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

# VSCM Manual

**Chris Cuevas**
**Benjamin Raydo**
**12 December 2011**

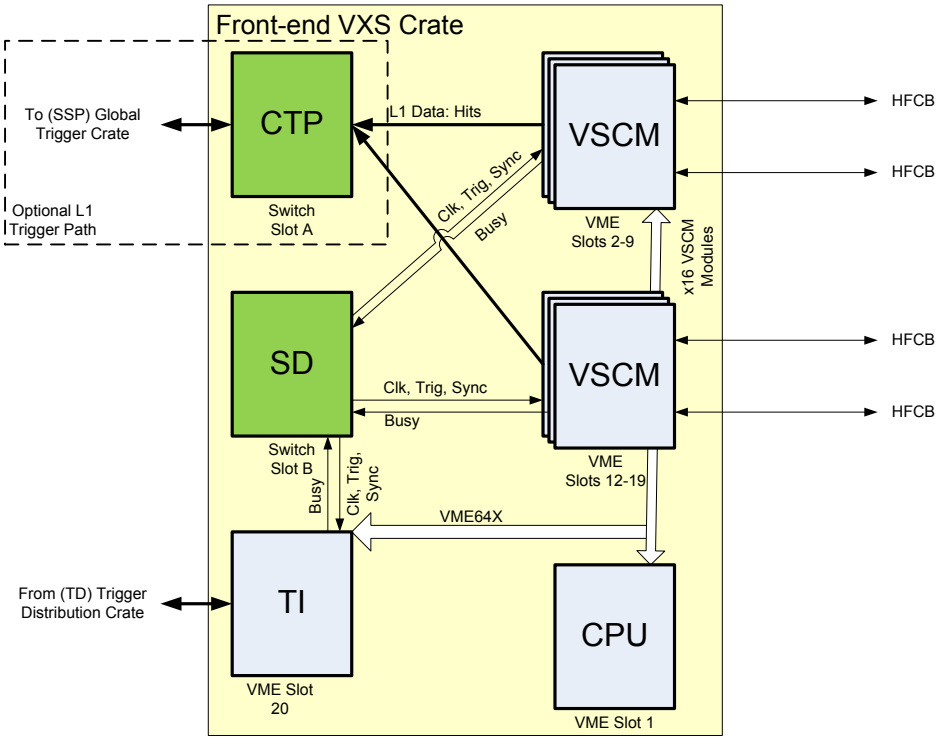
# Table of Contents


| Section | Title | Page |
|---|---|---|
| 1 | Introduction | 3 |
| 2 | Purpose of Module | 3 |
| 3 | Function Description | 4 |
| 4 | Specifications | 10 |
| 5 | PCB Layout View | 11 |
| 6 | VSCM Readout Data Format | 12 |
| 7 | VME Registers | 16 |

# 1 Introduction

The VSCM is a VME/VXS module that communicates with multiple FSSR2 chips on the Jefferson Lab Hybrid Flex Circuit Board (HFCB). The VSCM is responsible for configuring the FSSR2 registers, providing analog calibration pulses to the FSSR2 chips, delivering/monitoring proper control signals (clock, reset, status), and capturing serialized event data from the FSSR2. Each VSCM can interface to 2 HFCB modules and each HFCB has 4 FSSR2 chips. Up to 16 VSCM modules can reside in a VXS crate as shown in Figure 1a. When multiple VSCM modules are used additional cards are required to ensure event & timing synchronization. These additional modules are shown in Figure 1a and are the Trigger Interface (TI) and Signal Distribution (SD). Please refer to the particular module manual for further information. The VSCM does support a stand-alone mode useful when only 1 or 2 HFCB interfaces are used.

**Figure 1a: VSCM in the 20 Slot VXS Crate**



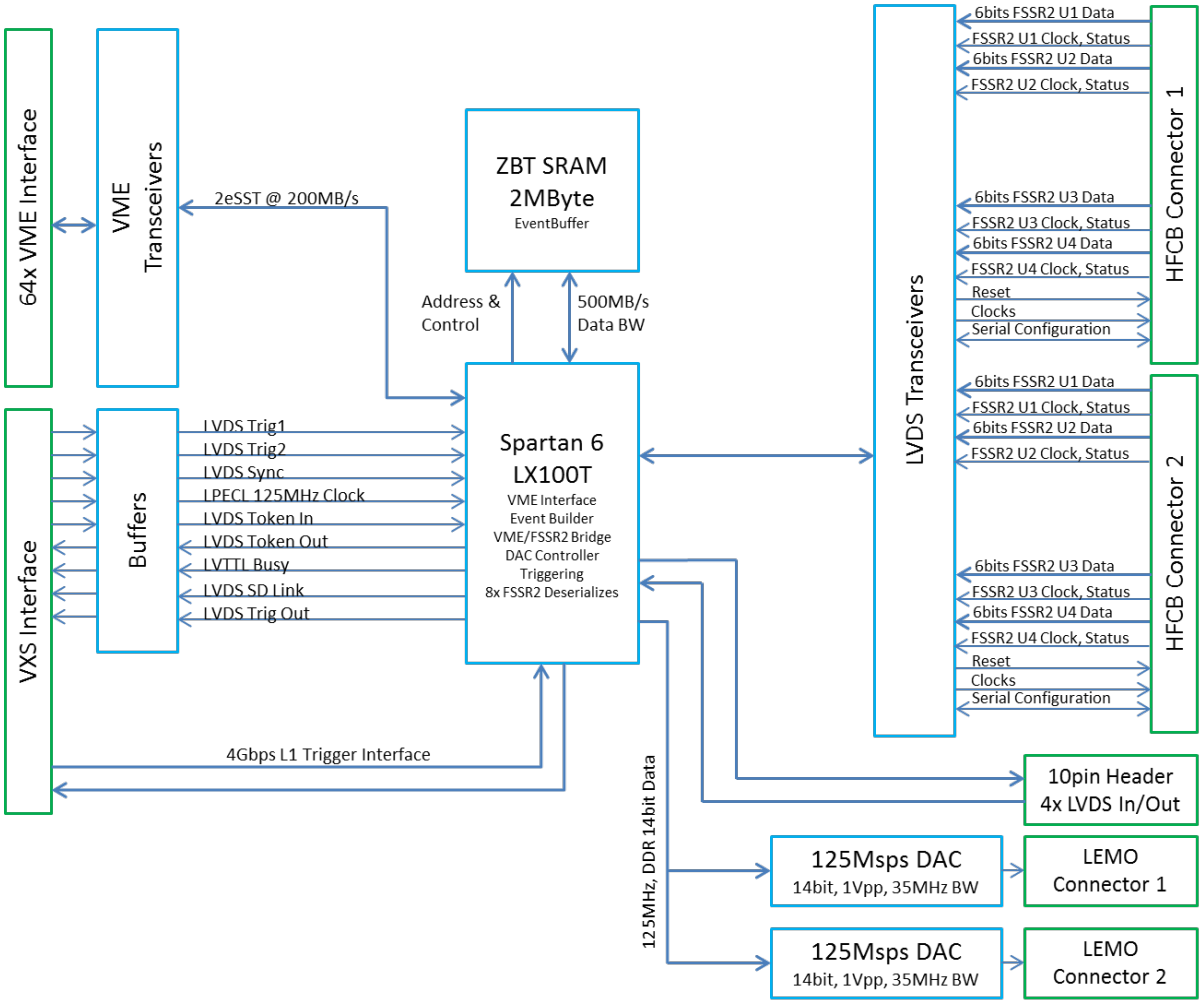

# 2. Purpose of the module

The main purpose of the VSCM is to convert the FSSR2 data-driven information stream into sparsified & triggered events that are correlated with external detectors. Idle status words are suppressed to minimize event size, but these status words are monitored to for diagnostic purposes of the individual FSSR2 chips. The event builder of the VSCM uses the BCO clock timestamp from each FSSR2 data word and matches it to the global system clock timestamp. Since the BCO clock is derived from the global system clock, triggers received by the VSCM will cause the event builder to extract only hits with specific BCO timestamps that correspond to a programmable time window where the physics event could have occurred on the HFCB. When a trigger is received the appropriate FSSR2 data words are copied into an event buffer and pushed into an event FIFO. These events can be readout in order with other modules in the system and event-level synchronization across all modules in the system is maintained.

## 3. Functional Description

In Figure 3a the block diagram of the VSCM is shown. The LX100T FPGA performs most of the work providing the VME interface, event building, event buffer logic, DAC waveform generator, L1 trigger logic, and synchronizes all FSSR2 data paths. Further details on each of these pieces are discussed below.

**Figure 3a: VSCM Hardware Block Diagram**




### 3.1 VME Interface

The VME interface is used to provide access to configuration registers on the VSCM, bridge access to the FSSR2 registers, and provide a high bandwidth interface to the CPU for event readout.

