



**Nuclear Physics Division**  
*Fast Electronics Group*

## **SSP Manual**

**Benjamin Raydo**  
**14 January 2014**

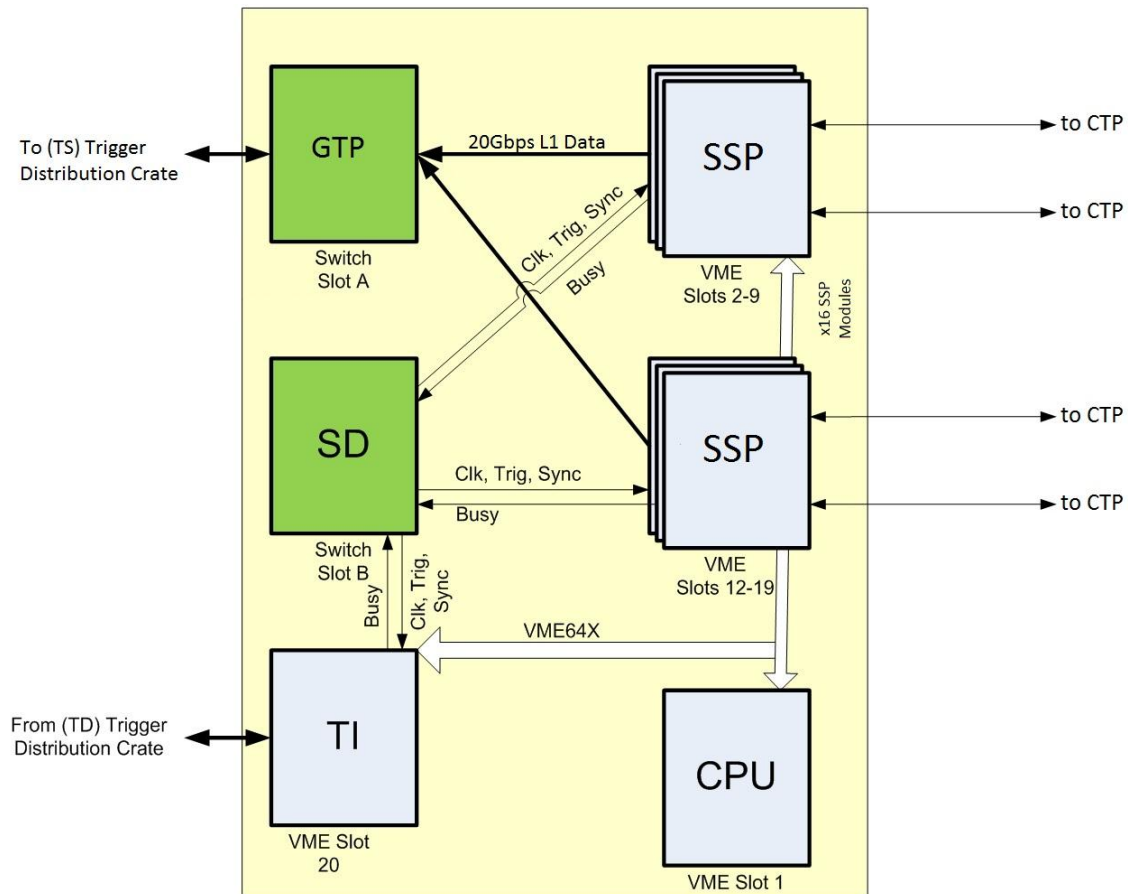
## Table of Contents

| <b>Section</b> | <b>Title</b>              | <b>Page</b> |
|----------------|---------------------------|-------------|
| 1              | Introduction              | 3           |
| 2              | Purpose of Module         | 3           |
| 3              | Function Description      | 4           |
| 4              | Specifications            | 6           |
| 5              | PCB Layout View           | 7           |
| 6              | VSCM Readout Data Format  | 8           |
| 7              | VME Registers             | 11          |
|                | Document Revision History | 31          |

## 1 Introduction

The Subsystem Processor (SSP) module participates in the Level 1 trigger logic for the new 12GeV experimental halls at Jefferson Lab. The SSP module receives Level 1 data streams from up to 8 crates (from the Crate Trigger Processor, CTP) for a single detector subsystem (multiple SSP modules can handle multiple detector subsystems). The SSP module processes the multiple Level 1 data streams to produce a single output stream that is passed directly to the Global Trigger Processor (GTP). Multiple SSP modules can be used to accommodate subsystems with more than 8 crates (in this case the GTP may need to perform a final computation to combine the multiple SSP streams). Figure 1a shows where this module sits along with some of the critical signals that it distributes to the various modules in the system. The Global Trigger Crate supports up to 8 SSP modules.

Figure 1a: SSP in the Global Trigger Crate



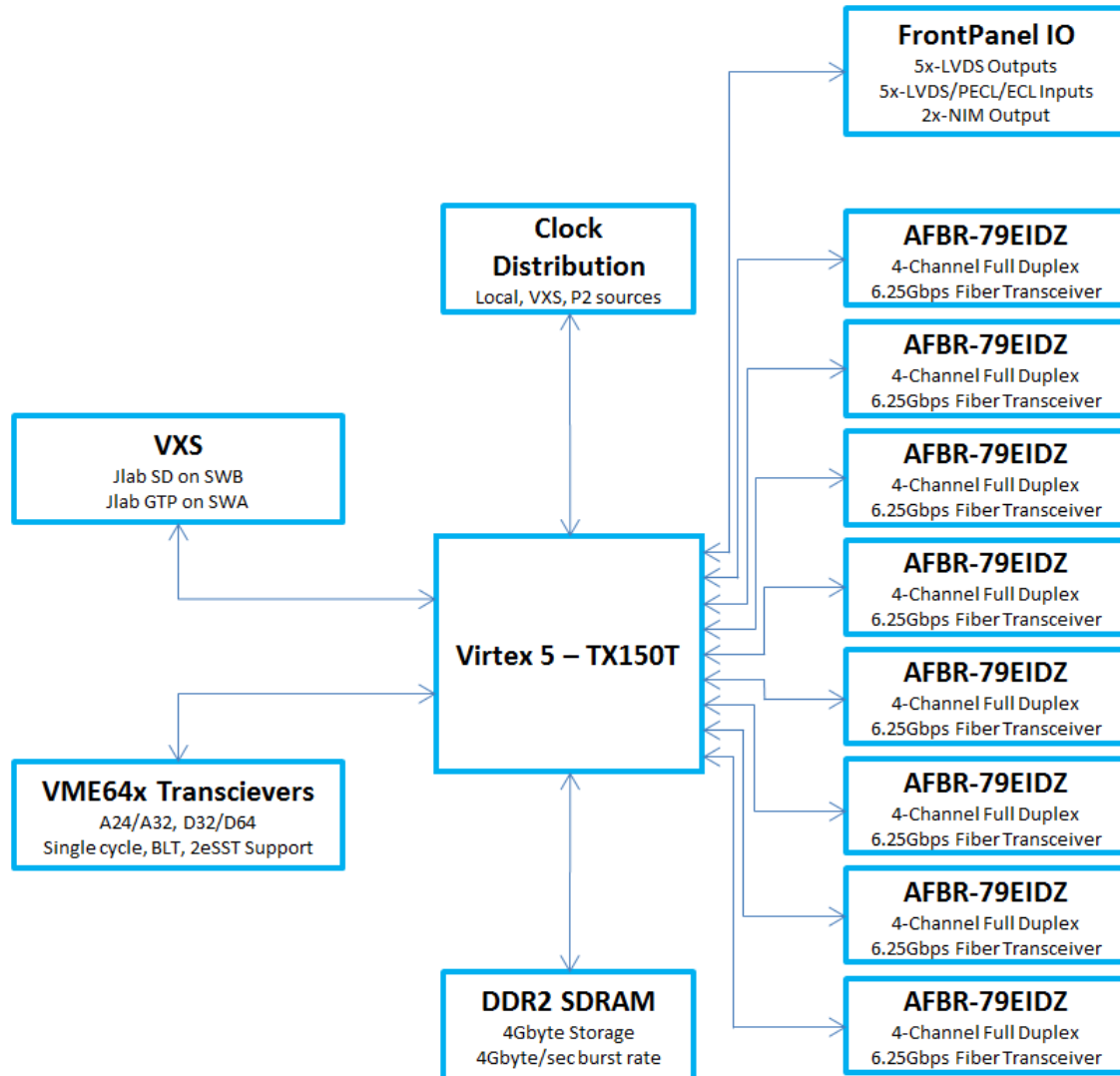
## 2. Purpose of the module

The SSP module communicates with the GTP and CTP using high speed serial links (2 lanes at 5Gbps for the GTP, 4 lanes at 2.5GBps for the CTP). For the CTP to SSP link a 4 lane full duplex fiber optic link is used to support the high data rate and long distance transmission (up to 150meters). The SSP to GTP link uses a 2 lane full duplex copper link over the VXS backplane. The links, after encoding, provide effectively 32bits of data every 4ns that can be used to transfer data that form the L1 trigger. The information sent on this link is specific to the experiment and detector which will be discussed in a separate document.

