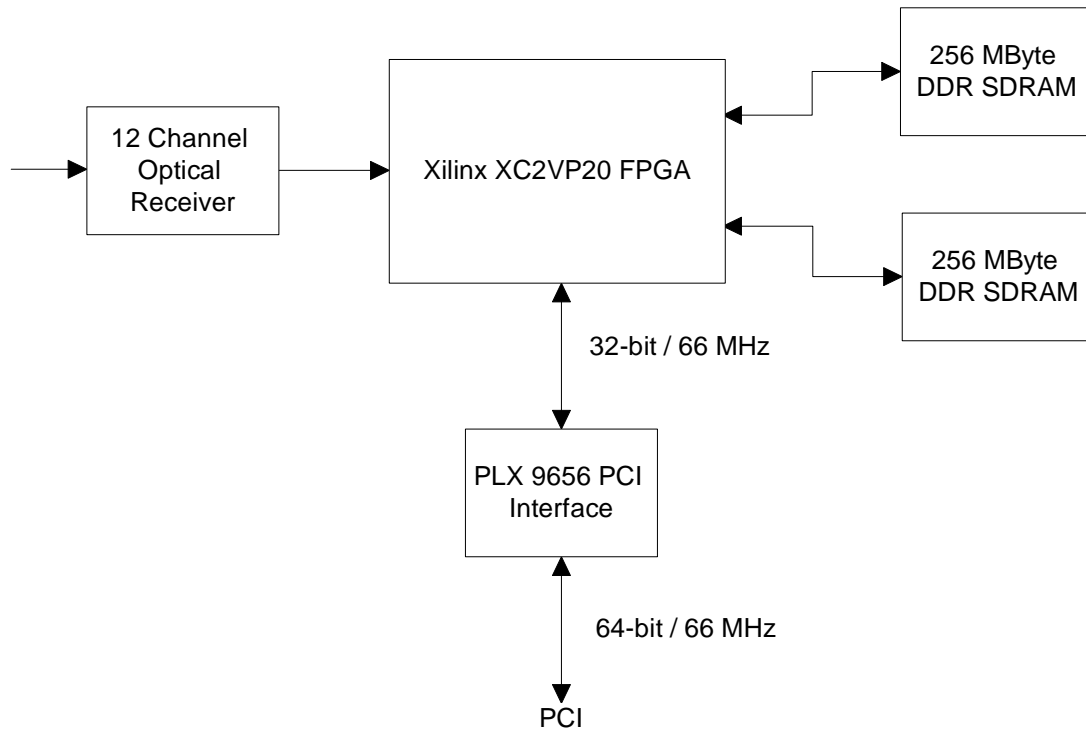


# Tevatron Ionization Profile Monitor (IPM) Buffer Module

08 March 2006  
Mark Bowden  
Richard Kwarciany

## Introduction

The IPM Buffer module receives data from the IPM QIE front end modules over fast serial optical links. The data is buffered and then read out through a standard 3.3 Volt PCI interface by a commercial PC. The difference in maximum input and output rates is approximately 10:1, so at least 10 seconds of data readout time are required for each second of data acquisition, or the data must be compressed/filtered. Additional time will be required for analysis, depending on the speed of the processor.



IPM Buffer Block Diagram

## Implementation

The IPM system will use 16 optical links to supply data. Standard optical components are available in 12 channel increments, meaning that two IPM Buffer modules with one optical receiver each will be used.



IPM Buffer PCI Card

## PCI Interface

The IPM Buffer uses the PLX PCI9656 interface. The PCI9656 runs at 64 bits/ 66MHz on the PCI side but is limited to 32 bit/ 66 MHz operation on the local bus. The maximum transfer rate is approximately 20 MBytes/sec, although it may be possible to run faster using two cards and transferring bursts of data in a “ping-pong” fashion.



PLX PCI9656

Advantages of the standalone PCI interface are;

- 1) it is active at power-up, and can be used to program the FPGA through PCI.
- 2) it has been used in previous Fermilab designs, so the development software exists and has been exercised.

PCI registers will be provided for configuration of the trigger, readout of the data, and diagnostic functions. In diagnostic mode, all of the main memory is accessible to the PCI host.

## ***Memory***

The IPM Buffer will hold two DDR SODIMM modules for a total of 0.5 GBytes of buffer memory. This represents approximately 0.4 seconds of continuous data.

The memory is arranged in two banks, with a total bandwidth of 2 GBytes/sec. This should be sufficient to handle the peak input data rate (1.5 GBytes/sec for 8 channels).

## ***FPGA***

The FPGA chosen for the IPM buffer is the Xilinx XC2VP20. This device has substantial internal resources (20,000+ registers and 1.5 Mbits of memory), allowing formatting and filtering to be done on-board the Buffer module before data is transferred to the PC. Each input link is decoupled by a 64 bit wide FIFO and the data is transferred to external memory in blocks of 32 64-bit words (2 KBytes).

## ***Deserializers***

The deserializers are included in the FPGA and are capable of accepting data rates in the range of 600-3000 Mbps. They require a local reference clock with a frequency very near (+/- 100 ppm) the intended data rate for initial lock. Following the initial frequency lock, the deserializers will relock to the actual frequency of the incoming data.

The local reference clock is provided by a crystal oscillator.

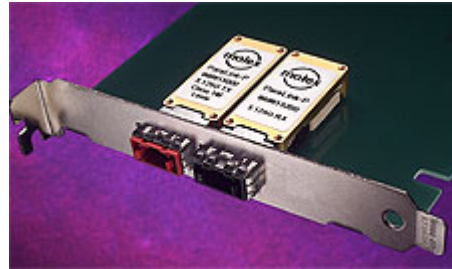
For reliable link operation, the data transmitter should run at a rate slightly faster than the required data transfer rate. This will cause insertion of an occasional idle character in the data stream, which helps to maintain word alignment in the receiver.

On command, the transmitter should also have the capability of sending a specific sync word (8B/10B comma character) for a period of at least 20 usec to establish receiver word synchronization.

All transmitters connected to the same IPM Buffer should operate at the same data rate (+/- 100 ppm).

## *Optical Links*

The IPM data sources are assumed to use multimode (850 nm) optics. The transmitter connector type is not specified, since fanout cables are available in most common types.



### Optical Receiver(s)

The receive end of the optical link uses the MTP/MPO style 12 channel connector. The receiver(s) are Infineon V23832R devices (second source Molex 86991). These receivers use a BGA array connector and are removable.



