = Discrete Output Hi

D29/D21 Cable Command D5 Setup Control

D21=0 / D29=0	Function = Tx Enabled
D21=0 / D29=1	Function = Discrete Input
D21=1 / D29=0	Function = Discrete Output Lo
D21=1 / D29=1	Function = Discrete Output Hi

D30/D22 Cable Command D6 Setup Control

D22=0 / D30=0	Function = Rx Enabled
D22=0 / D30=1	Function = Discrete Input
D22=1 / D30=0	Function = Discrete Output Lo
D22=1 / D30=1	Function = Discrete Output Hi

D23 Reserved.

D31 Test Mode Enable (Factory Test Only)
Disable driving Tx Enabled (Cable Cmd D5) and Rx Enabled (Cable Cmd D6). This prevents contention if two HPDI32B boards are cabled together during test.

3.3 BOARD STATUS REGISTER (Offset 0x00000008)

Current status of on-board signals, including the FIFO status flags and the Cable Command signals.

D0	Cable Command D0
D1	Cable Command D1
D2	Cable Command D2
D3	Cable Command D3
D4	Cable Command D4
D5	Cable Command D5
D6	Cable Command D6
D7	Start Transmit Indicates data transmission is in currently in progress
D8	Tx FIFO Not Empty Indicates the Tx FIFO is NOT Empty
D9	Tx FIFO Not Almost Empty Indicates the Tx FIFO is NOT Almost Empty
D10	Tx FIFO Not Almost Full Indicates the Tx FIFO is NOT Almost Full
D11	Tx FIFO Full Indicates the Tx FIFO is NOT Full
D12	Rx FIFO Not Empty Indicates the Rx FIFO is NOT Empty
D13	Rx FIFO Not Almost Empty Indicates the Rx FIFO is NOT Almost Empty
D14	Rx FIFO Not Almost Full Indicates the Rx FIFO is NOT Almost Full
D15	Rx FIFO Full Indicates the Rx FIFO is NOT Full
D17:D16	Reserved
D18	Transceiver Option 0 = RS-485 transceivers installed. 1 = PECL transceivers installed.
D20:D19	Reserved
D21	Tx OverRun Indicates an attempt to write to the Rx FIFO when the FIFO was Full. This bit is latched. It is cleared by writing a '1' to D21.
D22	Rx UnderRun Indicates an attempt to read from the Rx FIFO when the FIFO was Empty. This bit is latched. It is cleared by writing a '1' to D22.
D23	Rx OverRun Indicates an attempt to write to the Rx FIFO when the FIFO was full. This bit is latched. It is cleared by writing a '1' to D23.
D24:31	Reserved

3.4 TX ALMOST FLAG REGISTER (Offset 0x0000000C)

This register is sets the programmed values for the Tx FIFO Almost Empty and Almost Full Flags. This value is programmed following a Tx FIFO Reset. A Board Reset will reset this register to the default value (0x0010000F) and then program this value.

D15:0	Almost Empty Flag Value
D31:16	Almost Full Flag Value (# of available words remaining in FIFO – 1 when Flag asserted)

3.5 RX ALMOST FLAG REGISTER (Offset 0x00000010)

This register is sets the programmed values for the Rx FIFO Almost Empty and Almost Full Flags. This value is programmed following an Rx FIFO Reset. A Board Reset will reset this register to the default value (0x0010000F) and then program this value.

D15:0	Almost Empty Flag Value
D31:16	Almost Full Flag Value (# of available words remaining in FIFO – 1 when Flag asserted).

3.6 FEATURES REGISTER (Offset 0x00000014)

This Register indicates new features in each firmware version. This allows a driver to maintain compatibility across firmware and board revisions, while providing for new features to be added.

D31:D11	Unused
D10	Tx Clock Measurement
D9	Factory Testing Mode
D8	Local Interrupt disable
D7	Demand Mode DMA single cycle disable present
D6	General Purpose IO on Cable Command D1 Supported Stop on Tx bit present (D6)
D5	Tx/Rx Underrun & Overrun flags
D4	PLX DMA Channel 1 Supported
D3	General Purpose IO on Cable Command D6:2 Supported
D2	Level/Edge Triggered Interrupts Supported
D1	FIFO Count Registers Present
D0	FIFO Size Registers Present

3.7 TX FIFO/RX FIFO (Offset 0x00000018)

Transmit and receive FIFOs data. Writing loads data into the Tx FIFO for transmission, while reading gets incoming data from the Rx FIFO. The user must ensure that the Tx FIFO is not full when writing or data will be discarded. The user must also ensure that the Rx FIFO is not empty when reading or data returned will be indeterminate.