### 3. Functional Description

In Figure 3a the block diagram of the SSP is shown. The TX150T FPGA performs most of the work providing the VME interface, event building, event buffer logic, L1 trigger logic, SDRAM controller, and SerDes for fiber and VXS. In terms of the L1 trigger processing, data flows into the FPGA from the fiber transceivers and is processed then passed to the GTP through the VXS SWA port.

Figure 3a: SSP Hardware Block Diagram



#### 3.1 VME Interface

The VME interface is used to provide access to configuration registers on the SSP, bridge access to the fiber registers, and provide a high bandwidth interface to the CPU for event readout.

The A32 address space is dedicated to the event builder FIFO which is only used on some experiments where the SSP trigger data is used to tag each event read out. Depending on the application the FIFO exists inside the FPGA if it is relatively small or can exist in external SDRAM for very large buffers. Data can be read from the FIFO using single-cycle and block transfer VME protocols. Typically block transfer protocols will be used for event readout and specifically the 2eSST is intended for use to maximize performance. The 2eSST protocols provides nearly 200MB/s sustained transfer rate and supports the proprietary Jlab token-passing scheme that allows a single DMA operation on the CPU to transfer data from all SSP modules in a sequential manner eliminating overhead compared to individual board transfers.

The A24 address space is reserved for board register access. This address range does not support block transfer modes. Register access details will be provided in the board register description section discussed later.

### 3.2 VXS/Front Panel I/O

The VXS connection is used to interface to the trigger system without the need for loose cabling. This interface provides the following signals:

| Signal      | Description                                    | Direction    | Signal Type |
|-------------|--|--------------|-------------|
| Clock       | 250MHz System Synchronous Clock                | Input        | LVPECL      |
| Trig1       | L1 accept trigger bit, synchronous to clock    | Input        | LVPECL      |
| Trig2       | L1 accept trigger bit, synchronous to clock    | Input        | LVPECL      |
| Sync        | L1 synchronization bit, synchronous to clock   | Input        | LVPECL      |
| Busy        | Module busy signal                             | Output       | LVTTTL      |
| Token In    | Used in VME 2eSST token passing scheme         | Input        | LVDS        |
| Token Out   | Used in VME 2eSST token passing scheme         | Output       | LVDS        |
| Trigger Out | Module trigger bit                             | Output       | LVDS        |
| SD Link     | Undefined serial link to SD                    | Output       | LVDS        |
| L1 Trigger  | 5Gbps per lane (4) used to generate L1 trigger | Input/Output | CML         |

#### **Clock**

This clock signal is derived from the TI or Trigger Distribution Crate and is used to allow synchronous operation across multiple modules within a crate as well as across multiple crates.

#### **Trig1, Trig2**

These trigger bits tell the module when to capture and store an event.

#### **Sync**

The sync signal is used to align/start board timers at the same time as other boards in the crate and system.

#### **Busy**

Busy is normal held low, but if the SSP module event buffers become close to full the busy signal can be set high to signal that the trigger supervisor must stop sending triggers so the module buffers do not overflow. If buffer an overflow happens event synchronization from this module to another is lost.

#### **Token In/Token Out**

These are used by the VME interface when performing 2eSST transfers with token passing.



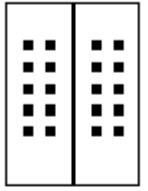
















#### **SDLink**

Currently a undefined serial link to the Signal Distribution board.

#### **L1 Trigger**

This is a high speed serial link to the GTP using the Aurora streaming protocol.

## 4. Specifications

|  |  |
|--|--|
| <b>JLAB<br/>SSP</b>  |  |
| <br><b>POWER</b><br><b>DTACK</b>  |  |
|   |  |
|   |  |
|   <b>1</b>     |  |
|   <b>2</b>     |  |
|   <b>3</b> |  |
|   <b>4</b> |  |
|   <b>5</b> |  |
|   <b>6</b> |  |
|   <b>7</b> |  |
|   <b>8</b> |  |
| <b>VXS</b><br><b>2-9,12-19</b>   |  |

**MECHANICAL**

- Single width VITA 41 Payload Module

**HIGH SPEED SERIAL P0 INPUTS/OUTPUTS:**

- 250MHz LVPECL Clock
- Trig 1, Trig 2, Sync LVPECL Inputs
- 4x 5Gbps Lanes to GTP (only 2 used for GTP)

**Front Panel INPUTS/OUTPUTS:**

- Ribbon-Fiber Optic transceiver
  - AFBR-79EIDZ
- 2x NIM LEMO Outputs
- 5x LVDS Outputs
- 5x AnyLevel Differential Inputs (LVPECL, ECL, LVDS)

**INDICATORS: (Front Panel)**

- Power OK – Green LED
- VME DTACK – Red LED
- Status – Yellow LED

**EVENT BUILDER:**

- Large event FIFO
- High trigger rate capable >200kHz
- 8μs maximum trigger latency

**PROGRAMMING:**

- On board JTAG Port, VME

**POWER REQUIREMENTS:**

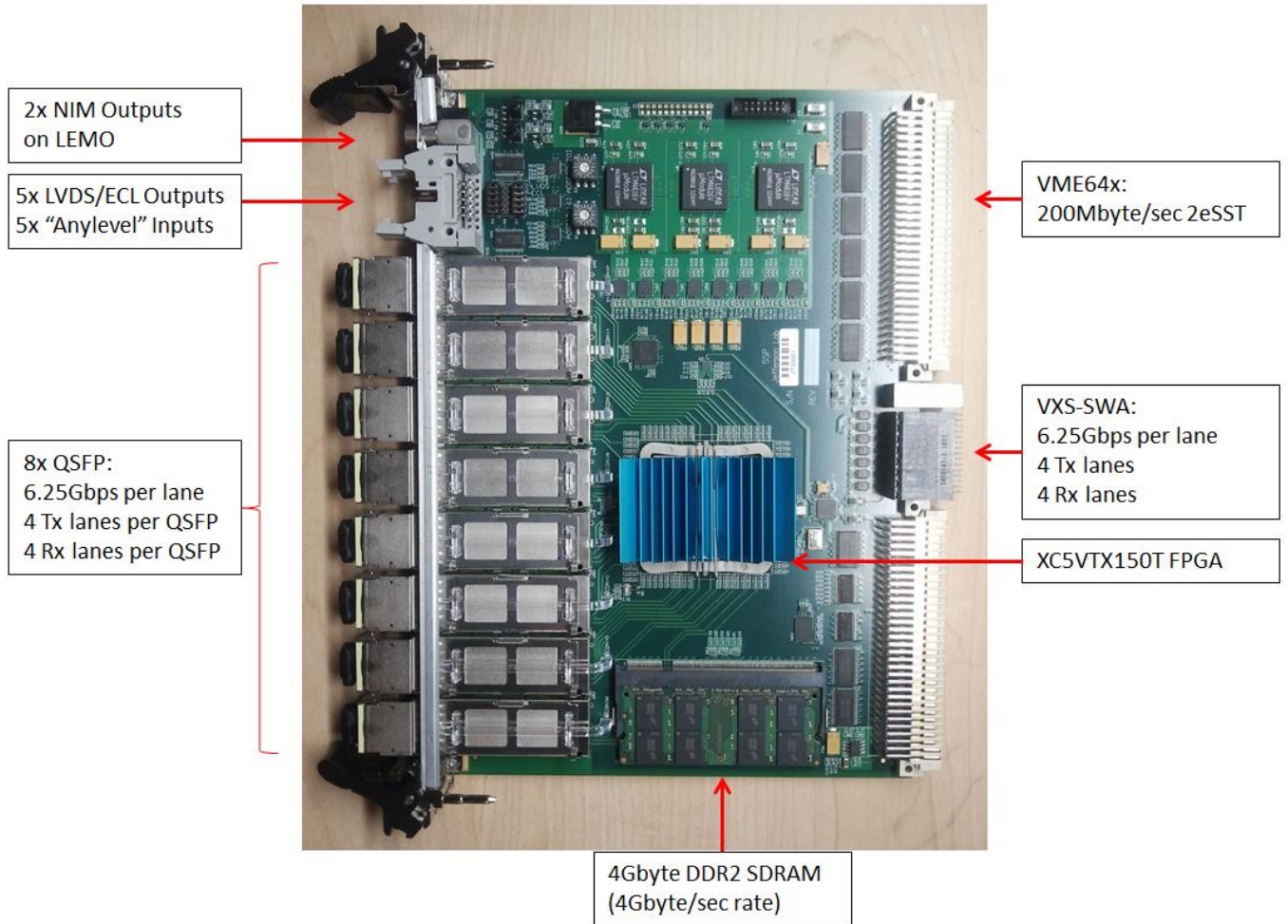
- +5.0v @ 7A Typ, 10A Peak

**ENVIRONMENT:**

- Commercial grade components (70 Celsius)
- Force air cooling required: TBD

## 5. PCB Assembly View

The SSP PCB is an 18-layer impedance controlled FR-370HR stackup



## 6. SSP Readout Data Format

The SSP readout data format utilizes the same encoding scheme defined for the JLAB FADC250. The word length for the readout data is 32bits. The event length is variable and depends on several factors (detector occupancy, headers, trailers, filler words).