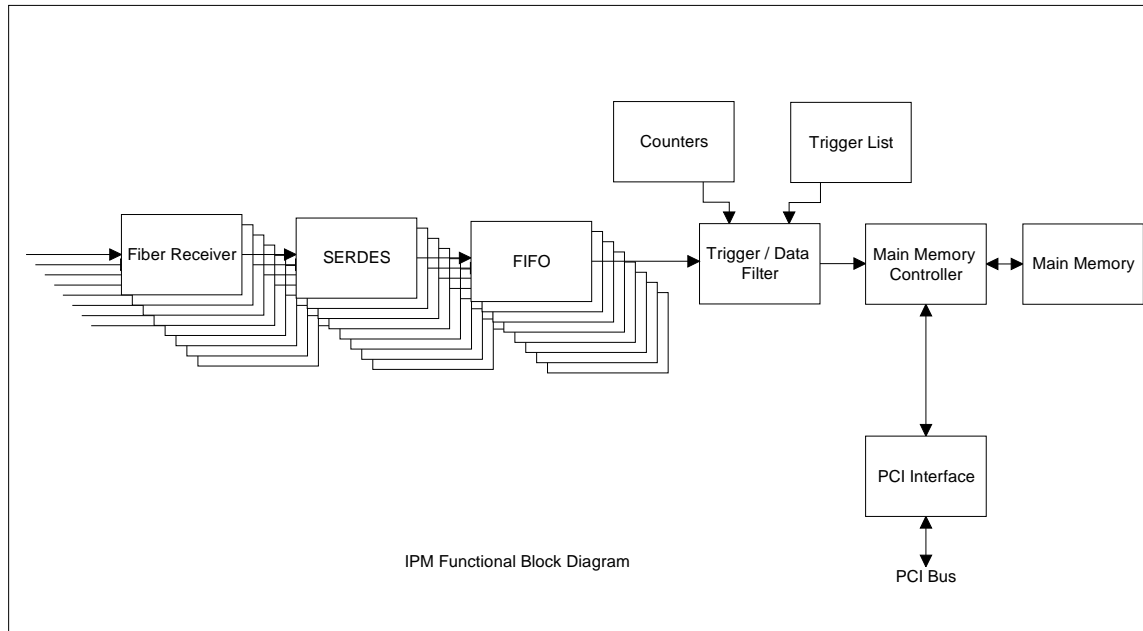
MTP (12 channel) parallel fiber

At the source end of the optical cable the 12 parallel fibers are split into individual fibers using either a short fanout adapter cable, or a fiber cassette.



Parallel fiber fanout cable.

## Functional Description



**Figure 1**

### *Fiber Receiver*

Referring to Figure 1, the Fiber Receivers are actually one discrete component which convert the optical input to a CML differential signal. The Infineon V23832R device used is a 12 channel device. Eight of the 12 channels are used in this implementation. Channels 8 through 11 are not connected.

### *SERDES*

Each CML input channel is deserialized by a SERDES device internal to the FPGA. The SERDES produces a 16-bit wide parallel data stream.

### *FIFO*

A two-stage FIFO memory on each input channel allows decoupling of the link clocks from the FPGA's clock, and also formatting of the incoming data into 64-bit wide data frames. The FIFOs are large enough to allow the data from all 8 serial links to be synchronized to each other. Since data on the links is organized into frames, synchronization of the eight links with each other is accomplished by comparing the frame header information on the links.

### *Trigger / Data Filter*

Configured through registers in the PCI interface, the Trigger / Data Filter block allows for selective acquisition of data from the front end electronics. Up to 318 samples per turn can be stored by populating the sample masks through PCI. For each bit set true in

the mask, data at that sample will be forwarded to the Memory Controller block for storage in memory. Individual channels are enabled or disabled in the Channel Mask Register. All data is stored if Data Filtering is disabled.

The Trigger / Data Filter block also controls triggering of data acquisition. The trigger logic is configured through PCI registers, and allows for saving a preset number of turns after an injection marker, or after a software data acquisition start command is issued. To acquire a preset number of turns, the number of turns is set in the N Turns register, then the N Turn Enable bit is set in the CSR register.

### *Counters*

The Counters block contains the Proton sample counter, the Pbar sample counter, and the Turn counter. The Proton sample counter is a modulo-318 counter that is reset by the proton marker bit in the data header, and incremented on each sample. The Pbar sample counter is similar, but is reset by the pbar marker bit. The Turn counter is a 16-bit counter that is reset by the injection marker bit, and increments when the proton counter wraps to zero. The values of each of the three counters is added to the header information and stored with the data during data acquisition.

### *Memory Controller*

The Memory Controller block provides for initialization and control of the DDR SDRAM memory that is external to the FPGA. The memory controller itself is internal to the FPGA. Write and Read requests are entertained from either the PCI interface, or from the Trigger / Data Filter blocks, depending on which mode the IPM Buffer is in. The mode is controlled via a bit in the CSR register. Two modes are defined. Acquisition Mode, and Readout Mode. When in Acquisition Mode, the Trigger / Data Filter block has exclusive access to the DDR SDRAM, and read attempts from PCI return DEADBEEF. Write attempts are ignored. When in Readout Mode, the PCI interface has exclusive access to the DDR SDRAM, and data from the input links is not stored.

### *Main Memory*

The main memory is discrete from the FPGA, and consists of two 256 MByte DDR SDRAM SODIMM modules. Each module is treated as a separate bank by the Memory Controller block, but is accessed as one large bank via PCI. Main memory can be read or written via PCI when the IPM Buffer is in Readout Mode. Placing the module into Acquisition Mode allows data from the input links to be stored in Main Memory.

### *PCI Interface*

The PCI interface is a dedicated PCI interface device that is discrete from the FPGA chip on the board. It supports 64-bit, 66 MHz PCI.

## ***Diagnostic Features***

### *Loopback Data Generator.*

Each input serial link is equipped with a pattern generator that can be configured to transmit a known data pattern into the serial input. When enabled data frames of the

required format are generated with an incrementing pattern to allow for confidence testing of the high speed serial links. The data generator transmits the following pattern:

#### *Force Error Registers*

To facilitate error reporting software development, the IPM Buffer module can be programmed to generate false error conditions. These error conditions are reported in the Link Error registers. Each of the individual links can be programmed to generate one or all of the 10 link error conditions by setting the mask bits in the Link Force Error Register.

#### *Front End Diagnostic Mode*

To facilitate debugging of front end boards, the IPM Buffer can be put into Front End Diagnostic Mode. When in this mode, raw footer data from the lowest four channel front end boards is copied into the IPM Buffer memory frame in place of the data for the highest four channel front end boards. Front End Diagnostic Mode is enabled by setting the Front End Diagnostic Mode bit in CSR0.

### ***Input Data Optical Connector***

The Input Data Optical Connector accepts a 12-fiber ribbon with an MTP connector. Fibers are numbered 0 – 11 with fiber 0 being defined as the bottom fiber when facing the IPM Buffer card from the rear of the host PC. Only fibers 0 through 7 are used in the IPM Buffer Module. Fibers 8 through 11 are not used.

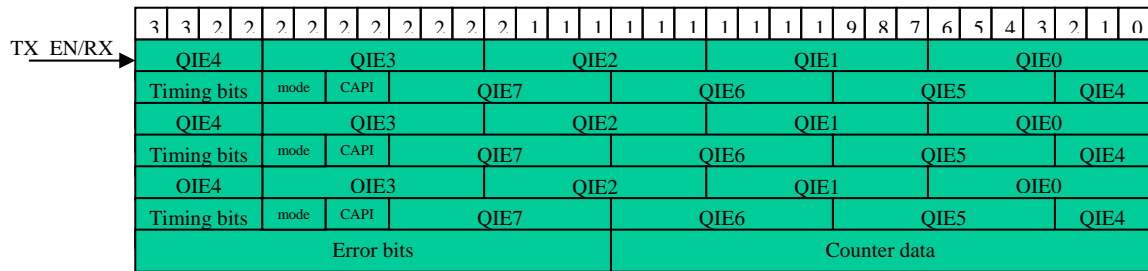
The FONetworks 12-fiber breakout adapter breaks the fiber ribbon out into 12 individual fibers with LC type connectors. The individual fibers are color coded as follows.

Fiber Color	Fiber Number
Dark Blue	Fiber 0
Orange	Fiber 1
Green	Fiber 2
Brown	Fiber 3
Gray	Fiber 4
White	Fiber 5
Red	Fiber 6
Black	Fiber 7
Yellow	Fiber 8
Purple	Fiber 9
Pink	Fiber 10
Light Blue	Fiber 11

### ***Input Data Format***

Link data received from the front-end boards adheres to the following encoding scheme:





IPM Buffer accepts data in the format above only. Data frames consist of seven contiguous 32-bit data words. Each frame contains data for three time slices, identified by the capID bits. Data frames may have fill characters between them, but a frame error will be set if fill a character occurs within a frame. Once each link's SERDES is locked, at least one fill character is required for IPM Buffer to frame synchronize, but once synchronized, a string of data frames with no fill character separating them is allowed. It is recommended that a fill character be added between data frames occasionally to allow IPM Buffer to check for synchronization errors. A counter is implemented to flag an error if a fill character is not received for more than approximately 100 contiguous data words.