D31:0	FIFO Data

3.8 TX STATUS LENGTH COUNT (Offset 0x0000001C)

Number of clocks the Status Valid signal will be asserted at the start of each transmit frame. This register is only valid if Cable Command D2 function is set as Status Valid (unused if GPIO).

D31:0 Tx Status Length Count Value

3.9 TX LINE VALID LENGTH COUNT (Offset 0x00000020)

Number of clocks that the Line Valid signal will be asserted after the Tx Line Invalid Length Counter has expired in a frame. The Line Valid signal will continue to alternate between the Tx Line Valid Length Counter and the Tx Line Invalid Length Counter until the end of the frame. This register is only valid if Cable Command D1 is set as Line Valid (unused if GPIO).

D31:0 Tx Line Valid Length Count Value

3.10 TX INVALID LENGTH COUNT (Offset 0x00000024)

Number of clocks that the Line Valid signal will be negated at the beginning of each transmit frame (following Status Valid count) or after the Tx Line Valid Length count has expired in a frame. The Line Valid signal will continue to pulse - alternating between the Tx Line Valid Length Counter and the Tx Line Invalid Length Counter until the end of the frame. This register is only valid if Cable Command D1 is set as Line Valid (unused if GPIO).

D15:0 Tx Line Invalid Length Count Value
D31:16 Reserved

3.11 RX STATUS LENGTH COUNTER (Offset 0x00000028)

Number of clocks Status Valid was asserted during the last received frame. This register will reset at the start of the next frame.

If Cable Command D2 is set as GPIIO (instead of Status Valid), this register will count the total number received words since Receive Enabled. This register will reset when receive is disabled.

D31:0 Rx Status Length Counter

3.12 RX LINE LENGTH COUNTER (Offset 0x0000002C)

Number of clocks Line Valid was asserted during the last received frame. This register is only valid if Cable Command D1 is set as Line Valid (unused if GPIO).

D31:0 Rx Line Length Counter

3.13 INTERRUPT CONTROL REGISTER (Offset 0x00000030)

Enables the Local Interrupt Sources to generate a Local Interrupt request.
See Section 3.3 for more detailed explanation of Interrupts.

D0	IRQ Source 0 - Enable Frame Valid (rising edge – Frame Start)
D1	IRQ Source 1 - Enable Frame Valid (falling edge - Frame End)
D2	IRQ Source 2 - Enable Cable Command D1 (rising edge)
D3	IRQ Source 3 - Enable Cable Command D2 (rising edge)
D4	IRQ Source 4 - Enable Cable Command D3 (rising edge)
D5	IRQ Source 5 - Enable Cable Command D4 (rising edge)
D6	IRQ Source 6 - Enable Cable Command D5 (rising edge)
D7	IRQ Source 7 - Enable Cable Command D6 (rising edge)
D8	IRQ Source 8 - Enable Tx FIFO Empty (rising edge - Tx FIFO Empty)
D9	IRQ Source 9 - Enable Tx FIFO Almost Empty (rising edge - Tx FIFO AEmpty)
D10	IRQ Source 10 - Enable Tx FIFO Almost Full (rising edge - FIFO AFull)
D11	IRQ Source 11 - Enable Tx FIFO Full (rising edge - Tx FIFO Full)
D12	IRQ Source 12 - Enable Rx FIFO Empty (falling edge - Rx FIFO Not Empty)
D13	IRQ Source 13 - Enable Rx FIFO Almost Empty (falling edge – Rx FIFO Not AEmpty)
D14	IRQ Source 14 - Enable Rx FIFO Almost Full (rising edge – Rx FIFO AFull)
D15	IRQ Source 15 - Enable Rx FIFO Full (rising edge – Rx FIFO Full)
D31:16	Reserved

3.14 INTERRUPT STATUS REGISTER (Offset 0x00000034)

Interrupt status for each of the interrupt sources. If an interrupt is enabled in the Interrupt Control register, the interrupt will be latched in this register until cleared via writing a '1' back to the respective bit. If an interrupt is not enabled, each bit will represent the current state of the interrupt source (although the interrupt will not be latched and will not generate a Local Interrupt request).

