	Document #	Date effective 24 May 2010
	Author(s) Ryan Herbst	Supersedes
	Draft Revision 0.24	
Document Title **EXO Electronics Functional Specification**		

Hard copies of this document are for REFERENCE ONLY and should not be considered the latest revision.

CHANGE HISTORY LOG

Revision	Effective Date	Description of Changes
0.04	03/21/2005	Updated to match current design.
0.05	02/21/2006	Updated to match design changes.
0.06	03/21/2006	Added memory debug registers & HV board registers.
0.07	05/24/2006	Added veto and calibration registers, added Veto data frames
0.08	01/18/2008	Added self trigger disable in conversion control.
0.09	12/10/2008	Added trigger rise time and hysteresis registers.
0.10	10/02/2009	Added new DAQ ready and Trigger FIFO Count Registers.
0.11	11/02/2009	Fixed typos.
0.23	01/06/2010	Fixed more typos. Modified Trigger and Veto frame formats.
0.24	05/24/2010	Moved HV board registers.

Hard copies of this document are for REFERENCE ONLY and should not be considered the latest revision.

1. DEFINITIONS AND ACRONYMS

The following terms, abbreviations, and acronyms are used in this document:

1.1 Acronyms

LVDS	**Low Voltage Differential Signaling**

2. REFERENCES

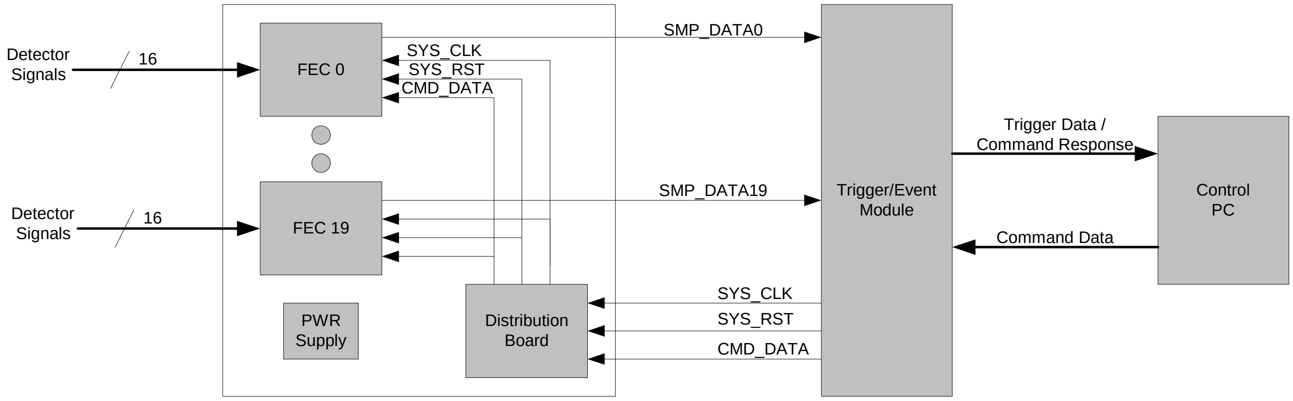
The list below provides documents that are to be used as references for this procedure:

2.1 Applicable Documents

Document Number	Description
PROCEDURES	

3. OVERVIEW

An overview of the EXO electronics is shown in the following figure. The various sensor wires arrive at a VME like chassis which contains up to 20 Front End Cards, each connected to 16 sensor wires. The first set of cards (up to 8) are connected to a series of Avalanche Photo Diodes (APD) and are known as the APD detector boards. The second set of Front End Cards (up to 6) are connected to a series of sensor wires biased at virtual ground and are known as the VG detector boards. The final set of Front End Cards (up to 6) are connected to a series of sensor wires biased at a high DC potential (~3KV) and are known as the HV detector boards.



EXO System Overview

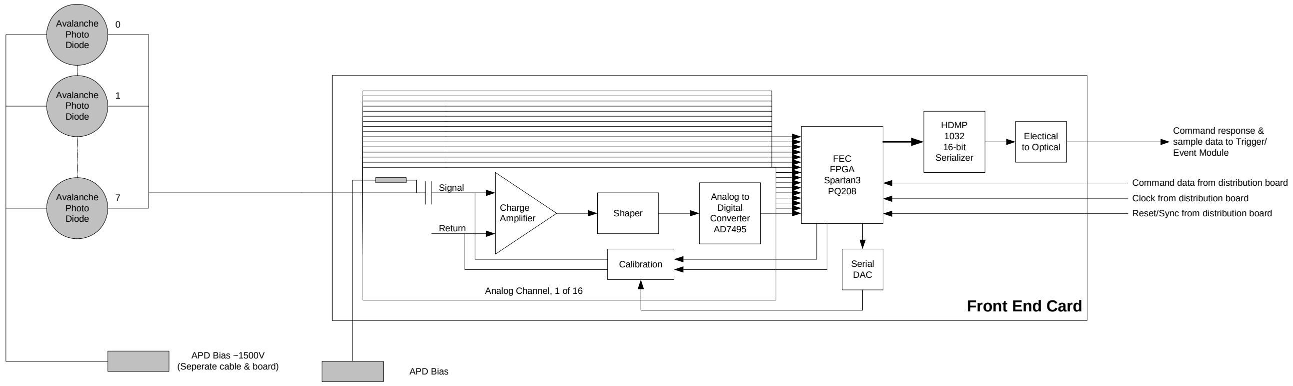
In addition to the Front End Cards the Front End Assembly (FEA) contains a single Front End Distribution Board (FEDB) which distributes the system clock, reset and command data signals received from the Trigger Event Module (TEM). These signals are received from the TEM over three fiber optic lines and are converted to LVDS signals for distribution across the Front End Backplane (FEB). Each of the 20 Front End Cards has its own fiber optic link to the Trigger Event Module which is used for sending back sample data as well as responses to received commands.

The Trigger Event Module (TEM) collects sample data from the Front End Cards, scans it for trigger points and stores the data in DDR memory. Once a trigger is detected the TEM will forward the trigger sample along with both pre-trigger and post-trigger data to the Control PC (CPC). The TEM is also responsible for distributing command traffic received from the CPC and collecting command responses from the Front End Cards for transmission back to the PC.

Another function of the TEM module is to add an accurate timestamp to each trigger that is sent to the control PC. The TEM has the ability to receive NTP timestamp data from the control PC which is used to calculate the time of each trigger event accurate to about 100ms.

3.1 Front End Card Overview

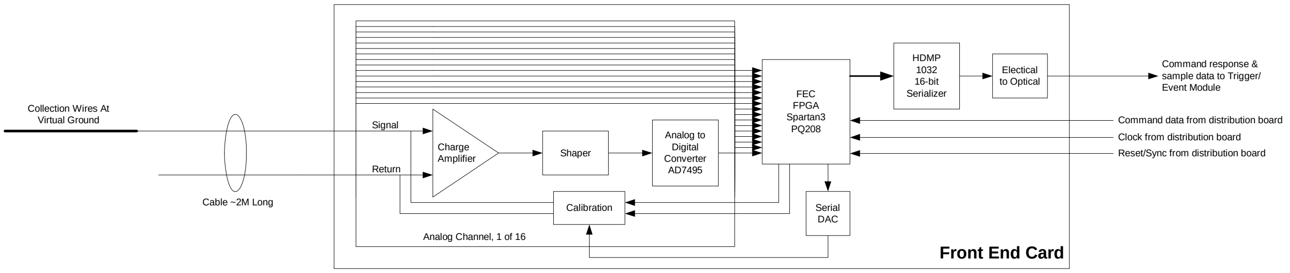
As described earlier the Front End Cards come in three flavors. A common PCB layout can be configured as an APD detector, a VG detector, a HV detector by changing the components loaded on the board. The following diagram shows an APD Front End Card connected to the APD detectors.


Avalanche Photo Diode
0
Avalanche Photo Diode
1
Avalanche Photo Diode
7
Signal
Return
Charge Amplifier
Shaper
Analog to Digital Converter AD7495
FEC FPGA Spartan3 PQ208
HDMP 1032 16-bit Serializer
Electical to Optical
Command response & sample data to Trigger/ Event Module
Command data from distribution board
Clock from distribution board
Reset/Sync from distribution board
Calibration
Serial DAC
Analog Channel, 1 of 16
Front End Card
APD Bias ~1500V (Seperate cable & board)
APD Bias

Front End Card Configured As APD Detector

The input signal from the APDs is held at a non-zero voltage in order to assure a specific gain on the APDs. This voltage is induced through a weak pull-up resistor on the APD circuit board. Each APD has two input points for this voltage, on for each group of eight channels on the board. The decoupled signal is then passed through a charge amplifier and then on to the signal shaper. The resulting signal is sampled at a 1-Mhz rate by an analog to digital converter (ADC) with a resolution of 12-bits. The FPGA on the Front End Card collects this data from the 16 detector channels and forms it into a single sample data frame. With each 12-bit sample stored in a 2-byte word, the resulting sample data rate on each Front End Card is 32-MBps.

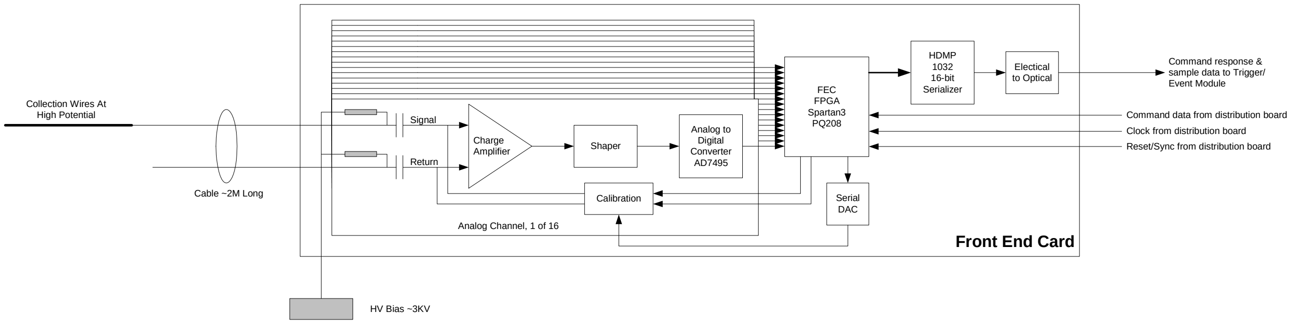
When configured as a VG detector the Front End Card is connected to a series of sense wires aligned horizontally in the chamber. Since these wires are held at a virtual ground potential they are connected directly into the charge amplifier on the Front End Card as shown in the following diagram.



Front End Card Configured As VG Detector

Each Front End Card configured as a VG detector can output 32-MBps similar to the APD detector.

The third and final type of Front End Card is the HV detector. As shown in the following diagram the individual sense wires aligned in the vertical direction are biased at a high DC potential (~3KV). This bias level is achieved through the use of weak pull up resistors on each of the Front End Cards that are connected to an external power source. A single de-coupling capacitor is inserted into the signal path to keep this high potential off of the charge amplifier input.



Front End Card Configured As HV Detector

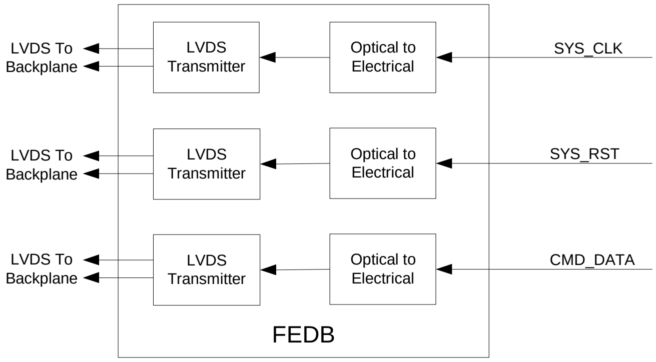
With 6 of these detectors in the EXO system the sample data rate is the same as the VG detector, 192-MBps.

Each Front End Card type contains a single digital to analog converter (DAC) and supporting logic that allows a defined charge to be inserted onto the signal wire. This allows the performance of the analog logic to be checked and characterized. A fixed voltage output from the calibration DAC is combined with a fixed length pulse from the FEC FPGA to generate a calibration signal. Each of the 16-channels on the FEC can be individually masked, enabling or disabling the calibration strobe to the channel. Each Front End Card also has the ability to send back a defined data pattern in its sample data frame. This allows software to test the operation of the digital logic in the system without the use of the calibration strobe.

Each Front End Card type has a single 16-bit serializer connected to its FPGA. This serializer is used to convert a 16-bit wide data word received from the FPGA aligned on a 20-Mhz clock to a 400-Mbps serial stream (320-Mbps payload + 80Mpbs overhead). Of the 16 bits passed from the FEC FPGA the lower 15-bits are used for transmitting sample data back to the TEM. The last bit, bit 15, is reserved for a serial communications channel that is used for sending command responses back through the TEM to the control PC.

3.2 Front End Distribution Board

The Front End Distribution Board (FEDB) interfaces to three fiber optical cables from the TEM and converts the received signal to LVDS for distribution to each of the 20 Front End Cards in the chassis. The following diagram describes the logic on the FEDB.

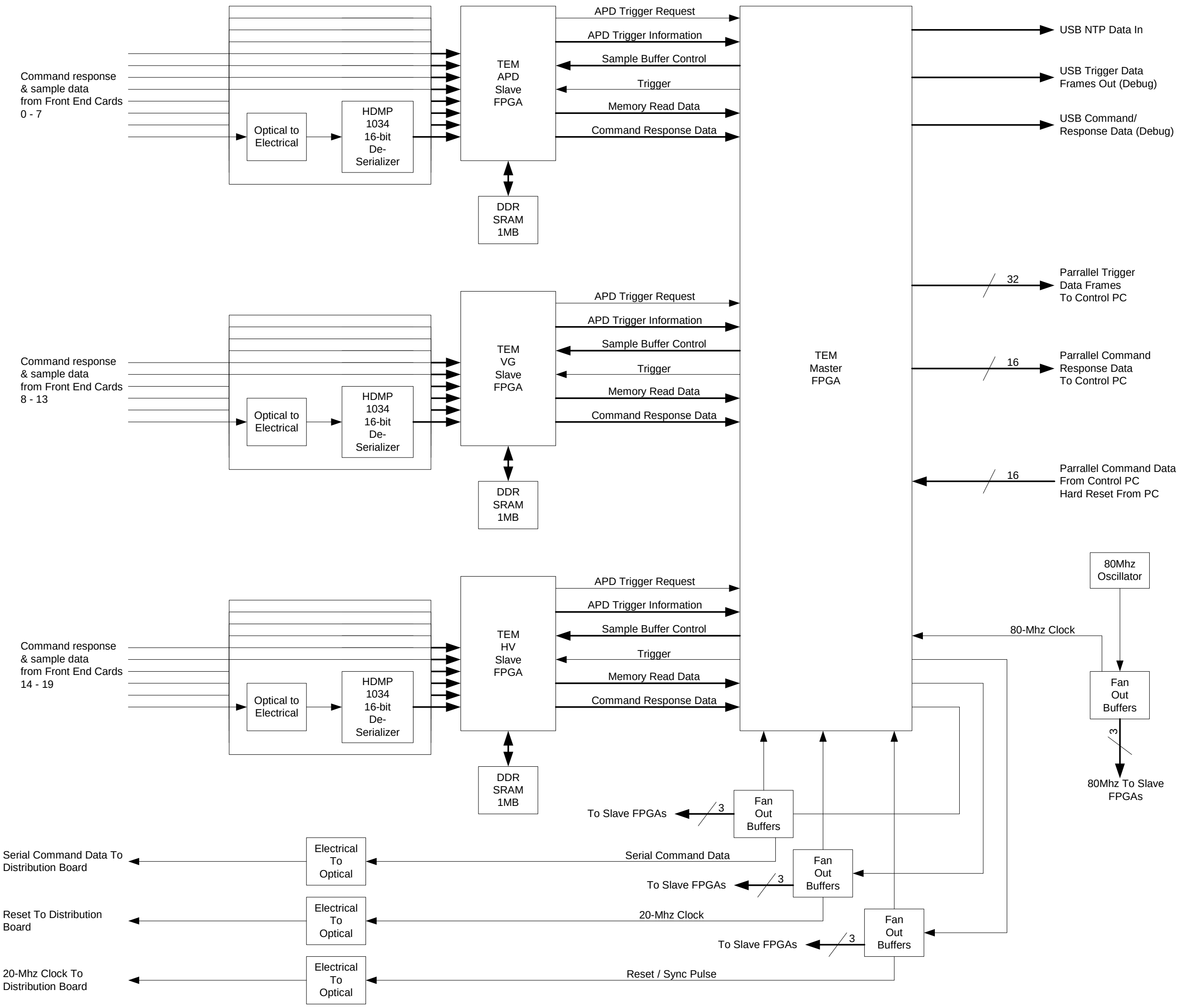


Front End Distribution Board

The Distribution Board uses the same PCB as the Front End Card.

3.3 Trigger Event Module

The Trigger Event Module (TEM) interfaces the 20 Front End Cards to the Control PC. A block diagram describing the logic on the TEM is shown below.



Trigger Event Module

The TEM receives sample data from the Front End Cards over 20 fiber optical links sending data at 400-Mbps. These serial streams are converted back into 16-bit 20-Mhz parallel data by the de-serializer chips. The de-serializer chip will align the received serial stream to the local 20-Mhz clock. Since the entire system is sourced from a single 80-Mhz clock source there will be no frequency shift in the recovered clock, only a fixed phase shift.

Due to the high number of incoming serial streams the sample processing logic is divided amongst three slave FPGAs on the TEM. Each slave FPGA is responsible for processing data from a particular detector group. Slave FPGA 0 processes data from the APD detectors, slave FPGA 1 processes data from the VG detectors and slave FPGA 2 processes data from the HV detectors.