IPM Buffer deserializers accept any 32-bit k character sequence as a fill character, but for deserializer clock correction, the clock correction sequence should be transmitted approximately every 5000 bytes. This allows the deserializer to add or remove clock correction characters as necessary to compensate for small clock rate differences between the transmitter and the receiver. The defined clock correction sequence for IPM Buffer is:

k28.5 d16.2 k28.5 d5.6 .

Data in the frame is defined as follows:

- QIE<sub>x</sub> Front End QIE pseudo exponential data.
- CAPID Two bit CAPID number from QIE chip.
- Mode Two bit Mode identifier from the QIE chip. Defined modes are as follows:
  - 00 Calibration
  - 01 Normal
  - 10 Debug/Testing
  - 11 RESET Marker
- Timing Bits Four front end timing bits defined as follows:
  - 31 Proton Injection
  - 30 PBar Injection
  - 29 Proton Marker
  - 28 PBar Marker
- Error Bits The following front end error bits are defined:
  - 30 – 25 Unused.
  - 24 PLL Lock Fail in Timeslice 2

- 23 PLL Lock Fail in Timeslice 1
- 22 PLL Lock Fail in Timeslice 0
- 21 CAPID error in Timeslice 2
- 20 CAPID error in Timeslice 1
- 19 CAPID error in Timeslice 0
- 18 FIFO Full in Timeslice 2
- 17 FIFO Full in Timeslice 1
- 16 FIFO Full in Timeslice 0

Counter Data 16-bit counter increments on each timeslice. This results in a count value incrementing by three for each data frame.

### ***Memory Data Format***

Data from the eight input links is combined and stored in main memory in 80 byte frames. Each frame contains data from one timeslice.

#### *Memory Data Frame Format*

63	0
Header Low	
Header High	
Link 0 Data	
Link 1 Data	
Link 2 Data	
Link 3 Data	
Link 4 Data	
Link 5 Data	
Link 6 Data	
Link 7 Data	

#### *Header Low*

63 – 56	55 – 54	53 – 52	51	50	49 - 41	40 - 32
PLL Lock Error	QIE Mode	Cap ID	Inter Board Sync Error	0	Proton Counter	PBar Counter

31 – 24	23 – 16	15 – 0
FIFO Full Error	Cap ID Error	Turn Counter

*Header High*

63 – 56	55 – 48	47 – 40	39 – 32	31 – 24	23 – 16	15 – 8	7 – 0
Counter Sync Error	Frame Error	QIE Mode Sync Error	Cap ID Sync Error	Proton Injection Error	PBar Injection Error	Proton Marker Error	PBar Marker Error

Header error bits in eight bit blocks contain one error bit for each link, with the lowest significant bit corresponding to the error occurring in link 0, the next lowest in link 1, etc.

Link x Data

QIE Data can be stored in Raw format (exactly as received from the front end board), or converted to an integer value using the conversion lookup table. The conversion lookup table is loaded and enabled through PCI (see: PCI Interface), and when enabled, causes the IPM Buffer to store an eight bit integer value into memory, instead of seven bit pseudo exponential value for each QIE channel.

*Link x Data (RAW)*

63 – 60	59 – 58	57 – 56	55 - 47	48 - 42	41 - 35	34 - 32
Timing	QIE Mode	Cap ID	QIE 7	QIE 6	QIE 5	QIE 4 (high)

31 – 28	27 – 21	20 – 14	13 – 7	6 – 0
QIE 4 (high)	QIE 3	QIE 2	QIE 1	QIE 0

*Link x Data (Integer Conversion Enabled)*

63 – 56	55 – 48	47 – 40	39 – 32	31 – 24	23 – 16	15 – 8	7 – 0
QIE 7	QIE 6	QIE 5	QIE 4	QIE 3	QIE 2	QIE 1	QIE 0

*Memory Data Frame Format when in Front End Diagnostic Mode*

63	0
Header Low	
Header High	
Link 0 Data	
Link 1 Data	
Link 2 Data	
Link 3 Data	
Link 0 RAW Front End Footer Data	

Link 1 RAW Front End Footer Data
Link 2 RAW Front End Footer Data
Link 3 RAW Front End Footer Data

*Link x RAW Front End Footer Data*

63 – 56	55 – 48	47 – 40	39 – 32
Frame Error, Hard Trigger and FE Init	Time Slice 2 Timing Mode and CapID	Time Slice 1 Timing Mode and CapID	Time Slice 0 Timing Mode and CapID

31 – 0
RAW Front End Footer

*Frame Error, Hard Trigger, and Front End Init*

63	62	61	60 – 56
Frame Error	Hard Trigger	Front End Init	Unused (0x00000)

*Time Slice x Timing Mode and CapID*

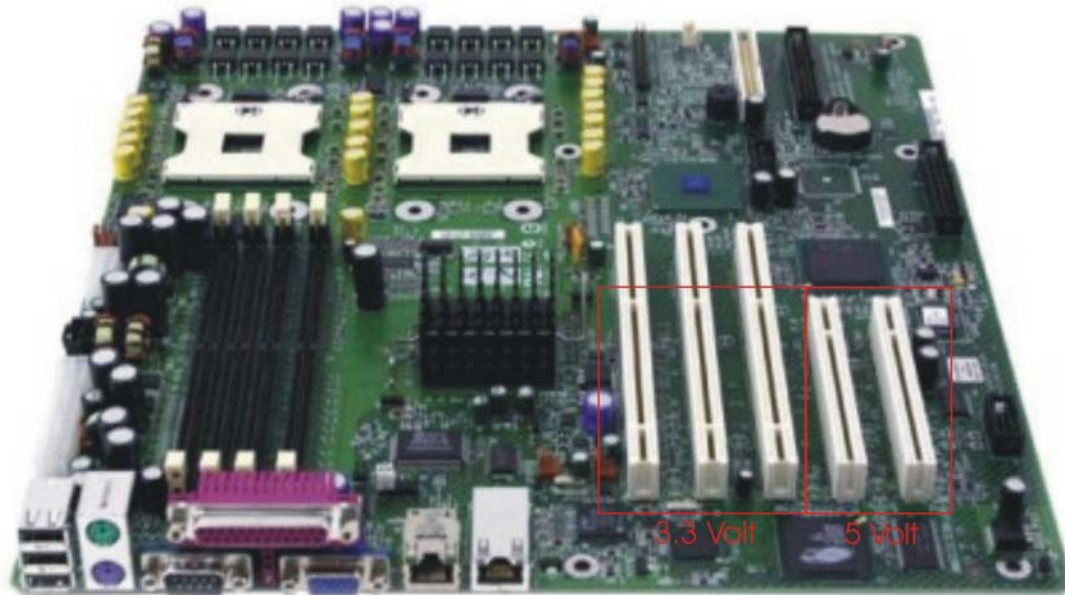
7	6	5	4	3 – 2	1 – 0
Proton Inj	PBar Inj	Proton Marker	PBar Marker	Mode	CapID

## ***Interboard Synchronization***

The IPM Buffer provides a mechanism for ensuring synchronization between two boards in the same host system. Connector J2 (top of pc board) provides for this function. A 20-pin ribbon cable jumper must be connected between the J2 connectors of the two boards. Once connected, one of the boards is designated a slave in software by setting the Interboard Slave Enable bit in the CSR register. Once properly configured and connected, the master IPM Buffer compares it's Timing bits, CapID bits, Proton Counter, and the lowest three bits of the Turn Counter, with those of the slave board. Any errors are flagged in bit 41 of the data frame header on the master board. Note that due to the time needed to do the compare, the error flag may not show true until the time slice frame after it actually occurs.

