Jefferson Science Associates

Thomas Jefferson National Accelerator Facility

Physics Division -- *Fast* Electronics Group

Description and Instructions for the Moller Firmware Processing Version 0x0D01 Control FPGA Version 0xA3 of Processing FPGA and Control FPGA Version of the ADC250V2 Boards

27 Feb 2018

Hai Dong

Modifications:

1) Change
"CL and CR and SL and SR delayed 0 to 152 nS"
to
"(CR and SR) and ((CL and SL) delayed 0 to 152 nS)"

2) Add Threshold 4 for CR Sum (shown in "**Pre-Processing (Processing FPGA)**" section.)
As shown in Appendix C, this is assigned to CONFIG10 which is mapped to Control Bus address 0x0033

3) Add "En CL OR Invert" to enable invert of OR output as shown in "FIG E: Scaler Coincident Matrix (SCM)"
As shown in Appendix D,"En CL OR Invert" is assigned to bit 30 of CONFIG 9-16

1.0 Introduction

The Hall A Moller polarimeter detects electrons scattered from a polarized iron foil. Since there are two electrons in the final state, the detector consists of two arms. Each arm contains 4 calorimeter blocks and 4 scintillator counters1.In one arm, the calorimeter blocks are arranged in a column. The electrons are swept along the column by a dipole magnet and their energies depend on the position. However, the HV on the PMTs are tuned to equalize the calorimeter outputs from all 4 blocks. The electron signal can be shared between two adjacent blocks. Typically, only one scintillator counter in the column is hit.

As shown in Figure A, the FADC250 receives 4 sets of signals: CL1-CL4, SL1-SL4, CR1-CR4, SR1-SR4 and Helicity and Helicity Flip Trigger. On the FADC250 board, CL, SL, CR, SR are received and processed by the processing FPGA. The Helicity and Helicity Flip Trigger are connected to the Control FPGA. The Processing FPGA processes the signals and sends the result to Control FPGA for further processing to be sent to host. Table A show the mapping of CL, SL, CR, SR to ADC input channels.

This document will describe the latest, upgraded firmware version of the processing FPGA on the FADC250 board for Hall A Moller Polarimeter Experiment. The new version of the firmware will provide more processing, include monitoring capabilities and reduces the processing time and data size. It has 6 unique functions as follows:

- Read Out Processing
- Trigger Processing
- Monitoring
- IC Configuration
- Miscellaneous
- Housekeeping.

The FADC250 board and this firmware are running nominally at 250MHz clock rate and all the time references listed in this document are Number of Samples * 4 nS. However the board and this firmware can also be ran at slower clock. In this case all the time references listed in this document are Number of Sample * 1/clock.

At power up, the firmware configures the ADC IC to handle negative going pulses. If positive going pulse is required, Appendix C shows how to configure ADC and DAC.