### Data Word Categories

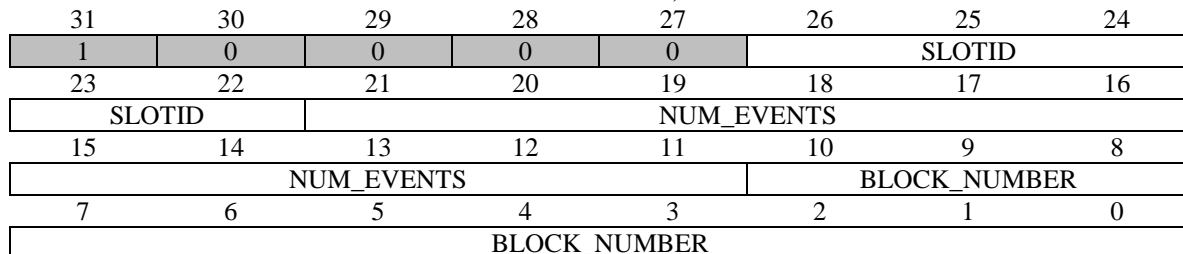
Data words from the module are divided into two categories: Data Type Defining (bit 31 = 1) and Data Type Continuation (bit 31 = 0). Data Type Defining words contain a 4-bit data type tag (bits 30 - 27) along with a type dependent data payload (bits 26 - 0). Data Type Continuation words provide additional data payload (bits 30 - 0) for the *last defined data type*. Continuation words permit data payloads to span multiple words and allow for efficient packing of various data types spanning multiple data words. Any number of Data Type Continuation words may follow a Data Type Defining word.

### Data Type List

|    |                               |
|----|-------------------------------|
| 0  | Block Header                  |
| 1  | Block Trailer                 |
| 2  | Event Header                  |
| 3  | Trigger Time                  |
| 4  | Reserved                      |
| 5  | Reserved                      |
| 6  | Reserved                      |
| 7  | Reserved                      |
| 8  | Reserved                      |
| 9  | Reserved                      |
| 10 | Reserved                      |
| 11 | Reserved                      |
| 12 | Reserved                      |
| 13 | Reserved                      |
| 14 | Data Not Valid (empty module) |
| 15 | Filler Word (non-data)        |

### Data Type: Block Header

Type: 0x0  
 Size: 1 word  
 Description: Indicates the beginning of a block of events. (High-speed readout of a board or a set of boards is done in blocks of events)



### BLOCK\_NUMBER:

Event block number (used to align blocks when building events)

### NUM\_EVENTS:

Number of events in block

### SLOTID:

Slot ID (set by VME64x backplane)



**Data Type: Block Trailer**

Type: 0x1  
 Size: 1 word  
 Description: Indicates the end of a block of events. The data words in a block are bracketed by the block header and trailer.

|           |    |           |    |    |        |    |    |
|-----------|----|-----------|----|----|--------|----|----|
| 31        | 30 | 29        | 28 | 27 | 26     | 25 | 24 |
| 1         | 0  | 0         | 0  | 1  | SLOTID |    |    |
| 23        | 22 | 21        | 20 | 19 | 18     | 17 | 16 |
| SLOTID    |    | NUM_WORDS |    |    |        |    |    |
| 15        | 14 | 13        | 12 | 11 | 10     | 9  | 8  |
| NUM_WORDS |    |           |    |    |        |    |    |
| 7         | 6  | 5         | 4  | 3  | 2      | 1  | 0  |
| NUM_WORDS |    |           |    |    |        |    |    |

**NUM\_WORDS:**

Total number of words in block of events

**SLOTID:**

Slot ID (set by VME64x backplane)

**Data Type: Event Header**

Type: 0x2  
 Size: 1 word  
 Description: Indicates the start of an event. The included trigger number is useful to ensure proper alignment of event fragments when building events. The 27bit trigger number (134M count) is not a limitation, as it will be used to distinguish events within event blocks, or among events that are concurrently being built or transported.

|                |    |    |    |    |                |    |    |
|----------------|----|----|----|----|----------------|----|----|
| 31             | 30 | 29 | 28 | 27 | 26             | 25 | 24 |
| 1              | 0  | 0  | 1  | 0  | TRIGGER_NUMBER |    |    |
| 23             | 22 | 21 | 20 | 19 | 18             | 17 | 16 |
| TRIGGER_NUMBER |    |    |    |    |                |    |    |
| 15             | 14 | 13 | 12 | 11 | 10             | 9  | 8  |
| TRIGGER_NUMBER |    |    |    |    |                |    |    |
| 7              | 6  | 5  | 4  | 3  | 2              | 1  | 0  |
| TRIGGER_NUMBER |    |    |    |    |                |    |    |

**TRIGGER\_NUMBER:**

Accepted event/trigger number

**Data Type: Trigger Time**

Type: 0x3  
 Size: 2 words  
 Description: Time of trigger occurrence relative to the most recent global reset. The time is measured by a 48bit counter that is clocked from the 125MHz system clock. The assertion of the global reset clears the counter. The de-assertion of global reset enables counter and thus sets t=0 for the module. The trigger time is necessary to ensure system synchronization and is useful in aligning event fragments when building events.

Word 1:

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| 1              | 0  | 0  | 1  | 1  | 0  | 0  | 0  |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| TRIGGER_TIME_H |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| TRIGGER_TIME_H |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| TRIGGER_TIME_H |    |    |    |    |    |    |    |

**TRIGGER\_TIME\_H:**

This is the upper 24bits of the trigger time

Word 2:

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| 0              | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| TRIGGER_TIME_L |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| TRIGGER_TIME_L |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| TRIGGER_TIME_L |    |    |    |    |    |    |    |

**TRIGGER\_TIME\_L:**

This is the lower 24bits of the trigger time

**Data Type: Data Not Valid**

Type: 0x14

Size: 1 word

Description: Module has no data available for readout. This can if the module is being read out too quickly after receiving (event building is in process and no data words have been put into the buffer yet) a trigger or if the module doesn't have any events to report.

|           |    |    |    |    |           |    |    |
|-----------|----|----|----|----|-----------|----|----|
| 31        | 30 | 29 | 28 | 27 | 26        | 25 | 24 |
| 1         | 1  | 1  | 1  | 0  | UNDEFINED |    |    |
| 23        | 22 | 21 | 20 | 19 | 18        | 17 | 16 |
| UNDEFINED |    |    |    |    |           |    |    |
| 15        | 14 | 13 | 12 | 11 | 10        | 9  | 8  |
| UNDEFINED |    |    |    |    |           |    |    |
| 7         | 6  | 5  | 4  | 3  | 2         | 1  | 0  |
| UNDEFINED |    |    |    |    |           |    |    |

**Data Type: Filler Word**

Type: 0x15

Size: 1 word

Description: Non-data word appended to the block of events. This is used to force the total number of 32-bit words read out of a module to be a multiple of 2 or 4 when

|           |    |    |    |    |           |    |    |
|-----------|----|----|----|----|-----------|----|----|
| 31        | 30 | 29 | 28 | 27 | 26        | 25 | 24 |
| 1         | 1  | 1  | 1  | 1  | UNDEFINED |    |    |
| 23        | 22 | 21 | 20 | 19 | 18        | 17 | 16 |
| UNDEFINED |    |    |    |    |           |    |    |
| 15        | 14 | 13 | 12 | 11 | 10        | 9  | 8  |
| UNDEFINED |    |    |    |    |           |    |    |
| 7         | 6  | 5  | 4  | 3  | 2         | 1  | 0  |
| UNDEFINED |    |    |    |    |           |    |    |

## 7. VME Registers

All SSP board registers can be accessed through the VME bus in the following mode:

- A24: single cycle accesses, 32bit aligned read or write access (register specific)

Event readout can be access through the VME bus in the following modes:

- A32: single cycle, BLT, MBLT, 2eVME, 2eSST

- Note: transfer rate for 2eSST is 200MB/s

- 

### Register Summary:

| Register Name                            | Description                  | Address Offset |
|--|------------------------------|----------------|
| <b>SspCfg peripheral (offset 0x0000)</b> |                              |                |
| <b>BoardId</b>                           | Board identification         | 0x0000         |
| <b>FirmwareRev</b>                       | Firmware revision            | 0x0004         |
| <b>SpiCtrl</b>                           | Non-volatile flash control   | 0x0008         |
| <b>SpiStatus</b>                         | Non-volatile flash status    | 0x000C         |
| <b>ICapCtrl</b>                          | FPGA configuration interface | 0x0010         |
| <b>ICapDataWr</b>                        | FPGA configuration interface | 0x0014         |
| <b>ICapDataRd</b>                        | FPGA configuration interface | 0x0018         |
| <b>ICapStatus</b>                        | FPGA configuration interface | 0x001C         |

|                                       |               |        |
|---------------------------------------|---------------|--------|
| <b>Clk peripheral (offset 0x0100)</b> |               |        |
| <b>Ctrl</b>                           | Clock control | 0x0000 |
| <b>Status</b>                         | Clock status  | 0x0004 |

|                                      |                    |        |
|--------------------------------------|--------------------|--------|
| <b>Sd peripheral (offset 0x0200)</b> |                    |        |
| <b>SrlSel[]</b>                      | Signal muxing      | 0x0000 |
| <b>PulserPeriod</b>                  | Pulser Period      | 0x0080 |
| <b>PulserLowCycles</b>               | Pulser low cycles  | 0x0084 |
| <b>PulserNPulses</b>                 | Pulser pulse count | 0x0088 |
| <b>PulserStart</b>                   | Pulser start       | 0x008C |
| <b>PulserDone</b>                    | Pulser status      | 0x0090 |
| <b>ScalerLatch</b>                   | Latch scalers      | 0x0100 |
| <b>Scalers[]</b>                     | Scalers            | 0x0104 |

|                                       |                               |        |
|---------------------------------------|-------------------------------|--------|
| <b>Trg peripheral (offset 0x0400)</b> |                               |        |
| <b>Ctrl</b>                           | Trigger control               | 0x0000 |
| <b>SumHistThr</b>                     | Sum histogram threshold       | 0x0014 |
| <b>SumHistWindow</b>                  | Sum histogram integral window | 0x0018 |
| <b>SumHistData</b>                    | Sum histogram bin data        | 0x0024 |

|  |                      |        |
|--|----------------------|--------|
| <b>Serdes peripheral (0x1000, 0x1100, 0x1200, 0x1300, 0x1400,0x1500,0x1600,0x1700,0x1800,0x1900)</b> |                      |        |
| <b>Ctrl</b>  | Control              | 0x0000 |
| <b>CtrlTile0</b>   | Tile 0 control       | 0x0004 |
| <b>CtrlTile1</b>   | Tile 1 control       | 0x0008 |
| <b>DrpCtrl</b>   | Drp control          | 0x000C |
| <b>Status</b>  | Status               | 0x0010 |
| <b>DrpStatus</b>   | Drp status           | 0x0014 |
| <b>ErrTile0</b>  | Tile 0 rx bit errors | 0x0018 |
| <b>ErrTile1</b>  | Tile 1 rx bit errors | 0x001C |
| <b>MonCtrl</b>   | Monitor control      | 0x0030 |
| <b>MonStatus</b>   | Monitor status       | 0x0034 |
| <b>MonMask[]</b>   | Monitor mask         | 0x0040 |
| <b>MonVal[]</b>  | Monitor match values | 0x0060 |
| <b>MonThr[]</b>  | Monitor thresholds   | 0x0080 |
| <b>MonData[]</b>   | Monitor capture data | 0x0090 |

## 7.1 SspCfg Peripheral Registers Section

Basic board information registers can be used to verify that this board is the SSP and check for the software revision, which should be checked for compatibility. Reprogramming the SSP firmware is also possible through these registers.

### Register: BoardId

Address Offset: 0x0000  
 Size: 32bits  
 Reset State: 0x53535020

|          |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| BOARD_ID |    |    |    |    |    |    |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| BOARD_ID |    |    |    |    |    |    |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| BOARD_ID |    |    |    |    |    |    |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| BOARD_ID |    |    |    |    |    |    |    |

### BOARD\_ID (RO):

0x53535020 = "SSP" in ASCII

### Register: FirmwareRev

Address Offset: 0x0004  
 Size: 32bits  
 Reset State: 0xFFFFFFFF

|                    |    |    |        |    |    |    |    |  |
|--------------------|----|----|--------|----|----|----|----|--|
| 31                 | 30 | 29 | 28     | 27 | 26 | 25 | 24 |  |
| -                  | -  | -  | SLOTID |    |    |    |    |  |
| 23                 | 22 | 21 | 20     | 19 | 18 | 17 | 16 |  |
| SSPTYPE            |    |    |        |    |    |    |    |  |
| 15                 | 14 | 13 | 12     | 11 | 10 | 9  | 8  |  |
| FIRMWARE_REV_MAJOR |    |    |        |    |    |    |    |  |
| 7                  | 6  | 5  | 4      | 3  | 2  | 1  | 0  |  |
| FIRMWARE_REV_MINOR |    |    |        |    |    |    |    |  |

### SLOTID(RO):

VME slot id

### SSPTYPE(RO):

Firmware build type [0x00 to 0xFF]

Defined types:  
 0x01 HallD

### FIRMWARE\_REV\_MAJOR (RO):

Major firmware revision number

### FIRMWARE\_REV\_MINOR (RO):

Minor firmware revision number

**Register: SpiCtrl**

Address Offset: 0x0008

Size: 32bits

Reset State: 0x00000080

|         |    |    |    |    |       |         |         |
|---------|----|----|----|----|-------|---------|---------|
| 31      | 30 | 29 | 28 | 27 | 26    | 25      | 24      |
| -       | -  | -  | -  | -  | -     | -       | -       |
| 23      | 22 | 21 | 20 | 19 | 18    | 17      | 16      |
| -       | -  | -  | -  | -  | -     | -       | -       |
| 15      | 14 | 13 | 12 | 11 | 10    | 9       | 8       |
| -       | -  | -  | -  | -  | START | NCS_CLR | NCS_SET |
| 7       | 6  | 5  | 4  | 3  | 2     | 1       | 0       |
| TX_DATA |    |    |    |    |       |         |         |

**TX\_DATA (RW):**

SPI tx data

**NCS\_SET (WO):**

Sets the NCS line of the SPI flash memory

**NCS\_CLEAR (WO):**

Clears the NCS line of the SPI flash memory

**START (WO):**

Begins a SPI transfer. Check SpiStatus DONE bit to determine when transaction is finished.

**Notes:**

- 1) This interface is used for firmware updates and general non-volatile parameter storage.

**Register: SpiStatus**

Address Offset: 0x000C

Size: 32bits

Reset State: 0xFFFFFFFF

|         |    |    |    |      |    |    |    |
|---------|----|----|----|------|----|----|----|
| 31      | 30 | 29 | 28 | 27   | 26 | 25 | 24 |
| -       | -  | -  | -  | -    | -  | -  | -  |
| 23      | 22 | 21 | 20 | 19   | 18 | 17 | 16 |
| -       | -  | -  | -  | -    | -  | -  | -  |
| 15      | 14 | 13 | 12 | 11   | 10 | 9  | 8  |
| -       | -  | -  | -  | DONE | -  | -  | -  |
| 7       | 6  | 5  | 4  | 3    | 2  | 1  | 0  |
| RX_DATA |    |    |    |      |    |    |    |

**RX\_DATA (RO):**

SPI rx data

**DONE (RO):**

'0' - SPI transfer in progress

'1' - SPI transfer is complete

**Register: ICap**

Address Offset: 0x0010  
Size: 32bits  
Reset State: 0xFFFFFFFF

**Notes:**

- 1) This interface provides direct access to the FPGA configuration interface. Only intended use is for a VME based FPGA reload after new firmware has been programmed into flash memory.

**Register: ICapDataWr**

Address Offset: 0x0014  
Size: 32bits  
Reset State: 0x00000000

**Notes:**

- 1) This interface provides direct access to the FPGA configuration interface. Only intended use is for a VME based FPGA reload after new firmware has been programmed into flash memory.

**Register: ICapDataRd**

Address Offset: 0x0018  
Size: 32bits  
Reset State: 0xFFFFFFFF

**Notes:**

- 1) This interface provides direct access to the FPGA configuration interface. Only intended use is for a VME based FPGA reload after new firmware has been programmed into flash memory.

**Register: ICapStatus**

Address Offset: 0x001C  
Size: 32bits  
Reset State: 0xFFFFFFFF

**Notes:**

- 1) This interface provides direct access to the FPGA configuration interface. Only intended use is for a VME based FPGA reload after new firmware has been programmed into flash memory.

## 7.2 Clk Peripheral Registers Section

Clock selection, reset, and status can be access through the following registers.