## ***PCI Interface***

***CAUTION: The IPM Buffer board supports 3.3-Volt PCI ONLY. The board will physically fit into a 5-Volt PCI slot, but powering a system with an IPM Buffer module in a 5-Volt slot WOULD BE BAD. Fatal damage to the IPM Buffer, the system motherboard, power supply, or all of the above will result if attempted.***



Note that 3.3 Volt PCI slots can be identified by the location of the key in the slot. 3.3 Volt PCI slots have the key near the back panel of the system, whereas 5 Volt PCI slots have the key nearest the end of the connector that is farthest from the back panel of the system. See figure above.

The following PCI registers are defined:

*PCI Configuration Space:*

00h	Device ID / Vendor ID (9656 10B5)
04h	Status / Command
08h	Class Code / Rev ID (068000AD)
10h	Base Address of Memory access to PLX registers (PCIBAR0)
14h	Base Address of I/O access to PLX registers (PCIBAR1)
18h	Base Address of Memory access to Readout Data Window (PCIBAR2)
1Ch	Base Address of I/O Space access to IPM CSR register block (PCIBAR3)
20h	Reserved
24h	Reserved
28h	Reserved
2Ch	PCI Subsystem ID / PCI Subsystem Vendor ID (24400003)
30h – 53h	Reserved (do not write)

*I/O space (I/O Base = contents of PCIBAR1):*

I/O Base + 00h	Control-Status register
I/O Base + 04h	Memory window page register

I/O Base + 08h	Master Link Select Mask / Trigger Mask
I/O Base + 0Ch	Event Data Address Counter register
I/O Base + 10h	SERDES Control 0
I/O Base + 14h	SERDES Control 1
I/O Base + 18h	Link Status Register
I/O Base + 1Ch	N Turn Count
I/O Base + 20h	Link Error Register 0
I/O Base + 24h	Link Error Register 1
I/O Base + 28h	Link Error Register 2
I/O Base + 2Ch	Link Error Register 3
I/O Base + 30h	Link Diagnostic Register
I/O Base + 34h	Peak Select Register
I/O Base + 38h	Peak 31 to 0
I/O Base + 3Ch	Peak 63 to 32
I/O Base + 40h – I/O Base + ACh	Reserved (DEADBEEF)
I/O Base + B0h	Running Sum Select
I/O Base + B4h – I/O Base + BCh	Reserved (DEADBEEF)
I/O Base + C0h	Running Sum 0
I/O Base + C4h	Running Sum 1
I/O Base + C8h	Running Sum 2
I/O Base + CCh	Running Sum 3
I/O Base + D0h	State Machine Low
I/O Base + D4h	State Machine Mid
I/O Base + D8h	State Machine High
I/O Base + DCh – I/O Base + ECh	Reserved (DEADBEEF)
I/O Base + F0h	Firmware ID code (date in mmddyyyy format)
I/O Base + F4h	FPGA Firmware JTAG TDO bit
I/O Base + F8h	IPM Buffer FIFO Full Error Flags
I/O Base + FCh	Reserved (DEADBEEF)
I/O Base + 100h	Proton Mask Register 0 (proton mask bits 31 to 0)
I/O Base + 104h	Proton Mask Register 1 (proton mask bits 63 to 32)
I/O Base + 108h	Proton Mask Register 2 (proton mask bits 95 to 64)
I/O Base + 10Ch	Proton Mask Register 3 (proton mask bits 127 to 96)
I/O Base + 110h	Proton Mask Register 4 (proton mask bits 159 to 128)
I/O Base + 114h	Proton Mask Register 5 (proton mask bits 191 to 160)
I/O Base + 118h	Proton Mask Register 6 (proton mask bits 223 to 192)
I/O Base + 11Ch	Proton Mask Register 7 (proton mask bits 255 to 224)
I/O Base + 120h	Proton Mask Register 8 (proton mask bits 287 to 256)
I/O Base + 124h	Proton Mask Register 9 (proton mask bits 317 to 288)
I/O Base + 128h – 13Ch	Reserved
I/O Base + 140h	PBar Mask Register 0 (PBar mask bits 31 to 0)

I/O Base + 144h	PBar Mask Register 1 (PBar mask bits 63 to 32)
I/O Base + 148h	PBar Mask Register 2 (PBar mask bits 95 to 64)
I/O Base + 14Ch	PBar Mask Register 3 (PBar mask bits 127 to 96)
I/O Base + 150h	PBar Mask Register 4 (PBar mask bits 159 to 128)
I/O Base + 154h	PBar Mask Register 5 (PBar mask bits 191 to 160)
I/O Base + 158h	PBar Mask Register 6 (PBar mask bits 223 to 192)
I/O Base + 15Ch	PBar Mask Register 7 (PBar mask bits 255 to 224)
I/O Base + 160h	PBar Mask Register 8 (PBar mask bits 287 to 256)
I/O Base + 164h	PBar Mask Register 9 (PBar mask bits 317 to 288)
I/O Base + 168h – 1FCh	Reserved
I/O Base + 200h – I/O Base + 3FFh	QIE Conversion Lookup Table

*Status / CSR Register (offset x00)*

31 – 16	15 – 0
Status (x0000)	Control

- Bit 31 – 28 RO Module Status (currently undefined x0000).
- Bit 27 RO IRQ On
- Bit 26 – 16 RO Module Status (currently undefined 000000000000).
- Bit 15 WO Master Reset (Reset all of FPGA including SERDES and link FIFOs)
- Bit 14 R/W Clear Errors
- Bit 13 R/W Clear IRQ  
IRQ Enabled when low.
- Bit 12 R/W Force IRQ  
When Bit 13 is low and 12 is high, the PCI Interrupt output is driven true.
- Bit 11 R/W Clear latched IRQ Status bit.
- Bit 10 WO Soft Reset (Reset all of FPGA except SERDES and link FIFOs)
- Bit 9 – 7 RO Unused (000).
- Bit 6 R/W Front End diagnostic mode enabled. When high, raw footer data from the front end modules on links 0 - 3 is written into the memory frame in the place of the data for links 4 – 7. In diagnostic mode, data from links 4 – 7 is not available.
- Bit 5 R/W RAW Mode Enable. When high, enables true RAW mode. Data from front end modules is stored in same format as received. Note that RAW mode is only available when Conversion Enable is not enabled. If Conversion Enable is true, the RAW Mode Enable bit has no affect.
- Bit 4 R/W Inter-Board Slave Enable.  
When high, board is placed into slave mode for interboard synchronization error checking. In slave mode, the board drives the inter-board outputs, and applies it's proton counter to them. When low, the board is in master mode, and samples Inter-board inputs. Inputs are compared with internal Proton counter bits and errors are flagged.
- Bit 3 R/W NTurn Enable  
When high, a trigger will cause the number of turns stored in the N Turn

- Count Register to be stored.
- Bit 2 R/W Sparsification Enable.  
When high, Sparsification mode is enabled. Only samples indicated in the Proton or PBar masks will be stored.
- Bit 1 R/W Conversion Enable.  
When high, the QIE conversion lookup table is enabled. Raw QIE data is used as an index into the conversion table. The output of the conversion table is stored in memory. When low, raw QIE data is stored.
- Bit 0 R/W Acquisition Enable.  
When high, the DAQ state machine controls the DDR memory. Data from the input links is stored. PCI access to the DDR memory is blocked. When low, PCI access of the DDR memory is enabled. Data from the Input links is not stored.