Figure A: Hall A Moller Set Up

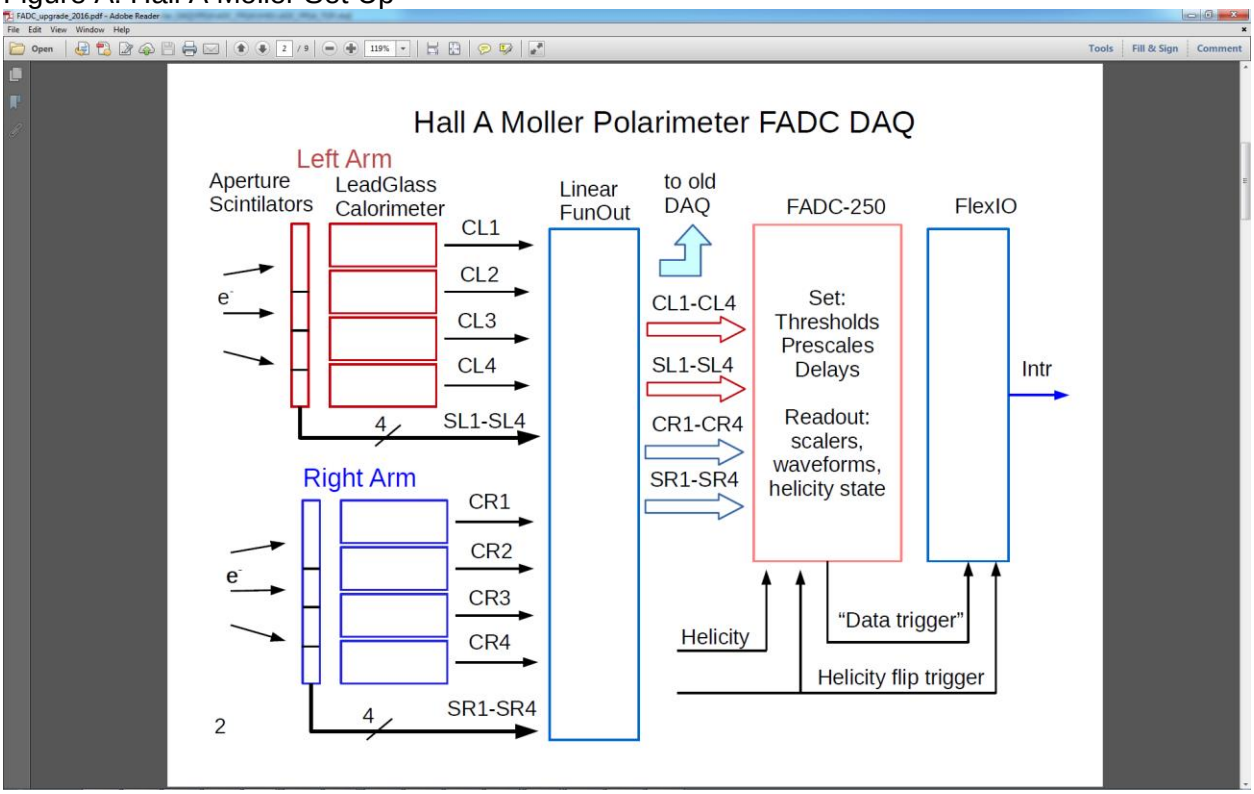


Table A: CL, SL, CR, SR to ADC channels mapping

ADC Channels	Polarimeter Signals
1	SL1
2	SL2
3	SL3
4	SL4
5	CL1
6	CL2
7	CL3
8	CL4
9	SR1
10	SR2
11	SR3
12	SR4
13	CR1
14	CR2
15	CR3
16	CR4

2.0 Read Out Processing

In Read Out Processing, the ADC samples are continuously stored in a 2048 sample Circular Raw Buffer (one per channel). When a trigger is received, a 12 bit Trigger Number counter is incremented and a number of ADC samples (Programmable Trigger Window) that are stored before (Programmable Lookback) the trigger is received is copied to the unprocessed Buffer. The Programmable Trigger Window (PTW) is user programmable from **5** to 511 (9 bits). The Programmable Lookback (PL) parameter is user programmable from 0 to 2047 (11 bits). After the PTW-number of ADC samples is copied to the unprocessed buffer, sample processing is based on the Processing Mode: Production Mode and Debug Mode.