### Register: Ctrl

Address Offset: 0x0000

Size: 32bits

Reset State: 0x00000000

|        |         |        |          |           |    |            |    |  |
|--------|---------|--------|----------|-----------|----|------------|----|--|
| 31     | 30      | 29     | 28       | 27        | 26 | 25         | 24 |  |
| CLKRST | -       | -      | -        | CLK_LOGIC |    | CLK_SERDES |    |  |
| 23     | 22      | 21     | 20       | 19        | 18 | 17         | 16 |  |
| -      | DRP_DEN | DRP_WE | DRP_ADDR |           |    |            |    |  |
| 15     | 14      | 13     | 12       | 11        | 10 | 9          | 8  |  |
| DRP_DI |         |        |          |           |    |            |    |  |
| 7      | 6       | 5      | 4        | 3         | 2  | 1          | 0  |  |
| DRP_DI |         |        |          |           |    |            |    |  |

#### DRP\_DI (RW):

DRP data input.

#### DRP\_ADDR (RW):

DRP data address.

#### DRP\_WE (RW):

DRP data write enable.

#### DRP\_DEN (RW):

DRP data enable.

#### CLK\_SERDES (RW):

Clock mux input selection:

“00” – disabled clock

“01” – VXS SWB SD clock

“10” – P2 clock

“11” – local clock

#### CLK\_LOGIC (RW):

Clock mux input selection:

“00” – disabled clock

“01” – VXS SWB SD clock

“10” – P2 clock

“11” – local clock

#### CLKRST (RW):

Write ‘1’ to this bit to reset clock PLL whenever clock source is changed (either due to register changes to signal loss). Write ‘0’ to release reset once input clock source is stable.

**Register: Status**

Address Offset: 0x0004

Size: 32bits

Reset State: 0x00000000

|        |    |    |    |    |    |        |         |
|--------|----|----|----|----|----|--------|---------|
| 31     | 30 | 29 | 28 | 27 | 26 | 25     | 24      |
| -      | -  | -  | -  | -  | -  | -      | -       |
| 23     | 22 | 21 | 20 | 19 | 18 | 17     | 16      |
| -      | -  | -  | -  | -  | -  | LOCKED | DRP_RDY |
| 15     | 14 | 13 | 12 | 11 | 10 | 9      | 8       |
| DRP_DO |    |    |    |    |    |        |         |
| 7      | 6  | 5  | 4  | 3  | 2  | 1      | 0       |
| DRP_DO |    |    |    |    |    |        |         |

**DRP\_DO (RO):**

DRP data output.

**DRP\_RDY (RO):**

'1' – DRP\_DO is valid

'0' – DRP\_DO invalid

**LOCKED (RO):**

'1' – SSP Global clock PLL is locked

'0' – SSP Global clock PLL not locked



### 7.3 Sd Peripheral Registers Section

Pulser setup and trigger, sync, general purpose I/O muxing are setup through the the following registers.

**Register: SrcSel[]**

Address Offset: 0x0000 + 4\*SD\_SRC\_x

Size: 32bits

Reset State: 0x00000000

|    |    |    |     |    |    |    |    |  |
|----|----|----|-----|----|----|----|----|--|
| 31 | 30 | 29 | 28  | 27 | 26 | 25 | 24 |  |
| -  | -  | -  | -   | -  | -  | -  | -  |  |
| 23 | 22 | 21 | 20  | 19 | 18 | 17 | 16 |  |
| -  | -  | -  | -   | -  | -  | -  | -  |  |
| 15 | 14 | 13 | 12  | 11 | 10 | 9  | 8  |  |
| -  | -  | -  | -   | -  | -  | -  | -  |  |
| 7  | 6  | 5  | 4   | 3  | 2  | 1  | 0  |  |
| -  | -  | -  | SRC |    |    |    |    |  |

**SRC (RW):**

Selects the signal source for the output signal (output signal is indicated by the index into SrcSel[])

The SD\_SRC\_x ID map is used to determine which index in the SrcSel register array to use:

| SD_SRC_x ID NAME   | Index in SrcSel | Description                    |
|--------------------|-----------------|--------------------------------|
| SD_SRC_LVDSOUT0    | 0               | Front Panel LVDS/ECL output #0 |
| SD_SRC_LVDSOUT1    | 1               | Front Panel LVDS/ECL output #1 |
| SD_SRC_LVDSOUT2    | 2               | Front Panel LVDS/ECL output #2 |
| SD_SRC_LVDSOUT3    | 3               | Front Panel LVDS/ECL output #3 |
| SD_SRC_LVDSOUT4    | 4               | Front Panel LVDS/ECL output #4 |
| SD_SRC_GPIO0       | 5               | Front Panel NIM output #0      |
| SD_SRC_GPIO1       | 6               | Front Panel NIM output #1      |
| SD_SRC_P2_LVDSOUT0 | 7               | P2 LVDS output#0               |
| SD_SRC_P2_LVDSOUT1 | 8               | P2 LVDS output#1               |
| SD_SRC_P2_LVDSOUT2 | 9               | P2 LVDS output#2               |
| SD_SRC_P2_LVDSOUT3 | 10              | P2 LVDS output#3               |
| SD_SRC_P2_LVDSOUT4 | 11              | P2 LVDS output#4               |
| SD_SRC_P2_LVDSOUT5 | 12              | P2 LVDS output#5               |
| SD_SRC_P2_LVDSOUT6 | 13              | P2 LVDS output#6               |
| SD_SRC_P2_LVDSOUT7 | 14              | P2 LVDS output#7               |
| SD_SRC_TRIG        | 15              | SSP internal trigger           |
| SD_SRC_SYNC        | 16              | SSP internal sync              |

Possible values for SRC contents of register:

| SD_SRC_SEL_x         | Value | Source Signal Description              |
|----------------------|-------|--|
| SD_SRC_SEL_0         | 0     | Drive constant '0'                     |
| SD_SRC_SEL_1         | 1     | Drive constant '1'                     |
| SD_SRC_SEL_SYNC      | 2     | VXS SWB Sync                           |
| SD_SRC_SEL_TRIG1     | 3     | VXS SWB Trig1                          |
| SD_SRC_SEL_TRIG2     | 4     | VXS SWB Trig2                          |
| SD_SRC_SEL_LVDSIN0   | 5     | Front panel LVDS input#0               |
| SD_SRC_SEL_LVDSIN1   | 6     | Front panel LVDS input#1               |
| SD_SRC_SEL_LVDSIN2   | 7     | Front panel LVDS input#2               |
| SD_SRC_SEL_LVDSIN3   | 8     | Front panel LVDS input#3               |
| SD_SRC_SEL_LVDSIN4   | 9     | Front panel LVDS input#4               |
| SD_SRC_SEL_P2LVDSIN0 | 10    | P2 LVDS input#0                        |
| SD_SRC_SEL_P2LVDSIN1 | 11    | P2 LVDS input#1                        |
| SD_SRC_SEL_P2LVDSIN2 | 12    | P2 LVDS input#2                        |
| SD_SRC_SEL_P2LVDSIN3 | 13    | P2 LVDS input#3                        |
| SD_SRC_SEL_P2LVDSIN4 | 14    | P2 LVDS input#4                        |
| SD_SRC_SEL_P2LVDSIN5 | 15    | P2 LVDS input#5                        |
| SD_SRC_SEL_P2LVDSIN6 | 16    | P2 LVDS input#6                        |
| SD_SRC_SEL_P2LVDSIN7 | 17    | P2 LVDS input#7                        |
| SD_SRC_SEL_PULSER    | 18    | Pulser output                          |
| SD_SRC_SEL_BUSY      | 19    | Event builder busy                     |
| SD_SRC_SEL_TRIGGER0  | 20    | SSP firmware specific trigger signal#0 |
| SD_SRC_SEL_TRIGGER1  | 21    | SSP firmware specific trigger signal#1 |
| SD_SRC_SEL_TRIGGER2  | 22    | SSP firmware specific trigger signal#2 |
| SD_SRC_SEL_TRIGGER3  | 23    | SSP firmware specific trigger signal#3 |
| SD_SRC_SEL_TRIGGER4  | 24    | SSP firmware specific trigger signal#4 |
| SD_SRC_SEL_TRIGGER5  | 25    | SSP firmware specific trigger signal#5 |
| SD_SRC_SEL_TRIGGER6  | 26    | SSP firmware specific trigger signal#6 |
| SD_SRC_SEL_TRIGGER7  | 27    | SSP firmware specific trigger signal#7 |

**Register: PulserPeriod**

Address Offset: 0x0080  
 Size: 32bits  
 Reset State: 0x00000000

|        |    |    |    |    |    |    |    |
|--------|----|----|----|----|----|----|----|
| 31     | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| PERIOD |    |    |    |    |    |    |    |
| 23     | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| PERIOD |    |    |    |    |    |    |    |
| 15     | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| PERIOD |    |    |    |    |    |    |    |
| 7      | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| PERIOD |    |    |    |    |    |    |    |

**PERIOD (R/W):**

Defines number of 4ns ticks for the pulser period

**Register: PulserLowCycles**

Address Offset: 0x0084  
 Size: 32bits  
 Reset State: 0x00000000

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| LOW_CYCLES |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| LOW_CYCLES |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| LOW_CYCLES |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| LOW_CYCLES |    |    |    |    |    |    |    |

**LOW\_CYCLES (R/W):**

Defines number of 4ns ticks of the pulser period the output stays low.