Each slave FPGA stores the received data and looks for a trigger. There are two types of triggers in each of the slave FPGAs. The first trigger type computes the difference between the current sum of the individual channels and the running average of the channel sum. If this difference is greater than one of 4 set thresholds then a trigger is asserted. The second trigger type finds the individual channel with the highest magnitude and compares it against 4 separate threshold values. When a trigger is found it will indicate the event to the master FPGA which will processes the trigger information further. The master FPGA also controls the allocation of event buffer resources for each slave FPGA to ensure all three FPGAs are kept in sync.

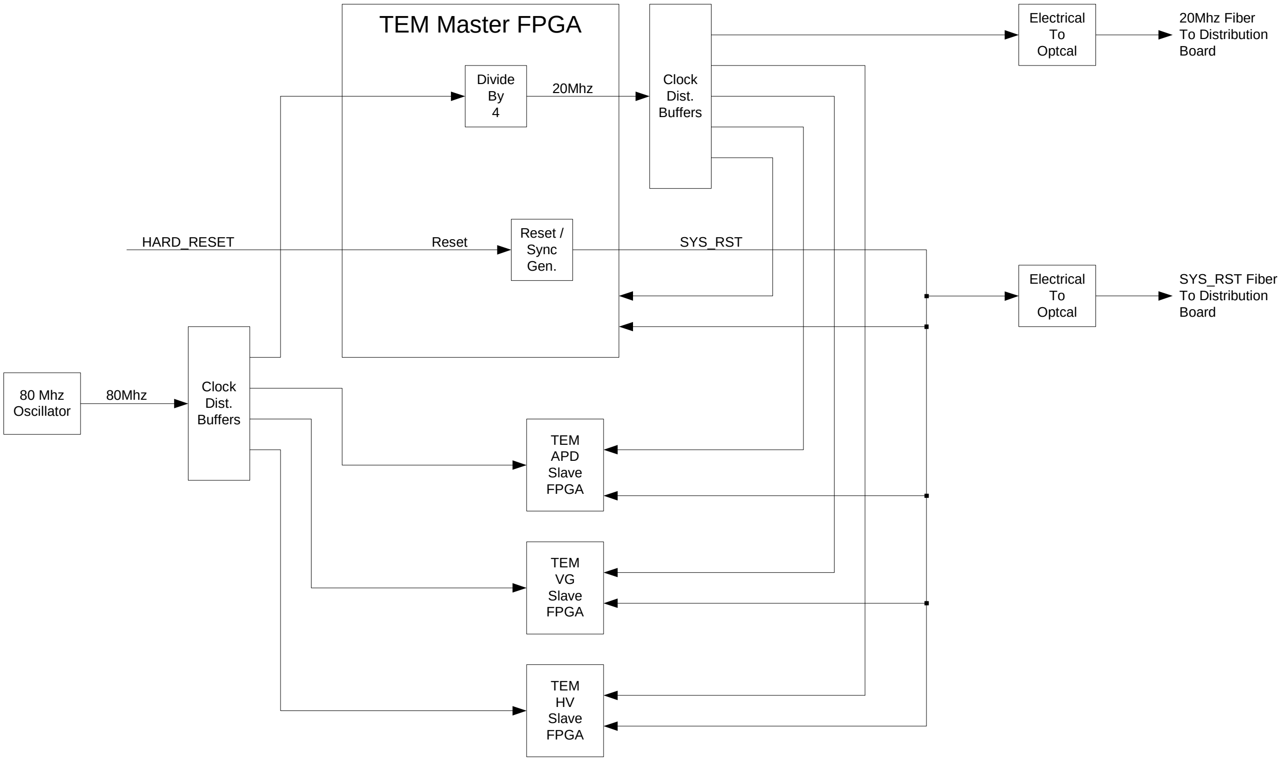
The TEM serves as the clock & reset source for the entire system. Clock and reset data are distributed in such a manner as to ensure that all four FPGAs on the TEM, as well as the FPGAs on the 20 Front End Cards, receive reset pulses and data at the same instance. The single reset line distributed from the TEM has the ability to function as both a sync pulse and a reset pulse. A detailed explanation of the system clocking follows in a later section.

The TEM interfaces to the Control PC through an interface card available from National Instruments. The PC will be outfitted with two of these PC cards, one for command traffic and the other for trigger data. The command interface board will be configured so that 16 of its 32-bits will be used for transmitting command frames to the EXO electronics while the remaining 16-bits will be configured as an input fro receiving response frames. The second interface board will have all 32-bits configured as inputs for receiving trigger data frames from the TEM.

4. SYSTEM CLOCKING

The clock and reset signals are distributed in such a way as to ensure that all four FPGAs on the TEM, as well as the FPGAs on the 20 Front End Cards, receive a common 20-Mhz system clock and a common synchronous reset. A common alignment of the distributed reset to the 20-Mhz system clock is necessary to ensure that all the locally generated 1-Mhz pulses and system timers are aligned. The locally generated 1-Mhz pulses are used to control the sample point of the analog to digital converter. The local system timer is used to add a local timestamp to the sample data.

The following diagram shows the clock & reset distribution logic present in the EXO electronics.



System Clock & Reset Distribution

All of the clocks in the EXO system are generated from a single 80Mz clock source. This clock is distributed locally on the TEM board to the master FPGA and all three of the slave FPGAs. The master FPGA uses this clock for its local logic and also passes the 80-Mhz clock to a divide by four block to generate the 20-Mhz system clock. The output of this divide by four logic is connected to a distribution buffer that is used to pass the clock back to the master FPGA, to the three slave FPGAs and down to the FPGAs on the 20 Front End Cards in the system. The 20-Mhz clock reaches the Front End Cards over a fiber optic line which carries the clock to the distribution board in the Front End Assembly.

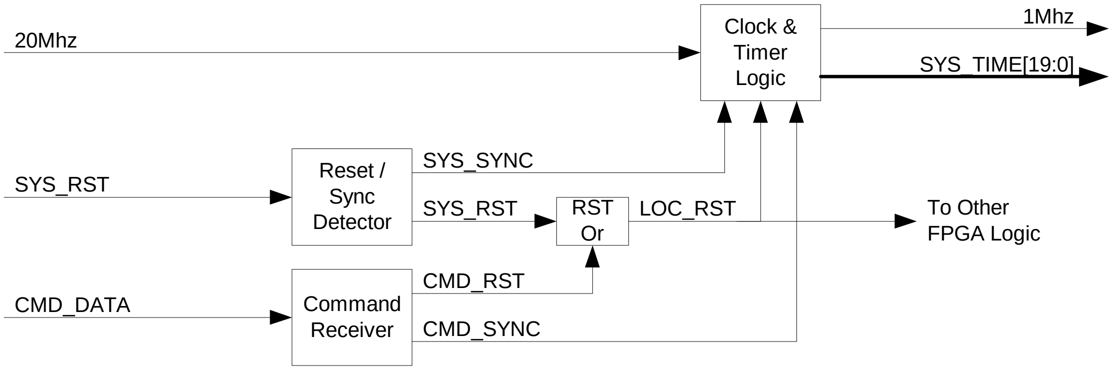
The synchronized SYS_RST signal is generated by the reset logic in the TEM master FPGA. This logic generates the signals that are passed over a shared reset signal to itself, the three slave FPGAs and all of the Front End Cards. Since the optical transmitter/receivers do not respond well to a signal with a constant value of ‘1’ or ‘0’, the reset signal must be an encoded signal with distinct patterns which represent the reset state of the system. In addition to supporting the required reset and not reset states the distributed signal also has the ability to support a sync state. This sync state has the same affect as the sync command send from the PC. The sync code will be sent down the reset line in response to a system time update received from the PC while the system is Idle and not taking data. This mechanism for generating the sync code is described in the next section. The sync code can also be used as a diagnostic tool and is currently generated when a jumper is inserted on the TEM board.

The encoded value send over the distributed reset line is an 8-bit value which will contain one of three codes depending on the state being transmitted over the reset line. The three codes sent over the reset line are shown below:

- 10101010 – Idle. This value indicates that the reset state is Idle and the device should be taken out of reset.
- 10011010 – Reset. This value indicates that the reset state is Reset and the device should be held in reset until the next Idle code is received.
- 10101010 – Sync. This value indicates that the reset state is Sync and an internal sync pulse should be generated within the device.

Upon power up each device will be in a reset state and will be receiving either a series of ‘0’s or ‘1’s on the reset line. Each device will stay in reset until the TEM master FPGA becomes active, transitions out of local reset and begins transmitting the Idle characters down the link. At this time the remaining FPGAs will be taken out of reset. Notice the similarity between the Reset and Sync codes in the above list. As the master FPGA transitions from sending a Reset code to an Idle code a single Sync will be detected by the receiving logic. This sync will have no affect since the local logic will not be taken out of reset until a true Idle code is detected on the link.

Each FPGA in the system contains a block of logic which receives the SYS_RST codes and generates local sync and reset signals. The following diagram shows the logic contained within each FPGA for local clock generation and system reset decoding.



Local Clock & Reset Logic

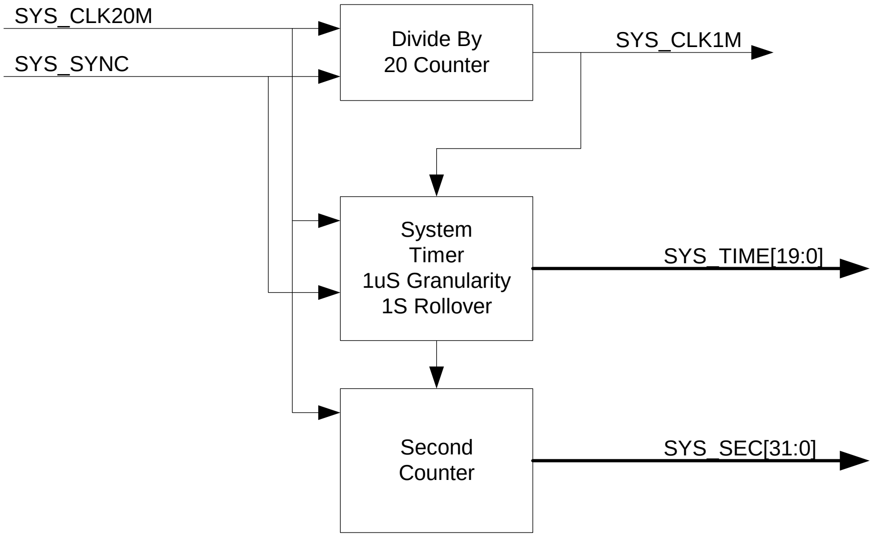
The 20-Mhz clock received from the clock generation logic in the TEM master FPGA is received by each FPGA and passed to the local clock & timer logic. Here a local 1-Mhz clock pulse is generated along with a 20-bit time stamp. The 20-bit time stamp is a running counter which increments each time the 1-Mhz clock pulse is asserted. The time stamp value will reset to zero every 1 second.

The fixed alignment of the 1-Mhz pulses and 20-bit system timers across the system is guaranteed through a variety of methods. When the entire system is reset both the counter generated the 1-Mhz pulse and 20-bit timer are reset to an initial value of zero. Since each unit in the entire system receives the same reset pulse, all of the timers in the system will be aligned when coming out of reset. In addition the command decoder logic supports a SYNC command which will reset the 20-bit timer and re-align the 1-Mhz clock pulses. Since each unit in the system receives commands with similar alignment every unit in the system will be aligned following a broadcast SYNC command. The last method for aligning the timer and 1-Mhz clock pulse is by sending a sync code down the SYS_RST line.

5. SYSTEM TIMERS

As each trigger event in the system is generated, a timestamp is added to the trigger data frame to identify the point in time that the trigger occurred. This trigger timestamp is a 52-bit NTP timestamp which indicates two values. The upper 32-bits of this 52-bit timestamp indicate the number of seconds that have occurred since 0:0:0 am on January 1, 1900. The lower 20 bits indicate the current sub-second count in microseconds. The seconds count is incremented each time the sub-second count rolls form a value of 999999 to 0. The true NTP timestamp is a 64-bit value with the sub-second counter having a precision of 232 picoseconds. It is important to note that the 32-bit second value in the NTP timestamp will saturate toward the end of February 7, 2036.

The following diagram shows the structure of the timers present in the TEM master FPGA.



System Timers

The first block shown generates the local 1-Mhz clock pulse and the second block is a 1-uS counter. These two blocks of logic are reset each time a SYS_SYNC is received either from the command decoder or the TEM master FPGA timing logic. Each time this 1uS counter reaches a value of 999999 it is reset and the seconds counter is incremented.

Software has the ability to set an initial seconds value into the TEM Master FPGA through a USB connection. Once this value is set the accuracy of the timestamp kept in the Master FPGA is dependent upon the accuracy of the local 80Mhz oscillator. It is expected that this oscillator will be slightly inaccurate and the TEM Master FPGA timer will drift over time from the true NTP timestamp as time passes since the last update of the NTP time by the PC.

In order to keep all of the uS counters in the system synced up the Master FPGA will generate a sync code on the reset line anytime the PC updates the NTP seconds value contained in the Master FPGA timer logic. Since this sync re-aligns all of the 1Mhz timer pulses the operation of the ADCs could be impacted. In order to prevent the ADCs from being interrupted the timer logic within the Master FPGA will operate in two modes.

If the system is currently Idle and not taking data, a sync pulse will be generated following a time update from the PC. Once this operation occurs the entire system will be in sync and the sub-seconds counters will be reset to a value of 0. The internal second counter in the TEM Master FPGA will be preset to this passed value.

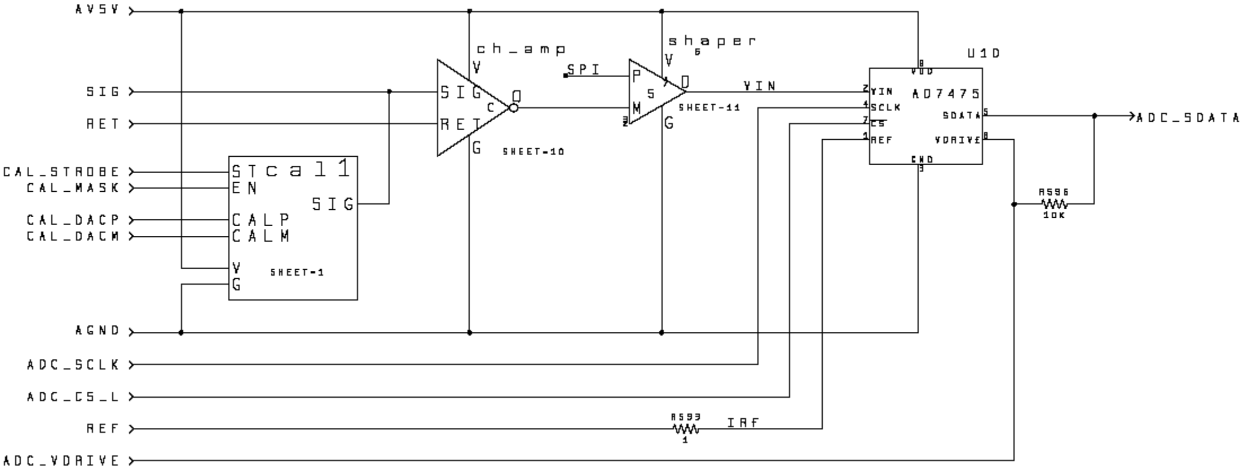
Once the system becomes active and data collection the PC will continue to write its NTP time at a fixed interval to the TEM. The TEM will not respond to this NTP time value write by resetting its local counters and sending a sync code over the reset line. Instead it will compute the difference between the local counters and the value which was sent from the PC. This difference is stored in a 32-bit drift register which will contain a signed integer value that is included in every trigger header. The combination of the NTP timestamp contained in the trigger header and this drift value can be used by software to compute the true time of the trigger event to the accuracy of the NTP client on the PC.

The clock drift measurement will be re-generated each time the PC sends an updated NTP time value to the TEM Master FPGA. The value contained it in the trigger value will be the last value that was calculated before the trigger event occurred. This signed 32-bit drift value allows for a maximum drift of 2,147,483,648 uSeconds (~2147 seconds or ~36Min) occurring during a continuous data taking run.

The following table indicates the amount of time that a single uninterrupted run can continue without saturating the drift value for various oscillator inaccuracies.