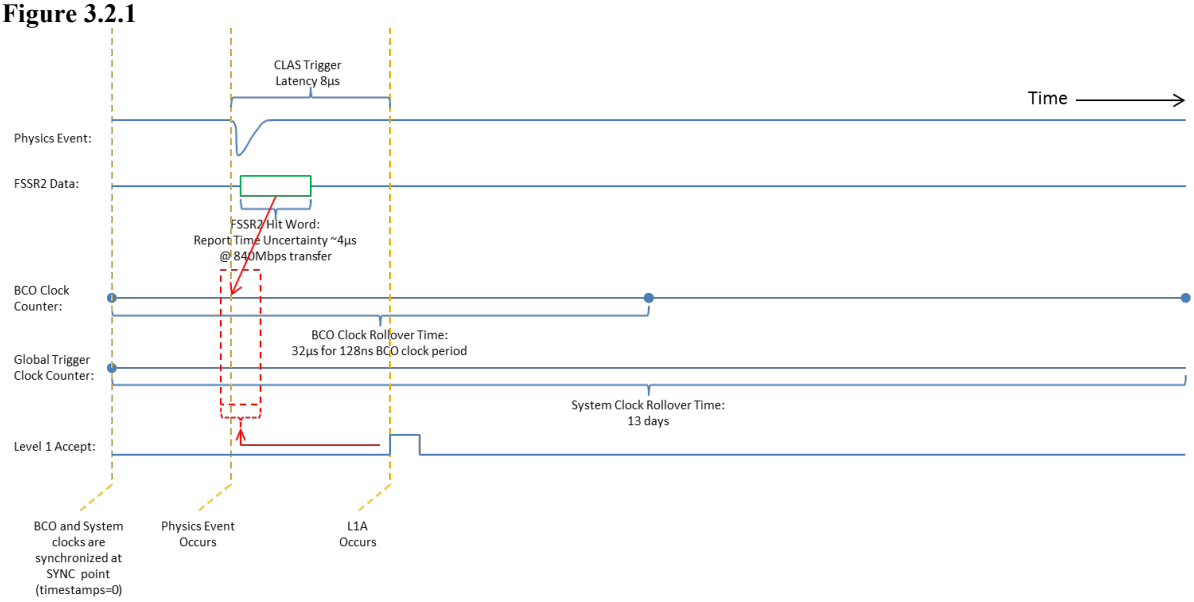
The A32 address space is dedicated to the event builder FIFO, roughly 2MB in size on each VSCM, and can be read using single-cycle and block transfer VME protocols. Typically block transfer protocols will be used for event readout and specifically the 2eSST is intended for use to maximize performance. 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 VSCM 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 Event Builder

In CLAS12, the trigger latency is expected to be around 8µs. The VSCM is setup to extract event data within a programmable look-back window relative to the received trigger. Figure 3.2.1 shows a general timing diagram the may help to see how the event builder is able to select the correct FSSR2 data words that should be associated for a given trigger.

**Figure 3.2.1**




### 3.2.1 Determining which BCO numbers to readout on Trigger

All readout modules in the trigger system are derived from the same global clock (FSSR2 use a BCO clock that is derived from this same clock). All onboard global clock counters and FSSR2 BCO clock counters are synchronously reset.

The trigger data window, defined by the dashed red-line, has an associated start and stop BCO number. After the release of a global reset/sync, the start number is initialized to the value the BCO number would be to match the desired look-back time. The stop number is initialized to the value of the start number plus the desired time window size where FSSR2 hits are desired to capture. For example:

**Operating conditions (All units in system clock tick periods of 8ns):**
BCOPeriod: 16 (=128ns)
Lookback: 1,000 (=8,000ns)
Window: 25 (=200ns)

**Calculated initialization values for the BCO trigger window:**
So the trigger start BCO counter is initialized to:
TRIG_BCO_START = 256 – Lookback / BCOPeriod
TRIG_BCO_START_CNT = (BCOPeriod – Lookback % BCOPeriod) % BCOPeriod

The trigger stop BCO counter is initialized to:
TRIG_BCO_STOP = 256 – (Lookback – Window + 1) / BCOPeriod
TRIG_BCO_STOP_CNT = (BCOPeriod –(Lookback – Window + 1) % BCOPeriod) % BCOPeriod

**Which gives:**
TRIG_BCO_START = 193
TRIG_BCO_START_CNT = 8
TRIG_BCO_STOP = 195

TRIG_BCO_STOP_CNT = 0

The TRIG_BCO_START_CNT and TRIG_BCO_STOP_CNT values are counter that increment at the system clock rate (125MHz) and rollover to 0 when they hit the BCO clock period count (in system clock ticks). When a rollover happens, the corresponding BCO number (TRIG_BCO_START or TRIG_BCO_STOP) will increment, which means the trigger window moves along in time at the system clock rate and tags the BCO start and stop times the trigger window touches.

When synchronization reset is released these start/stop BCO parameters will increment with a frequency of the BCO clock so that the look-back window tracks the desired time and this makes trigger processing easier since the next stages work in terms of BCO numbers. When a trigger is received by the VSCM, all FSSR2 hits reported in the BCO trigger start to stop time range are used to build an event corresponding to that trigger.

In the following table, the BCO numbers that correspond to hits that happened 8,000ns to 7,800ns ago is highlighted indicated the example numbers from above (at T=0):

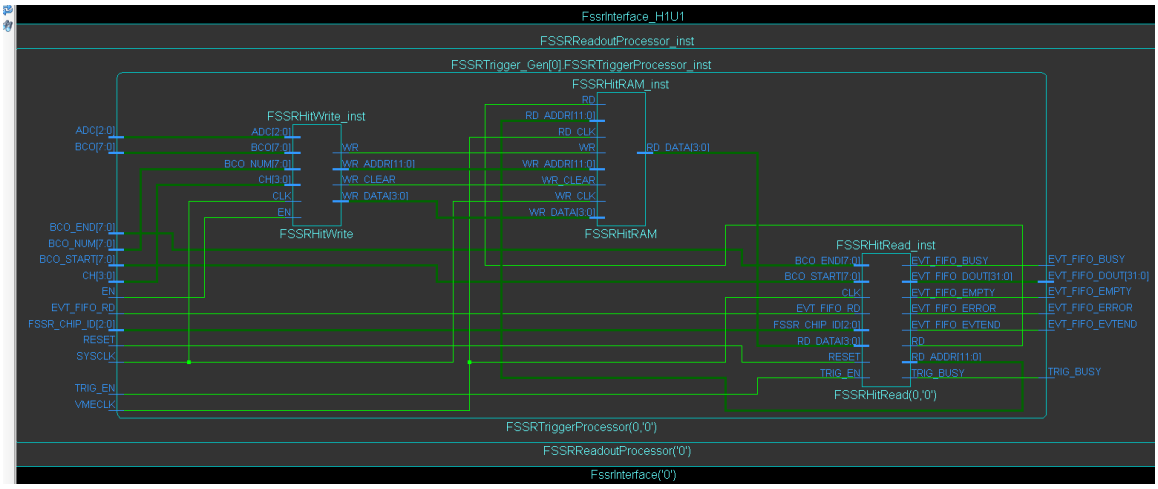
| BCONUM | TrigStartTime(ns) | TrigStopTime(ns) |
|---|---|---|
| 189 | -8576 | -8449 |
| 190 | -8448 | -8321 |
| 191 | -8320 | -8193 |
| 192 | -8192 | -8065 |
| 193 | -8064 | -7937 |
| 194 | -7936 | -7809 |
| 195 | -7808 | -7681 |
| 196 | -7680 | -7553 |
| 197 | -7552 | -7425 |
| 198 | -7424 | -7297 |
| 199 | -7296 | -7169 |

Note that all BCO numbers that span time inside the trigger window are tagged for readout. Stepping this window at the system clock rate of 125MHz provides an important feature that reduces overall occupancies. It is expected that the overall uncertainty of trigger to SVT hit is less than a BCO clock period. By specifying a window that covers this uncertainty you can notice, from the BCO start and stop trigger calculations, that when the window of uncertainty falls within a BCO clock period only 1 BCO clock period of data will be used to extract an event, but if this window overlaps times that fall into 2 or more BCO windows then those events will extract 2 or more BCO windows worth of data,

### 3.2.2 Extracting Triggered Event Hits

There are 8 FSSR2 chips per VSCM and each has a dedicated deserializer inside the XC6SLX100T FPGA. After deserialization, the status words are used for monitoring purposes and the event words are split up into 64 parallel trigger processors that each handle 16 strips out of the total 1,024 per VSCM.

A block-level schematic representation (Xilinx Planahead VHDL-to-schematic translator) for 1 of the identical 64 trigger processors is shown here:




In this schematic, the first parallel trigger processor of the VSCM is shown, which handles strips 0-15 of the first FSSR2.

**FSSRHitWrite**

The "FSSRHitWrite_inst" component receives the strip channel, BCO number, and ADC for any hit the FSSR2 reports for strips 0-15. This forms a 12bit address from the strip number (4bits) and BCO number (8bits) and writes to a 4bit x 4kword dual port RAM shown as the component "FSSRHitRAM_inst". The value written to this address is the 3bit ADC value for the hit and a 1bit valid flag indicating the sample is valid.

The "FSSRHitWrite_inst" component also continuously clears old data in the dual port RAM by clearing the 1bit valid flag of each address with BCO equal to the current BCO clock - 128 (BCO_NUM-128). For an 8MHz BCO clock this means that only the most recent 16µs of RAM is valid (128 BCO clocks at 8MHz = 16µs).