**Register: PulserNPulses**

Address Offset: 0x0088  
 Size: 32bits  
 Reset State: 0x00000000

|       |    |    |    |    |    |    |    |
|-------|----|----|----|----|----|----|----|
| 31    | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| COUNT |    |    |    |    |    |    |    |
| 23    | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| COUNT |    |    |    |    |    |    |    |
| 15    | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| COUNT |    |    |    |    |    |    |    |
| 7     | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| COUNT |    |    |    |    |    |    |    |

**COUNT (R/W):**

0x00000000: disable pulser output  
 0x00000001 to 0xFFFFFFFF: number of periods to deliver pulser output  
 0xFFFFFFFF: infinite cycle count for pulser output

**Notes:**

- 1) When using fixed count of pulses the pulser must be trigger to start by writing to the **PulserStart** register

**Register: PulserStart**

Address Offset: 0x008C  
 Size: 32bits  
 Reset State: 0x00000000

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| PULSER_START |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| PULSER_START |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| PULSER_START |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| PULSER_START |    |    |    |    |    |    |    |

**PULSER\_START (WO):**

Write any value to start pulser operation. The pulse number counter is cleared.

**Register: PulserDone**

Address Offset: 0x0090  
Size: 32bits  
Reset State: 0x00000000

|    |    |    |    |    |    |    |      |
|----|----|----|----|----|----|----|------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24   |
| -  | -  | -  | -  | -  | -  | -  | -    |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16   |
| -  | -  | -  | -  | -  | -  | -  | -    |
| 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8    |
| -  | -  | -  | -  | -  | -  | -  | -    |
| 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0    |
| -  | -  | -  | -  | -  | -  | -  | DONE |

**DONE (RO):**

'0' – pulser is still delivering pulses as defined in **PulserNPulses**  
'1' – pulser is is not active (either disabled or has finished fixed pulse count)

**Register: ScalerLatch**

Address Offset: 0x0100  
Size: 32bits  
Reset State: 0x00000000

|    |    |    |    |    |    |    |         |
|----|----|----|----|----|----|----|---------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24      |
| -  | -  | -  | -  | -  | -  | -  | -       |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16      |
| -  | -  | -  | -  | -  | -  | -  | -       |
| 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8       |
| -  | -  | -  | -  | -  | -  | -  | -       |
| 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0       |
| -  | -  | -  | -  | -  | -  | -  | DISABLE |

**DISABLE (RW):**

'1' – disables non-buffered scalers. Do this prior to reading out scalers  
'0' – enables non-buffered scalers. Do this after readout is complete. This will also reset the scaler contents to 0.

**Register: Scalers[]**

Address Offset: 0x0104  
Size: 32bits  
Reset State: 0x00000000

|        |    |    |    |    |    |    |    |
|--------|----|----|----|----|----|----|----|
| 31     | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| SCALER |    |    |    |    |    |    |    |
| 23     | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| SCALER |    |    |    |    |    |    |    |
| 15     | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| SCALER |    |    |    |    |    |    |    |
| 7      | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| SCALER |    |    |    |    |    |    |    |

**SCALER (RO):**

32bit scaler value. Refer to map index to determine what signal is mapped to which offset in **Scalars[]**. Scalers will stop counting once 0xFFFFFFFF is reached – this value can be considered an indicator of overflow.

| Scaler Name           | Index in Scalers[] | Description  |
|-----------------------|--------------------|--|
| SD_SCALER_SYSCLK      | 0                  | 50MHz counts since latch latch. Use for Reference. |
| SD_SCALER_GCLK        | 1                  | 250MHz global clock. May not always be on.         |
| SD_SCALER_SYNC        | 2                  | VXS SWB Sync input edges                           |
| SD_SCALER_TRIG1       | 3                  | VXS SWB Trig1 input edges                          |
| SD_SCALER_TRIG2       | 4                  | VXS SWB Trig2 input edges                          |
| SD_SCALER_GPIO0       | 5                  | FP NIM output#0 edges                              |
| SD_SCALER_GPIO1       | 6                  | FP NIM output#1 edges                              |
| SD_SCALER_LVDSIN0     | 7                  | FP LVDS input#0 edges                              |
| SD_SCALER_LVDSIN1     | 8                  | FP LVDS input#1 edges                              |
| SD_SCALER_LVDSIN2     | 9                  | FP LVDS input#2 edges                              |
| SD_SCALER_LVDSIN3     | 10                 | FP LVDS input#3 edges                              |
| SD_SCALER_LVDSIN4     | 11                 | FP LVDS input#4 edges                              |
| SD_SCALER_LVDSOUT0    | 12                 | FP LVDS output#0 edges                             |
| SD_SCALER_LVDSOUT1    | 13                 | FP LVDS output#1 edges                             |
| SD_SCALER_LVDSOUT2    | 14                 | FP LVDS output#2 edges                             |
| SD_SCALER_LVDSOUT3    | 15                 | FP LVDS output#3 edges                             |
| SD_SCALER_LVDSOUT4    | 16                 | FP LVDS output#4 edges                             |
| SD_SCALER_BUSY        | 17                 | Busy assertion edges seen                          |
| SD_SCALER_BUSYCYCLES  | 18                 | Number of 50MHz cycles busy was high               |
| SD_SCALER_P2_LVDSIN0  | 19                 | P2 LVDS input#0 edges                              |
| SD_SCALER_P2_LVDSIN1  | 20                 | P2 LVDS input#1 edges                              |
| SD_SCALER_P2_LVDSIN2  | 21                 | P2 LVDS input#2 edges                              |
| SD_SCALER_P2_LVDSIN3  | 22                 | P2 LVDS input#3 edges                              |
| SD_SCALER_P2_LVDSIN4  | 23                 | P2 LVDS input#4 edges                              |
| SD_SCALER_P2_LVDSIN5  | 24                 | P2 LVDS input#5 edges                              |
| SD_SCALER_P2_LVDSIN6  | 25                 | P2 LVDS input#6 edges                              |
| SD_SCALER_P2_LVDSIN7  | 26                 | P2 LVDS input#7 edges                              |
| SD_SCALER_P2_LVDSOUT0 | 27                 | P2 LVDS output#0 edges                             |
| SD_SCALER_P2_LVDSOUT1 | 28                 | P2 LVDS output#1 edges                             |
| SD_SCALER_P2_LVDSOUT2 | 29                 | P2 LVDS output#2 edges                             |
| SD_SCALER_P2_LVDSOUT3 | 30                 | P2 LVDS output#3 edges                             |
| SD_SCALER_P2_LVDSOUT4 | 31                 | P2 LVDS output#4 edges                             |
| SD_SCALER_P2_LVDSOUT5 | 32                 | P2 LVDS output#5 edges                             |
| SD_SCALER_P2_LVDSOUT6 | 33                 | P2 LVDS output#6 edges                             |
| SD_SCALER_P2_LVDSOUT7 | 34                 | P2 LVDS output#7 edges                             |

## 7.4 Trg Peripheral Registers Section

The following registers setup the trigger processing of the SSP.

### Register: Ctrl

Address Offset: 0x0000

Size: 32bits

Reset State: 0x00000000

|          |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -        | -  | -  | -  | -  | -  | -  | -  |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -        | -  | -  | -  | -  | -  | -  | -  |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| GTP_SRC  |    |    |    |    |    |    |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| FIBER_EN |    |    |    |    |    |    |    |

### FIBER\_EN (R/W):

Bit x:

'1' enables fiber port x in trigger processor.

'0' disables fiber port x in trigger processor.

### GTP\_SRC (R/W):

Determines the trigger data sent to the GTP over the VXS backplane:

0 – Fiber port 0 data

1 – Fiber port 1 data

2 – Fiber port 2 data

3 – Fiber port 3 data

4 – Fiber port 4 data

5 – Fiber port 5 data

6 – Fiber port 6 data

7 – Fiber port 7 data

8 – Sum of all fiber ports enabled by FIBER\_EN bits

**Register: SumHistThr**

Address Offset: 0x0014  
 Size: 32bits  
 Reset State: 0x00000000

|     |    |    |    |    |    |    |    |
|-----|----|----|----|----|----|----|----|
| 31  | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| THR |    |    |    |    |    |    |    |
| 23  | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| THR |    |    |    |    |    |    |    |
| 15  | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| THR |    |    |    |    |    |    |    |
| 7   | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| THR |    |    |    |    |    |    |    |

**THE (R/W):**

Threshold applied to Sum trigger data. When Sum trigger data go above this threshold the histogrammer will integrate a window (defined by SumHistWindow) and increment the bin corresponding to this integral value.