True Master Oscillator Frequency	Drift / Day	Maximum Run Time
80Mhz + 50PPM	-4.320 seconds / day	248.564 Days
80Mhz + 25PPM	-2.160 seconds / day	497.115 Days
80Mhz	0 seconds / day	Infinite
80Mhz – 25PPM	2.160 seconds / day	497.115 Days
80Mhz – 50PPM	4.320 seconds / day	248.564 Days

6. ANALOG FRONT END



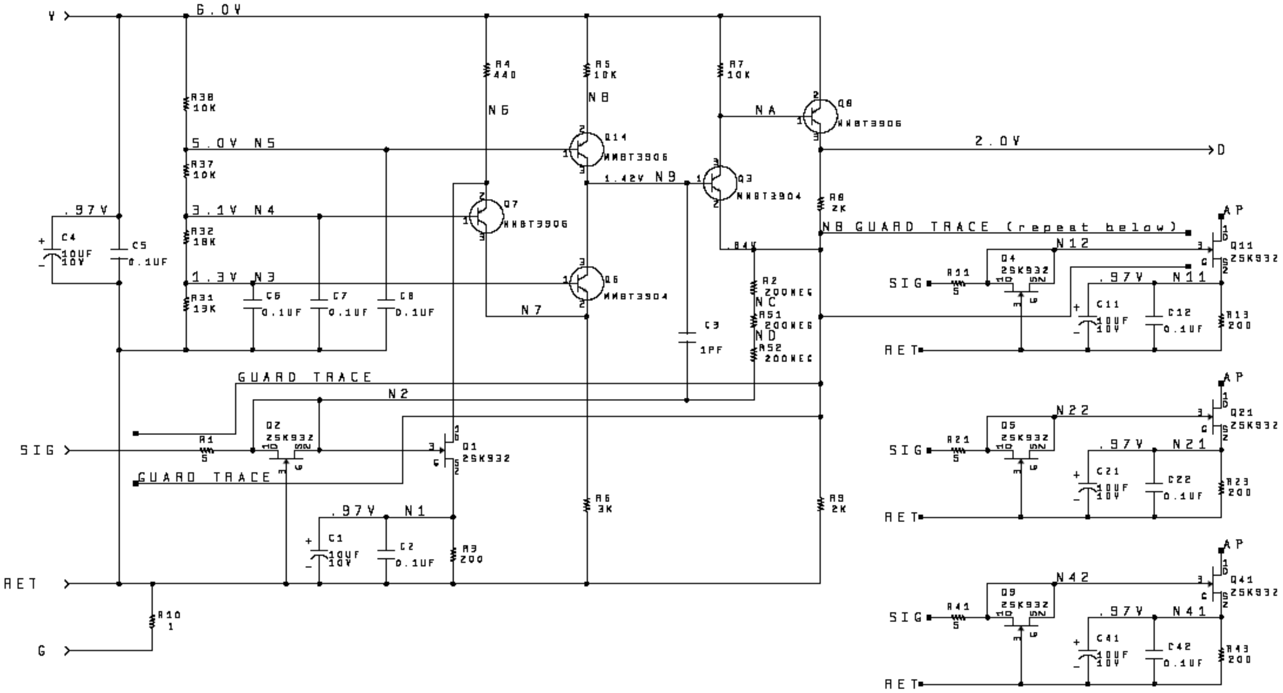
Analog Front End

The analog portion of the Front End Card is shown in the previous figure. The incoming signal passes through a charge amplifier and a shaper before being sampled by the 1Mhz analog to digital converter. A calibration block is used to insert a test charge into the circuit in order to characterize the operation of the circuit.

6.1 Charge Amplifier

The input FET (2SK932, gm = 30 mS at Id = 5 mA) drives node N6. The input impedance of the common base stage Q7 is low compared to the load resistor R4 = 440 Ohm (must get value from simulation), insuring efficient transfer of the signal into Q7. This is followed by a second common base stage Q6. The reason for the cascaded common base stages is to transform the high impedance at node N9 (> 100 K) to a much lower one (<100 Ohm) at N6. This large an impedance transformation was not possible with a single grounded base stage.

The measured open loop gain at DC is ~10,000. (Must determine as function of frequency in simulation and measurement). This gives an effective capacitance at the input of ~10,000 pF for a 1 pF feedback capacitor. A large effective capacitance is important in order to achieve low cross coupling between signals. The values of the feedback capacitor and feedback resistor will be adjusted to the requirements (decay time constant and effective capacitance at the input).



Charge Amplifier

The high impedance at N9 is achieved by drawing a small current (~30 uA) in Q14 and Q6 and by providing a buffer Q3/Q8 with a high input impedance.

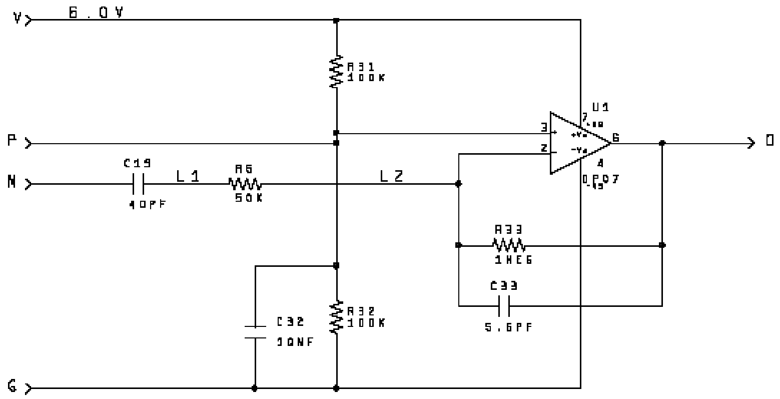
The AC feedback is taken from node N9 for optimum stability, avoiding the phase shift in buffer Q3/Q8. The DC feedback is taken from the unity gain node NB, which also acts as a level shift. (For the high impedance DC feedback, there is no problem with phase shift). The charge signal is taken from the gain=2 node "O". This extra gain reduces the requirements on the noise performance for the subsequent shaper/gain stage.

The input protection is a proposed scheme which needs to be tested. Anecdotal evidence is that the unprotected FET breaks "quite often". That sounds as if some protection is needed, but perhaps not a very powerful one. Resistor R1 may be made a short. A large value in that position would provide good protection, but would introduce too much noise. A value of 30 Ohm, equivalent to the noise source in the FET, would increase the noise by a factor 1.4. I propose to use the gates of the 2SK932 as protection devices. The gate of the input FET opens for a positive signal, and the gate of Q2 for any negative excursion. The FET gate should meet the requirement of low leakage in the off-state while being able to handle large currents (~100mA) in the on-state. Alternatively, back to back protection diodes of a type to be determined, could be used.

The input FET with the biasing resistor in the source line is repeated up to four times, while R4 will be reduced by a factor equal to the number of FETs. The current in each FET is stabilized by the negative feedback in the source line. The four FET option will be used for the APD system, which presents a 1000 pF load. Four FETs in parallel have been shown to give 1000 el. noise at 6 us shaping time. For the wire signals, two FETs may be advantageous.

A few guard traces, connected to the gain=1 output, are indicated in the schematic. These are added to the board lay-out and may be helpful in reducing leakage to the high impedance input to the charge amplifier.

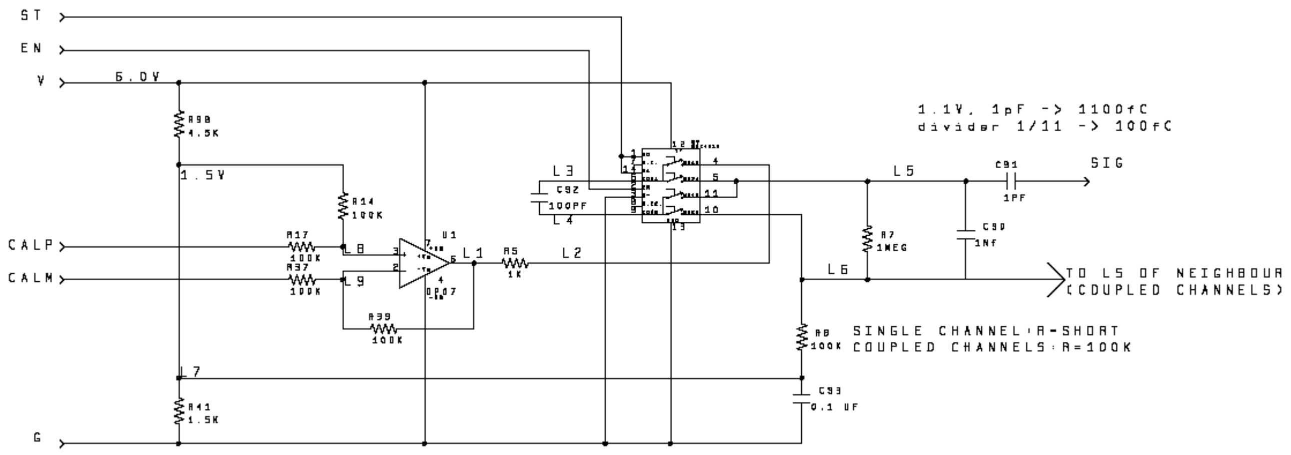
6.2 Shaper



Shaper

The shaper serves as differentiator, integrator and gain stage. The differentiation time constant is given by the product of C19*R6. It can be made very long by choosing C19 large. The integration time constant is the product of R33*C33 and again can be modified by changing C33. The gain is given by the ratio of R33/R6. The opamp used for the tests was OP07. It appeared to have sufficiently low noise. We will also try the AD8519, which has lower noise.

6.3 Calibrator



CALP	CALM	L1	L6
0.0	0	1.5	1.5
1.5	0	3.0	1.5
3.0	0	4.5	1.5

Calibrator

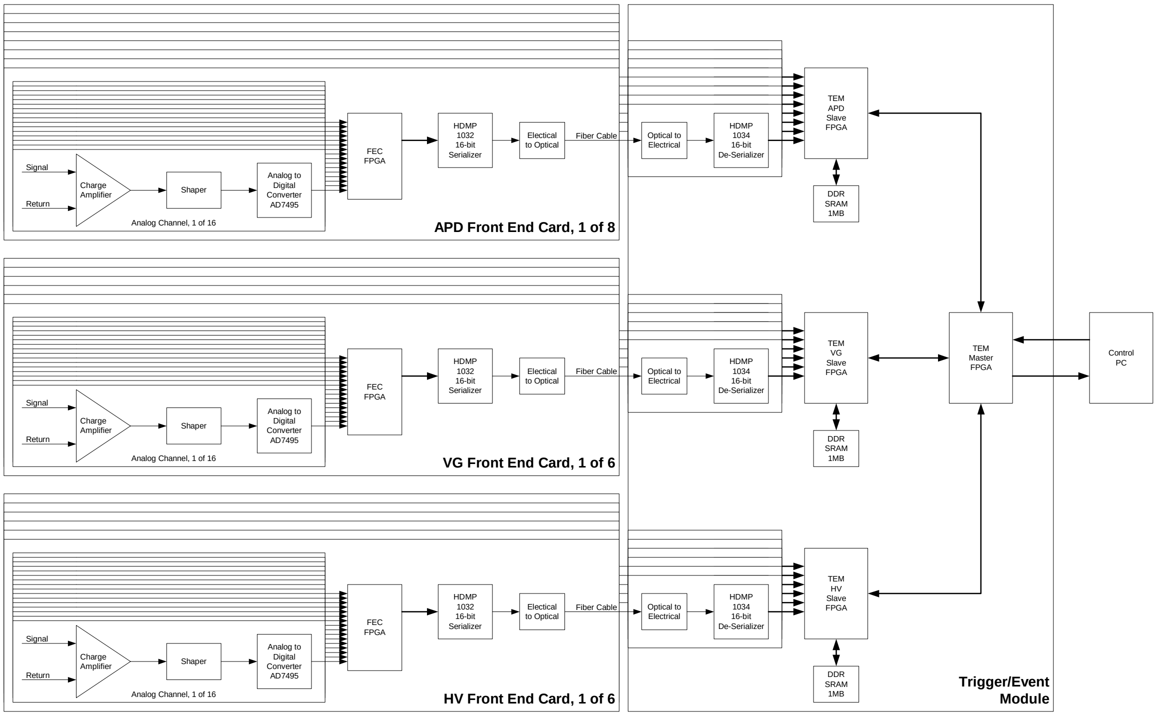
The calibrator injects charge through a switched capacitor arrangement. Each channel has a dedicated buffer which receives DC from a (shared) DAC through a pair of differential lines. This ensures that the amplifier channels do not talk to each other via the crucial DC calibration levels. The full scale signal at 100 fC is fairly small. Since, for good reproducibility, one would like to use a full scale DC level of ~1V and a capacitor not less than 1 pF, that amounts to 1 pC. A capacitive divider is implemented to reduce the calibration signal.

Two different options can be realized by loading different components in position R8 as indicated in the schematic. The most straightforward one is "Single Channel". A second option "Coupled Channels" is provided to test pairs of amplifiers connected through a signal provided by the charge on a floating capacitor. This is to simulate the signal flowing between the x- and y-wires and through a pair of amplifiers.

Alternatively, a board with only the calibrator loaded, could be connected to a mock-up of the drift chamber so that the test signal is driven to the amplifiers through the cable, a more realistic model for the signal flow in the experiment.

7. DATA FLOW

The following diagram shows the path of the sample data from its origin at the analog to digital converter all the way to the control PC. The analog to digital converter connects directly to the FPGA on the Front End Card (FEC) which processes the data from the 16 analog channels and forms it into a frame that is sent to the Trigger Event Module (TEM). Each group of detectors (APD, VG or HV) has its own slave FPGA on the TEM for processing the received data. These slave FPGAs store the sample data and perform the trigger processing for its group.



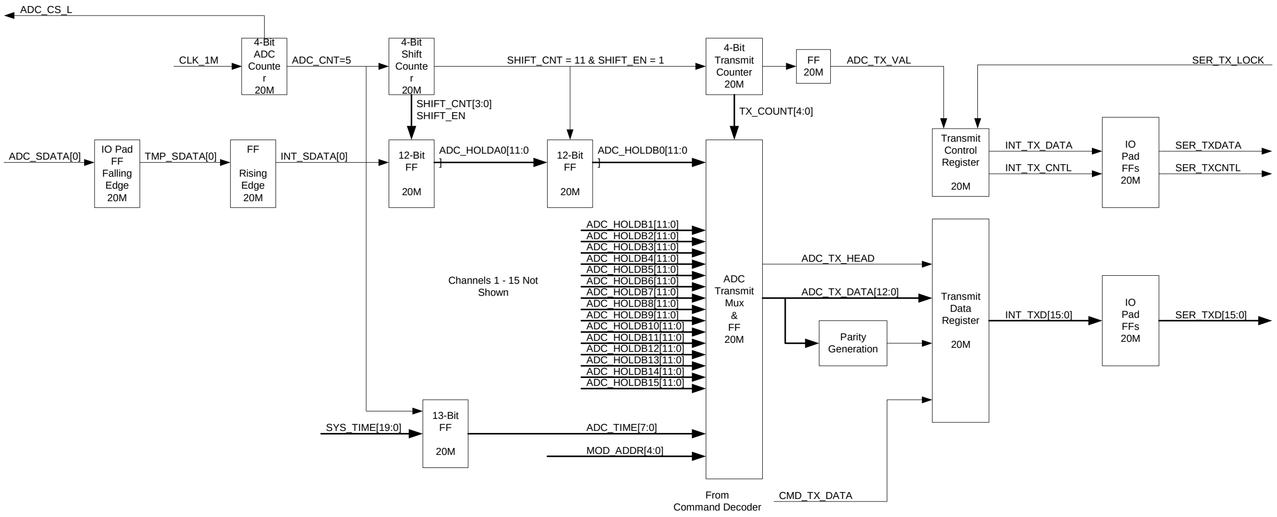
EXO Data Flow Overview

Although the slave FPGA is connected directly to the DDR SRAM used to store the sample data, the master FPGA on the TEM controls when the writes will take place and to which of eight samples buffers the data will be written. Reads from the sample memory are also controlled by the master FPGA and are timed to ensure that the slave FPGA can keep up with the incoming sample data. This read data will be formed into a trigger even frame by the master FPGA when a trigger event is detected.

7.1 Front End Card Sample Processing

The FPGA on the Front End Card (FEC) is responsible for controlling the timing of the analog to digital converter (ADC) and reading that sample data from it. Sample data from the analog to digital converter is received in a serial fashion, with four empty bits followed by 12 samples bits with the MSB of the 12-bit sample arriving first. The sample sequence is started by asserting the ADC_CS_L signal low.

The following figure shows the sample processing logic in the Front End Card FPGA. For clarity some logic for ADC channels 1-15 has been removed.



Front End Card Sample Logic

The timing of the sample sequence is determined by the local 1-Mhz timing pulse. This timing pulse is derived from the 20-Mhz system clock and is synchronized throughout the EXO system. The entire sample sequence is started with the arrival of the 1Mhz system pulse. The arrival of this pulse is followed by the assertion of the external chip select to the ADCs and the start of a local sequence counter to control the sampling of the incoming data. The incoming serial data from the ADC is then converted into a 13-bit word containing the 12-sample bits and a single sample parity bit. This parity bit will remain with the sample data until it reaches the control PC.

The parallel data from the 16 ADC channels is then formed into a sample data frame, shown below, for transmission to the Trigger/Event Module (TEM). A header is added to the front of the sample data which contains the lower 8 bits of the timestamp, the module address, a header parity bit and a flag bit to indicate the start of the sample data frame.