D15:0	IRQ 15:0 Status
D31:16	Reserved

3.15 TRANSMIT CLOCK DIVIDER REGISTER (Offset 0x00000038)

Allow the On-board Transmit clock to be dived down based on this register is the clock divider value. If this register contains 0 (default) or 1, the On-board clock is used as the Transmit clock.

D15:0	Tx Clock Divide Value
D31:16	Reserved

3.16 TRANSMIT FIFO SIZE (Offset 0x00000040)

Transmit FIFO depth. This is the true FIFO depth, regardless of the FIFO data width.

D19:0	Tx FIFO Depth
D31:0	Reserved

3.17 RECEIVE FIFO SIZE (Offset 0x00000044)

Receive FIFO depth. This is the true FIFO depth, regardless of the FIFO data width.

D19:0 Rx FIFO Depth
D31:0 Reserved

3.18 TRANSMIT FIFO WORDS (Offset 0x00000048)

Current number of words in the Transmit FIFO.

D19:0 Current Number of Words in Tx FIFO
D31:20 Reserved

3.19 RECEIVE FIFO WORDS (Offset 0x0000004C)

This register will track the current number of words in the Receive FIFO.

D19:0 Current Number of Words in Rx FIFO
D31:0 Reserved

3.20 INTERRUPT EDGE/LEVEL REGISTER (Offset 0x00000050)

Defines the interrupt source as Level Hi/Lo or Rising Edge/ Falling Edge triggered (along with the Interrupt Hi/Lo register) . 1 = Edge triggered (default) ; 0 = Level triggered.

D15:0 IRQ 15:0 Edge Trigger Enable
D31:16 Reserved

3.21 INTERRUPT HI/LO REGISTER (Offset 0x00000054)

Defines the interrupt source as Level Hi/Lo or Rising Edge/ Falling Edge triggered (along with the Interrupt Edge/Level register) . 1 = Active Hi (default) ; 0 Active Lo..

D15:0 IRQ 15:0 Active Hi Enable
D31:16 Reserved

3.22 TEST REGISTER (Offset 0x00000064)

Future resister to provides test bits for factory testing. Do not set any bits in this register.

3.23 TX CLOCK REGISTER (Offset 0x00000064)

Measured frequency of Tx Clock (with sub-divider) in MHz.

CHAPTER 4: PCI INTERFACE

4.0 PCI INTERFACE REGISTERS

A PCI9080 I/O Accelerator from PLX Technology handles the PCI Interface. The PCI interface is compliant with the 5V, 33MHz 32-bit PCI Specification 2.1. The PCI9080 provides dual DMA controllers for fast data transfers to and from the on-board FIFOs. Fast DMA burst accesses provide for a maximum burst throughput of 132MB/s to the PCI interface. To reduce CPU overhead during DMA transfers, the controller also implements Chained (Scatter/Gather) DMA, as well as Demand Mode DMA.

Since many features of the PCI9080 are not utilized in this design, it is beyond the scope of this document to duplicate the PCI9080 User's Manual. Only those features, which will clarify areas specific to the PCI-X are detailed here. Please refer to the PCI9080 User's Manual (See Related Publications) for more detailed information. Note that the BIOS configuration and software driver will handle most of the PCI9080 interface. Unless the user is writing a device driver, the details of this PCI Interface Chapter may be skipped.

4.1 PCI REGISTERS

The PLX 9080 contains many registers, many of which have no effect on the HPDI32B performance. The following section attempts to filter the information from the PCI9080 manual to provide the necessary information for a HPDI32B specific driver.

The HPDI32B uses an on-board serial EEPROM to initialize many of the PCI9080 registers after a PCI Reset. This allows board specific information to be preconfigured.

4.1.1 PCI CONFIGURATION REGISTERS

The PCI Configuration Registers allow the PCI controller to identify and control the cards in a system.

PCI device identification is provided by the Vendor ID/Device ID (Addr 0x0000) and Sub-Vendor ID/Sub-Device ID Registers (0x002C). The following definitions are unique to the General Standards HPDI32B boards. All drivers should verify the ID/Sub-ID information before attaching to this card. These values are fixed via the Serial EEPROM load following a PCI Reset, and cannot be changed by software.