*DDR Memory Page Register (offset x04)*

31 - 29	28 - 20	19 - 0
000	Page	00000000000000000000

*Master Link Select Mask / Trigger Mask (offset x08)*

31 - 24	23 - 16	15 - 4	3 - 0
x00	Master Link Select Mask	x000	Trigger Mask

- Bit 31 - 24 RO Unused (x00)
- Bit 23 R/W Select Link 7 as Master Link  
When high, IPM buffer's front end counter synchronizes itself to the front end counter of this link. Only one link may be selected as master.
- Bit 22 R/W Select Link 6 as Master Link  
Similar to bit 23 for Link 6.
- Bit 21 R/W Select Link 5 as Master Link  
Similar to bit 23 for Link 5.
- Bit 20 R/W Select Link 4 as Master Link  
Similar to bit 23 for Link 4.
- Bit 19 R/W Select Link 3 as Master Link  
Similar to bit 23 for Link 3.
- Bit 18 R/W Select Link 2 as Master Link  
Similar to bit 23 for Link 2.
- Bit 17 R/W Select Link 1 as Master Link  
Similar to bit 23 for Link 1.
- Bit 16 R/W Select Link 0 as Master Link  
Similar to bit 23 for Link 0.
- Bit 3 R/W Proton Injection Bit  
When high, acquisition triggers on receipt of a Proton Injection flag from the front end.
- Bit 2 R/W PBar Injection Bit  
When High, acquisition triggers on receipt of a PBar Injection flag from



- the front end.
- Bit 1 R/W Proton Marker Bit  
When High, acquisition triggers on receipt of a Proton Marker flag from the front end.
- Bit 0 R/W PBar Marker Bit  
When High, acquisition triggers on receipt of a PBar Marker flag from the front end.

*DDR Address Counter Register (offset x0C)*

31 – 29	19 - 0
Unused	Address Counter

- Bit 31 – 29 RO Unused (000).
- Bit 19 – 0 R/W Address Counter Register.  
When written, causes DAQ to begin writing at this address when acquisition is enabled. When read, shows the next address to be written by the DAQ state machine. Note that since DAQ state machine always writes 64-bit words, low three bits are always zero.

*SERDES Control 0 (offset x10)*

31 – 24	23 – 16	15 – 8	7 – 0
PowerDown	TXInhibit	RXReset	TXReset

- Bit 31 – 24 R/W Power Down.  
When high, powers off corresponding SERDES.
- Bit 23 – 16 R/W TX Inhibit.  
When high, corresponding SERDES transmit is disabled.
- Bit 15 – 8 R/W RX Reset.  
When high, corresponding SERDES receiver is in reset.
- Bit 7 – 0 R/W TX Reset.  
When high, corresponding SERDES transmitter is in reset.

*SERDES Control 1 (offset x14)*

31 – 16	15 – 8	7 – 0
Loopback Control	TX Pattern Enable	Link Enable

- Bit 31 – 16 R/W SERDES loopback control.  
Note that each SERDES uses two bits for loopback control. Bits 17 and 16 correspond to SERDES 0 and so on. The truth table for loopback control is; 00 = Normal non-loopback mode, 01 = Parallel loopback mode, 10 = Serial loopback mode. Mode 11 is undefined.
- Bit 15 – 8 R/W TX Pattern Enable.  
When high SERDES Transmit pattern generator is enabled for corresponding link.
- Bit 7 – 0 R/W Link Enable.  
When high, corresponding link is enabled.

*Link Status Register (offset x18)*

31 – 26	25 – 24	23 – 16	15 – 8	7 - 0
000000	Paroli	Link Data Activity	Frame Locked	SERDES Locked

- Bit 31 – 26 RO Unused (000000).  
 Bit 25 RO Paroli Power bad latched.  
 Set by Paroli optical power on link 0 falling below minimum required for reliable data transmission. Cleared by Clear Errors bit, Reset bit, or hard Reset.  
 Bit 24 Paroli Power good.  
 Not latched. High if Paroli link 0 optical power is above the minimum threshold required for reliable data transmission. Low if not.  
 Bit 23 – 16 Link Data Activity.  
 High if data is being received on corresponding input link.  
 Bit 15 – 8 Frame Locked.  
 When high, IPM data frames are being received and the IPM buffer input state machine for the corresponding link recognizes the frame as a valid IPM frame.  
 Bit 7 – 0 SERDES Locked.  
 When high the input SERDES for the corresponding link is locked onto the 8b-10b data received from the front end. Data does not have to be valid IPM data for SERDES lock to be true.

*N Turn Count Register (offset x1C)*

31 – 16	15 – 0
Unused	N Turn Count

- Bit 31 – 16 RO Unused (x0000).  
 Bit 15 – 0 R/W N Turn Count Register.  
 Sets the number of turns stored after a trigger.

*Link Error 0 Register (offset x20)*

31 – 24	23 – 16	15 – 8	7 – 0
Illegal Character latched	Comma Error Latched	Frame Error Latched	SERDES Lock Error Latched

- Bit 31 – 24 RO Illegal Character Latched.  
 Set high for corresponding link when SERDES has received an illegal character. Cleared by Clear Errors bit, Reset bit, or hard reset.  
 Bit 23 – 16 RO Comma Error Latched.  
 Set high for corresponding link when a comma character has not been received for more than 100 words. Cleared by the Clear Errors Bit, Reset bit, or hard reset.  
 Bit 15 – 8 RO Frame Error Latched.  
 Set high for corresponding link when the link control state machine

detects an IPM frame error. IPM frames are defined as seven consecutive 32-bit data words. One or more comma characters between frames is optional as long as at least one comma is received at least every 100 words.

Bit 7 – 0 RO SERDES Lock Error Latched.  
Set high for corresponding link when the SERDES loses lock. Cleared by Clear Errors bit, Reset bit, or hard reset.

*Link Error 1 Register (offset x24)*

31 – 24	23 – 16	15 – 8	7 – 0
Proton Marker Error Latched	PBar Marker Error Latched	QIE Mode Error Latched	Link Frame Error

Bit 31 – 24 RO Proton Marker Error Latched.  
Set high for corresponding link when the Proton Marker bit does not agree with other links. Cleared by Clear Errors bit, Reset bit, or hard reset.

Bit 23 – 16 RO PBar Marker Error Latched.  
Set high for corresponding link when PBar Marker bit does not agree with other links. Cleared by Clear Errors bit, Reset bit, or hard reset.

Bit 15 – 8 RO QIE Mode Error Latched.  
Set high for corresponding link when QIE Mode bit does not agree with other links. Cleared by Clear Errors bit, Reset bit, or hard reset.

Bit 7 – 0 RO Link Frame Error.  
High for corresponding link when the link control state machine has detected an IPM frame error.

*Link Error 2 Register (offset x28)*

31 – 24	23 – 16	15 – 8	7 – 0
FIFO Full Error Latched	Cap ID Error Latched	Proton Injection Error Latched	PBar Injection Error Latched

Bit 31 – 24 RO FIFO Full Error Latched.  
Set when the data link FIFO for the corresponding link reaches the full state. Cleared by Clear Errors bit, Reset bit, or hard reset.

Bit 23 – 16 RO Cap ID Error Latched.  
Set for corresponding link when front end logic has detected a CAP ID Error. Cleared by Clear Errors bit, Reset bit, or hard reset.

Bit 15 – 8 RO Proton Injection Error latched.  
Set for corresponding link when Proton Injection bit does not agree with other links. Cleared by Clear Errors bit, Reset bit, or hard reset.

Bit 7 – 0 RO PBar Injection Error Latched.  
Set for corresponding link when PBar bit does not agree with other links. Cleared by Clear Errors bit, Reset bit, or hard reset.