	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
US Data Header	1	HP	Module Address[4:0]					Sample_Time[7:0]							
Channel 0 Data	0	0	SP	Sample_Data[11:0]											
Channel 1 Data	0	0	SP	Sample_Data[11:0]											
Channel 2 Data	0	0	SP	Sample_Data[11:0]											
Channel 3 Data	0	0	SP	Sample_Data[11:0]											
Channel 4 Data	0	0	SP	Sample_Data[11:0]											
Channel 5 Data	0	0	SP	Sample_Data[11:0]											
Channel 6 Data	0	0	SP	Sample_Data[11:0]											
Channel 7 Data	0	0	SP	Sample_Data[11:0]											
Channel 8 Data	0	0	SP	Sample_Data[11:0]											
Channel 9 Data	0	0	SP	Sample_Data[11:0]											
Channel 10 Data	0	0	SP	Sample_Data[11:0]											
Channel 11 Data	0	0	SP	Sample_Data[11:0]											
Channel 12 Data	0	0	SP	Sample_Data[11:0]											
Channel 13 Data	0	0	SP	Sample_Data[11:0]											
Channel 14 Data	0	0	SP	Sample_Data[11:0]											
Channel 15 Data	0	0	SP	Sample_Data[11:0]											

HP = Header Parity, Even   SP = Sample Parity, Even

Sample Data Frame

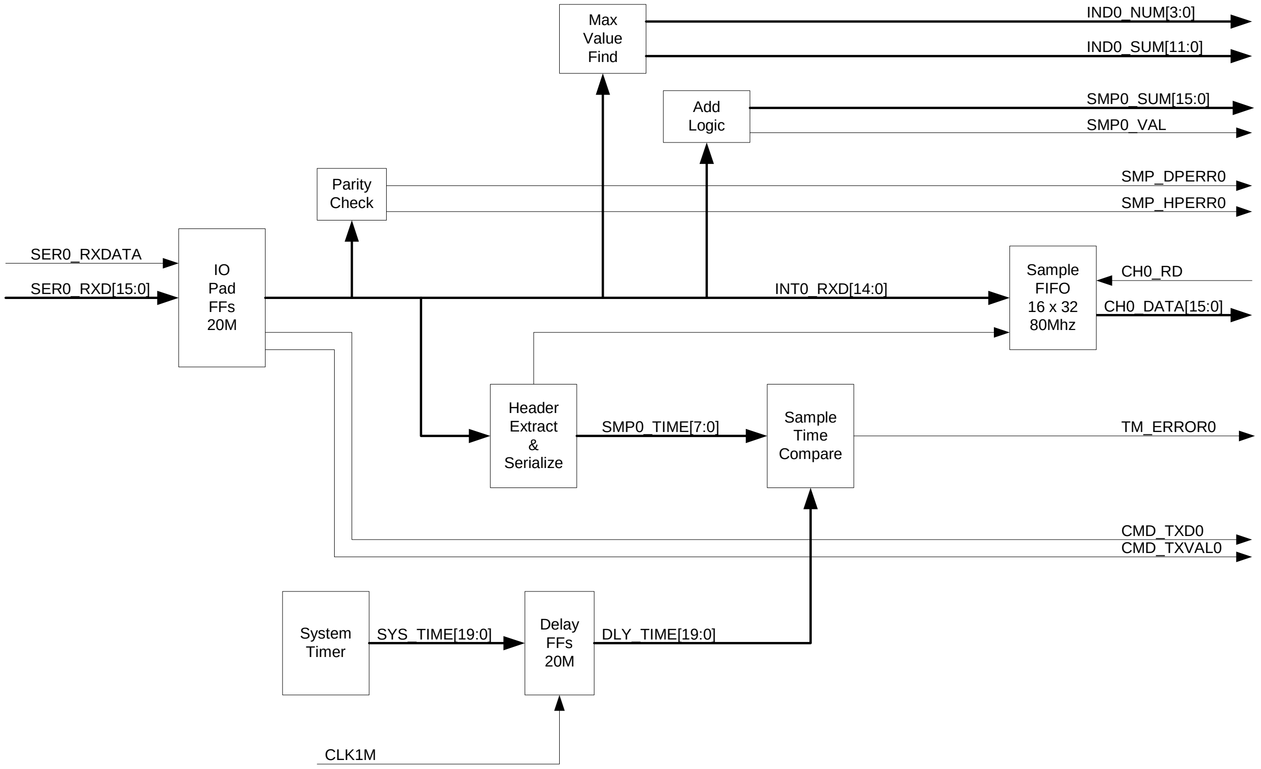
Once the data has been packed into a frame for transmission to the TEM the data is passed to an external serializer device. This device takes in 16-bit words arriving at 20Mhz and converts them into a 400Mbs (320Mbs data + 80Mbs overhead) serial stream. 15 of these bits come from the FEC FPGA sample processing logic while the 16th bit is used for a serial communications channel containing command response data.

The Front End Card has the ability to override the ADC sample data with a fixed data pattern for testing purposes. When the TEST_DATA bit is set in the conversion control register the 12-bit sample from the ADCs are replaced with test data as shown below:

ADC Channel	Sample Bits[11:8]	Sample Bits[7:0]
0	0x0	SYS_TIME[7:0]
1	0x1	SYS_TIME[15:8]
2	0x2	~SYS_TIME[7:0]
3	0x3	~SYS_TIME[15:8]
4	0x4	SYS_TIME[7:0]
5	0x5	SYS_TIME[15:8]
6	0x6	~SYS_TIME[7:0]
7	0x7	~SYS_TIME[15:8]
8	0x8	SYS_TIME[7:0]
9	0x9	SYS_TIME[15:8]
10	0xA	~SYS_TIME[7:0]
11	0xB	~SYS_TIME[15:8]
12	0xC	SYS_TIME[7:0]
13	0xD	SYS_TIME[15:8]
14	0xE	~SYS_TIME[7:0]
15	0xF	~SYS_TIME[15:8]

7.2 Trigger/Event Module Sample Processing

The 400Mbs serial stream received by the Trigger Event Module (TEM) is converted back into a sequence of 16-bit words. The de-serializer device will align the received data to the local 20-Mhz clock on the TEM and present this data to one of the three slave FPGA devices. Each slave FPGA device is responsible for a specific group of detectors. Slave FPGA 0 is responsible for Front End Cards configured as APD detectors, while FPGAs 1 and 2 are responsible for Front End Cards configured as VG & HV detectors respectively. All three FPGAs will contain the exact same logic and samples from the any missing Front End Cards will appear as NULL values.

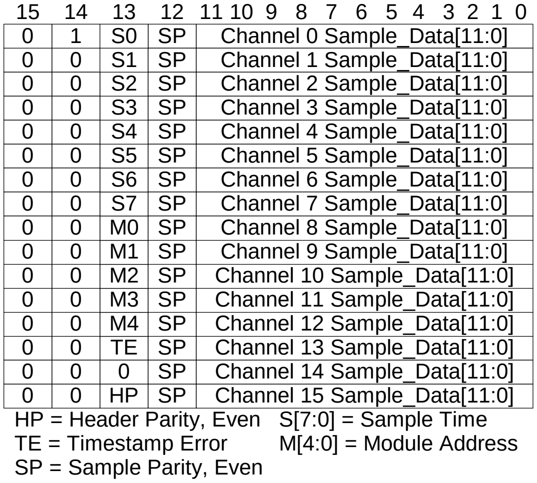


TEM Sample Processing

Only one of the 8 sample processing channels is shown in the previous figure for simplicity.

As mentioned earlier the lower 15-bits of the 16-bit word received from the de-serializer contain the sample data frame sent from the Front End Card. Bit 15 of the 16-bit word contains a serial communications channel used for command response frames. The TEM slave FPGA will separate these two traffic streams, passing the serial channel to the command response processing logic while passing the sample data frames to the trigger processing logic described below.

As the sample frame is received the header word will be extracted from the head of the frame. This data will be converted to a serial stream which will be inserted into bit 13 of the converted frame. This is done to keep the size of the stored frame at a power of 2 in order to conserve memory space. As the header is extracted the timestamp field will be compared against a delayed copy of the local timer on the slave FPGA. If a mismatch is found a timestamp error bit will be set in the outgoing serialized header data. The following diagram shows the format of the sample data frame which will be stored in the slave FPGA's memory.



15	14	13	12	11 10 9 8 7 6 5 4 3 2 1 0
0	1	S0	SP	Channel 0 Sample_Data[11:0]
0	0	S1	SP	Channel 1 Sample_Data[11:0]
0	0	S2	SP	Channel 2 Sample_Data[11:0]
0	0	S3	SP	Channel 3 Sample_Data[11:0]
0	0	S4	SP	Channel 4 Sample_Data[11:0]
0	0	S5	SP	Channel 5 Sample_Data[11:0]
0	0	S6	SP	Channel 6 Sample_Data[11:0]
0	0	S7	SP	Channel 7 Sample_Data[11:0]
0	0	M0	SP	Channel 8 Sample_Data[11:0]
0	0	M1	SP	Channel 9 Sample_Data[11:0]
0	0	M2	SP	Channel 10 Sample_Data[11:0]
0	0	M3	SP	Channel 11 Sample_Data[11:0]
0	0	M4	SP	Channel 12 Sample_Data[11:0]
0	0	TE	SP	Channel 13 Sample_Data[11:0]
0	0	0	SP	Channel 14 Sample_Data[11:0]
0	0	HP	SP	Channel 15 Sample_Data[11:0]

HP = Header Parity, Even S[7:0] = Sample Time
TE = Timestamp Error M[4:0] = Module Address
SP = Sample Parity, Even

Stored Sample Frame

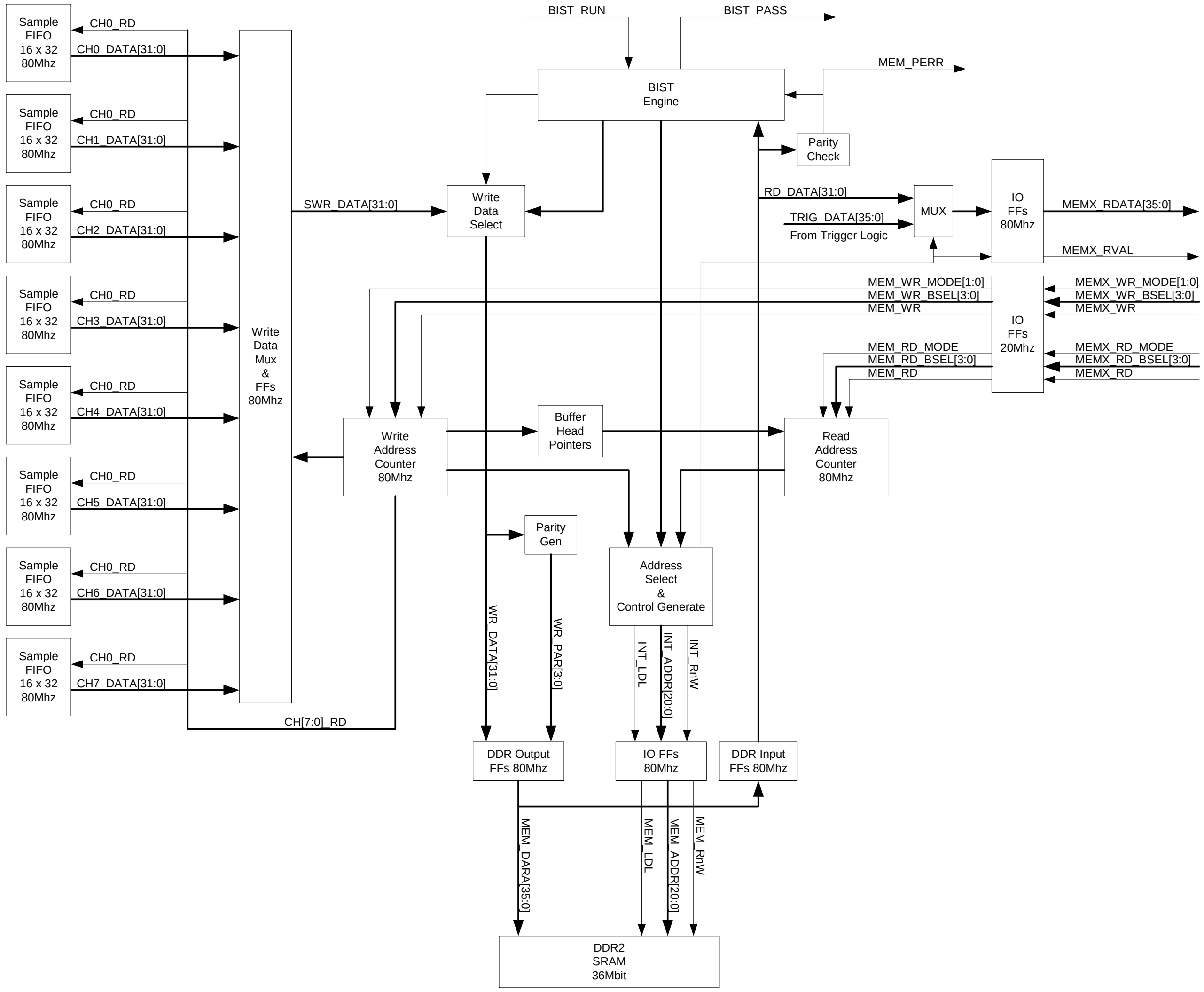
Once the sample timestamp has been compared, the TEM slave FPGA will process the sample words received in the sample frame. As each word is received it will be passed through adder logic to create a 16-bit sum of the 16 samples received in the frame. In order to ensure that a noisy channel does not cause over-triggering of the system, an enable is provided for each of the 128 channels processed by the slave FPGA. Software must be sure to adjust the trigger thresholds accordingly when removing channels from the sum equation. In addition to generating a sum of all of the individual, the local logic will also determine which of the 16 received channels has the highest magnitude. This magnitude as well as the number of the channel it is associated with will be passed on to the trigger logic.

The sample data is then written into a shallow burst FIFO which acts as the barrier between the local 20Mhz logic and sample store logic which operates at 80Mhz. Each write to the burst FIFO will be a 32-bit word generated by grouping the received samples into pairs.

7.3 Trigger Event Module Sample Store

The TEM sample store logic pulls the data from the 8 burst FIFOs and stages it for writing into the external DDR2 SRAM. A single sample frame will be read from each of the 8 burst FIFOs in starting from FIFO 0 and moving on to FIFO7. If any of the FIFOs are empty due to lack of incoming samples frames a NULL value will be substituted for the missing data.

The following diagram shows the TEM sample store logic.



TEM Sample Store

The write operation of the sample store logic is controlled directly by the TEM master FPGA. The master FPGA passes a write buffer select vector, a write buffer mode vector, and a buffer write signal to the slave FPGA. The sequence is started by setting the write buffer select to the selected buffer. During the first write pulse the write buffer mode is set to reset. This results in the head pointer being reset and the address counter in the write logic being reset. As each subsequent write to the sample store memory is performed the write buffer mode vector indicates how the head pointer should be update. The first possible buffer write mode indicates the buffer is in pre-trigger mode and has not yet stored enough samples to fill the pre-trigger buffer. During this state the address counter is incremented on each write but the head pointer value is kept at its reset value. Once the pre-trigger buffer has filled up the buffer write mode vector will change indicating that the head pointer should be incremented each time a write is performed. This results in a circular buffer where the head pointer always lags the write address value by the size of the pre-trigger buffer. Once a trigger has been detected by the master FPGA the write buffer mode vector will be updating indicating that the head pointer should now stop incrementing preserving the pre-trigger samples while storing new post-trigger samples.

Although each data sample is already protected by a sample parity bit the sample store logic will store parity bits along with the data sample. This parity will be checked upon read and a MEM_PERR signal in the slave FPGAs status register will be set if a parity error is detected.

The master FPGA starts a read sequence by setting the read buffer select to the buffer from which it wishes to read data. The read buffer mode signal will then be set to a reset state during the first read from the slave FPGA memory. This results in the read address counter being pre-set to the head pointer value for the selected buffer. Subsequent reads from the slave memory will be accompanied by a read buffer mode indicating that the address counter should be incremented on each read normally. A compile-time configuration in the FPGA firmware will allow data from unused Front End Card channels to be skipped in order to keep the trigger frame size as small as possible.

For each read strobe that the master FPGA asserts a single sample (8 32-bit words) will be read from external memory. The read data is checked for parity errors and is passed directly to the master FPGA at a rate of 80-Mhz and a width of 32-bits.

For each 1us period the external memory can process 80 read or write cycles to external memory.

Since each read or write cycle consists of a 4-byte access, the external memory has a total bandwidth of 320MBs. The first 64 clock periods (256MBs) will be dedicated to sample writes to the memory. The next 8 clock cycles (32MBs) are then dedicated to read accesses if required. The remaining 8 clock cycles (32MBs) are left idle. With three slave FPGAs the total read bandwidth allocated is 96MBs which is more data bandwidth than the PC interface can handle.

The external DDR2 SRAM is a single 18bit x 2M part which provides a total memory size of 4Mbytes. Each sample being stored will consume 2 bytes of memory space allowing 2M samples to be stored in the external memory. Each buffer partitioned in memory is required to store 2048 time slices of samples from its attached Front End Cards. Each slave has 8 Front End Cards attached each providing 16 samples per time slice. This equates to 128 samples for each time slice and 256K samples per buffer. This allows eight separate trigger event buffers to be created in the chosen memory. The design of both the master and slave FPGAs support the doubling of the external memory size and the use of 16 separate trigger event buffers.

When the system is started data will stored in sample buffer 0. Once a trigger occurs the system will continue to write to buffer 0 until 1024 additional samples have been taken. The master FPGA will then switch buffer and direct the slave FPGA to begin storing data in buffer 1. As each trigger comes the current buffer is filled with post trigger data and a new buffer is selected. If a large number of triggers are occurring and the PC can not handle them at their incoming rate all of the buffers will eventually fill. For example in high trigger rate situation the PC may be reading data from buffer 0 which the system is currently storing samples in buffer 7. Buffers 1 – 6 are filled with event data from previous triggers. If a trigger occurs at this time the system will post fill buffer 7 with 1024 samples following the trigger. Once the post fill is complete the system needs a new buffer in which to store samples. If the PC has not finished reading from buffer 0 then the system is force to flush the sample data from buffer 7 and begin writing new sample data to it. The data that was in buffer 7 is a lost trigger event. The system will increment the lost trigger counter and an empty trigger event frame will be added to the trigger queue for the last trigger.