Vendor ID	0x10B5	PLX Technology
Device ID	0x9080	PCI9080
Sub-Vendor ID	0x10B5	PLX Technology
Sub-Device ID	0x2400	GSC HPDI32

The configuration registers also setup the PCI IO and Memory mapping for the HPDI32B. The PCI9080 is setup to use PCIBAR0 and PCIBAR1 to map the internal PLX registers into PCI Memory and IO space respectively. PCIBAR2 will map the Local Space Registers into PCI memory space, and PCIBAR3 is unused. Typically, the OS will configure the PCI configuration space.

For further information of the PCI configuration registers, consult the PLX Technology PCI9080 Manual.

4.1.2 LOCAL CONFIGURATION REGISTERS

The Local Configuration registers give information on the Local side implementation. These include the required memory size. The HPDI32B memory size is initialized to 4k Bytes. All other Local Registers initialize to the default values described in the PCI9080 Manual.

4.1.3 RUNTIME REGISTERS

The Runtime registers consist of mailbox registers, doorbell registers, and a general-purpose control register. The mailbox and doorbell registers are not used and serve no purpose on the HPDI32B. All other Runtime Registers initialize to the default values described in the PCI9080 Manual.

4.1.4 DMA REGISTERS

The Local DMA registers are used to setup the DMA transfers to and from the on-board FIFOs. DMA is supported only to the four FIFO locations. The HPDI32B supports both Demand (DREQ# controlled) and Non-Demand mode DMA. Both Channel 0 and Channel 1 DMA are supported.

4.1.4.1 DMA CHANNEL MODE REGISTER: (PCI 0x80 / 0x94)

The DMA Channel Mode register must be setup to match the hardware implementation

Bit	Description	Value	Notes
D1:0	Local Bus Width	11 = 32 bit	
D5:2	Internal Wait States	0000 = Unused	
D6	Ready Input Enable	1 = Enabled	
D7	Bterm# Input Enabled	0 = Unused	
D8	Local Burst Enable	1 = Supported	Bursting allows fast back-to-back accesses to the FIFOs to speed throughput
D9	Chaining Enable (Scatter Gather DMA)	X	DMA source addr, destination addr, and byte count are loaded from memory in PCI Space.
D10	Done Interrupt Enable	X	DMA Done Interrupt
D11	Local Addressing Mode	1 = No Increment	DMA to/from FIFOs only
D12	Demand Mode Enable	X	Demand Mode DMA is supported for FIFO accesses on the HPDI32B. (See Section 3.3)
D13	Write & Invalidate Mode	X	
D14	DMA EOT Enable	0 = Unused	
D15	DMA Stop Data Transfer Enable	0 = BLAST terminates DMA	
D16	DMA Clear Count Mode	0 = Unused	
D17	DMA Channel Interrupt Select	X	
D31:18	Reserved	0	

CHAPTER 5: HARDWARE CONFIGURATION

5.0 BOARD LAYOUT

The following figure is a drawing of the physical components of the PMC-HPDI32B:

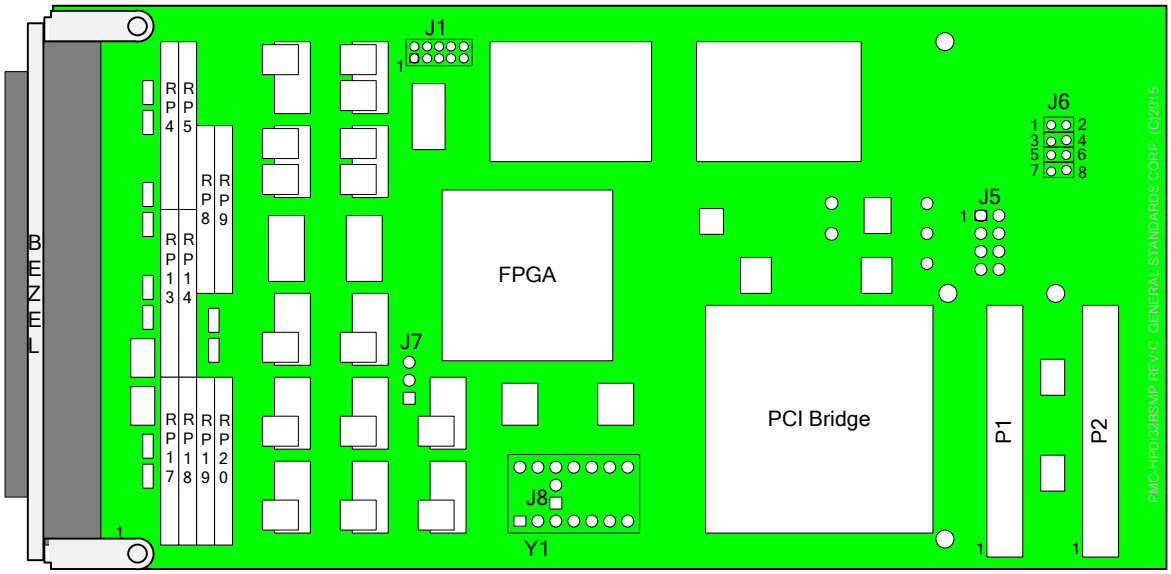


Figure 5-1: Board Layout – Top

5.1 HARDWARE JUMPERS

There are no user configurable jumpers present on the PMC-HPDI32B. Jumper J6 is for future enhancement.

5.2 TERMINATION RESISTORS

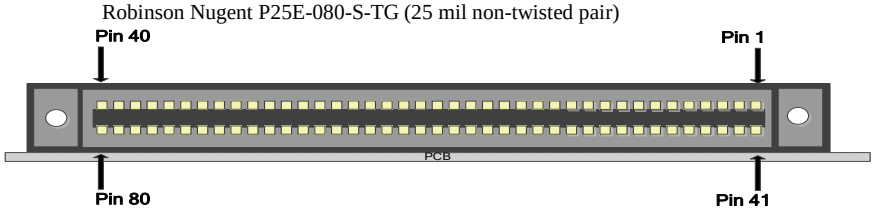
The PMC-HPDI32B is designed with socketed parallel termination resistors for all cable signals. These termination resistors are 8 pin SIPs. There are 10 termination SIPs – RP4, RP5, RP8, RP9, RP13, RP14, RP17, RP18, RP19, and RP20 . Refer to Figure 5-1 for resistor pack locations.

The factory installed parallel resistors for RS422/RS485 termination are 120 Ω. Contact General Standards Corporation if a different termination value is required.

For PECL, two termination variations are available. The -P1 Ordering Option incorporates a 220 Ohm Output Termination / 120 Ohm Input Termination for each signal. The -P2 Ordering Option us unterminated. (This corresponds to previous PMC-HPDI32A PECL Options).

5.3 CABLE INTERFACE CONNECTOR

User I/O Connector: 80 pin IO connector (female)
Part Number: Robinson Nugent P50E-080-P1-SR1-TG
Mating Connector: Robinson Nugent P50E-080-S-TG (50 mil twisted pair)
Robinson Nugent P25E-080-S-TG (25 mil non-twisted pair)



Pin #	Signal Name	Pin #	Signal Name
1	CLKp	41	D12p
2	CLKn	42	D12n
3	CMD0p	43	D13p
4	CMD0n	44	D13n
5	CMD1p	45	D14p
6	CMD1n	46	D14n
7	CMD2p	47	D15p
8	CMD2n	48	D15n
9	CMD3p	49	D16p
10	CMD3n	50	D16n
11	CMD4p	51	D17p
12	CMD4n	52	D17n
13	CMD5p	53	D18p
14	CMD5n	54	D18n
15	CMD6p	55	D19p
16	CMD6n	56	D19n
17	D0p	57	D20p
18	D0n	58	D20n
19	D1p	59	D21p
20	D1n	60	D21n
21	D2p	61	D22p
22	D2n	62	D22n
23	D3p	63	D23p
24	D3n	64	D23n
25	D4p	65	D24p
26	D4n	66	D24n
27	D5p	67	D25p
28	D5n	68	D25n
29	D6p	69	D26p
30	D6n	70	D26n
31	D7p	71	D27p
32	D7n	72	D27n
33	D8p	73	D28p
34	D8n	74	D28n
35	D9p	75	D29p
36	D9n	76	D29n
37	D10p	77	D30p
38	D10n	78	D30n
39	D11p	79	D31p
40	D11n	80	D31n

CHAPTER 6: ORDERING INFORMATION

6.0 ORDERING INFORMATION

Since the PMC-HPDI32B is designed to fit a variety of high-speed digital interface needs, there are several options that must be specified when ordering the HPDI32B board. Please consult our sales department with your application requirements to decide on the correct ordering options.