*Link Error 3 Register (offset x2C)*

31 – 24	23 – 16	15 – 8	7 – 0
---------	---------	--------	-------

Cap ID Sync Error Latched	QIE Mode Error Latched	Counter Error Latched	PLL Lock Error Latched
---------------------------	------------------------	-----------------------	------------------------

- Bit 31 – 24 RO Cap ID Sync Error Latched.  
Set for corresponding link when Cap ID value does not match value in other links. Note that the Cap ID values must match for at least four links for these values to be valid. If the Cap ID values do not match for at least four links, then error bits will be set, but it is not possible to determine which are errors and which are not. Latched errors are cleared by the Clear Errors bit, Reset bit, or a hard reset.
- Bit 23 – 16 RO QIE Mode Error Latched.  
Set for corresponding link when the QIE Mode value does not match the QIE Mode value in other links. Note that the QIE Mode values must match for at least four links for these values to be valid. If the QIE Mode values do not match for at least four links, then error bits will be set, but it is not possible to determine which are errors, and which are not. Latched errors are cleared by the Clear Errors bit, Reset bit or a hard reset.
- Bit 15 – 8 RO Counter Error.  
Set for corresponding link when front end counter value does not match IPM Buffer counter value. Cleared by Clear Errors bit, Reset bit, or hard reset.
- Bit 7 – 0 RO PLL Lock Error Latched.  
Set for corresponding link when front end logic has detected a PLL Lock Error. Cleared by Clear Errors bit, Reset bit, or hard reset.

*Link Diagnostic (Force Error) Register (offset x30)*

31 – 24	23 – 16	15 – 11	10 - 0
Unused	Link Select	Unused	Force Error Mask

- Bit 31 – 24 RO Unused (x00).
- Bit 23 – 16 R/W Link Select.  
When high, corresponding link will generate error selected in Force Error Mask.
- Bit 15 – 11 RO Unused (00000).
- Bit 10 – 0 R/W Force Error Mask.
- Bit 10 Force CAP ID Sync Error.
  - Bit 9 Force Counter Error.
  - Bit 8 Force PLL Lock Error.
  - Bit 7 Force FIFO Full Error.
  - Bit 6 Force CAP ID Error.
  - Bit 5 Force Proton Injection Error.
  - Bit 4 Force PBar Injection Error.
  - Bit 3 Force Proton Marker Error.
  - Bit 2 Force PBar Marker Error.
  - Bit 1 Force QIE Mode Error.
  - Bit 0 Force Frame Error.

*Peak Select Register (offset x34)*

31 - 24	23 - 21	20 - 18	17 - 15	14 - 12	11 - 9	8 - 6	5 - 3	2 - 0
Unused (x00)	Link7	Link6	Link5	Link4	Link3	Link2	Link1	Link0

- Bit 31 – 24 RO Unused (x00)
- Bit 23 – 21 R/W Peak Select for Link 7. Select one of the eight channels on Link 7 for Peak Finder to connect to. Setting Bits 23 – 21 to b000 results in Link 7 channel 0 connected to the Peak Finder for Link 7. A binary value of 001 connects Link 7 channel 1 to the Link 7 Peak Finder, and so on.
- Bit 20 – 18 R/W Peak Select for Link 6. Functions similarly to the Peak Select for Link 7.
- Bit 17 – 15 R/W Peak Select for Link 5. Functions similarly to the Peak Select for Link 7.
- Bit 14 – 12 R/W Peak Select for Link 4. Functions similarly to the Peak Select for Link 7.
- Bit 11 – 9 R/W Peak Select for Link 3. Functions similarly to the Peak Select for Link 7.
- Bit 8 – 6 R/W Peak Select for Link 2. Functions similarly to the Peak Select for Link 7.
- Bit 5 – 3 R/W Peak Select for Link 1. Functions similarly to the Peak Select for Link 7.
- Bit 2 – 0 R/W Peak Select for Link 0. Functions similarly to the Peak Select for Link 7.

*Peak 31 – 0 (offset x38)*

31 – 24	23 – 16	15 – 8	7 – 0
Peak Link 3	Peak Link 2	Peak Link 1	Peak Link 0

- Bit 31 – 24 RO Peak value for the Link 3 Peak Finder.
- Bit 23 - 16 RO Peak value for the Link 2 Peak Finder.
- Bit 15 - 8 RO Peak value for the Link 1 Peak Finder.
- Bit 7 - 0 RO Peak value for the Link 0 Peak Finder.

The Peak Finder in each link returns the peak value of the last completed turn for the channel it is connected to. This value for links 0 through 3 can be read in the above bit fields.

*Peak 63 - 32 (offset x3C)*

31 – 24	23 – 16	15 – 8	7 – 0
Peak Link 7	Peak Link 6	Peak Link 5	Peak Link 4

- Bit 31 – 24 RO Peak value for the Link 7 Peak Finder.
- Bit 23 - 16 RO Peak value for the Link 6 Peak Finder.
- Bit 15 - 8 RO Peak value for the Link 5 Peak Finder.

Bit 7 - 0 RO Peak value for the Link 4 Peak Finder.

The Peak Finder in each link returns the peak value of the last completed turn for the channel it is connected to. This value for links 4 through 7 can be read in the above bit fields.

*Running Sum Select (offset xB0)*

31 – 28	27 – 24	23 – 20	19 – 16	15 – 12	11 – 8	7 – 4	3 – 0
Link7	Link6	Link5	Link4	Link3	Link2	Link1	Link0

- Bit 31 – 28 R/W Running Sum Select for Link 7. Selects which of the eight QIE channels on link 7 to connect to the Link 7 running sum counter. Valid codes are 0 – 7. Codes 8 – 15 are reserved.
- Bit 27 – 24 R/W Running Sum Select for Link 6.
- Bit 23 – 20 R/W Running Sum Select for Link 5.
- Bit 19 – 16 R/W Running Sum Select for Link 4.
- Bit 15 – 12 R/W Running Sum Select for Link 3.
- Bit 11 – 8 R/W Running Sum Select for Link 2.
- Bit 7 – 4 R/W Running Sum Select for Link 1.
- Bit 3 – 0 R/W Running Sum Select for Link 0.

*Running Sum 0 (offset xC0)*

31 – 16	15 – 0
Link 1	Link 0

- Bit 31 – 16 RO Running Sum value for Link 1.
- Bit 15 – 0 RO Running Sum value for Link 0.

*Running Sum 1 (offset xC4)*

31 – 16	15 – 0
Link 3	Link 2

- Bit 31 – 16 RO Running Sum value for Link 3.
- Bit 15 – 0 RO Running Sum value for Link 2.

*Running Sum 2 (offset xC8)*

31 – 16	15 – 0
Link 5	Link 4

- Bit 31 – 16 RO Running Sum value for Link 5.
- Bit 15 – 0 RO Running Sum value for Link 4.

*Running Sum 3 (offset xCC)*

31 – 16	15 – 0
Link 7	Link 6

- Bit 31 – 16 RO Running Sum value for Link 7.
- Bit 15 – 0 RO Running Sum value for Link 6.

*State Machine Low (offset xD0)*

31 – 24	23 – 16	15 – 8	7 – 0
Event DDR Address	Event DDR Data	PCI DDR Cache	PCI DDR State

- Bit 31 – 24 RO Instantaneous value of the Event DDR Address State Machine.
- Bit 23 – 16 RO Instantaneous value of the Event DDR Data State Machine.
- Bit 15 – 8 RO Instantaneous value of the PCI DDR Cache Controller State Machine.
- Bit 7 – 0 RO Instantaneous value of the PCI DDR State Machine.