7.4 TEM Trigger Data Frame Transmission

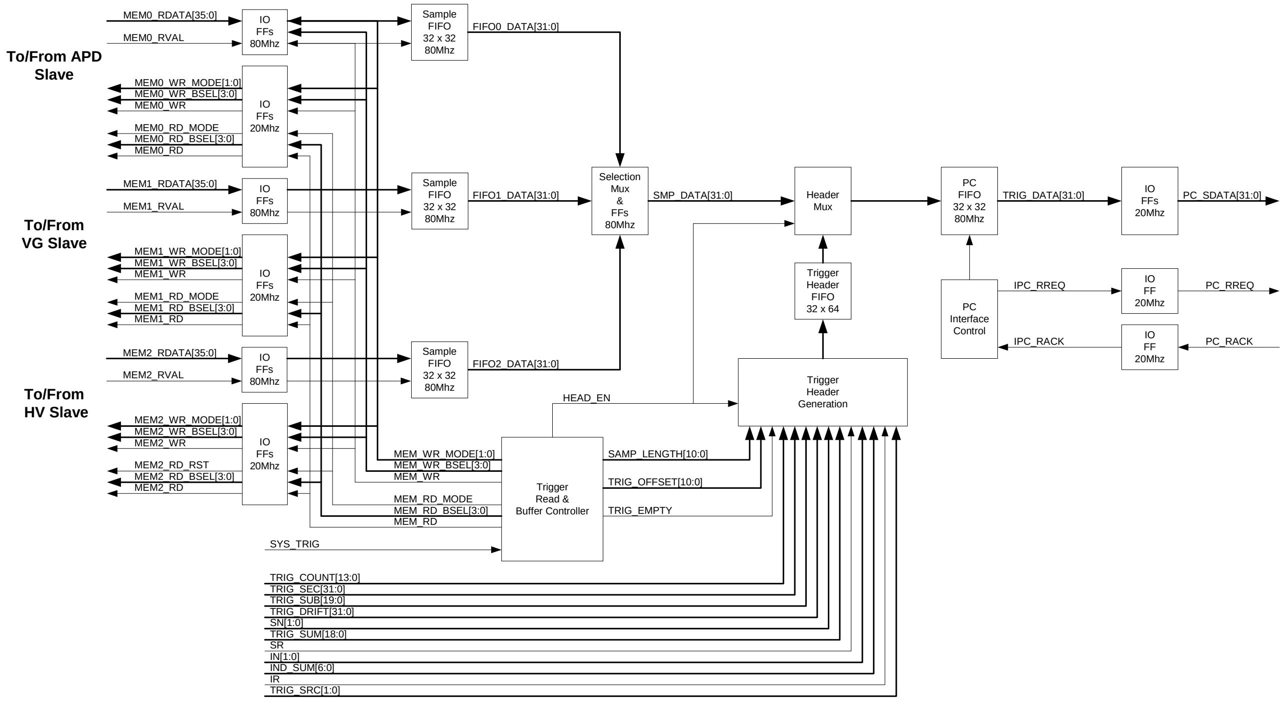
Once a trigger has been detected the trigger event header is created by the master FPGA. This trigger event header contains the following information:

- A trigger count value indicating the serial number of the trigger being generated.
- A timestamp for the trigger event including the following data:
  - 32-bit NTP Seconds timestamp.
  - 20-bit NTP Sub-Seconds timestamp.
  - 32-bit signed NTP drift indication
- The source of the trigger that was detected (APD, VG or HV).
- Sum trigger request.
- Sum trigger threshold number.
- Sum trigger value.
- Individual trigger request.
- Individual trigger threshold number.
- Individual trigger channel indication.
- An indication of the sample length (1-2048).
- An indication of a empty sample data.
- The location of the trigger sample within the sample frame. (1-2048)

Since there are 8 possible trigger buffers this trigger event header must be stored so that previous trigger events can be sent to the PC first. A trigger header FIFO is provided for storing the headers for up to 64 trigger events.

In a situation where a trigger event buffer must be re-used due to buffer exhaustion a trigger header will still be added to the trigger header FIFO. The sample length field for this header will be set to zero indicating that no associated sample data is available for this trigger. When the logic reading from the trigger header FIFO encounters this header it will know not to read data from the slave FPGA and the header will be sent by itself to the control PC.

The logic responsible for generating the trigger data frame is shown in the following diagram:



TEM Trigger Data Frame Generation

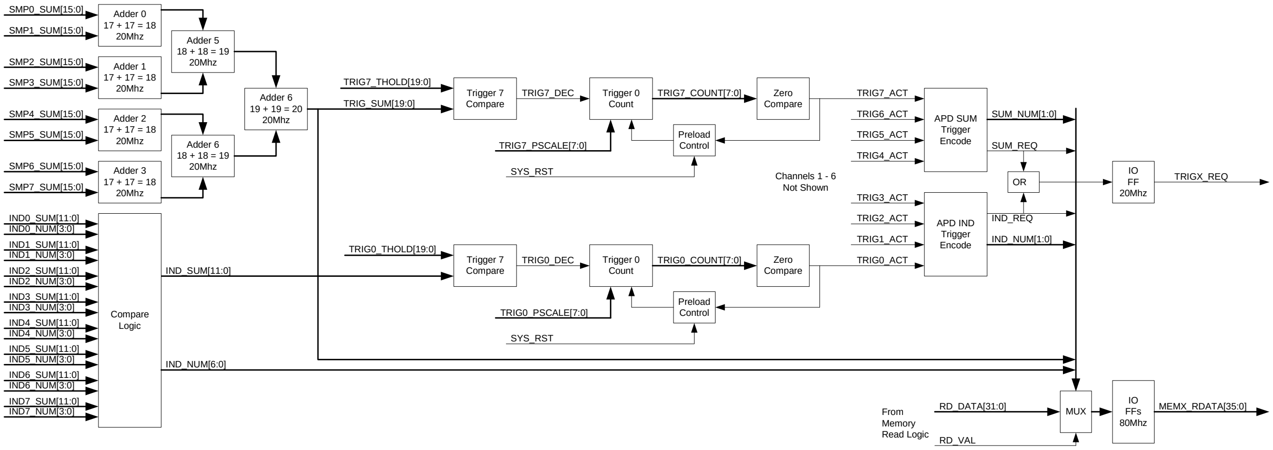
When the read logic encounters a trigger header with a non-zero sample length value it will begin the process of reading the sample data from the memories attached to the slave FPGAs. Incoming data from the three slave FPGAs is first deposited into one of three burst FIFOs. The FIFOs are then read from once the associated trigger even header has been added to the PC FIFO. The format of the trigger data frame is described in section 13.1.2.

Reads from the slave FPGAs will be controlled so that the three sample FIFOs and the pc FIFO do not fill up. These FIFOs will be as large as possible in order to help free up space in the external DDR2 SRAM memory.

8. TRIGGER PROCESSING

Each group of detectors (APD, VG or HV) in the EXO system has its own set of 8 triggers. Each trigger has its own threshold value as well as a pre-scale counter. The threshold is self explanatory and the pre-scale value determines the number of times the sum must exceed the threshold before a true trigger occurs.

The 8 triggers are divided into two groups of 4. The first 4 triggers compare a preset threshold against the channel which has the highest magnitude of the enabled channels. These four triggers are referred to as the individual trigger thresholds. The second set of 4 triggers compare a preset threshold against the sum of all of the enabled channels.



TEM Trigger Sum Compare

The logic for each of the 8 receive channels on the slave FPGA pass three values to the trigger detection logic. These values include the sum of the enabled samples received in the sample frame, the magnitude of the largest of the enabled channels and an indication of which of the 16 channels received has the highest magnitude. The sum values from each of the 8 receiving channels are added together to get a sum of all of the channels connected to the slave FPGA.

An average of the computed sum value over the past N samples is calculated. This average value is then subtracted from the current sum to generate a new value which is passed to the upper four trigger detection channels in the trigger logic. Similarly the maximum values from the 8 receive channels are passed through a comparison block to determine which of the 8 incoming values is the largest. The resulting value is then passed to the lower 4 trigger detection channels in the trigger logic. The generated trigger sum as well as the channel number of the largest magnitude sample are then passed on for inclusion in the trigger header if a trigger is detected.

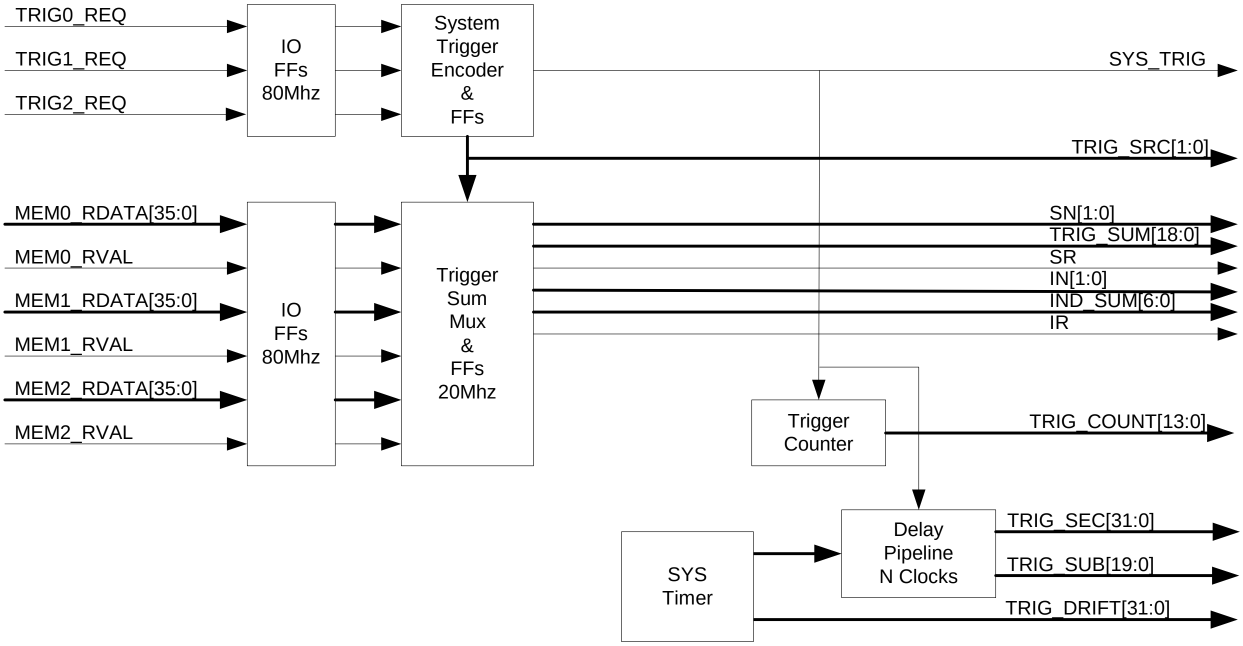
The values passed to each of the 8 trigger channels are compared against the trigger threshold for the channel. If the incoming value is greater or equal to the defined threshold a decrement flag is passed to the pre-scale counter. The pre-scale counter is pre-set to the configured value when the conversion cycle is started or when its value reaches zero. Each time the passed value exceeds the threshold a decrement flag is passed to the pre-scale counter. If the counter value is zero when the decrement flag is passed the trigger output for that particular channel is asserted. The pre-scale counter is then re-loaded with the pre-scale value. If the count was not already zero the current count value will be reduced by 1. In order to keep the pre-scale counter from decrementing multiple times in a single event a pre-scale hold off setting defines the number of samples to skip before the next pre-scale decrement is allowed. If any of the pre-scale counters in a group of 4 triggers is decremented then the other three are held off for this defined time period as well.

The outputs of the lower 4 trigger channels are passed to an encoder to determine if an individual trigger is occurring. If a trigger is detected then its number is encoded in a value to determine which of the 4 trigger thresholds was exceeded. In addition a request is output which is passed directly to the external trigger request line in addition to being included in the trigger data. Similarly the upper 4 trigger channels have a similar encoder with an output request and a trigger number value.

The outputs of the two encoders are or’ed together to generate a trigger request signal to the master FPGA. In addition to sending out the request to the master FPGA, the slave FPGA also passes trigger information that will be added to a trigger header if a local trigger is asserted. This data is passed over the MEM_RDATA lines any time read data is not being passed from the slave FPGA memory to the master FPGA. This passed trigger data includes the sum value of all enabled channels, the channel number which contains the highest sample value, the request signal from each of the two trigger groups as well as the trigger number from each of the two trigger groups.

It is important to note that only one of the 4 trigger thresholds in each group can be active at any given time. For example if a value is passed which exceeds two of the 4 trigger thresholds then only the highest number of the two will be enabled. If the value exceeds trigger 0 and 1 then only the pre-scale counter on trigger channel 1 is decremented. Software should configure the triggers so that the threshold value increases as the trigger channel number increases.

The trigger outputs from the three slave FPGAs are passed to the trigger logic in the master FPGA shown below.



Master Trigger Logic

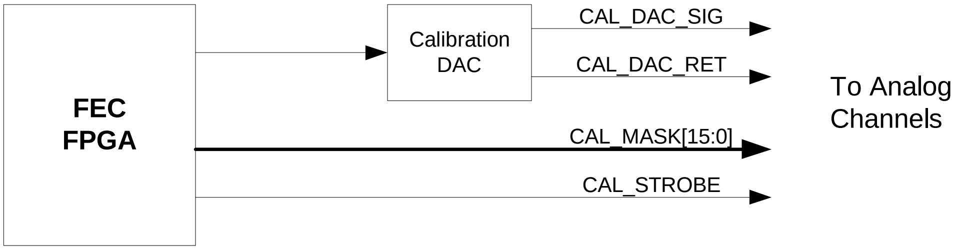
The trigger outputs from the three slave FPGAs are connected to a system trigger encoder. This encoder determines if any of the three are active and chooses one as the active trigger. If any of the values are active the post trigger counter will be started and the source of the trigger will be registered in the TRIG_SRC value. In addition the associated trigger data passed over the MEM_RDATA lines will also be registered. If more than module has an active trigger the lowest source will be encoded.

Once a trigger is detected up to 1024 samples will be stored (including the trigger sample) before the data is passed to the PC. In some cases a new trigger may appear before all 1024 samples are stored. A software settable dead time counter determines the minimum number samples that must be taken after a trigger before another trigger can be detected. Once this set number of samples is stored triggers are re-enabled. A new trigger will result in the data no longer being stored in the current buffer. The new trigger sample and its subsequent data will be stored in the next available buffer.

The TEM master FPGA has the ability to override the trigger event logic and generate a fake trigger event every 5 seconds. This feature is enabled by setting the TEST_TRIG bit in the conversion control register. In addition software can force a single trigger to occur immediately by setting the FORCE_TRIG bit in the conversion control register.

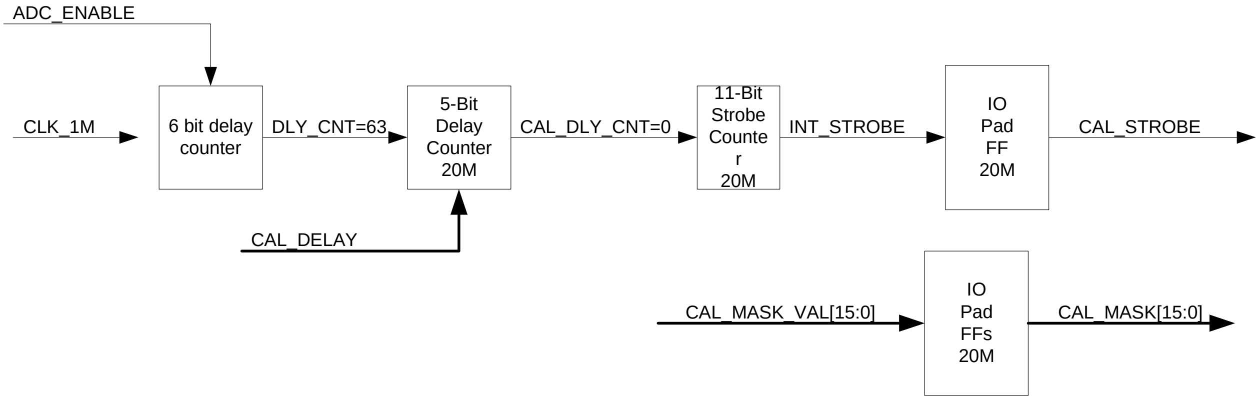
9. CALIBRATION

The Front End Card has the ability to inject a calibration strobe into each of the 16 front end analog channels. This calibration pulse is generated using a fixed voltage output of a digital to analog converter (DAC), a mask bit for each analog channel and a single strobe bit as shown below.



Calibration Logic

The calibration cycle is started by first setting a fixed DC output on the calibration DAC. This output value is set by writing a value to the Calibration Data Register. Next the 16-bit mask value will be written to determine which channels should receive the calibration charge. A delay count is then set in the Calibration Delay Register to determine the number of 20 MHz (50us) clock periods to delay before the calibration pulse is asserted.



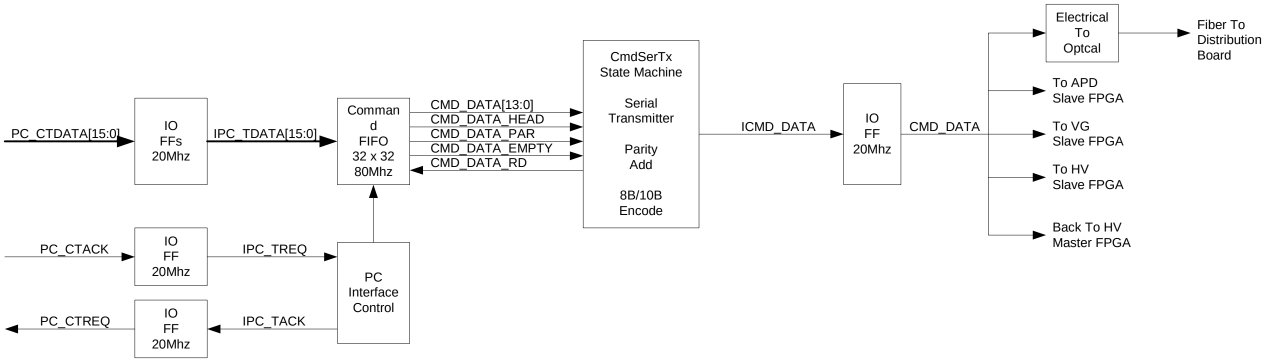
Calibration Strobe Generation

Once the calibration logic is triggered a calibration strobe will be generated N 1Mhz cycles after the conversion logic is enabled, where N is configured in the 1Mhz delay setting register. If the delay count is set to 0, the calibration strobe will immediately follow the 1Mhz clock. Otherwise the delay count will control the number of 20Mhz clocks to wait after the 1Mhz clock is received. The strobe counter will then keep the pulse asserted for 100us.

10. COMMAND & RESPONSE

Command frames are sent from the control PC to the various FPGAs in the EXO system. A 16-bit interface exists between the control PC and the TEM master FPGA for passing this command data. The 16-bit data from the PC will be loaded into a command FIFO before being passed to a command serializer. This command serializer will convert the 16-bit data into a 8B/10B encoded serial stream that is processed by the receive logic in the various FPGAs. The 8B/10B encoding is required to keep a proper one's density on the fiber optic link.

The command data transmission logic is shown below.