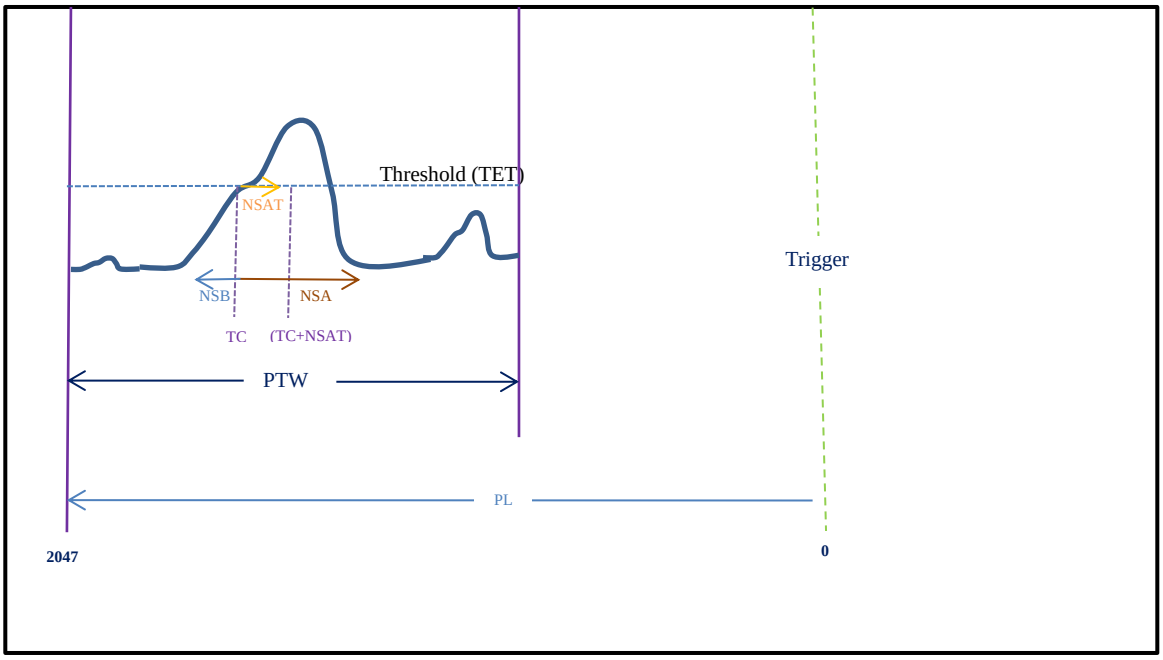


Figure 1 : Read Out Processing

1. Production Mode (9).

In this mode, the algorithm detects pulses within the PTW number of samples stored in the unprocessed buffer. The samples of pulse(s) are summed and the time at which the pulse occurred within the PTW is calculated.

a. **Pulse Definition**. Pulse identification is initiated if a number of consecutive samples (NSAT) are above a programmable threshold (TET). The pulse duration (data set) includes the number of ADC samples before (NSB) and after (NSA) the first sample crossed threshold (TC). NSA includes TC. The data set that defines the pulse includes only samples within the trigger window (PTW), even if NSB and NSA would extend beyond the window boundaries.

i. Up to four distinct pulses may be detected within PTW samples. The Maximum Number of Pulses (MNoP) is programmable from 1 to 4. If PTW contains more than MNoP, only MNop pulses are reported.

ii. NSA is programmable from Min **2** to 511 (9bits).

iii. NSB is a 4 bit programmable parameter. When bit 3 is zero, bits 0-2 indicates the number of samples before TC to be included in Pulse Duration. When the bit 3 is one, bits 0-1 indicate the number of samples after TC to be excluded from Pulse Duration.

iv. **NSA has to be greater than NSB when bit 3 is one**. When bit 3 is one only bits 0-1 are effective (bit 2 is ignored).

v. Pulse duration = samples from MAX((TC-NSB),1) to MIN((TC+NSA-1), PTW) when NSB bit 3 = 0

vi. Pulse duration = samples from (TC+NSB) to MIN((TC+NSB+NSA-1), PTW) when NSB bit 3 = 1.

vii. Pulse number is counting from 1.

viii. Pulse has to have at least one sample below (less than) TET to be considered the end to allow detection of a next pulse.

ix. If the Pulse starts less than NSAT + 1 before the end of PTW, it is **NOT** counted as a pulse.

x. When NSB is negative, if the pulse starts after [NSB] + 2 samples before end of window, it is **NOT** counted as a pulse.

xi. When NSB is negative, NSA – (NSB bits 1, 0) has to be greater than 3.

xii. If first sample is above threshold it does not count as NSAT and hence TC will never has a value of one.