*State Machine Mid (offset xD4)*

31 – 10	9 – 8	7 – 4	3 – 0
Unused (zeros)	DDR Controller Write State	DDR Controller Read State	DDR Controller State

- Bit 9 – 8 RO Instantaneous value of the DDR Controller Write State Machine.
- Bit 7 – 4 RO Instantaneous value of the DDR Controller Read State Machine.
- Bit 3 – 0 RO Instantaneous value of the DDR Controller State Machine.

*State Machine High (offset xD8)*

31 – 28	27 – 24	23 – 20	19 – 16	15 – 12	11 – 8	7 – 4	3 – 0
Link 7	Link 6	Link 5	Link 4	Link 3	Link 2	Link 1	Link 0

- Bit 31 – 28 RO Instantaneous value of the Link 7 state machine.
- Bit 27 – 24 RO Instantaneous value of the Link 6 state machine.
- Bit 23 – 20 RO Instantaneous value of the Link 5 state machine.
- Bit 19 – 16 RO Instantaneous value of the Link 4 state machine.
- Bit 15 – 12 RO Instantaneous value of the Link 3 state machine.
- Bit 11 – 8 RO Instantaneous value of the Link 2 state machine.
- Bit 7 – 4 RO Instantaneous value of the Link 1 state machine.
- Bit 3 – 0 RO Instantaneous value of the Link 0 state machine.

*Firmware ID (offset xF0)*

31 – 24	23 – 16	15 – 0
Month	Day	Year

- Bit 31 – 24 RO Firmware revision month in BCD.
- Bit 23 – 16 RO Firmware revision day in BCD.
- Bit 15 – 0 RO Firmware revision year in BCD.

*FPGA Firmware JTAG TDO bit (offset xF4)*

31 – 1	0
Unused (all zeros)	TDO

Bit 31 – 1 RO Unused (all zeros)  
 Bit 0 RO FPGA Firmware JTAG TDO bit

*IPM Buffer FIFO Full Error Flags (offset xF8)*

31 - 29	28 – 0
000	FIFO Full Error Flags

Bit 31 – 29 RO Unused (zeros)  
 Bit 28 RO DDR Output (to external memory) Data FIFO Full Error  
 Bit 27 RO DDR Command FIFO Full Error  
 Bit 26 RO DDR Input (from external memory) Data FIFO Full Error  
 Bit 25 RO PCI to DDR Data FIFO Full Error  
 Bit 24 RO PCI to DDR Address FIFO Full Error  
 Bit 23 – 16 RO Link Formatted Data FIFO Full Error  
 Bit 15 – 8 RO Link Formatted Footer FIFO Full Error  
 Bit 7 – 0 RO Link RAW Data FIFO Full Error

FIFO Full Error flags should not set under normal running conditions and when true indicate a firmware flaw in the IPM Buffer. Errors are set when the error condition occurs, and are cleared by setting the Clear Errors Bit in CSR0, or by Reset.

*Proton Sparsification Mask Registers (offset x100 – x124)*

31 – 0
Proton Sparsification Mask

Bit 31 – 0 R/W Proton Sparsification Mask.  
 When high, corresponding sample will be stored in memory on each turn if Sparsification Enable bit is set. If Sparsification Enable bit is cleared, then Proton Mask is not used. Note: Register at offset x40 contains mask bits for samples 0 through 31. Register at offset x44 contains mask bits for samples 63 through 32, etc.

*PBar Sparsification Mask Registers (offset x140 – x164)*

31 – 0
Proton Sparsification Mask

Bit 31 – 0 R/W PBar Sparsification Mask.  
 When high, corresponding sample will be stored in memory on each turn if Sparsification Enable bit is set. If Sparsification Enable bit is cleared, then PBar Mask is not used. Note: Register at offset x80 contains mask bits for samples 0 through 31. Register at offset x84 contains mask bits for samples 63 through 32, etc.



*QIE Conversion Lookup Table (I/O Base + 200h – 3FFh)*

31 - 9	8 - 2	1 - 0
Don't care	Conversion Value	Don't care

- Bit 31 – 9      RO    Unused (zeros).
- Bit 8 – 2      R/W   Conversion table Value.  
Value written will be stored in DDR memory QIE raw data of this table entry offset from the table base address. Table is read-only when Acquisition Enable bit is high.
- Bit 1 – 0      RO    Unused (00).

*Memory Space:*

Mem Base + 00000000h -- Mem Base + 000FFFFFh	Event Memory Window (1 Mbyte)
----------------------------------------------------	-------------------------------

***In-System FPGA Programming***

The Xilinx FPGA and its associated configuration PROMs can be programmed using the Xilinx software and a Xilinx JTAG device programming cable. The PROMs can also be programmed through the PCI interface without removing the IPM Buffer board from the host system. This is done by first using the Xilinx Impact software to generate a .svf formatted file. This file is then converted into a .xsvf formatted file using the SVF2XSVF502.EXE program:

```
c:\> svf2xsvf02 -r 0 -i ipmfpga.svf -o ipmfpga.xsvf
```

The IPM Buffer control program can then be used to program the FPGA flash memory on the buffer board. Once programmed, the buffer board host system must be power cycled for the changes to take affect.

*In System FPGA Programming technical details*

In system programming of the FPGA flash memories is done by accessing the JTAG signals on the board through the PCI interface. When in-system programming is enabled, the external JTAG connector on the board is disabled. Enabling in-system programming is done by driving the USERo pin on the PLX PCI interface chip low. Once low, the JTAG pins can be accessed through the PCI interface. Details of programming can be found in Xilinx XAPP058: <http://www.xilinx.com>. Once programming is complete, the USERo output should be returned to a high state.

In-system pin mapping is as follows. Refer to the PLXPCI9656BA user manual for register locations:

USERo output enable:            PLX CNTRL(19)

USERo output:	PLX CNTRL(16)
JTAG_TCLK:	PLX CNTRL(24)
JTAG_TMS:	PLX CNTRL(25)
JTAG_TDI:	PLX CNTRL(26)
JTAG_TDO:	CSRBase + xF4(0)

Programming sequence:

Set USERo output to a known state:

CNTRL(16) <= 0x1

Enable USERo output:

CNTRL(19) <= 0x1

Set USERo output to enable in-system programming:

CNTRL(16) <= 0x0

Once enabled, the JTAG signals on the board can be accessed via register bits. By sequencing the JTAG signals as described in XAPP058, the two FPGA flash memories can be programmed.

Upon completion of the programming operation, disable in-system programming:

CNTRL(16) <= 0x1

After the flash memories have been programmed, the IPM Buffer board and its host system must be power cycled for the changes to take affect. Simply rebooting the board will not cause the FPGA to reload its program.

If in-system programming is interrupted or fails for some reason, it is possible to leave the flash memories with corrupted or incomplete data in them. If this happens, the IPM Buffer board will remain functional with the previous FPGA code running, but once the module is power cycled, it will go into an unusable state. To reflash an IPM Buffer in this state, DIP Switch SW-1(3) must be set to diagnostic mode (off) and the flash memories programmed using the Xilinx cable.

*Using the IPM BUF Control program to re-program the IPM Buffer FPGA*

The IPMBUF Control program supports in-system programming of the IPM Buffer module. To program a buffer board, select the JTAG tab, browse to the desired .xsvf data file, and press the “program” button. Once programming is complete, power cycle the host system when prompted.

*Creating an in-system programming FPGA data file*

The data file required by the IPMBUF Control program is in the XSVF format. To create this file, the Xilinx Impact software must first create .MCS files.

Once a successful Place and Route is complete, the MCS files can be created using Impact as follows.