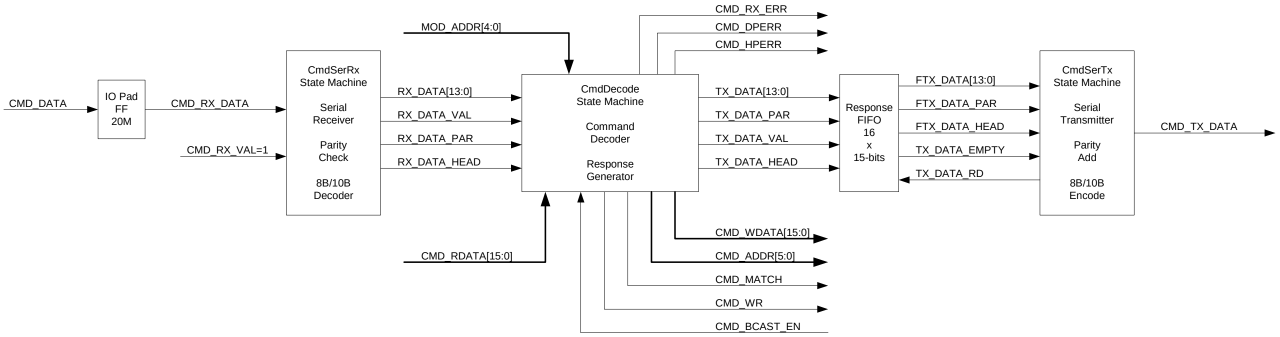
TEM Command Frame Generation

The serial command data is passed to an optical module for transmission down to the 20 Front End Cards. The serial command data is also presented to the local slave and master FPGAs on the TEM.

In order to convert the serial data back into a 1 16-bit parallel word the serial transmitter will send a special comma character whenever the link is idle which is used detect the 16-bit alignment. This special comma character is a special sequence that can never be detected on the serial link for any combination of the other 8B10B characters.

The serial data on the command link is passed to a serial receiver. This receiver detects the special command characters and converts the serial stream into parallel words. These parallel words are passed through an 8B/10B decoder then along to the command decoder. The serial receiver outputs a 14-bit data word, a data valid indication, a data header indication and a parity bit.

The logic used to process the received command data in each FPGA is shown below.



FEC Command Frame Reception & Processing

The parallel data is passed to a command decoder block. The command decoder will extract the received read or write command and performs the necessary local action. Once the header word is received the command decoder checks to see if the command is a read or write command and extracts the register address. If the command is a write it will wait for the two data payload words that are to follow. If the command is a read it will attempt to process the command immediately.

The address in the command header is checked to see if it matches the local module address or if it is a broadcast address. If the address is local the match signal will be asserted by the command decoder. If the address in the header is a broadcast the command decoder will wait for the broadcast enable flag to be asserted by external logic. The external logic will generate this broadcast enable bit using the address that is output by the command decoder.

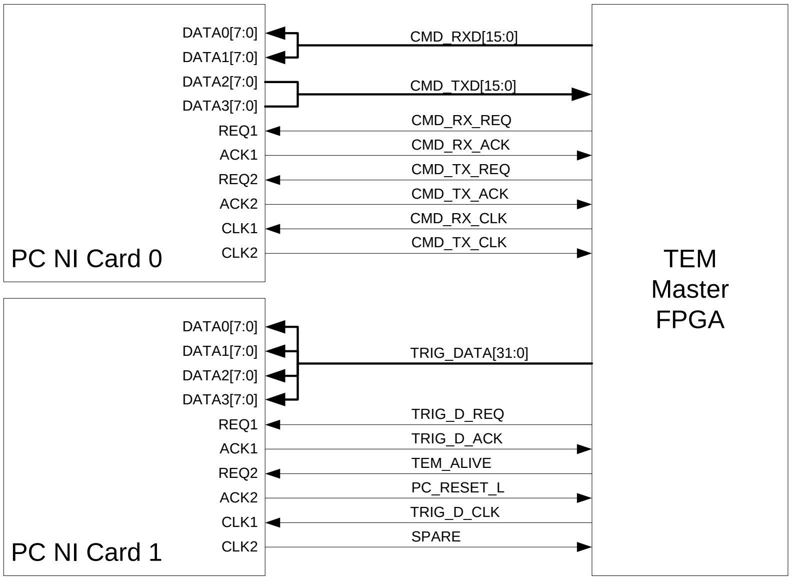
If a header parity error or data parity error is detected by the command decoder the command will be terminated

The command decoder will then generate a command response frame for transmission back to the control PC. The three word response frame will be written to a response FIFO. A serializer block will read each command word from the response FIFO and send it out as an 8B/10B encoded serial stream of 16-bits.

Command response traffic received by the TEM master FPGA is queue for transmission to the control PC over a dedicated 16-bit interface. At each point where more two or more response channels are funneled together a simple round robin arbiter is used to store the response traffic in a local FIFO before it is sent onward.

11. PC INTERFACE

The connection between the National Instruments cards and the TEM master FPGA is shown below.



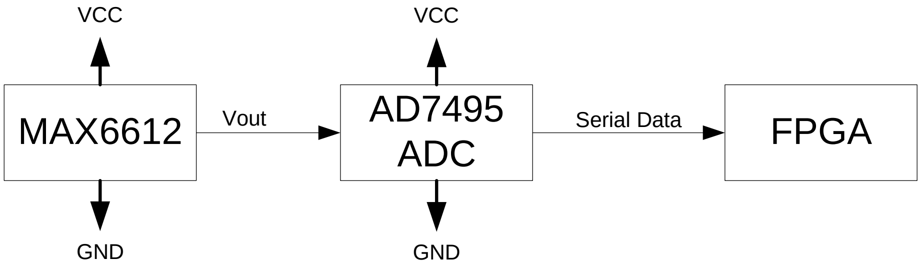
PC Interface Connections

The first NI card is configured into two groups. Group 0 will serve as the input of response data back from the TEM. Group 0 shall be configured in burst mode with reverse clocking as it will receive a clock from the TEM along with data. Group 1 on NI card 0 will serve as the output of command data to the TEM. It shall be configured in burst mode with reverse clocking as it will send a clock to the TEM along with the command data. The REQ & ACK signals of both groups shall be configured as active high.

The second NI card will be configured as a single 32-bit group. This single group will receive trigger data frames from the TEM. It shall be configured in burst mode with reverse clocking as it will receive a clock from the TEM along with data. REQ1 & ACK1 shall be assigned to the group and configured as active high signal. ACK2 is unused. REQ2. CLK2 is reserved for future use.

12. ENVIRONMENTAL MONITORING

Both the Front End Cards and the Trigger Event Module have the ability to sample both the ambient temperature and their input voltage. Each Front End Card has its own temperature sensor allowing the current heat distribution in the Front End Assembly to be characterized. The TEM contains an identical temperature monitoring circuit to detect an overhead condition withing the TEM assembly. The following diagram describes the circuit used to pass the temperature data back to the PC.



Temperature Sensor

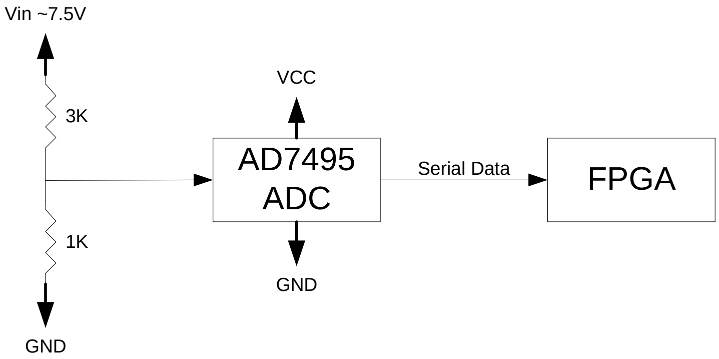
The MAX6612 device outputs a voltage that is proportional to its current die temperature. Knowing the outputs voltage, the current temperature can be calculated with the following equation:

$$T\ (C) = (V - 0.4) / 0.01953$$

The output of this device is passed to an AD7495 analog to digital converter. This device will sample the output voltage of the MAX6612 and send it in serial format to the local FPGA. There it is converted into a parallel value which can be read by the PC in the “Temperature ADC Register”. The AD7495 is a 12-bit ADC with an internal reference voltage of 2.5V. The PC will compute the current temperature from the 12-bit ADC value (D) with the following equation:

$$T = (((D/4095) * 2.5) - 0.4) / 0.01953$$

The input voltage monitoring circuit is very similar to the temperature circuit described above. The input voltage is passed through a voltage divider and then sampled by an AD7495 ADC. The circuit used to sample the 7.5V input voltage to the Front End Cards is shown below.

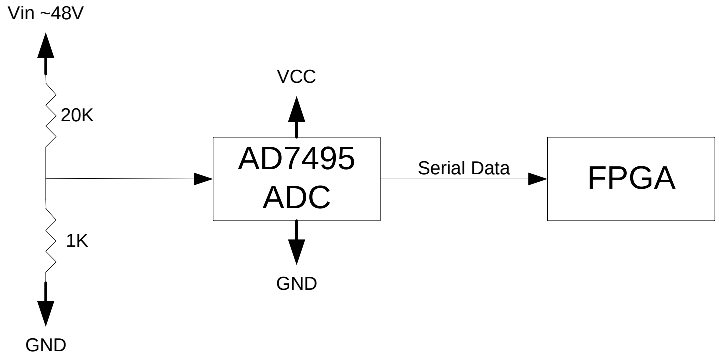


Front End Card Voltage Sensor

The sampled voltage is converted into a 12-bit value which can be read by the PC in the “Power ADC Register”. The AD7495 is a 12-bit ADC with an internal reference voltage of 2.5V. The PC will compute the input voltage from the 12-bit ADC value (D) with the following equation:

$$V = ((D/4095) * 2.5) * 4$$

The circuit used to sample the 48V input voltage to the Trigger Event Module is shown below.



Trigger Event Module Voltage Sensor

The PC will compute the input voltage from the 12-bit ADC value (D) with the following equation:

$$V = ((D/4095) * 2.5) * 21$$

13. FRAME FORMATS

Frames sent to and from the EXO electronics include command frames, response frames & trigger event frames. Command frames are sent from the control PC to the various FPGAs in the system, response frames are sent from the FPGAs to the control PC and trigger event frames are sent from the TEM to the control PC.

13.1 Command Frames

Command frames in the EXO system are either a write command frame or a read command frame. The two frame types both support a common frame header format that consists of a header parity bit, a read/write flag, the module address and a register address. The format of a write command frame in the EXO system is shown below.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DS Data Header	1	P	RW = 1	0	RA6	Module Address[4:0]					Register Address[5:0]					
MSB Write Data	0	P	Write_Data[27:22]						Write_Data[15:8]							
LSB Write Data	0	P	Write_Data[21:16]						Write_Data[7:0]							

P = Parity Bit, Even RW = Read / Write Flag

RA6 = Register Address[6]

Write Command Frame

The common header has its RW bit set to ‘1’ indicating a write frame. The 16-bit value being written is spit between the two payload words as shown in the above figure. Each half of the value being written is protected with a single even parity bit.

Read frames are somewhat simpler in that they only contain the address of the module and register being accessed. The read frame is shown in the following figure.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DS Data Header	1	P	RW = 0	0	RA6	Module Address[4:0]					Register Address[5:0]					

H = Parity Bit, Even RA6 = Register Address[6]

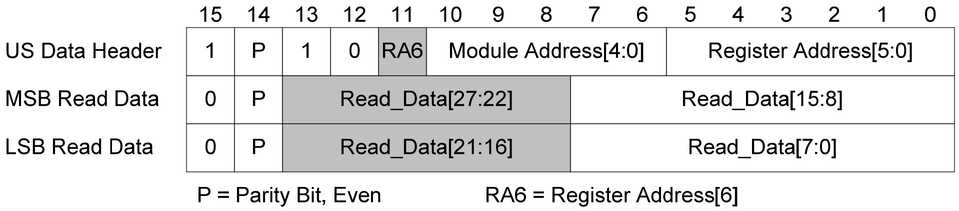
Read Command Frame

The read frame header is identical to the write frame header except that the RW bit is set to ‘0’ to indicate a read frame. No payload data is present in the read frame.

Command frames are sent between the Trigger/Event Module and the Front End Card over a dedicated Fiber link using a simple serial protocol. This protocol includes a single start bit followed by a 16-bit word. Data sent in the downstream direction is transmitted at 20Mbs.

13.1.1 Command Response Frame

All read commands are followed by a response frame sent from the destination module back to the control PC. For all read commands a response frame will be sent containing the contents of the register being read. The format of the response frame is shown in the following figure.



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
US Data Header	1	P	1	0	RA6	Module Address[4:0]					Register Address[5:0]					
MSB Read Data	0	P	Read_Data[27:22]						Read_Data[15:8]							
LSB Read Data	0	P	Read_Data[21:16]						Read_Data[7:0]							

P = Parity Bit, Even RA6 = Register Address[6]

Response Frame Format

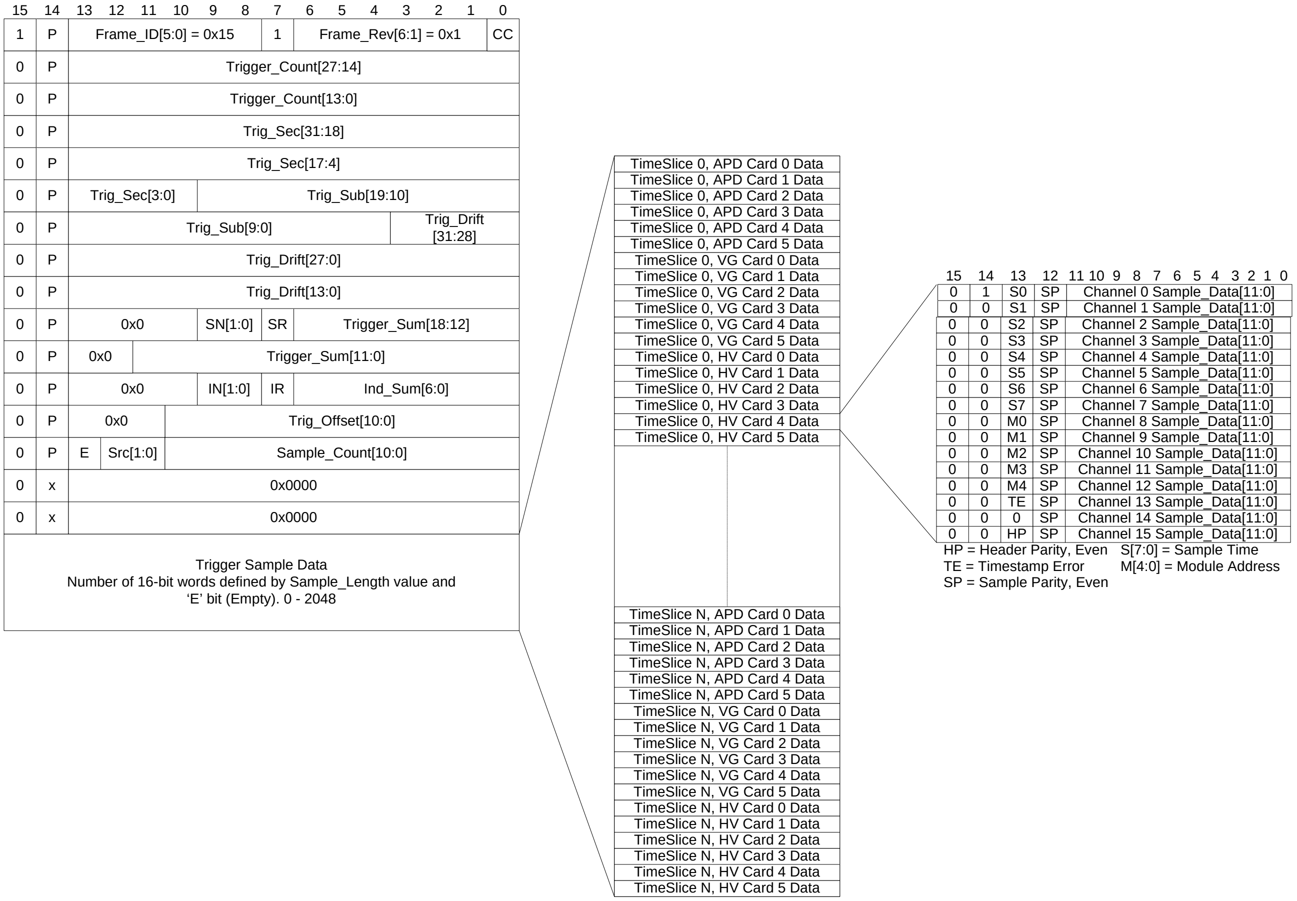
The module address field contains the address of the module that is sending the response frame. The module address field for a response frame will never contain a broadcast address. The register address is an exact copy of the address contained in the original command which triggered this response.

Upstream response frames are sent between the Front End Card and the Trigger/Event Module are sent serially using a stolen bit from the link between the two systems. The data rate for the response link is 20Mbs.

13.1.2 Trigger Data Frame

The trigger data frame is used to transfer all the data associated with a trigger event to the control

PC. The trigger data frame is a larger frame that contains the trigger data followed by the sample data sent up by the front end cards. The following diagram shows the format of the trigger data frame.