**Register: SumHistWindow**

Address Offset: 0x0018  
 Size: 32bits  
 Reset State: 0x00000000

|     |    |    |    |    |    |    |    |
|-----|----|----|----|----|----|----|----|
| 31  | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -   | -  | -  | -  | -  | -  | -  | -  |
| 23  | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| NSA |    |    |    |    |    |    |    |
| 15  | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -   | -  | -  | -  | -  | -  | -  | -  |
| 7   | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| NSB |    |    |    |    |    |    |    |

**NSA(R/W):**

The number of samples to integrate the Sum trigger data after it crosses threshold

**NSB(R/W):**

The number of samples to integrate the Sum trigger data before it crosses threshold

**Register: SumHistData**

Address Offset: 0x0024  
 Size: 32bits  
 Reset State: 0x00000000

|      |    |    |    |    |    |    |    |
|------|----|----|----|----|----|----|----|
| 31   | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| DATA |    |    |    |    |    |    |    |
| 23   | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| DATA |    |    |    |    |    |    |    |
| 15   | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DATA |    |    |    |    |    |    |    |
| 7    | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DATA |    |    |    |    |    |    |    |

**DATA(R/O):**

Read this register 32 times after the histogram has been disabled. The 1st data corresponds to bin 0, the 512<sup>th</sup> read corresponds to bin 31. The bins are scaled log<sub>2</sub> to maintain a large dynamic range over the range of pulse energy than may be obtained.

## 7.5 Serdes Peripheral Registers Section

The following registers setup/monitor each of the Serdes (Fiber and VXS) of the SSP.

### Register: Ctrl

Address Offset: 0x0000  
 Size: 32bits  
 Reset State: 0x00000203

|          |          |    |          |        |         |         |          |
|----------|----------|----|----------|--------|---------|---------|----------|
| 31       | 30       | 29 | 28       | 27     | 26      | 25      | 24       |
| -        | -        | -  | -        | -      | -       | -       | -        |
| 23       | 22       | 21 | 20       | 19     | 18      | 17      | 16       |
| -        | -        | -  | -        | -      | -       | -       | -        |
| 15       | 14       | 13 | 12       | 11     | 10      | 9       | 8        |
| -        | -        | -  | -        | ERR_EN | ERR_RST | RESET   | TXENPRBS |
| 7        | 6        | 5  | 4        | 3      | 2       | 1       | 0        |
| TXENPRBS | RXENPRBS |    | LOOPBACK |        |         | GTRESET | PWRDN    |

#### PWRDN (R/W):

- '1' disable power of the transceiver
- '0' enable power of the transceiver

#### GTRESET (R/W):

- '1' reset transceiver
- '0' release reset of transceiver

#### LOOPBACK (R/W):

- "000" – keep at this value

#### RXENPRBS (R/W):

- "00" – keep at this value

#### TXENPRBS (R/W):

- "00" – keep at this value

#### RESET (R/W):

- '1' reset link
- '0' release reset of link

#### ERR\_RST (R/W):

- '1' reset error counts
- '0' release reset error counts

#### ERR\_EN (R/W):

- '1' enable error counts
- '0' disable error counts (error counters should be disabled when read)



**Register: CtrlTile0**

Address Offset: 0x0004  
 Size: 32bits  
 Reset State: 0x05420542

|                |    |                |    |             |    |                |    |
|----------------|----|----------------|----|-------------|----|----------------|----|
| 31             | 30 | 29             | 28 | 27          | 26 | 25             | 24 |
| -              | -  | -              | -  | TXDIFFCTRL1 |    | TXBUFDIFFCTRL1 |    |
| 23             | 22 | 21             | 20 | 19          | 18 | 17             | 16 |
| TXBUFDIFFCTRL1 |    | TXPREEMPHASIS1 |    |             |    | RXEQMIX1       |    |
| 15             | 14 | 13             | 12 | 11          | 10 | 9              | 8  |
| -              | -  | -              | -  | TXDIFFCTRL0 |    | TXBUFDIFFCTRL0 |    |
| 7              | 6  | 5              | 4  | 3           | 2  | 1              | 0  |
| TXBUFDIFFCTRL0 |    | TXPREEMPHASIS0 |    |             |    | RXEQMIX0       |    |

Do not modify values on this register.

**Register: CtrlTile1**

Address Offset: 0x0008  
 Size: 32bits  
 Reset State: 0x05420542

|                |    |                |    |             |    |                |    |
|----------------|----|----------------|----|-------------|----|----------------|----|
| 31             | 30 | 29             | 28 | 27          | 26 | 25             | 24 |
| -              | -  | -              | -  | TXDIFFCTRL1 |    | TXBUFDIFFCTRL1 |    |
| 23             | 22 | 21             | 20 | 19          | 18 | 17             | 16 |
| TXBUFDIFFCTRL1 |    | TXPREEMPHASIS1 |    |             |    | RXEQMIX1       |    |
| 15             | 14 | 13             | 12 | 11          | 10 | 9              | 8  |
| -              | -  | -              | -  | TXDIFFCTRL0 |    | TXBUFDIFFCTRL0 |    |
| 7              | 6  | 5              | 4  | 3           | 2  | 1              | 0  |
| TXBUFDIFFCTRL0 |    | TXPREEMPHASIS0 |    |             |    | RXEQMIX0       |    |

Do not modify values on this register.

**Register: DrpCtrl**

Address Offset: 0x000C  
 Size: 32bits  
 Reset State: 0x00000000

|         |          |    |    |    |           |           |     |
|---------|----------|----|----|----|-----------|-----------|-----|
| 31      | 30       | 29 | 28 | 27 | 26        | 25        | 24  |
| -       | -        | -  | -  |    | DEN_TILE1 | DEN_TILE0 | DWE |
| 23      | 22       | 21 | 20 | 19 | 18        | 17        | 16  |
| -       | DRP_ADDR |    |    |    |           |           |     |
| 15      | 14       | 13 | 12 | 11 | 10        | 9         | 8   |
| DRP_DIN |          |    |    |    |           |           |     |
| 7       | 6        | 5  | 4  | 3  | 2         | 1         | 0   |
| DRP_DIN |          |    |    |    |           |           |     |

Do not modify values on this register.

**Register: Status**

Address Offset: 0x0010

Size: 32bits

Reset State: 0x00000000

|         |         |         |         |          |          |          |          |
|---------|---------|---------|---------|----------|----------|----------|----------|
| 31      | 30      | 29      | 28      | 27       | 26       | 25       | 24       |
| -       | -       | -       | -       | -        | -        | -        | -        |
| 23      | 22      | 21      | 20      | 19       | 18       | 17       | 16       |
| -       |         |         |         |          |          |          |          |
| 15      | 14      | 13      | 12      | 11       | 10       | 9        | 8        |
|         | SRCRDYN | TXLOCK  | CHUP    | RXPOL3   | RXPOL2   | RXPOL1   | RXPOL0   |
| 7       | 6       | 5       | 4       | 3        | 2        | 1        | 0        |
| LANEUP3 | LANEUP2 | LANEUP1 | LANEUP0 | HARDERR3 | HARDERR2 | HARDERR1 | HARDERR0 |

**HARDERRx (RO):**

'0' – no hard error on lane x

'1' – hard error occurred on lane x. Reset is required.

**LANEUPx (RO):**

'0' – lane x is down.

'1' – lane x is up

**RXPOLx (RO):**

'0' – rx polarity is normal

'1' – rx polarity is inverted

**CHUP (RO):**

'0' – channel is down

'1' – channel is up

**TXLOCK (RO):**

'0' – TX PLL not locked (likely due to missing 250MHz global clock)

'1' – TX PLL is locked

**SRCRDYN (RO):**

'0' – Data is current being received over channel

'1' – Data is not being received over channel

**Register: CrateId**

Address Offset: 0x0020  
 Size: 32bits  
 Reset State: 0x00000000

|         |    |    |    |    |    |    |    |
|---------|----|----|----|----|----|----|----|
| 31      | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -       | -  | -  | -  | -  | -  | -  | -  |
| 23      | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -       | -  | -  | -  | -  | -  | -  | -  |
| 15      | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| CRATEID |    |    |    |    |    |    |    |
| 7       | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CRATEID |    |    |    |    |    |    |    |

**CRATEID (RO):**

The first lower 16bit word received on the transceiver which has been defined to be the CrateId. This is updated any time the link goes from ILDE->DATA (normally only when the system SYNC goes low)

**Register: MonCtrl**

Address Offset: 0x0030  
 Size: 32bits  
 Reset State: 0x00000000

|    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -  | -  | -  | -  | -  | -  | -  | -  |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -  | -  | -  | -  | -  | -  | -  | -  |
| 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -  | -  | -  | -  | -  | -  | -  | -  |
| 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| -  | -  | -  | -  | -  | -  | -  | EN |