- 1) Start Impact and select “Prepare Configuration File” from the “Operation Mode Selection” dialog.
- 2) Select “PROM File” from the “Prepare Configuration Files” dialog.
- 3) At the “Prepare PROM Files” dialog, select “Parallel PROM”, and enter “ipmfpga” (no quotes) in the “PROM File name” box.
- 4) Select “512K” for the “Parallel PROM Density”, and press the “Add” button twice to add two devices.
- 5) Press “Next” at the summary dialog, and press “Add File...” at the “Add Device File” dialog.
- 6) Browse to the “ipmfpga.bit” file and select it.
- 7) Select “No” to add another design file.
- 8) Press “finish”, and “Yes” to generate the file now.
- 9) Two new files will be created. ipmfpga\_0.mcs, and ipmfpga\_1.mcs. Close Impact.

To create the .XSVF file from the .MCS files, continue.

- 1) Start Impact. Select “Mode” from the pull-down bar and select “File Mode”.
- 2) Select the “SVF-STAPL-XSVF” tab.
- 3) Right-click in the upper pane and select “add Xilinx device”.
- 4) When prompted, select “Create” and “XSVF”.
- 5) Enter the .xsvf file name: ipmfpga.xsvf, and press “Save”. Press “OK” to start adding devices.
- 6) Browse to and add the ipmfpga\_0.mcs file. Select “xc18v04\_vq44” for the part name and press “OK”.
- 7) Right-click in the upper pane and select: “Add Xilinx Device”. Select “ipmfpga\_1.mcs. Select “xc18v04\_vq44” for the part name and press “OK”.
- 8) Right-click on the first Xilinx chip shown (labeled: ipmfpga\_0.mcs) and select “program”.
- 9) Make sure “Erase Before Programming”, “Verify” and “Parallel Mode” are checked. Make sure “Load FPGA” is not checked. Press “OK”.
- 10) Repeat Steps 8 and 9 for the second Xilinx chip (labeled: ipmfpga\_1.mcs).
- 11) Select “Output” from the pull-down menu, and select “XSVF” file and “Stop Writing to XSVF File”.
- 12) Close Impact.

### ***DIP Switch Settings***

The IPM Buffer module is equipped with an eight-switch DIP switch to allow enabling of diagnostic features. These switches do not need to be changed for normal operation of the module. Note the the switches are number 1 to 8 beginning at pin 1 of SW1.

DIP Switch SW-1 switches are defined as follows:

SW1	Reserved (off)
SW2	Reserved (off)

SW3	IPM Buffer diagnostic/recovery mode enable
SW4	Reserved (off)
SW5	Reserved (off)
SW6	Reserved (off)
SW7	Reserved (off)
SW8	Diagnostic Boundary Scan Enable

- SW1 – 2      Reserved. Must be set to the Off position.
- SW3          When set to On, the Buffer module is in normal mode. The J3 JTAG port is connected to the two FPGA Flash memories only (U8 and U9). When set to Off, the Buffer module is in diagnostic mode. The J3 JTAG port daisy chains the two FPGA Flash memories, the FPGA, and the PLX PCI Interface chip in the JTAG chain. In-system access to the JTAG port is disabled, and the J3 connector is forced enabled.
- SW4 – 7      Reserved. Must be set to the Off position.
- SW8          Diagnostic Boundary Scan Enable. When On, the IPM Buffer module is in boundary scan diagnostic mode, and the module will not respond to PCI accesses. When Off, boundary scan testing of the module is disabled, and normal operation is enabled.

### ***Indicator LEDs***

The rear panel of the IPM Buffer contains five multi-color LEDs. These LEDs are labeled D9, D10, D11, D15, and D16 with D9 being the top-most LED. There are also five more multi-color LEDs on the board that are visible only when looking inside the host system. These are labeled D1 through D5 with D1 nearest the rear panel of the host system. The function of the LEDs is as follows:

D1	FPGA PCI access
D2	DDR Memory Activity
D3	Reserved
D4	SERDES Channel 0 through 3 data activity
D5	SERDES Channel 4 through 7 data activity
D9	Paroli Channel 1 Power Good
D10	SERDES Channel 0 through 3 locked status
D11	SERDES Channel 4 through 7 locked status
D15	Link Channel 0 through 3 data frame lock status
D16	Link Channel 4 through 7 data frame lock status

- D1      FPGA PCI Access. LED will light blue when the FPGA is being accessed through the PCI interface. Note that registers accessed on the PLX PCI interface chip will not cause D1 to light.
- D2      DDR Memory Activity. D2 will light Green when the DDR memory is being read or written.
- D3      Reserved. Always dark.

- D4 SERDES Channel 0 through 3 data activity. D4 lights blue when recognizable data is seen on any of channels 0 through 3. D4 lights red if data errors are seen on any of channels 0 through 3.
- D5 SERDES Channel 4 through 7 data activity. D4 lights blue when recognizable data is seen on any of channels 4 through 7. D4 lights red if data errors are seen on any of channels 4 through 7.
- D9 Paroli Channel 1 Power Good. Lights blue if the light power detected on channel zero of the Paroli optical receiver is good.
- D10 SERDES Channel 0 through 3 locked status. Lights green if the SERDES PLLs are locked on channels 0 through 3. Lights red if none of the PLLs on channels 0 through 3 are locked, and lights orange if at least one, but not all four channels are locked.
- D11 SERDES Channel 4 through 7 locked status. Lights green if the SERDES PLLs are locked on channels 4 through 7. Lights red if none of the PLLs on channels 4 through 7 are locked, and lights orange if at least one, but not all four channels are locked.
- D15 Link Channel 0 through 3 data frame lock status. Lights green if the IPM Buffer receive logic recognizes the data frames coming on all four of channels 0 through 3. Lights red if none of these four channels have recognizable data, and lights orange if at least one, but not all of these four channels have recognizable data.
- D16 Link Channel 4 through 7 data frame lock status. Lights green if the IPM Buffer receive logic recognizes the data frames coming on all four of channels 4 through 7. Lights red if none of these four channels have recognizable data, and lights orange if at least one, but not all of these four channels have recognizable data.

### ***Initial Board Configuration***

Upon first powering up a newly assembled IPM Buffer board, the following steps should be taken:

1. With DDR memory not installed, set DIP switches to switch 3 on and all others off.
2. Ohm all power supplies to ground to verify that no power planes are shorted to ground.
3. Insert board into 3.3 Volt PCI slot and turn on power.
4. Verify all voltages. Note that VREF\_1.25V will read high until Xilinx is programmed.
5. Connect Xilinx programming cable to J4 and program the CPLD.
6. Turn off power and set DIP switches to switch 8 on and all others off.
7. Install boundary scan SODIMM and connect boundary scan JTAG connector to J3.
8. Install boundary scan jumpers at J1 and J2. Note that the empty pins on the J1 connector face the pin 1 end of the connector.
9. Connect boundary scan controller to the network and turn it on.

10. Turn on power to the IPM Buffer host system, and start “Scanplus Runner”.
11. Load and run test: IPM\_interconnect\_withU5.tsp. If test fails view log and debug problem.
12. Turn off power and remove boundary scan connections.
13. Set DIP switch 3 to on and all others off.
14. Turn on power and connect Xilinx programming cable to J3.
15. Using Impact, program the two Xilinx PROMs with the current FPGA microcode.
16. Run PLXMon and program the PLX’s serial EPROM using the utility included in PLXMon. Use file: eedata8.eep.
17. Reboot system and verify that IPM Buffer board functions correctly.

### ***Power Requirements***

Xilinx X2VP20 5.5 Watts

Infineon V23832-R 1.3 Watts

Infineon HYS64D64020GDL-7-A, 5 Watts (x 2)

PLX PCI9656BA 830 mW

Total estimated maximum power requirements: 19 Watts.