<table>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>1</td><td>P</td><td colspan="6">Frame_ID[5:0] = 0x15</td><td>1</td><td colspan="6">Frame_Rev[6:1] = 0x1</td><td>CC</td></tr>
<tr><td>0</td><td>P</td><td colspan="14">Trigger_Count[27:14]</td></tr>
<tr><td>0</td><td>P</td><td colspan="14">Trigger_Count[13:0]</td></tr>
<tr><td>0</td><td>P</td><td colspan="14">Trig_Sec[31:18]</td></tr>
<tr><td>0</td><td>P</td><td colspan="14">Trig_Sec[17:4]</td></tr>
<tr><td>0</td><td>P</td><td colspan="4">Trig_Sec[3:0]</td><td colspan="10">Trig_Sub[19:10]</td></tr>
<tr><td>0</td><td>P</td><td colspan="10">Trig_Sub[9:0]</td><td colspan="4">Trig_Drift [31:28]</td></tr>
<tr><td>0</td><td>P</td><td colspan="14">Trig_Drift[27:0]</td></tr>
<tr><td>0</td><td>P</td><td colspan="14">Trig_Drift[13:0]</td></tr>
<tr><td>0</td><td>P</td><td colspan="4">0x0</td><td colspan="2">SN[1:0]</td><td>SR</td><td colspan="7">Trigger_Sum[18:12]</td></tr>
<tr><td>0</td><td>P</td><td colspan="2">0x0</td><td colspan="12">Trigger_Sum[11:0]</td></tr>
<tr><td>0</td><td>P</td><td colspan="4">0x0</td><td colspan="2">IN[1:0]</td><td>IR</td><td colspan="7">Ind_Sum[6:0]</td></tr>
<tr><td>0</td><td>P</td><td colspan="3">0x0</td><td colspan="11">Trig_Offset[10:0]</td></tr>
<tr><td>0</td><td>P</td><td>E</td><td colspan="2">Src[1:0]</td><td colspan="11">Sample_Count[10:0]</td></tr>
<tr><td>0</td><td>x</td><td colspan="14">0x0000</td></tr>
<tr><td>0</td><td>x</td><td colspan="14">0x0000</td></tr>
<tr><td colspan="16">Trigger Sample Data
Number of 16-bit words defined by Sample_Length value and
‘E’ bit (Empty). 0 - 2048</td></tr>
</table>

TimeSlice 0, APD Card 0 Data
TimeSlice 0, APD Card 1 Data
TimeSlice 0, APD Card 2 Data
TimeSlice 0, APD Card 3 Data
TimeSlice 0, APD Card 4 Data
TimeSlice 0, APD Card 5 Data
TimeSlice 0, VG Card 0 Data
TimeSlice 0, VG Card 1 Data
TimeSlice 0, VG Card 2 Data
TimeSlice 0, VG Card 3 Data
TimeSlice 0, VG Card 4 Data
TimeSlice 0, VG Card 5 Data
TimeSlice 0, HV Card 0 Data
TimeSlice 0, HV Card 1 Data
TimeSlice 0, HV Card 2 Data
TimeSlice 0, HV Card 3 Data
TimeSlice 0, HV Card 4 Data
TimeSlice 0, HV Card 5 Data
TimeSlice N, APD Card 0 Data
TimeSlice N, APD Card 1 Data
TimeSlice N, APD Card 2 Data
TimeSlice N, APD Card 3 Data
TimeSlice N, APD Card 4 Data
TimeSlice N, APD Card 5 Data
TimeSlice N, VG Card 0 Data
TimeSlice N, VG Card 1 Data
TimeSlice N, VG Card 2 Data
TimeSlice N, VG Card 3 Data
TimeSlice N, VG Card 4 Data
TimeSlice N, VG Card 5 Data
TimeSlice N, HV Card 0 Data
TimeSlice N, HV Card 1 Data
TimeSlice N, HV Card 2 Data
TimeSlice N, HV Card 3 Data
TimeSlice N, HV Card 4 Data
TimeSlice N, HV Card 5 Data

<table>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td>1</td><td>S0</td><td>SP</td><td colspan="12">Channel 0 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>S1</td><td>SP</td><td colspan="12">Channel 1 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>S2</td><td>SP</td><td colspan="12">Channel 2 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>S3</td><td>SP</td><td colspan="12">Channel 3 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>S4</td><td>SP</td><td colspan="12">Channel 4 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>S5</td><td>SP</td><td colspan="12">Channel 5 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>S6</td><td>SP</td><td colspan="12">Channel 6 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>S7</td><td>SP</td><td colspan="12">Channel 7 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>M0</td><td>SP</td><td colspan="12">Channel 8 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>M1</td><td>SP</td><td colspan="12">Channel 9 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>M2</td><td>SP</td><td colspan="12">Channel 10 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>M3</td><td>SP</td><td colspan="12">Channel 11 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>M4</td><td>SP</td><td colspan="12">Channel 12 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>TE</td><td>SP</td><td colspan="12">Channel 13 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>SP</td><td colspan="12">Channel 14 Sample_Data[11:0]</td></tr>
<tr><td>0</td><td>0</td><td>HP</td><td>SP</td><td colspan="12">Channel 15 Sample_Data[11:0]</td></tr>
</table>

HP = Header Parity, Even S[7:0] = Sample Time
TE = Timestamp Error M[4:0] = Module Address
SP = Sample Parity, Even

Trigger Data Frame

The trigger data frame contains a varying number of data samples before and after the trigger sample. At most the trigger data frame will contain 2047 samples. The number of samples contained in the trigger data frame is defined by the Sample_Count[10:0] value contained in the trigger event header. The offset of the trigger sample within the frame is defined by the Trig_Offset[10:0] value in the trigger event header.

13.1.3 Veto Frame

The Veto frame is used to transfer veto information to the control PC.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	HP	Veto_Frame_ID[5:0] = 0x33						1	Veto_Frame_Rev[6:0] = 0x01						
0	0	0x0000													
0	1	P	Veto_Sec[31:19]												
0	0	P	Veto_Sec[18:6]												
0	0	P	Veto_Sec[5:0]						Veto_Sub[19:13]						
0	0	P	Veto_Sub[12:0]												
0	0	P	Veto_Mask[63:51]												
0	0	P	Veto_Mask[50:38]												
0	0	P	Veto_Mask[37:25]												
0	0	P	Veto_Mask[24:12]												
0	0	P	Veto_Mask[11:0]												TE
0	0	P	Veto_Serial[12:0]												
0	0	0x0000													
0	0	0x0000													
0	0	0x0000													
0	0	0x0000													

Veto Frame

14. EXO ADDRESS MAP

The following table defines the module address for the various FPGAs in the EXO system.

Address	Module
0x00	APD Card 0
0x01	APD Card 1
0x02	APD Card 2
0x03	APD Card 3
0x04	APD Card 4
0x05	APD Card 5
0x06 – 0x07	Unused
0x08	VG Card 0
0x09	VG Card 1
0x0A	VG Card 2
0x0B	VG Card 3
0x0C	VG Card 4
0x0D	VG Card 5
0x0E – 0x0F	Unused
0x10	HV Card 0
0x11	HV Card 1
0x12	HV Card 2
0x13	HV Card 3
0x14	HV Card 4
0x15	HV Card 5
0x16	Unused
0x17	Calibration Board
0x18	High Voltage Board 0
0x19	High Voltage Board 1
0x1A	Veto Board
0x1B	TEM APD Slave FPGA
0x1C	TEM VG Slave FPGA
0x1D	TEM HV Slave FPGA
0x1E	TEM Master FPGA
0x1F	Broadcast Address

EXO System Module Address Map

14.1 EXO Register Map

The following table describes the register map for the EXO system. Each FPGA in the system supports a subset of the registers defined below. The following table describes the overall register map as well as which FPGAs support each register. An FPGA will only respond to broadcast commands for registers that it supports.

Address	FEC	HV	Veto	Cal	Slave	Master	Name
0x00	Yes	Yes	Yes	Yes	Yes	Yes	Reset/Version Register
0x01	Yes	Yes	Yes	Yes	Yes	Yes	System Status Register
0x02	No	No	No	No	Yes	No	Slave Status Register
0x03	No	Yes	No	No	No	No	HV Status Register
0x04	Yes	Yes	Yes	Yes	Yes	Yes	System Timer Register
0x05	No	No	No	No	No	Yes	DAQ Ready Register
0x06	No	No	No	No	No	Yes	System Clock Register 0
0x07	No	No	No	No	No	Yes	System Clock Register 1
0x08	Yes	No	No	Yes	No	No	Calibration Data Register
0x09	Yes	No	No	No	No	No	Calibration Mask Register
0x0A	Yes	No	No	Yes	No	No	Calibration Delay Register
0x0B	No	No	No	No	Yes	No	Memory Error Address
0x0C	Yes	No	Yes	Yes	Yes	Yes	Conversion Control Register
0x0D	No	No	No	No	Yes	No	Memory Control Register
0x0E	No	No	No	No	Yes	No	Memory Error Data 0
0x0F	No	No	No	No	Yes	No	Memory Error Data 1
0x10	No	No	No	No	No	Yes	Trigger Count Register
0x11	No	No	No	No	No	Yes	Trigger Drop Count Register
0x12	No	No	No	No	Yes	No	Trigger Background Count Register
0x13	No	No	No	No	No	Yes	Trigger Dead Time Count Register
0x14	No	No	No	No	Yes	No	Trigger Rise Time Register
0x15	No	No	No	No	Yes	No	Trigger Hysteresis Register
0x16-0x1D	No	No	No	No	No	No	Unused
0x1E	Yes	Yes	Yes	No	No	Yes	Power In ADC Register
0x1F	Yes	Yes	Yes	No	No	Yes	Temperature ADC Register
0x20	No	No	No	No	Yes	No	Individual Trigger 0 Threshold Register
0x21	No	No	No	No	Yes	No	Individual Trigger 0 Pre-Scale Register
0x22	No	No	No	No	Yes	No	Individual Trigger 1 Threshold Register

0x23	No	No	No	No	Yes	No	Individual Trigger 1 Pre-Scale Register
0x24	No	No	No	No	Yes	No	Individual Trigger 2 Threshold Register
0x25	No	No	No	No	Yes	No	Individual Trigger 2 Pre-Scale Register
0x26	No	No	No	No	Yes	No	Individual Trigger 3 Threshold Register
0x27	No	No	No	No	Yes	No	Individual Trigger 3 Pre-Scale Register
0x28	No	No	No	No	Yes	No	Sum Trigger 0 Threshold Register
0x29	No	No	No	No	Yes	No	Sum Trigger 0 Pre-Scale Register
0x2A	No	No	No	No	Yes	No	Sum Trigger 0 Threshold Register
0x2B	No	No	No	No	Yes	No	Sum Trigger 0 Pre-Scale Register
0x2C	No	No	No	No	Yes	No	Sum Trigger 0 Threshold Register
0x2D	No	No	No	No	Yes	No	Sum Trigger 0 Pre-Scale Register
0x2E	No	No	No	No	Yes	No	Sum Trigger 0 Threshold Register
0x2F	No	No	No	No	Yes	No	Sum Trigger 0 Pre-Scale Register
0x30	No	No	No	No	Yes	No	FEC 0 Sum Disable Register
0x31	No	No	No	No	Yes	No	FEC 1 Sum Disable Register
0x32	No	No	No	No	Yes	No	FEC 2 Sum Disable Register
0x33	No	No	No	No	Yes	No	FEC 3 Sum Disable Register
0x34	No	No	No	No	Yes	No	FEC 4 Sum Disable Register
0x35	No	No	No	No	Yes	No	FEC 5 Sum Disable Register
0x36	No	No	No	No	Yes	No	FEC Enable Register
0x37	No	No	No	No	No	Yes	Trigger FIFO Register
0x38	No	Yes	No	No	No	No	4KV Control DAC
0x39	No	Yes	No	No	No	No	2KV Control DAC
0x3A	No	Yes	No	No	No	No	APD0 200V Control DAC Register
0x3B	No	Yes	No	No	No	No	APD1 200V Control DAC Register
0x3C	No	Yes	No	No	No	No	APD2 200V Control DAC Register
0x3D	No	Yes	No	No	No	No	APD3 200V Control DAC Register
0x3E	No	Yes	No	No	No	No	APD4 200V Control DAC Register
0x3F	No	Yes	No	No	No	No	APD5 200V Control DAC Register
0x40	No	No	Yes	No	No	No	Veto Enable 0 Register
0x41	No	No	Yes	No	No	No	Veto Enable 1 Register
0x42	No	No	Yes	No	No	No	Veto Enable 2 Register
0x43	No	No	Yes	No	No	No	Veto Enable 3 Register
0x44 – 0x4F	No	No	No	No	No	No	Unused
0x50	No	No	No	Yes	No	No	Calibration Card Relay Enable
0x51 – 0x5F	No	No	No	No	No	No	Unused
0x60	No	Yes	No	No	No	No	4KV ADC Value

0x61	No	Yes	No	No	No	No	2KV ADC Value
0x62	No	Yes	No	No	No	No	APD0 ADC 0 Value
0x63	No	Yes	No	No	No	No	APD0 ADC 1 Value
0x64	No	Yes	No	No	No	No	APD1 ADC 0 Value
0x65	No	Yes	No	No	No	No	APD1 ADC 1 Value
0x66	No	Yes	No	No	No	No	APD2 ADC 0 Value
0x67	No	Yes	No	No	No	No	APD2 ADC 1 Value
0x68	No	Yes	No	No	No	No	APD3 ADC 0 Value
0x69	No	Yes	No	No	No	No	APD3 ADC 1 Value
0x6A	No	Yes	No	No	No	No	APD4 ADC 0 Value
0x6B	No	Yes	No	No	No	No	APD4 ADC 1 Value
0x6C	No	Yes	No	No	No	No	APD5 ADC 0 Value
0x6D	No	Yes	No	No	No	No	APD5 ADC 1 Value
0x6E – 0x7F	No	No	No	No	No	No	Unused

EXO System Register Map

14.2 EXO Register Descriptions

14.2.1 0x00 – Reset Register

This register has no real format. Writes to this register initiate a reset of the addressed module(s). All internal timers, counters, status bits and state machines are reset.

Reads from this register return the version ID from the selected FPGA. The format of the 16-bit version ID is shown below:

Bit(s)	Name	Description
15:12	FPGA_TYPE	Type of FPGA. 0x1 = Calibration Card 0x2 = Veto Card 0x3 = High Voltage Board 0x4 = Front End FPGA 0x8 = TEM Slave 0xC = TEM Master
11:4	FPGA_REV	Revision.
3:0	FPGA_BLD_ID	Build ID. This value is modified anytime a new FPGA is built

0x00 – Reset Register

14.2.2 0x01 – System Status Register

This register is used to read the status of the addressed FPGA(s). Reads from this address return the data as described below. A write to this address sets the register to its default state. The data contained in the write is ignored.

Bit(s)	Name	Description
15:8	CMD_COUNT[7:0]	The number of command frames addressed to this FPGA, including broadcast commands and this register read. Saturating counter, Default = 0.
7:4	RSVD	Reserved, read as 0.
3	TX_NLOCK	The serial transmitter TX_READY bit has transitioned to a low state since last clear. Sticky Bit, default = 0. (FEC & Veto FPGAs Only, Read as zero for other FPGAs)
2	CMD_HPERR	A header parity error has been detected in a command frame since the last clear. Sticky bit, default = 0.
1	CMD_DPERR	A data parity error has been detected in a command frame since the last clear. Sticky bit, default = 0.
0	RX_ERROR	A serial protocol error has been detected by the serial receiver since the last clear. Sticky bit, default = 0.

0x01 – System Status Register

14.2.3 0x02 – Slave Status Register

This register is used to read the status of the addressed FPGA(s). This register contains bits that are specific to the TEM Slave FPGAs. Reads from this address return the data as described below. A write to this address sets the register to its default state. The data contained in the write is ignored.

The LINK_ERR bits are used to indicate if an error has occurred on the receive link from the front end card. If any of the SMP_PERR, TM_ERROR, or FEC_ERR errors occur on a link the link's error bit will be set.

Bit(s)	Name	Description
15	HV_ERR	An HV Link Error Has Occurred. APD & HV FPGAs
14	VETO_ERR	A VETO Link Error Has Occurred. APD FPGA.
13:8	LINK_ERR[5:0]	Link error status for Front End Cards.
7	Reserved	Unused, read as 0.
6	LINK_ERR	FEC or Veto Link Error.
5	SD_ERR	FEC or Veto Signal Detect Error.

4	SLIP_ERR	FEC or Veto Slip Error.
3	RDY_ERR	FEC or Veto Ready Error.
2	RX_ERR	Fec or Veto Receive Error.
1	SMP_PERR	A frame parity error has occurred on a FEC link.Default = 0.
0	TM_ERROR	Sample time mismatch has occurred on a FEC link. Default = 0.

0x02 – Slave Status Register

14.2.4 0x03 – High Voltage Status Register

This register is used to read the status of the addressed High Voltage board(s).

Bit(s)	Name	Description
15:13	Reserved	Unused, read as 0.
12	V12_PRES	12V Input Is Present.
11:0	V12_ADC	Level of 12V input. 0xFFF = 15V

0x03 – HV Status Register

14.2.5 0x04 – System Timer Register

This register contains the system timer. A read from this register will return the current timer value. A write to this register sets the SYS_TIME value to 0x0000 and resets the alignment of the 1Mhz system clock to the 20Mhz system clock. The resolution of this timer is 1us.

Bit(s)	Name	Description
19:0	SYS_TIME[19:0]	Current value of system timer. Default = 0.

0x04 – System Timer Register

14.2.6 0x05 – DAQ Ready Register

This register is used as a software watchdog. Software must refresh the count in this value periodically to ensure it never reaches zero. A DAQ_READY hardware line is asserted whenenver this counter is non-zero.

Bit(s)	Name	Description
27:0	DAQ_RTIME[27:0]	DAQ Ready Timer Refresh Value. Default = 0.

0x05 – DAQ Ready Register

14.2.7 0x06 – System Clock Register 0

This register serves two purposes. On read it returns the lower 16 bits of the current timer drift counter. The drift counter tracks the drift between the TEM oscillator and the NTP time client on the PC. On writes this register is used to update the NTP time value inside the TEM. The set of the NTP value does not take place until a write occurs to System Clock Register 1. Write the lower 16 bits of the NTP time value to this register.