**FSSRHitRead**

The "FSSRHitRead_inst" component receives receives the trigger window parameters (BCO_START and BCO_END) for each trigger that is accepted by the VSCM. When the TRIG_EN signal goes high, the "FSSRHitRead_inst" component extracts all valid hits from the dual port RAM that is in the range of BCO_START and BCO_END and puts these hits into an event FIFO and terminates the end of event with a special word to indicate to the next state event builder consolidator.

The event FIFO signals (EVT_FIFO_*) are used by the event builder consolidator to combine data from 8 trigger processors into a single event/FIFO and this is done again until all 1,024 channels are processed into a single FIFO at the VME readout buffer level. Header and trailer words are put into each event and blocks of events for upstream event builders to ensure event level synchronization.

## 3.3 Event Buffer

The event buffer uses the 2MB external SRAM to form a large event FIFO. This buffer is large enough to hold several hundred events assuming an unrealistic 100% occupancy. This allows the event builder to create blocks of many events, which decreases data overhead. The bandwidth (500MB/s) into the SRAM is enough to support continuous readout at the VME 2eSST 200MB/s bandwidth while simultaneously writing into the buffer at that rate. Also, the VME 2eSST 200MB/s is across typically 10+ boards so the average readout bandwidth per board is much smaller.

### 3.4 Calibration Pulser

The calibration pulser circuit provides a 1Vpp dynamic range, up to 125Msps, and 14bit resolution (for pulse height steps to be in sub mV increments). The bandwidth is sufficient to allow ~10ns rise times to be delivered over several feet of 50ohm triax cable terminated with 50ohms. Two independent outputs are provided to drive both HCFB modules. The pulser signal phase can be placed with a deterministic phase relationship to the BCO clock that drives the FSSR2.

### 3.5 FSSR2 Data Path

The FSSR2 chips provide event data through up to 6 LVDS pairs with a source synchronous clock. The VSCM supports receiving data from all 6 LVDS pairs from each FSSR2. The VSCM is fully capable of receiving DDR data at up to 70MHz (840Mbps per FSSR2). The LVDS signals are done using discrete transceivers with much better common mode range and ESD performance than any FPGA LVDS I/O.

### 3.6 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 | 125MHz 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 | LVTTL |
| 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 link 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. Upon receipt of a trigger signal, the VSCM will parse the front-end data and extract physics event data that matches a time-window related to the received Trig1/Trig2 time.

**Sync**
The sync signal is used to align/start board timers at the same time as other boards in the crate and system. This signal will also be used to reset the FSSR2 chips to ensure the BCO clock timer are synchronized to the VSCM global timer.

**Busy**
Busy is normal held low, but if the VSCM 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 to not overflow. If buffer overflows happen event synchronization from this module to another is lost.

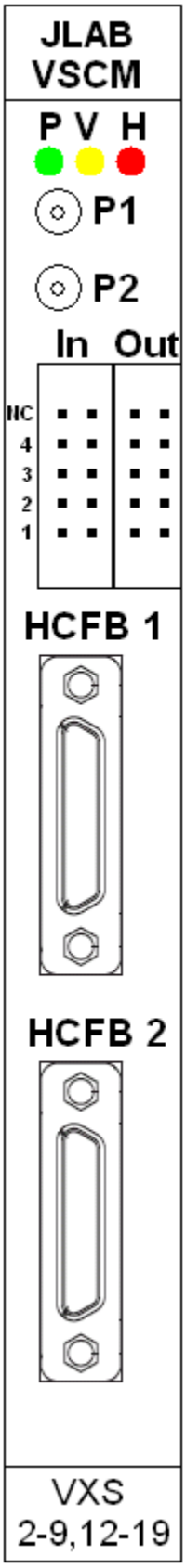
**Token In/Token Out**
These are used by the VME interface when performing 2eSST transfers with token passing.

**SDLink**
Currently a undefined serial link to the Signal Distribution board.

**L1 Trigger**
This is a high speed serial link to the CLAS12 Crate Trigger Processor (CLAS12 CTP) used to send processed front-end data to the Global Trigger System that can be correlated with other modules/detectors to generate a L1 trigger accept. Currently the SVT is not planned to be part of the L1 trigger, but the VSCM supports this feature if it is desired in the future. Potentially some level of track reconstruction on the SVT could be done in hardware to provide a high level trigger that would benefit experiments. The large 8μs trigger decision time in CLAS12 allows the relatively long latency of FSSR2 data to make it into the L1 trigger decision.

The CLAS12 CTP would be capable of receiving all the FSSR2 hits from an entire crate (roughly 4Gbps bandwidth is need for each VSCM). The CTP could then identify hits that have corresponding hits on the stereo axis and ship this information upstream through a ~20Gbps fiber optic link to the Global Trigger Crate where all regions of the SVT are consolidated in a single model where hits in the different regions can be linked together to identify tracks. The SSP is a potential board in the Global Trigger Crate that could be used for the consolidation of data, but depending on required performance and resources the SSP may or may not be a suitable candidate. Further research will be required in the future to determine the needs at the Global Trigger Crate if a need for using the SVT in the L1 trigger arises; however, the VSCM implementation allows all information to flow to the following trigger stages and therefore sets no bottleneck.

## 4. Specifications




MECHANICAL
- Single width VITA 41 Payload Module

HIGH SPEED SERIAL P0 INPUTS/OUTPUTS:
- 125MHz LVPECL Clock
- Trig 1, Trig 2, Sync Inputs
- 4x 2.5-3.125Gbps Lanes to CTP

Front Panel INPUTS/OUTPUTS:
- 2x HCFB Microdot Connectors
  - 6x LVDS 70MHz DDR lines per FSSR2
  - Serial Configuration
  - 3-10MHz programmable BCO clock
  - 0-70Mhz programmable RCLK
- 2x Triax DAC Pulser Calibration Outputs
- 4x LVDS Outputs
- 4x AnyLevel Differential Inputs (LVPECL, ECL, LVDS)

INDICATORS: (Front Panel)
- Power OK – Red LED
- VME DTACK – Green LED
- Status – Yellow LED

EVENT BUILDER:
- 2MB Event FIFO
- High trigger rate capable >>10kHz
- 16µs maximum trigger latency

DIAGNOSTICS:
- Per channel scalers (1,024)
- FSSR2 data integrity monitor
- FSSR2 chip status
- Event builder status

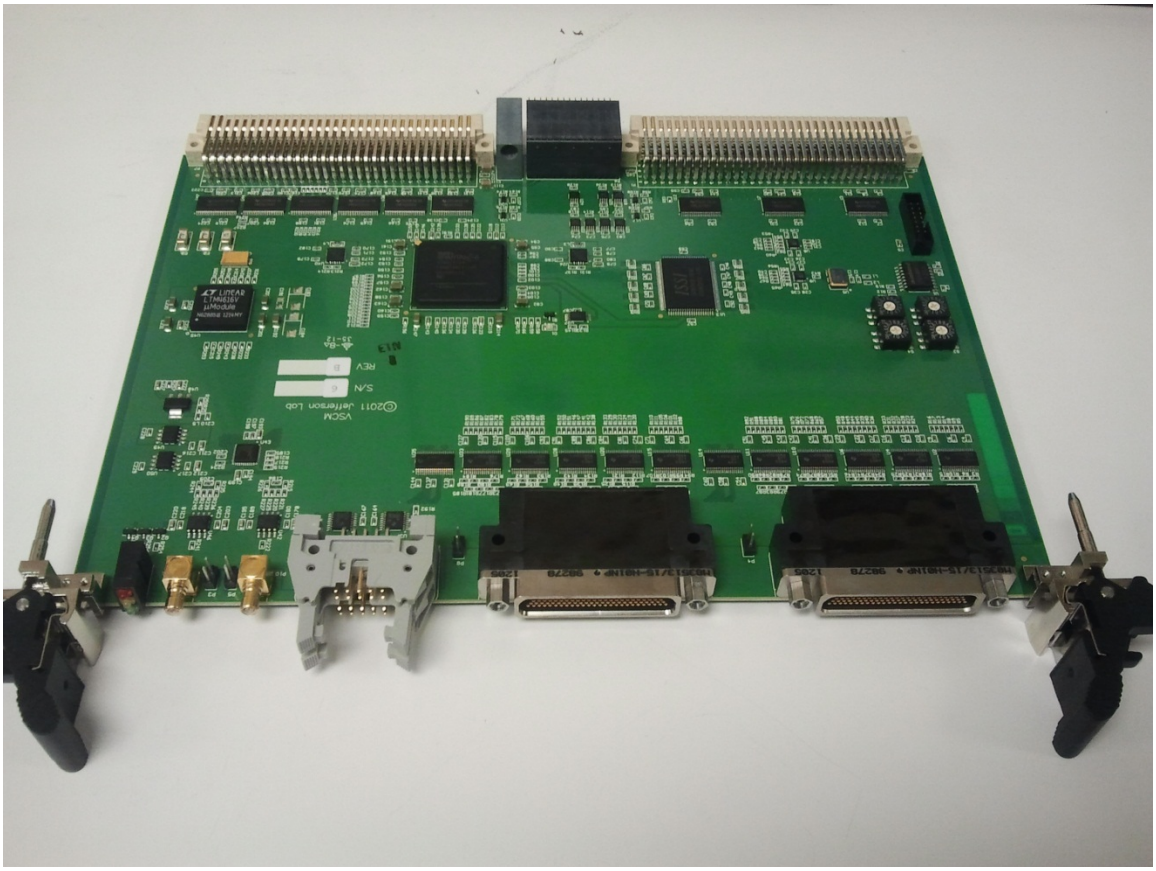
PROGRAMMING:
- On board JTAG Port, VME

POWER REQUIREMENTS:
- +5.0v @ 4A Typ

ENVIRONMENT:
- Commercial grade components (70 Celsius)

## 5. PCB Assembly View

The VSCM PCB is a 12 layer FR-370HR design. It is mostly digital, but does contain an analog section for the DAC pulser and jitter/noise sensitive differential pairs used for clocks and Giga-bit transceiver channels.