**EN (RW):**

'0' – disable serdes data monitor (must be done for readout)  
 '1' – enable serdes data monitor

**Register: MonStatus**

Address Offset: 0x0034  
 Size: 32bits  
 Reset State: 0x00000000

|         |    |    |    |        |        |        |        |
|---------|----|----|----|--------|--------|--------|--------|
| 31      | 30 | 29 | 28 | 27     | 26     | 25     | 24     |
| LATENCY |    |    |    |        |        |        |        |
| 23      | 22 | 21 | 20 | 19     | 18     | 17     | 16     |
| LATENCY |    |    |    |        |        |        |        |
| 15      | 14 | 13 | 12 | 11     | 10     | 9      | 8      |
| -       | -  | -  | -  | CRCOK3 | CRCOK2 | CRCOK1 | CRCOK0 |
| 7       | 6  | 5  | 4  | 3      | 2      | 1      | 0      |
| -       | -  | -  | -  | -      | -      | -      | RDY    |

**RDY (RO):**

'0' – serdes data monitor has not triggered  
 '1' – serdes data monitor has triggered and data is ready for readout

**CRCx (RO):**

'0' – lane x has failed CRC verification during last sync frame  
 '1' – lane x has passed CRC verification during last sync frame

**LATENCY (RO):**

Latency, in 4ns ticks, from SYNC to receipt of first trigger word on serdes

**Register: MonMask**

Address Offset: 0x0040,...  
Size: 32bits  
Reset State: 0x00000000

Used for debugging.

**Register: MonVal**

Address Offset: 0x0060,...  
Size: 32bits  
Reset State: 0x00000000

Used for debugging.

**Register: MonThr**

Address Offset: 0x0080,...  
Size: 32bits  
Reset State: 0x00000000

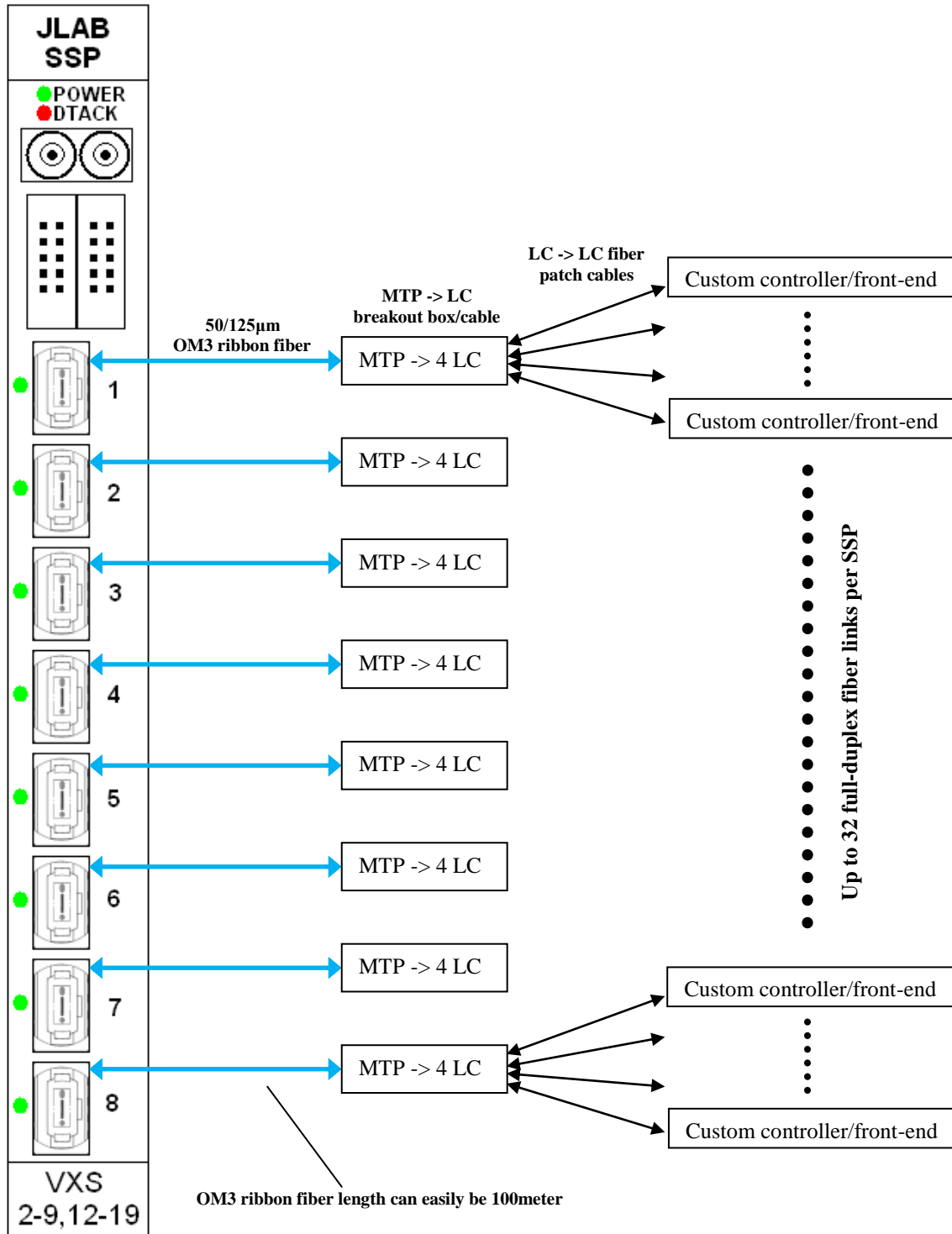
Used for debugging.

**Register: MonData**

Address Offset: 0x0090,...  
Size: 32bits  
Reset State: 0x00000000

Used for debugging.

# Appendix A: Alternate use – front-end readout mode



## General Notes:

- 1) SSP has 32 full-duplex fiber links that can operate at up to 6.25Gbps. These could be partitioned in a few different ways. Shown above would allow each SSP to connect to 32 different controllers; however, if higher bandwidth is needed it may make more sense to deliver the full MTP connection to each controller (in which case only 8 controllers per SSP would be supported, but at >20Gbps per controller).
- 2) Listed lines rates would be reduced to 80% for usable data transfer rates after 8b/10b encoding rules have been applied (to satisfy maximum run length/AC coupling requirements).
- 3) If the fiber optic links deliver enough bandwidth, all front-end data could be stream to the SSP where the readout data trigger matching, data suppression, and event building could be performed. This option would minimize the new hardware developments. Alternatively much of this work could be performed on the front-end boards and the fiber links would be for delivering trigger and synchronization signals in one direction, and triggered data and status in the other direction.
- 4) A synchronizing clock will likely be needed and could be derived from the SSP fiber links. This approach has a few potential problems for the front-end sampling:
  - a. Recovered SerDes clock will be jittery (on the order of 10's of ps RMS jitter). This can be reduced/filtered by a jitter cleaning PLL.
  - b. The recovered SerDes clock will have phase variations each time a link is established. This would result in a timing uncertainty of the recovered clock period, typically 6.4ns at 3.125Gbps (depending on the SerDes it could be a few times higher though). This can be reduced by with some tricks in a configurable SerDes to achieve sub-nanosecond stability, but requires firmware development and testing to quantify the achievable stability.

## Document Revision History

### 8/15/2013 (V2.0 firmware):

- 1) Initial document released with Hall D firmware features implemented.

### 1/13/2014 (V2.1 firmware):

#### SD Peripheral:

- 1) Updated Pulser clock source to use the 250MHz global clock (as registers indicated, 50MHz local clock was incorrectly used before)
- 2) ScalerLatch changed to R/W. Must be set to enable scalers to count, disabled to stop scalers so they can be read. This includes histograms

#### Clk Peripheral:

- 1) Changed clock selection bits register Ctrl. These changes also provide readable values so clock source can be read back.

#### Cfg Peripheral

- 1) added ICapDataWr, ICapDataRd, ICapStatus registers for full FPGA ICAP primitive support to allow FPGA reconfiguration via VME registers.

#### Serdes Peripheral

- 1) In Serdes peripheral: added CrateId registers for fiber transceiver ports

#### Trg Peripheral

- 1) In Trigger peripheral: SumHistWindow NSA/NSB changed to 8bit values
- 2) Removed SumHistCtrl: histogram is now controlled by Sd->ScalerLatch
- 3) Removed SumHistTime: histogram is now controlled by Sd->ScalerLatch, so Sd clk scalers can be used for histogram normalization
- 4) SumHistData changed from 512bins to 32bins (bin 0 =  $2^0$ , bin 1 =  $2^1$ , ...bin 31 =  $2^{31}$ )

#### Cfg Peripheral

- 1) Changed FirmwareRev register to shirk SSPTYPE to 8bits and add the VME SLOTID.