Bit(s)	Name	Description
15:0	NTP_DRIFT[15:0] or NTP_TIME[15:0]	Lower 16 bits of drift value on read or lower 16 bits of NTP time value on write.

0x06 – System Clock Register 0

14.2.8 0x07 – System Clock Register 1

This register serves two purposes. On read it returns the upper 16 bits of the current timer drift counter. The drift value is a 32-bit signed integer. The drift counter tracks the drift between the TEM oscillator and the NTP time client on the PC. On writes this register is used to update the NTP time value inside the TEM. The set of the NTP value does not take place until a write occurs to this register. Write the upper 16 bits of the NTP time value to this register.

Bit(s)	Name	Description
15:0	NTP_DRIFT[31:16] or NTP_TIME[31:16]	Upper 16 bits of drift value on read or upper 16 bits of NTP time value on write.

0x07 – System Clock Register 1

14.2.9 0x08 – Calibration Data Register

This register is used to set the output level of the FEC calibration DAC. A write to this register will set the value in the shadow register within the FPGA. Once the write to the internal FPGA is completed the FPGA will start the write sequence to the external DAC to set the value within the device. A read from the register will return the contents of the shadow register in the FPGA, not the actual contents within the calibration DAC.

Bit(s)	Name	Description
15:0	CAL_DATA[15:0]	Calibration DAC output value. Default = 0.

0x08 – Calibration Data Register

14.2.10 0x09 – Calibration Mask Register

This register is used to enable the calibration strobe for the 16-channels present on the front end card. A value of ‘1’ set to a particular bit enables the calibration strobe for the associated channel. A read from this register returns the current calibration mask value.

Bit(s)	Name	Description
15:0	CAL_MASK[15:0]	Set to ‘1’ to enable the calibration strobe for the associated channel. Default = 0.

0x09 – Calibration Mask Register

14.2.11 0x0A – Calibration Delay Register

This register is used to set the delay between the reception of a “Start Conversion” command and when the calibration strobe is output from the FEC FPGA. The upper bits set this delay value in 1us increments, the lower bits set the delay in 50ns increments. This register also sets the calibration delay for the calibration board.

Bit(s)	Name	Description
15:14	Unused	Read as zero.
13:8	CAL_DELAY1M	Delay value in 1us increments. Default = 0.
7:5	Unused	Read as zero.
4:0	CAL_DELAY[4:0]	Delay value in 50ns increments. Default = 0.

0x0A – Calibration Delay Register

14.2.12 0x0B – Memory Error Address Register

This register contains a portion of the memory address at which the internal BIST engine detected an error.

Bit(s)	Name	Description
15:14	MEM_PASS[1:0]	Pass of memory test where error was detected, 0 -3.
13:0	MEM_ADDR[21:8]	Address of memory test where error was detected.

0x0B – Memory Error Address Register

14.2.13 0x0C – Conversion Control Register

This register is used to start & stop the data conversion process. The conversion processes is enabled and the trigger is armed by setting the EN_CONVERT bit to '1'. If the TEST_DATA bit is set (FEC cards only) the Front End Card will substitute a known data pattern for the ADC sample data. When the TEST_TRIG bit is set (TEM Slave Only) a trigger event is generated every 5 seconds. The FORCE_TRIG bit can be set to trigger immediately. A read from this register will return the current state of the defined bits.

Bit(s)	Name	Description
15:10	RSVD	Reserved, read as 0.
9:8	TEST_TRIG_RATE	Set test trigger rate when TEST_TRIG='1'. 0x0 = 0.25Hz 0x1 = 1Hz 0x2 = 10Hz 0x3 = 20Hz
7	SELF_TRIG_DIS	Disable self triggers. (Slave Only). Default = 0.
6	CAL_BOARD_ARM	Calibration Arm (Cal Board Only). Default = 0.
5	VETO_FORCE	Veto force enable. (Veto Board Only). Default = 0. (Force veto once per second).
4	CAL_ARM	Calibration Arm Bit. (FEC Only). Default = 0.
3	FORCE_TRIG	Trigger immediately. (Slave Only). Default = 0.
2	TEST_TRIG	Enable test trigger. (Slave Only). Default = 0.
1	TEST_DATA	Enable test sample data. (FEC only). Default = 0.
0	EN_CONVERT	Set to '1' to enable conversion. Default = 0.

0x0C – Conversion Control Register

14.2.14 0x0D – Memory Control Register

This register is used to control the BIST engine on each of the slave FPGAs as well as monitor the parity error status of each memory. The MEM_PERR bit is set anytime a parity error is detected when reading from the external memory. This bit can be cleared by writing a '1' back to this bit. The BIST engine is started by setting the BIST_RUN bit to '1'. When the BIST engine completes it will set the BIST_DONE bit and the BIST_PASS bit will be set if the engine completed successfully.

The BIST_RUN bit must be cleared before normal memory operations can proceed.

Bit(s)	Name	Description
15:8	MEM_ADDR[7:0]	Address of memory test where error was detected.
7	RSVD	Reserved, read as 0.
6	BIST_PASS	BIST Run Passed, Valid when BIST_RUN and BIST_DONE are set.
5	BIST_DONE	BIST Run Is Done, Valid when BIST_RUN is set.
4	BIST_RUN	Enable BIST Run. Default = 0.
3:1	RSVD	Reserved, read as 0.
0	MEM_PERR	A Memory Parity Error has occurred; write ‘1’ to clear bit. Default = 0.

0x0d – Memory Control Register

14.2.15 0x0E – Memory Error Data 0 Register

This register contains the data read at the failing address during a BIST test.

Bit(s)	Name	Description
15:0	MEM_DATA[15:0]	Data read from memory at failing BIST address.

0x0E – Memory Error Data 0 Register

14.2.16 0x0F – Memory Error Data 1 Register

This register contains the data read at the failing address during a BIST test.

Bit(s)	Name	Description
15:0	MEM_DATA[31:16]	Data read from memory at failing BIST address.

0x0F – Memory Error Data 1 Register

14.2.17 0x10 – Trigger Count Register

This register is used to monitor the number of trigger events that have occurred in the EXO system. A read from this register returns the current count. This register does not saturate. This register is reset when the run is stopped.

Bit(s)	Name	Description
15:14	RSVD	Unused, Read as 0.
13:0	TRG_CNT[13:0]	Trigger event count, rollover counter. Default = 0.

0x10 – Trigger Count Register

14.2.18 0x11 – Trigger Drop Count Register

This register is used to monitor the number of triggers events that were dropped in the EXO system. A read from this register returns the current count. This register does not saturate. This register is reset when the run is stopped.

Bit(s)	Name	Description
15:14	RSVD	Unused, Read as 0.
13:0	TRG_DCNT[13:0]	Trigger event drop count, saturating counter. Default = 0.

0x11 – Trigger Drop Count Register

14.2.19 0x12 – Trigger Sum Count Register

This register control the number of background samples that are included in the running background average. The background average is subtracted from the current sum before it is compared against the 4 sum threshold values. A value of 0 means that no background sum is computed.

Bit(s)	Name	Description
15:4	RSVD	Unused, Read as 0.
3:0	SUM_COUNT[3:0]	Sum Count: 0x 0 = 0 Samples 0x 1 = 1 Sample 0x 2 = 2 Samples 0x 3 = 4 Samples 0x 4 = 8 Samples ….. 0xA = 512 Samples 0xB = 1024 Samples

0x12 – Trigger Sum Count Register

14.2.20 0x13 – Trigger Dead Time Register

This register defines the number of samples that must be taken following the trigger sample before a new trigger is accepted.

Bit(s)	Name	Description
15:0	TRIG_DEAD[15:0]	Trigger Dead Time Value. Default = 0.

0x13 – Trigger Dead Time Register

14.2.21 0x14 – Sum Trigger Hysteresis Register

This register defines the hysteresis value which determines how low below threshold the signal must go before the trigger is de-asserted.

Bit(s)	Name	Description
15:0	TRIG_SUM_HYST	Sum trigger hysteresis value. Default=0.

0x14 – Sum Trigger Hysteresis Register

14.2.22 0x15 – Individual Trigger Hysteresis Register

This register defines the hysteresis value which determines how low below threshold the signal must go before the trigger is de-asserted.

Bit(s)	Name	Description
15:0	TRIG_IND_HYST	Individual trigger hysteresis value. Default=0.

0x15 – Individual Trigger Hysteresis Register

14.2.23 0x1E – Voltage ADC Register

This register contains the output of the voltage input ADC.

Bit(s)	Name	Description
15:12	RSVD	Unused, Read as 0.
11:0	PWR_ADC[11:0]	Power input ADC Value.

0x1E – Power ADC Register

14.2.24 0x1F – Temperature ADC Register

This register contains the output of the temperature input ADC.

Bit(s)	Name	Description
15:12	RSVD	Unused, Read as 0.

11:0	TEMP_ADC[11:0]	Temperature input ADC Value.

0x1F – TEMP ADC Register

14.2.25 0x20 – Individual Trigger 0 Threshold Register

This register contains the 12-bit threshold value for APD/VG/HV individual Trigger 0. If the largest channel value is equal to or greater than the ITRG0_THOLD value and the ITRG0_EN bit in the following register is set, individual trigger 0 is asserted.

Each detector group has 4 possible individual trigger thresholds each with its own pre-scale counter. This register is replicated at addresses 0x22, 0x24 & 0x26 for the remaining 3 trigger thresholds. Reads from this register return the current threshold value, while a write access will update the value.

Bit(s)	Name	Description
15:0	ITRG0_THOLD[15:0]	Individual Trigger 0 Threshold, (12-bits used). Default = 0.

0x20 – Trigger 0 Threshold Register

14.2.26 0x21 – Individual Trigger 0 Pre-Scale Register

This register contains the 8-bit pre-scale value for individual Trigger 0. This register defines the number of individual Trigger 0 assertions to skip before a trigger is generated. A value of 0 is this register will allow every Trigger 0 assertion to result in a trigger. The ITRG0_EN bit determines if this trigger threshold is enabled.

Each detector group has 4 possible individual trigger thresholds each with its own pre-scale counter. This register is replicated at addresses 0x23, 0x25 & 0x27 for the remaining 3 trigger pre-scale values. Reads from this register return the current pre-scale value, while a write access will update the value.

Bit(s)	Name	Description
15	ITRG0_EN	Individual Trigger 0 Enable, Default = 0.
14:11	RSVD	Unused, Read as 0.
10:8	ITRG0_THOLD[18:16]	Individual Trigger 0 Threshold, Default = 0. (unused)

7:0	ITRG0_PSCALE[7:0]	Individual Trigger 0 Pre-Scale, Default = 0.

0x21 – Individual Trigger 0 Pre-Scale Register

14.2.27 0x28 – Sum Trigger 0 Threshold Register

This register contains the lower 16-bits of the 19-bit threshold value for APD/VG/HV sum Trigger 0. If the sum is equal to or greater than the STRG0_THOLD value and the STRG0_EN bit in the following register is set, sum trigger 0 is asserted.

Each detector group has 4 possible sum trigger thresholds each with its own pre-scale counter. This register is replicated at addresses 0x2A, 0x2C & 0x2E for the remaining 3 sum trigger thresholds. Reads from this register return the current threshold value, while a write access will update the value.

Bit(s)	Name	Description
15:0	STRG0_THOLD[15:0]	Sum Trigger 0 Threshold, Default = 0.

0x28 – Sum Trigger 0 Threshold Register

14.2.28 0x29 – Sum Trigger 0 Pre-Scale Register

This register contains the 8-bit pre-scale value for sum Trigger 0 as well as the upper 3 bits of the 19-bit threshold value. This register defines the number of sum Trigger 0 assertions to skip before a trigger is generated. A value of 0 is this register will allow every sum Trigger 0 assertion to result in a trigger. The STRG0_EN bit determines if this trigger threshold is enabled.

Each detector group has 4 possible sum trigger thresholds each with its own pre-scale counter. This register is replicated at addresses 0x2B, 0x2D & 0x2F for the remaining 3 sum trigger pre-scale values. Reads from this register return the current pre-scale value, while a write access will update the value.

Bit(s)	Name	Description
15	STRG0_EN	Sum Trigger 0 Enable, Default = 0.
14:11	RSVD	Unused, Read as 0.
10:8	STRG0_THOLD[18:16]	Sum Trigger 0 Threshold, Default = 0.

7:0	STRG0_PSCALE[7:0]	Sum Trigger 0 Pre-Scale, Default = 0.

0x29 – Sum Trigger 0 Pre-Scale Register

14.2.29 0x30 – FEC 0 Sum Disable Register

This register allows software to remove particular FEC channels from the summing equation. When the bit associated with a particular channel is set a zero value will be substituted for the channel's sample when calculating the sample sum for a particular FEC. There exists one of these registers for each of the FECs connected to a particular slave FPGA. Address 0x31 is for FEC 1, address 0x32 is for FEC 2 and so on.

Bit(s)	Name	Description
15:0	SUM_DISABLE[15:0]	Disable bit for each FEC channel. Default = 0.

0x30 – FEC 0 Sum Disable Register

14.2.30 0x36 – FEC Enable Register

This register allows software to enable & disable receive data from any non-TEM card in the system.

Bit(s)	Name	Description
15:9	Unused	Unused.
8	HV_ENABLE	HV Board Enable. (APD & VG Slave Only). Default = 0.
7	Unused	Unused.
6	VETO_ENABLE	Veto Card Enable (APD Slave Only). Default = 0.
5:0	FEC_ENABLE[5:0]	Front End Card Enable bits. Default = 0.

0x36 – FEC Enable Register

14.2.31 0x37 – Trigger/Veto FIFO Entry Count Register

This register contains the number of 32-bit words present in the trigger FIFO.

Bit(s)	Name	Description
15:12	Unused	Unused.
11	VETO_NE	Veto FIFO is not empty.
10	TRIG_NE	Trigger FIFO is not empty.
9:0	TRG_CNT[9:0]	Trigger FIFO Count.

0x37 – Trigger/Veto FIFO Entry Count Register

14.2.32 0x38 – 4KV Control DAC Register

This register is used to set the DAC value to control the 4KV power supply.

Bit(s)	Name	Description
15:0	4KV_DAC[15:0]	4KV DAC Value.

0x38 – 4KV Control DAC Register

14.2.33 0x39 - 2KV Control DAC Register

This register is used to set the DAC value to control the 2KV power supply.

Bit(s)	Name	Description
15:0	2KV_DAC[15:0]	2KV DAC Value.

0x39 – 2KV Control DAC Register

14.2.34 0x3A - APD0 200V Control DAC Register

This register is used to set the DAC value to control the APD0 200V power supply. Address locations 0x3B – 0x3F correspond to APDs 1 – 5.

Bit(s)	Name	Description
15:0	APD_200V_DAC[15:0]	APD0 200V DAC Value..

0x3A – APD0 200V Control DAC Register

14.2.35 0x40 - Veto Mask 0 Register

This register enables veto inputs 0-15.

Bit(s)	Name	Description
15:0	VETO_MASK[15:0]	Veto Mask Bits. Default = 0.

0x40 – Veto Mask 0 Register

14.2.36 0x41 - Veto Mask 1 Register

This register enables veto inputs 16-31.

Bit(s)	Name	Description
15:0	VETO_MASK[31:16]	Veto Mask Bits. Default = 0.

0x41 – Veto Mask 1 Register

14.2.37 0x42 - Veto Mask 2 Register

This register enables veto inputs 32-47.

Bit(s)	Name	Description
15:0	VETO_MASK[47:32]	Veto Mask Bits. Default = 0.

0x42 – Veto Mask 2 Register

14.2.38 0x43 - Veto Mask 3 Register

This register enables veto inputs 48-63.

Bit(s)	Name	Description
15:0	VETO_MASK[63:48]	Veto Mask Bits. Default = 0.

0x43 – Veto Mask 3 Register

14.2.39 0x50 – Cal Relay Control Register

This register enables calibration channels for the calibration card.

Bit(s)	Name	Description
15:0	CAL_RELAY[15:0]	CAL Relay Control

0x50 – Cal Relay Control Register

14.2.40 0x60 – 4KV ADC Register

This register is used to read the ADC value for the 4KV voltage measurement. A write to this register is required to initiate a conversion cycle. The upper 4-bits return the serial number of the conversion.

Bit(s)	Name	Description
15:12	4KV_SER[3:0]	4KV ADC Acquisition Serial Number.
11:0	4KV_ADC[11:0]	4KV ADC Value.

0x60 – 4KV ADC Register

14.2.41 0x61 - 2KV DAC Register

This register is used to read the ADC value for the 2KV voltage measurement. A write to this register is required to initiate a conversion cycle. The upper 4-bits return the serial number of the conversion.

Bit(s)	Name	Description
15:12	2KV_SER[3:0]	2KV ADC Acquisition Serial Number.
11:0	2KV_ADC[11:0]	2KV ADC Value.

0x61 – 2KV ADC Register

14.2.42 0x62 - APD0 200V ADC0 Register

This register is used to read the ADC0 value for the APD0 200V current sense. Address locations 0x64, 0x66, 0x68, 0x6A, 0x6C correspond to APDs 1 – 5. A write to this register is required to initiate a conversion cycle. The upper 4-bits return the serial number of the conversion.

Bit(s)	Name	Description
15:12	APD_200V_SER[3:0]	APD0 200V ADC Acquisition Serial Number..
11:0	APD_200V_ADC[11:0]	APD0 200V ADC Value..

0x62 – 2KV ADC Register

14.2.43 0x63 - APD0 200V ADC1 Register

This register is used to read the ADC1 value for the APD0 200V current sense. Address locations 0x65, 0x67, 0x69, 0x6B, 0x6D correspond to APDs 1 – 5. A write to this register is required to initiate a conversion cycle. The upper 4-bits return the serial number of the conversion.

Bit(s)	Name	Description
15:12	APD_200V_SER[3:0]	APD0 200V ADC Acquisition Serial Number..
11:0	APD_200V_ADC[11:0]	APD0 200V ADC Value..

0x63 – 2KV ADC Register