## 6. VSCM Readout Data Format

The VSCM 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 | BCO Start/Stop Time |
| 5 | Reserved |
| 6 | Reserved |
| 7 | Reserved |
| 8 | FSSR2 Strip Hit |
| 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)

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

**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: BCO Start/Stop Time**

Type: 0x4
Size: 1 word
Description: Indicates the trigger time window used to extract FSSR2 hits in terms of the BCO clocks.

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

**BCOSTART_TIME:**
BCO number for the beginning of the capture window (which includes this number)
**BCOSTOP_TIME:**
BCO number for the end of the capture window (which DOES NOT include this number)

**Data Type: FSSR2 Strip Hit**

Type: 0x8
Size: 1 word
Description: FSSR2 strip hit words. A separate word is used for each strip hit that is inside the trigger window.

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

**ADC:**
3bit ADC value for the strip hit determined by the FSSR2

**BCONUM:**
BCO number the hit occurred. This can be used with the TRIGGER_TIME and/or BCOSTART_TIME to know where the hit occurred relative to the trigger window.

**STRIPID:**
0-127 value indicating which strip was hit on the FSSR2 chip.

**CHIPID:**
Chip ID as reported by the FSSR2 event.

**HFCB:**
'0' – HFCB1 interface reported the hit
'1' – HFCB2 interface reported the hit

**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 VSCM 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 |
|---|---|---|
| **Board Information** | | |
| **A_FIRMWARE_REV** | Firmware revision | 0x0000 |
| **A_BOARDID** | Board identification | 0x0004 |
| **Debug/upgrade Configuration** | | |
| **A_ICAP** | FPGA configuration interface | 0x0008 |
| **A_SPI_FLASH** | Non-volatile flash interface | 0x0014 |
| **A_SRAM_DBG_ADR** | Sram test address | 0x0060 |
| **A_SRAM_DBG_DATA** | Sram test data | 0x0064 |
| **Readout Configuration** | | |
| **A_TRIG_WINDOW** | Trigger readout window | 0x0148 |
| **A_CLOCK_CFG** | Clock setup | 0x0030 |
| **A_TRIG_LATENCY** | Trigger processing latency | 0x0038 |
| **A_TRIG_BUSY_THR** | Busy threshold | 0x003C |
| **A_ADR32M** | Multi-board readout address | 0x0034 |
| **A_AD32** | Readout address | 0x0044 |
| **A_INTERRUPT** | Interrupt setup | 0x0048 |
| **A_INTERRUPT_ACK** | Interrupt acknowledge | 0x004C |
| **A_GEO** | Geographical address | 0x0050 |
| **A_RESET** | Soft reset | 0x0068 |
| **A_EVT_FIFO_STATUS** | Event FIFO status | 0x007C |
| **A_FIFO_WORD_CNT** | Event FIFO word count | 0x0054 |
| **A_FIFO_EVENT_CNT** | Event FIFO event count | 0x0058 |
| **A_READOUT_CFG** | Readout interrupt setup | 0x005C |
| **A_BLOCK_CFG** | Event blocking setup | 0x0028 |
| **I/O Configuration** | | |
| **A_TRIG** | Trigger setup | 0x0100 |
| **A_SYNC** | Sync setup | 0x0104 |
| **A_SWAGPIO** | VXS-SWA GPIO setup | 0x0108 |
| **A_SWBGPIO** | VXS-SWB GPIO setup | 0x010C |
| **A_FPOUTPUT_0** | Front panel output 0 setup | 0x0138 |
| **A_FPOUTPUT_1** | Front panel output 1 setup | 0x013C |
| **A_FPOUTPUT_2** | Front panel output 2 setup | 0x0140 |
| **A_FPOUTPUT_3** | Front panel output 3 setup | 0x0144 |
| **A_TOKENIN_CFG** | Readout token input setup | 0x0018 |
| **A_TOKENOUT_CFG** | Readout token output setup | 0x001C |
| **A_SDLINK_CFG** | VXS-SDLINK setup | 0x0020 |
| **A_TRIGOUT_CFG** | VXS-TRIGOUT setup | 0x0024 |

| Pulser Configuration | | |
|---|---|---|
| **A_PULSER_PERIOD** | Pulser period | 0x0120 |
| **A_PULSER_LOW** | Pulser high cycle time | 0x0124 |
| **A_PULSER_NPULSES** | Pulser pulse count | 0x0150 |
| **A_PULSER_START** | Pulser start | 0x0154 |
| **A_PULSER_STATUS** | Pulser status | 0x0158 |

| FSSR Configuration | | |
|---|---|---|
| **A_FSSR_CLK_CFG** | FSSR BCO clock period | 0x006C |
| **A_FSSR_MASK** | FSSR got hit mask | 0x00AC |
| **A_FSSR_ADDR_H1** | FSSR chip address HFCB 1 | 0x00A4 |
| **A_FSSR_ADDR_H2** | FSSR chip address HFCB 2 | 0x00A8 |
| **A_MCLK_STATUS** | MCLK PLL status | 0x0070 |
| **A_MCLK_CFG** | MCLK PLL interface | 0x0074 |

| FSSR Serial Interface | | |
|---|---|---|
| **A_FSSR_SER_CFG** | FSSR serial configuration | 0x0080 |
| **A_FSSR_SER_CLK** | FSSR serial clock sync | 0x00A0 |
| **A_FSSR_SER_DATA0** | FSSR serial data | 0x0084 |
| **A_FSSR_SER_DATA1** | FSSR serial data | 0x0088 |
| **A_FSSR_SER_DATA2** | FSSR serial data | 0x008C |
| **A_FSSR_SER_DATA3** | FSSR serial data | 0x0090 |

| DAC Pulser | | |
|---|---|---|
| **A_DAC_TRIG** | DAC trigger setup | 0x014C |
| **A_DAC_CFG** | DAC setup | 0x0094 |
| **A_DAC_CH0** | DAC data ch0 | 0x0098 |
| **A_DAC_CH1** | DAC data ch1 | 0x009C |

| Scalers | | |
|---|---|---|
| **A_SCALER_LATCH** | Latch scalers | 0x0078 |
| **A_SCALER_FP_OUTPUT0** | Front panel output 0 scaler | 0x0FC8 |
| **A_SCALER_FP_OUTPUT1** | Front panel output 1 scaler | 0x0FCC |
| **A_SCALER_FP_OUTPUT2** | Front panel output 2 scaler | 0x0FD0 |
| **A_SCALER_FP_OUTPUT3** | Front panel output 3 scaler | 0x0FD4 |
| **A_SCALER_FP_INPUT0** | Front panel input 0 scaler | 0x0FD8 |
| **A_SCALER_FP_INPUT1** | Front panel input 1 scaler | 0x0FDC |
| **A_SCALER_FP_INPUT2** | Front panel input 2 scaler | 0x0FE0 |
| **A_SCALER_FP_INPUT3** | Front panel input 3 scaler | 0x0FE4 |
| **A_SCALER_BUSY** | Busy scaler | 0x0FE8 |
| **A_SCALER_BUSY_CYCLES** | Busy time scaler | 0x0FEC |
| **A_SCALER_VMECLK** | VME clock scaler | 0x0FF0 |
| **A_SCALER_SYNC** | VXS-SYNC input scaler | 0x0FF4 |
| **A_SCALER_TRIG1** | VXS-TRIG1 input scaler | 0x0FF8 |
| **A_SCALER_TRIG2** | VXS-TRIG2 input scaler | 0x0FFC |

| FSSR Chip Scalers (x: 1-2, y: 1-4) | | |
|---|---|---|
| **A_FSSR_LAST_STAT_HxUy** | FSSR last status word | 0x1000+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_STAT_CNT_HxUy** | FSSR status word scaler | 0x1004+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_EVENT_CNT_HxUy** | FSSR event word scaler | 0x1008+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_WORD_CNT_HxUy** | FSSR word scaler | 0x100C+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_IDLE_CNT_HxUy** | FSSR idle scaler | 0x1010+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_AQBCO_CNT_HxUy** | FSSR AQBCO scaler | 0x1014+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_MARKERR_CNT_HxUy** | FSSR mark error scaler | 0x1018+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_ENCERR_CNT_HxUy** | FSSR encoding error scaler | 0x101C+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_CHPIDERR_CNT_HxUy** | FSSR chip ID error scaler | 0x1020+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_HIST_CFG_HxUy** | FSSR histogram setup | 0x1024+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_GOTHIT_CNT_HxUy** | FSSR gothit scaler | 0x1028+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_HIST_CNT_HxUy** | FSSR histogram data | 0x102C+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_REF_CNT_HxUy** | FSSR reference scaler | 0x1030+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_HIST_TIME_HxUy** | FSSR histogram time | 0x1034+0x400*(x-1)+0x100*(y-1) |
| **A_FSSR_CORETALK_CNT_HxUy** | FSSR coretalking scaler | 0x1038+0x400*(x-1)+0x100*(y-1) |

## 7.1 Board Information Registers Section

Basic board information registers can be used to verify that this board is the VSCM and check for the software revision, which should be checked for compatibility.