PMC - HPDI32B - <FIFO Size> - <Interface>

Option		Value	Description
FIFO Size		64K	8K x 32 FIFOs (Rx FIFO = 32KB / Tx FIFO = 32KB)
		256K	32K x 32 FIFOs (Rx FIFO = 128KB / Tx FIFO = 128KB)
		1M	256K x 32 FIFOs (Rx FIFO = 512KB / Tx FIFO = 512KB)
Interface	RS-485	(blank)	RS-485
	PECL	P1	PECL - 220 Ohm Output Termination / 120 Ohm Input Termination
		P2	PECL - No Termination

<FIFO SIZE>

FIFOs depths can range from 8k words to 128k words. Larger FIFO depth is important for faster interfaces to reduce the risk of software overhead.
Standard configuration of the HPDI32B contains 32k word deep FIFOs.

<BUS INTERFACE>

RS485/422 Interface
The RS485/RS422 interface - bus clock speeds up to 25MHz (100 Mbytes per sec).

PECL Interface
The PECL (Pseudo-ECL) interface.

Note: LVDS and TTL versions of the HPDI32A/B family are available in the PMC64-HPDI32ALT version of the board. Please visit our website for more information.

6.1 INTERFACE CABLE

General Standards Corporation has standard an interface cables for the HPDI32B board. This cable is twisted pair for increased noise immunity. Several standard cable lengths are offered, or the cable length can be custom ordered to the user's needs. Versions of the cable are available with connectors on both ends, or the cable may be ordered with a single connector to allow the user to adapt the other end for a specific application. Please consult factory for more information on cabling options.

6.2 CUSTOM APPLICATIONS

Although the HPDI32B board provides extensive flexibility to accommodate most user applications, custom interfaces exist, which may not exactly conform to the HPDI32B interface standard. General Standards Corporation has worked with many customers to provide customized versions based on the HPDI32B board. Please consult our sales department with your specifications to inquire about a custom application.

APPENDIX A – FIRMWARE REVISIONS

PMC-HPDI32A

v 209 (2/00)	Add Tx_enabled and Rx_enabled to Cable Cmd D5 and D6
v20A (8/00)	Invert Cable start (Rx_Cable_Cmd3 copied
v20B (8/00)	Fix IRQ0 /1 to hold IRQ for 2 clks. Removed IRQ sources 2-6. Sync all Rx_Command
v20C (10/00)	Change Tx to remove one clock delay on TxEmpty
v20D (10/00)	Fix Tx to tristate Tx_Cmd_D[6..5] - were GND causing contention when driven
v20E (2/01)	Fix Demand Mode DMA to do single requests until empty
v20F (2/01)	Add new features from PCI version
v210 (2/01)	Update Interrupt defaults
v211 (8/01)	Make Interrupts all edge triggered, fix ECL Xcvr enables
v212 (7/02)	Fix Tx bug where FIFO goes empty an Tx SM stops and restarts - loses one data word
v213 (5/03)	Fix FF Size Count when only Rx or TX populated (also requires a wire mod for full flag)
v214 (6/03)	Fix Cable throttle, GPIO Input default (and Frame Valid GPIO), irq latching, overrun clear - sync with PCI version 206
v215 (8/03)	Update tx (TXDONE), change default of GPIO back - sync with PCI32 v208
v216 (9/05)	Change FW Rev to 0xFF (changed back in v217)
v217 (9/05)	Fix FW Reg to show PMC form factor
v218 (4/09)	Major revamp - Add internal FIFO to sync FLAGS (Not shipped)
v219 (6/09)	Add Test Reg to disable Interrupt (Not Shipped)

PMC-HPDI32B

v220 (10/18)	Initial HPDI32B release
v221 (6/20)	Change Board status register to exactly match HPDI32A
v222 (6/20)	Change Tx Clock Divider to exactly match HPDI32ALT
v223 (12/20)	Frame valid always enabled if set to GPIO (add bug in HPDI32A).