**Register: A_FIRMWARE_REV**

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

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

**FIRMWARE_REV_MAJOR (RO):**
Major firmware revision number
**FIRMWARE_REV_MINOR (RO):**
Minor firmware revision number

**Register: A_BOARDID**

Address Offset: 0x0004
Size: 32bits
Reset State: 0x5653434D

| 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):**
0x5653434D = “VSCM” in ASCII

## 7.2 Debug/upgrade Configuration Registers Section

The registers in this section are not intended to be used by a typical user of this board. These provide access to the configuration flash memory and the direct FPGA configuration. The SRAM registers provide a simple interface to test the external memory, which bypasses the FIFO logic the event builder uses when accessing this memory.

**Register: A_ICAP**

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

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

**DATA (R/W):**
SPARTAN6_ICAP data port
**WRITE (WO):**
SPARTAN6_ICAP write port
**CLK (WO):**
SPARTAN6_ICAP clk port
**CE (WO):**
SPARTAN6_ICAP ce port
**BUSY (RO):**
SPARTAN6_ICAP busy port
**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: A_SPI_FLASH**

Address Offset: 0x0014
Size: 32bits
Reset State: 0x00000007

| 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 |
| - | - | - | - | MISO | NCS | CLK | MOSI |

**MOSI, CLK, NCS (WO):**
SPI interface outputs to flash memory
**MISO (RO):**
SPI interface input from flash memory
**Notes:**

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

**Register: A_SRAM_DBG_ADR**

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

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

**ADDR (WO):**

Address to use for read/write when A_SRAM_DBG_DATA is accessed

**Notes:**

1) This interface is used for testing read/write accesses to the external SRAM, which is normally controlled by the event builder FIFO.

**Register: A_SRAM_DBG_DATA**

Address Offset: 0x0064
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/W):**

When read (written), this reads (writes) data from (to) the external SRAM at the address specified in A_SRAM_DBG_ADR

**Notes:**

1) This interface is used for testing read/write accesses to the external SRAM, which is normally controlled by the event builder FIFO.

## 7.3 Readout Configuration Registers Section

**Register: A_TRIG_WINDOW**

Address Offset: 0x0148
Size: 32bits
Reset State: 0x00000000

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

**WINDOW_START_I (R/W):**

When the trigger system SYNC is released, this register initializes the trigger logic BCO start capture counter (integer start BCO counter).

**WINDOW_START_R (R/W):**

When the trigger system SYNC is released, this register initializes the trigger logic BCO start capture counter (remainder start BCO counter).

**WINDOW_STOP_I (R/W):**

When the trigger system SYNC is released, this register initializes the trigger logic BCO stop capture counter (integer stop BCO counter).

**WINDOW_STOP_R (R/W):**

When the trigger system SYNC is released, this register initializes the trigger logic BCO stop capture counter (remainder stop BCO counter).

**Notes:**

1) These counters together define the effective capture window width and lookback time when a trigger is received. These also relate the global trigger system time to the slower BCO clock counters of the FSSR2.

**Register: A_CLOCK_CFG**

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 |
| - | - | - | - | - | - | CLKRST | CLKSEL |

**CLKSEL (R/W):**

'0' – selects onboard local 125MHz oscillator source for triggering/FSSR reference
'1' – selects VXS-SWB 125MHz oscillator source for triggering/FSSR reference

**CLKRST (R/W):**

'0' – clock system reset cleared
'1' – clock system reset set

**Notes:**

1) When changing CLKSEL, the CLKRST must be set and cleared to ensure.
2) A soft reset must be applied after changing clock sources. See A_RESET register.

**Register: A_TRIG_LATENCY**

Address Offset: 0x0038
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 |
| - | - | - | - | - | - | LATENCY | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| LATENCY | | | | | | | |

**LATENCY (R/W):**

When a trigger is received it is timestamped. The LATENCY register is the minimum time elapsed from receipt of trigger to when it is processed by the event builder (i.e. TRIGGER_TIMESTAMP + LATENCY > CURRENT_TIME). These units are in 8ns ticks, proving a latency of up to 8us.

**Notes:**

1) The use of this register is required when triggering readout data where the latency from the FSSR2 outputs cause data to show up after a trigger is received. This is a function of the trigger data window start/stop times and FSSR2 readout bandwidth.

**Register: A_TRIG_BUSY_THR**

Address Offset: 0x003C
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 |
| - | - | - | - | - | - | - | - |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BUSY_THRESHOLD | | | | | | | |

**BUSY_THRESHOLD (R/W):**

Defines the number of outstanding triggers the event builder has to process before a BUSY status is asserted. This can be used to inhibit triggers at the global trigger distribution.

**Register: A_ADR32M**

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

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

**ADR_MIN (R/W):**
Low A32 address (bits 31:23) used for multi-board, token-passing readout

**ADR_MAX (R/W):**
High A32 address (bits 31:23) used for multi-board, token-passing readout

**EN (R/W):**
'0' – disables VME A32 multi-board addressing mode
'1' – enabled VME A32 multi-board addressing mode

**FIRST (R/W):**
'0' – not first board in token passing scheme
'1' – first board in token passing scheme

**LAST (R/W):**
'0' – not last board in token passing scheme
'1' – last board in token passing scheme

**TAKE (R/W):**
'0' – does nothing
'1' – gives board token

**STATUS (R/W):**
'0' – board does not have token
'1' – board does have token

**Register: A_ADR32**

Address Offset: 0x0044
Size: 32bits
Reset State: 0xXXXX8000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| - | - | - | - | - | - | - | - |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| A32_BASE | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| A32_BASE | - | - | - | - | - | - | A32_EN |

**A32_BASE (R/W):**
A32 base address (bits 31:23)

**A32_EN (R/W):**
'0' – disables VME A32 addressing mode
'1' – enabled VME A32 addressing mode

**Register: A_INTERRUPT**

Address Offset: 0x0048
Size: 32bits
Reset State: 0x00000000

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

**INT_ID (R/W):**
VME bus interrupt ID
**INT_LEVEL (R/W):**
VME bus interrupt level
**INT_EN (R/W):**
VME bus interrupt enable

**Register: A_INTERRUPT_ACK**

Address Offset: 0x004C
Size: 32bits
Reset State: 0xXXXXXXXX

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

**INTERRUPT_ACK (WO):**
Writing to this register will acknowledge any outstanding interrupt. This will allow further interrupt from this module to interrupt on the VME bus if any interrupting condition persists or occurs in the future.

**Register: A_GEO**

Address Offset: 0x0050
Size: 32bits
Reset State: 0xXXXXXXXX

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

**VME_ADDR (RO):**

VME address switch settings. The lower 8 bits form the A24 base address. The upper 8 bits are not used in the firmware, but are available to the user for any purpose desired (for example, the user could read this and set the A32_BASE to this value to use dip switch controlled A32 VME addressing).

**SLOTID (RO):**

VME geographical addressing slot. Slot 30 will be reported on parity error.

**Notes:**

1) Geographical addressing is only support when module is used on aVME64X compatibly crate. A parity error will be generated on non-VME64X compatible crates.

**Register: A_RESET**

Address Offset: 0x0068
Size: 32bits
Reset State: 0xXXXXXXXX

| 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 |
| - | - | - | - | - | - | - | RESET |

**RESET (WO):**

'0' – clear "soft" reset
'1' – set "soft" reset. This will clear all event builder FIFOs. To ensure reliable reset, no triggers should be delivered during the reset period.

**Register: A_EVT_FIFO_STATUS**

Address Offset: 0x007C
Size: 32bits
Reset State: 0xXXXXXXXX

| 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 |
| ERR7 | ERR6 | ERR5 | ERR4 | ERR3 | ERR2 | ERR1 | ERR0 |

**ERRx (RO):**

'0' – no FIFO overrun on FSSR2 chip x trigger processing
'1' – FIFO overrun seen on FSSR2 chip x trigger processing

**Notes:**

1) A soft reset, writing to A_RESET, or hardware reset must be applied to clear this condition.
2) An error here indicates data is dropping due to very high occupancies, large trigger windows, very high trigger rates, and/or too slow bandwidth on VME bus.

**Register: A_FIFO_WORD_CNT**

Address Offset: 0x0054
Size: 32bits
Reset State: 0xXXXXXXXX

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

**WORD_CNT (RO):**

The number of 32bit event builder words currently residing in readout FIFO.

**Register: A_FIFO_EVENT_CNT**

Address Offset: 0x0058
Size: 32bits
Reset State: 0xXXXXXXXX

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

**EVENT_CNT (RO):**

The number of event builder events currently residing in readout FIFO.

**Register: A_READOUT_CFG**

Address Offset: 0x005C
Size: 32bits
Reset State: 0x00000000

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

**EVT_WORD_INT_LEVEL (R/W):**

Range: 0 to 65535. Sets the 32bit word interrupt threshold for the event builder. If the number of 32bit event words inside the event builder FIFO is greater-than or equal to this value an interrupt will be generated if enabled by the A_INTERRUPT register.

**EVT_NUM_INT_LEVEL (R/W):**

Range: 0 to 32767. Sets the event count interrupt threshold for the event builder. If the number of events inside the event builder FIFO is greater-than or equal to this value an interrupt will be generated if enabled by the A_INTERRUPT register.

**BERREN (R/W):**

'0' – disable VME bus error assertion for end-of-event signaling (user must know event size or parse readout contents to ensure event synchronization/alignment)
'1' – enables VME bus error assertion for end-of-event signaling

**Register: A_BLOCK_CFG**

Address Offset: 0x0028
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 |
| - | - | - | - | - | BLOCK_SIZE | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BLOCK_SIZE | | | | | | | |

**BLOCK_SIZE (R/W):**

Number of events per block to be built by the event builder.

## 7.4 I/O Configuration Registers Section

This section covers the registers control how I/O signals interact with the firmware. Most of the signals support multiplexing to provide flexibility to aid in various configurations or debug. The multiplexor inputs are all the same for all signal that have support for multiplexed inputs and is defined as follows:

| Multiplexor Input | Signal | Description |
|---|---|---|
| 0 | 0 | Constant '0' |
| 1 | 1 | Constant '1' |
| 2 | PULSER_OUTPUT | Onboard pulser output |
| 3 | FP_INPUT0 | Front panel LVDS input 0 |
| 4 | FP_INPUT1 | Front panel LVDS input 1 |
| 5 | FP_INPUT2 | Front panel LVDS input 2 |
| 6 | FP_INPUT3 | Front panel LVDS input 3 |
| 7 | VXS_SWB_SYNC | VXS Switch B Sync input |
| 8 | VXS_SWB_TRIG1 | VXS Switch B Trig1 input |
| 9 | VXS_SWB_TRIG2 | VXS Switch B Trig2 input |
| 10 | VXS_SWA_GPIO0 | VXS Switch A GPIO Input 0 |
| 11 | VXS_SWA_GPIO1 | VXS Switch A GPIO Input 1 |
| 12 | VXS_SWB_GPIO0 | VXS Switch B GPIO Input 0 |
| 13 | VXS_SWB_GPIO1 | VXS Switch B GPIO Input 1 |
| 14 | BUSY | Internal busy |
| 15 | FSSR_GOTHIT | FSSR GotHit pin status OR'd across enabled chips |
| 16 | DAC_TRIGGER | DAC Trigger (no delay) |
| 17 | DAC_TRIGGER_DELAYED | DAC Trigger (with delay) |
| 18 | BCOCLK | BCOCLK |
| 19 | TOKENOUT | VME Token Out |
| 20 | TOKENIN | VXS Switch B Token input |
| 31-21 | Reserved | Reserved |

**Register: A_TRIG**

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 |
| - | - | - | MUX_SRC | | | | |

**MUX_SRC (R/W):**

Selects trigger input source: see MUX_SRC description at the beginning of this section for source signal description.

**Register: A_SYNC**

Address Offset: 0x0104
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 |
| - | - | - | MUX_SRC | | | | |

**MUX_SRC (R/W):**

Selects sync input source: see MUX_SRC description at the beginning of this section for source signal description.

**Register: A_SWAGPIO**

Address Offset: 0x0108
Size: 32bits
Reset State: 0x01000100

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | DIR1 |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| - | - | - | MUX_SRC1 | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| - | - | - | - | - | - | - | DIR0 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | - | - | MUX_SRC0 | | | | |

**MUX_SRCx (R/W):**

Selects VXS-Switch-A-SEx driver source: see MUX_SRC description at the beginning of this section for source signal description.

**DIRx (R/W):**

'0' – Set as output (drives VXS-Switch-A-SEx line)
'1' – Set as input (weak pull-up on VXS-Switch-A-SEx line)

**Notes:**

1) Default values should be left unchanged.

**Register: A_SWBGPIO**

Address Offset: 0x010C
Size: 32bits
Reset State: 0x01000100

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | DIR1 |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| - | - | - | MUX_SRC1 | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| - | - | - | - | - | - | - | DIR0 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | - | - | MUX_SRC0 | | | | |

**MUX_SRCx (R/W):**

Selects VXS-Switch-B-SEx driver source: see MUX_SRC description at the beginning of this section for source signal description.

**DIRx (R/W):**

'0' – Set as output (drives VXS-Switch-B-SEx line)
'1' – Set as input (weak pull-up on VXS-Switch-B-SEx line)

**Notes:**

1) Default values should be left unchanged.

**Register: A_FPOUTPUT_0 -> A_FPOUTPUT_3**

Address Offset: 0x0138, 0x013C, 0x0140, 0x0144
Size: 32bits
Reset State: 0x00000000

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

**MUX_SRC (R/W):**

Selects LVDS output driver source: see MUX_SRC description at the beginning of this section for source signal description.

**DELAY (R/W):**

0-8191: Delayed output in 8ns ticks (0 to 65,528ns range)

**Register: A_TOKENIN_CFG**

Address Offset: 0x0018
Size: 32bits
Reset State: 0x00000014

| 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 |
| - | - | - | MUX_SRC | | | | |

**MUX_SRCx (R/W):**

Selects readout token input source: see MUX_SRC description at the beginning of this section for source signal description.

**Notes:**

1) Default values should be left unchanged.

**Register: A_TOKENOUT_CFG**

Address Offset: 0x001C
Size: 32bits
Reset State: 0x00000013

| 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 |
| - | - | - | MUX_SRC | | | | |

**MUX_SRCx (R/W):**

Selects readout token input source: see MUX_SRC description at the beginning of this section for source signal description.

**Notes:**

1) Default values should be left unchanged.

**Register: A_SDLINK_CFG**

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 |
| - | - | - | - | - | - | - | - |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | - | - | MUX_SRC | | | | |

**MUX_SRCx (R/W):**

Selects readout token input source: see MUX_SRC description at the beginning of this section for source signal description.

**Notes:**

1) Default values should be left unchanged.

**Register: A_TRIGOUT_CFG**

Address Offset: 0x0024
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 |
| - | - | - | MUX_SRC | | | | |

**MUX_SRCx (R/W):**

Selects readout token input source: see MUX_SRC description at the beginning of this section for source signal description.

**Notes:**

1) Default values should be left unchanged.

## 7.5 Pulser Configuration Registers Section

**Register: A_PULSER_PERIOD**

Address Offset: 0x0120
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 8ns ticks for the pulser period

**Register: A_PULSER_LOW**

Address Offset: 0x0124
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 8ns ticks of the pulser period the output stays low.

**Register: A_PULSER_NPULSES**

Address Offset: 0x0150
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 0xFFFFFFFE: 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 A_PULSER_START register

**Register: A_PULSER_START**

Address Offset: 0x0154
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: A_PULSER_STATUS**

Address Offset: 0x0158
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 A_PULSER_NPULSES
'1' – pulser is is not active (either disabled or has finished fixed pulse count)

## 7.6 FSSR Configuration Registers Section

**Register: A_FSSR_CLK_CFG**

Address Offset: 0x006C
Size: 32bits
Reset State: 0x00000010

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| SER_MODE7 | | SER_MODE6 | | SER_MODE5 | | SER_MODE4 | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| SER_MODE3 | | SER_MODE2 | | SER_MODE1 | | SER_MODE0 | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| - | - | - | - | - | - | - | - |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BCOCLK_PERIOD | | | | | | | |

**BCOCLK_PERIOD (R/W):**

0x02 to 0xFE: BCO clock period defined in 8ns ticks. Must be an even number.

**SER_MODEx (R/W):**

"00" – 6bit readout mode
"01" – 4bit readout mode
"10" – 2bit readout mode
"11" – 1bit readout mode

**Notes:**

1) BCO clock periods should range between 32 (3.90625MHz) and 16 (7.8125MHz)
2) The slow control serial interface between the FSSR2 and VSCM is designed to work up to 12.5MHz (BCO clock period of 10 ticks). BCO clock frequencies beyond 8MHz on the FSSR2 have not been part of the chip specifications and so impacts on timing reliability for the FSSR2 are unknown by the VSCM developer.
3) SER_MODE0 corresponds to HFCB1-U1
   SER_MODE1 corresponds to HFCB1-U2
   …
   SER_MODE4 corresponds to HFCB2-U1
   …

**Register: A_FSSR_MASK**

Address Offset: 0x00AC
Size: 32bits
Reset State: 0x000000FF

| 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 |
| GOTHIT7 | GOTHIT6 | GOTHIT5 | GOTHIT4 | GOTHIT3 | GOTHIT2 | GOTHIT1 | GOTHIT0 |

**GOTHITx (R/W):**

'0' – disables GOTHIT signal from OR that feed the MUX input "FSSR_GOTHIT"
'1' – enables GOTHIT signal from OR that feed the MUX input "FSSR_GOTHIT"

**Notes:**

1) GOTHIT0 corresponds to HFCB1-U1
   GOTHIT1 corresponds to HFCB1-U2
   …
   GOTHIT4 corresponds to HFCB2-U1

…

**Register: A_FSSR_ADDR_H1**

Address Offset: 0x00A4
Size: 32bits
Reset State: 0x0C0B0A09

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| - | - | - | CHIP_ADDR_U4 | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| - | - | - | CHIP_ADDR_U3 | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| - | - | - | CHIP_ADDR_U2 | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | - | - | CHIP_ADDR_U1 | | | | |

**CHIP_ADDR_Ux (R/W):**

Defines 5bit chip address used in slow control serial interface on HFCB 1 connector. This chip ID is also used for error checking of the event words reported by the FSSR2.

**Register: A_FSSR_ADDR_H2**

Address Offset: 0x00A8
Size: 32bits
Reset State: 0x0C0B0A09

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| - | - | - | CHIP_ADDR_U4 | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| - | - | - | CHIP_ADDR_U3 | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| - | - | - | CHIP_ADDR_U2 | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | - | - | CHIP_ADDR_U1 | | | | |

**CHIP_ADDR_Ux (R/W):**

Defines 5bit chip address used in slow control serial interface on HFCB 2 connector. This chip ID is also used for error checking of the event words reported by the FSSR2.

**Register: A_MCLK_STATUS**

Address Offset: 0x0070
Size: 32bits
Reset State: 0xXXXXXXXX

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

**DATA (RO):**
MCLK PLL configuration data output
**RDY (RO):**
'0' - MCLK PLL configuration not ready
'1' - MCLK PLL configuration ready
**LOCKED (RO):**
'0' - MCLK PLL is not locked
'1' - MCLK PLL is locked

**Register: A_MCLK_CFG**

Address Offset: 0x0074
Size: 32bits
Reset State: 0x00000000

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

**DATA (R/W):**
MCLK PLL configuration data input
**ADDR (R/W):**
MCLK PLL configuration address input
**DEN (R/W):**
'0' - MCLK PLL configuration data not valid
'1' - MCLK PLL configuration data valid
**DWE (R/W):**
'0' - MCLK PLL configuration data read
'1' - MCLK PLL configuration data write
**RST (R/W):**
'0' - MCLK PLL not reset
'1' - MCLK PLL reset

## 7.7 FSSR Serial Interface Registers Section

**Register: A_FSSR_SER_CFG**

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

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

**REG_ADDR (R/W):**
FSSR2 register address

**CMD (R/W):**
FSSR2 register command:
"001" - write, "010" - set, "100" - read, "101" - reset, "110" - default

**RDY (RO):**
'0' – shift register transfer is still active
'1' – shift register transfer is idle/complete

**RUN (RO):**
'0' – does nothing
'1' – starts shift register transfer (auto-clearing bit)

**NBITS (R/W):**
Number of bits to transfer (doesn't include chip, register, or command bit length)

**CHIP_SEL (R/W):**
"0000" – Select HFCB1, U1
"0001" – Select HFCB1, U2
"0010" – Select HFCB1, U3
"0011" – Select HFCB1, U4
"0100" – Select HFCB2, U1
"0101" – Select HFCB2, U2
"0110" – Select HFCB2, U3
"0111" – Select HFCB2, U4
"1000" – Select HFCB1 U1-U4, HFCB2 U1-U4 (Wild Chip ID used)
"1001" – Select HFCB1, U1-U4 (Wild Chip ID used)
"1010" – Select HFCB2, U1-U4 (Wild Chip ID used)

**RESET (R/W):**
'0' - FSSR2 MasterReset not asserted
'1' - FSSR2 MasterReset asserted

**Register: A_FSSR_SER_CLK**

Address Offset: 0x00A0
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 |
| - | - | - | - | - | - | - | SYNC |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BCONUM | | | | | | | |

**BCONUM (R/W):**

FSSR2 BCONUM to issue command on

**SYNC (R/W):**

'0' – do not synchronize issued command with a particular BCONUM
'1' – synchronize issued command with BCONUM

**Notes:**

1) This register is used to deliver commands exactly on specified BCO clock number. This allows BCO clock synchronization checks across chips and provides the mechanism for synchronizing BCO counters across FSSR2 chips even if commands are issues to chips one at a time.

**Register: A_FSSR_SER_DATA0 -> A_FSSR_SER_DATA3**

Address Offset: 0x0084, 0x0088, 0x008C, 0x0090
Size: 32bits
Reset State: 0x00000000

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

**DATAx (R/W):**

FSSR2 shift register data input/output.

**Notes:**

1) Writing this register sets the initial contents to load into the FSSR2 serial shift register transmitter. Reading this register provides the received shift register data from the FSSR2 at the end of a read.
2) Four 32bit registers are used to support the maximum shift register data transfer of 128bits.

## 7.8 DAC Pulser Registers Section

**Register: A_DAC_TRIG**

Address Offset: 0x0014C
Size: 32bits
Reset State: 0x00000000

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

**MUX_SRCx (R/W):**
Selects DAC waveform trigger input source: see MUX_SRC description at the beginning of this section for source signal description.

**BCO_NUM (R/W):**
When BCOEN=1, the DAC pulser will synchronize to that BCO number.

**DELAY (R/W):**
Amount of delay after the DAC is triggered, in 8ns ticks, that the waveform output begins. In SYNC=1, this delay happens after the trigger has been synchronized with the BCOCLK.

**SYNC (R/W):**
'0' – does not synchronize the DAC waveform with the BCOCLK
'1' – synchronizes the DAC waveform with the BCOCLK

**BCOEN (R/W):**
'0' – does not synchronize the DAC waveform to a specific BCO number
'1' – synchronizes DAC waveform using BCOCLK and BCO number (BCO_NUM)

**Register: A_DAC_CFG**

Address Offset: 0x00094
Size: 32bits
Reset State: 0x00000000

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

**DATA (R/W):**
8bit data for DAC serial configuration

**ADDR (WO):**
8bit DAC serial address for configuration

**IDLE_DAC_CODE (R/W):**
14bit DAC code used when no waveform is being output.

**START (WO):**
'0' – does nothing
'1' – starts serial transfer for DAC configuration (auto clearing bit)

**Register: A_DAC_CH0**

Address Offset: 0x00098
Size: 32bits
Reset State: 0x00000000

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

**DATA (WO):**
14bit DAC waveform data
**ADDR (WO):**
9bit DAC waveform address
**LEN (R/W):**
9bit DAC waveform length
**Notes:**
1) Writing to this register will load the sample a DAC waveform sample (DATA) into the waveform address location (ADDR). LEN should match the address of the last waveform sample.

**Register: A_DAC_CH1**

Address Offset: 0x0009C
Size: 32bits
Reset State: 0x00000000

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

**DATA (WO):**
14bit DAC waveform data
**ADDR (WO):**
9bit DAC waveform address
**LEN (R/W):**
9bit DAC waveform length
**Notes:**
1) Writing to this register will load the sample a DAC waveform sample (DATA) into the waveform address location (ADDR). LEN should match the address of the last waveform sample.

## 7.9 Scalers Registers Section

**Register: A_SCALER_LATCH**

Address Offset: 0x00078
Size: 32bits
Reset State: 0xXXXXXX00

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| GLOBAL | - | - | - | - | - | - | - |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| - | - | - | - | - | - | - | - |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| - | - | - | - | - | - | - | - |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| H2U4 | H2U3 | H2U2 | H2U1 | H1U4 | H1U3 | H1U2 | H1U1 |

**HxUy (R/W):**

'0' – disabled histogram bin filler for FSSR2 HxUy chip
'1' – enables histogram bin filler for FSSR2 HxUy chip
'1'->'0' – latches all scalers for FSSR2 HxUy chip

**GLOBAL (WO):**

'0' – does nothing
'1' – latches all scalers, except ones related to FSSR2 chip

**Notes:**

1) Scalers are all buffered and auto-cleared when latched for readout.
2) Histograms are currently not buffered and are only cleared when read. These must be disabled during readout.

**Register: A_SCALER_FP_OUTPUT0 -> A_SCALER_FP_OUTPUT3**

Address Offset: 0x0FC8, 0x0FCC, 0x0FD0, 0x0FD4
Size: 32bits
Reset State: 0xXXXXXXXX

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

**SCALER_FPOUTPUTx (RO):**

Number of rising edges seen on FPOUTPUTx since last scaler latch

**Register: A_SCALER_FP_INPUT0 -> A_SCALER_FP_INPUT3**

Address Offset: 0x0FD8, 0x0FDC, 0x0FE0, 0x0FE4
Size: 32bits
Reset State: 0xXXXXXXXX

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

**SCALER_FPOUTPUTx (RO):**

Number of rising edges seen on FPINPUTx since last scaler latch

**Register: A_SCALER_BUSY**

Address Offset: 0x0FE8
Size: 32bits
Reset State: 0xXXXXXXXX

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

**SCALER_BUSY (RO):**

Number of rising edges seen on internal BUSY since last scaler latch

**Register: A_SCALER_BUSY_CYCLES**

Address Offset: 0x0FEC
Size: 32bits
Reset State: 0xXXXXXXXX

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

**SCALER_BUSY_CYCLES (RO):**

Number of clock cycles (125MHz) internal BUSY was seen high since last scaler latch.

**Register: A_SCALER_VMECLK**

Address Offset: 0x0FF0
Size: 32bits
Reset State: 0xXXXXXXXX

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

**SCALER_VMECLK (RO):**

Number of VMECLK clock edges (50MHz) seen since last scaler latch.

**Notes:**

1) This scaler can be used to normalize other scalers latched by the GLOBAL.

**Register: A_SCALER_SYNC**

Address Offset: 0x0FF4
Size: 32bits
Reset State: 0xXXXXXXXX

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

**SCALER_SYNC (RO):**

Number of rising edges seen on the VXS-SWB-SYNC input since last scaler latch

**Register: A_SCALER_TRIG1**

Address Offset: 0x0FF8
Size: 32bits
Reset State: 0xXXXXXXXX

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| SCALER_TRIG1 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| SCALER_TRIG1 | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| SCALER_TRIG1 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SCALER_TRIG1 | | | | | | | |

**SCALER_TRIG1 (RO):**

Number of rising edges seen on the VXS-SWB-TRIG1 input since last scaler latch

**Register: A_SCALER_TRIG2**

Address Offset: 0x0FFC
Size: 32bits
Reset State: 0xXXXXXXXX

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| SCALER_TRIG2 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| SCALER_TRIG2 | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| SCALER_TRIG2 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SCALER_TRIG2 | | | | | | | |

**SCALER_TRIG2 (RO):**

Number of rising edges seen on the VXS-SWB-TRIG2 input since last scaler latch

**Register: A_SCALER_TRIGGER**

Address Offset: 0x0FC0
Size: 32bits
Reset State: 0xXXXXXXXX

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

**SCALER_TRIGGER (RO):**

Number of triggers seen at the event builder trigger input (counted whether event build accepts the trigger or not)

**Register: A_SCALER_TRIGGER_ACCEPTED**

Address Offset: 0x0FC4
Size: 32bits
Reset State: 0xXXXXXXXX

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

**SCALER_TRIGGER_ACCEPTED (RO):**

Number of seen and accepted at the event builder trigger input (difference between A_SCALER_TRIGGER and A_SCALER_TRIGGER_ACCEPTED gives the number of lost triggers due to the event builder being busy).

## 7.10 FSSR Chip Scalers Registers Section

**Register: A_FSSR_LAST_STAT_HxUy**

Address Offset: 0x1000, 0x1100,…,0x1700
Size: 32bits
Reset State: 0xXXXXXXXX

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

**LAST_STAT_HxUy (RO):**

Last FSSR2 status word seen on readout interface from FSSR2 HxUy chip since last latch of HxUy scalers.

**Register: A_FSSR_STAT_CNT_HxUy**

Address Offset: 0x1004, 0x1104,…,0x1704
Size: 32bits
Reset State: 0xXXXXXXXX

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

**STAT_CNT_HxUy (RO):**

Number of FSSR2 status word seen on readout interface from FSSR2 HxUy chip since last latch of HxUy scalers.

**Register: A_FSSR_EVENT_CNT_HxUy**

Address Offset: 0x1008, 0x1108,…,0x1708
Size: 32bits
Reset State: 0xXXXXXXXX

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

**EVENT_CNT_HxUy (RO):**

Number of FSSR2 event word seen on readout interface from FSSR2 HxUy chip since last latch of HxUy scalers.

**Register: A_FSSR_WORD_CNT_HxUy**

Address Offset: 0x100C, 0x110C,…,0x170C
Size: 32bits
Reset State: 0xXXXXXXXX

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

**WORD_CNT_HxUy (RO):**

Number of FSSR2 words seen (status+event) on readout interface from FSSR2 HxUy chip since last latch of HxUy scalers.

**Register: A_FSSR_IDLE_CNT_HxUy**

Address Offset: 0x1010, 0x1110,…,0x1710
Size: 32bits
Reset State: 0xXXXXXXXX

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

**IDLE_CNT_HxUy (RO):**

Number of idle clock cycles (125MHz domain) where no FSSR2 words seen on readout interface from FSSR2 HxUy chip since last latch of HxUy scalers.

**Register: A_FSSR_AQBCO_CNT_HxUy**

Address Offset: 0x1014, 0x1114,…,0x1714
Size: 32bits
Reset State: 0xXXXXXXXX

| 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 |
| AQBCO_CNT_HxUy | | | | | | | |

**AQBCO_CNT_HxUy (RO):**

Number of FSSR2 status words seen on readout interface with AQBCO set from FSSR2 HxUy chip since last latch of HxUy scalers.

**Register: A_FSSR_MARKERR_CNT_HxUy**

Address Offset: 0x1018, 0x1118,…,0x1718
Size: 32bits
Reset State: 0xXXXXXXXX

| 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 |
| MARKERR_CNT_HxUy | | | | | | | |

**MARKERR_CNT_HxUy (RO):**

Number of FSSR2 words seen on readout interface with having a mark error from FSSR2 HxUy chip since last latch of HxUy scalers.

**Register: A_FSSR_ENCERR_CNT_HxUy**

Address Offset: 0x101C, 0x111C,…,0x171C
Size: 32bits
Reset State: 0xXXXXXXXX

| 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 |
| ENCERR_CNT_HxUy | | | | | | | |

**ENCERR_CNT_HxUy (RO):**

Number of FSSR2 event words seen on readout interface with invalid channel from FSSR2 HxUy chip since last latch of HxUy scalers.

**Register: A_FSSR_CHIPIDERR_CNT_HxUy**

Address Offset: 0x1020, 0x1120,…,0x1720
Size: 32bits
Reset State: 0xXXXXXXXX

| 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 |
| CHIPIDERR_CNT_HxUy | | | | | | | |

**CHIPIDERR_CNT_HxUy (RO):**

Number of FSSR2 status words seen on readout interface with invalid chip ID from FSSR2 HxUy chip since last latch of HxUy scalers.

**Register: A_FSSR_HIST_CFG_HxUy**

Address Offset: 0x1024, 0x1124,…,0x1724
Size: 32bits
Reset State: 0xXXXXXXXX

| 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 |
| - | - | - | - | - | - | - | MODE |

**MODE (R/W):**

'0' – Histogram for FSSR2 HxUy chip data in channel hit mode
'1' – Histogram for FSSR2 HxUy chip data in latency mode

**Register: A_FSSR_GOTHIT_CNT_HxUy**

Address Offset: 0x1028, 0x1128,…,0x1728
Size: 32bits
Reset State: 0xXXXXXXXX

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

**GOTHIT_CNT_HxUy (RO):**

Number of FSSR2 gothit rising edges seen from FSSR2 HxUy chip since last latch of HxUy scalers.

**Register: A_FSSR_HIST_CNT_HxUy**

Address Offset: 0x102C, 0x112C,…,0x172C
Size: 32bits
Reset State: 0xXXXXXXXX

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

**HIST_CNT_HxUy (RO):**

After FSSR2 HxUy histograms are disabled for readout the bin pointer is reset to 0. Each read of HIST_CNT_HxUy increments the bin address pointer so that subsequent reads to this address read out the following bins in sequential order.

Each time a bin is read it is also cleared. All histogram bins used should be read to clear them for the next latch period.

**Register: A_FSSR_REF_CNT_HxUy**

Address Offset: 0x1030, 0x1130,…,0x1730
Size: 32bits
Reset State: 0xXXXXXXXX

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

**REF_CNT_HxUy (RO):**

Number of 125MHz clock edges seen since last scaler latch of HxUy scalers.

**Notes:**

1) This scaler can be used to normalize other scalers latched by the HxUy latch bit.

**Register: A_FSSR_HIST_TIME_HxUy**

Address Offset: 0x1034, 0x1134,…,0x1734
Size: 32bits
Reset State: 0xXXXXXXXX

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

**HIST_TIME_HxUy (RO):**

Number of 125MHz clock edges seen during the histogram enable period since last scaler latch of HxUy scalers.

**Notes:**

1) This scaler can be used to normalize the histogram bins that are enabled/disabled by the HxUy latch bit.

**Register: A_FSSR_CORETALK_CNT_HxUy**

Address Offset: 0x1038, 0x1138,…,0x1738
Size: 32bits
Reset State: 0xXXXXXXXX

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

**CORETALK_CNT_HxUy (RO):**

Number of FSSR2 coretalking rising edges seen from FSSR2 HxUy chip since last latch of HxUy scalers.

**Register: A_FSSR_LAST_DATA_HxUy**

Address Offset: 0x103C, 0x113C,…,0x173C
Size: 32bits
Reset State: 0xXXXXXXXX

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

**LAST_DATA_HxUy (RO):**

Last FSSR2 data word seen on readout interface from FSSR2 HxUy chip since last latch of HxUy scalers.

## 8. Document Revision History

**5/23/2013:**

1) Modified register A_TRIG_LATENCY. Limited maximum latency to 8us.
2) Added A_SCALER_TRIGGER and A_SCALER_TRIGGER_ACCEPTED

**10/15/2013 (Firmware V2.8):**

1) Modified register A_DAQ_TRIG.Support pulsing on specific BCO number.
2) Modified register A_FSSR_CLK_CFG. Support 1b, 2b, 4b readout modes in addition tostandard 6b mode.
3) Added register A_FSSR_LAST_DATA_HxUy.