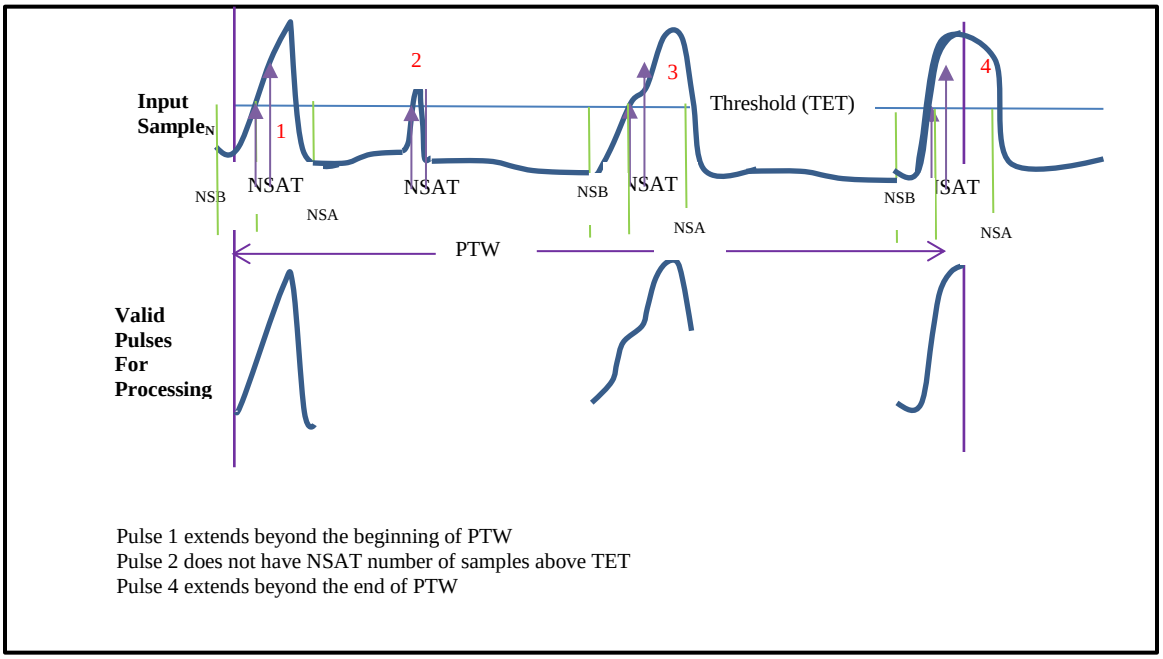


Figure 2 : Pulse Definition

~~b.~~ **Pulse Sum** is sum of samples of the defined Pulse Duration. The resulting Sum is an 18 bit quantity. When a pulse falls outside PTW, the condition is reported in the Integral quality bits (3 bits). When the sum is overflowed, all sum bits are oneIf reported Coarse Time is 1,2,3,4,or 5 the integral calculated for any NSAT other than zero is invalid. This will be fixed in the next version.

i. When any samples before a pulse is underflow, integral quality bit 9 of integral word for that pulse will be set.

ii. When any samples before a pulse is overflow, integral quality bit 10 of integral word for that pulse will be set

iii. When NSA falls outside the ending of PTW, integral quality bit 11 of integral word will be set.

iv. Time over threshold counter: The number of samples where the FADC amplitude is larger than the readout threshold (TET) in the integration window will be reported in the data using 9 bits.

v. If Coarse Time is 1,2,3,4, or 5 the integral calculated for any NSAT other than zero is invalid. This will be fixed in the next version

Sum Quality Bit	Functions
0	Set when any samples before a pulse is underflow.
1	Set when any samples before a pulse is overflow
2	Set When NSA falls outside of window ending

c. **Pulse Pedestal** is the sum of a number of samples (NPED) after the beginning of PTW. NPED is programmable from 4 to 15 samples. When any of NPED samples is greater than a programmable max pedestal threshold (MaxPed) or is underflow or over flow, the Pedestal Quality bit 14 (shown in Appendix A) of Pedestal Sum is set.

d. **Time to Digital Converter (TDC)**

i. The time reported represents the time on the pulse's leading edge where half of its maximum sampled amplitude is reached. The algorithm for computing this time is described below. Exceptional cases where the algorithm cannot be applied are also discussed. Whenever the algorithm fails, the reported time is the threshold crossing time TC. Information is returned that identifies these cases to the user.

ii. A baseline amplitude (VMIN) is determined for the entire trigger window by averaging the first 4 samples of the trigger window. A pulse with threshold crossing sample number TC is identified in the manner discussed in the section on pulse definition. The peak amplitude (VPEAK) is determined by finding a sample beyond TC for which the sample value first decreases. The algorithm will search for VPEAK beyond the expected end of the pulse (TC + NSA). Cases for which no VPEAK is detected are discussed below.

iii. The half amplitude (VMID = (VPEAK + VMIN) / 2)) of the pulse is computed. The sample number N1 is found on the leading edge of the pulse that satisfies:
   1. $V(N1) \leq VMID < V(N1+1)$
   2. where V(N1) and V(N+1) are the sample values of adjacent samples N1 and N1+1. N1 is reported as the coarse time.

iv. The estimated time of occurrence of VMID between samples N1 and N1+1 is determined by a linear interpolation using their sample values V(N1) and V(N1+1). The time between samples (4 ns) is divided into 64 subsamples (62.5 ps each). In essence,
   1. $TF = 64*(VMID – V(N1)) / (V(N1+1) – V(N1))$.
   2. TF is reported as the fine time with values from 0 to 63. TF is reported in Fine Time bits 20-15 (Appendix A).
   3. Coarse Time is the time of V(N1) and it is reported in Coarse Time bits 29-21 (Appendix A).

v. If any of the first 5 samples is greater than MaxPed but less than TET, the TDC algorithm will proceed and Time Quality Bit 0 will be set to 1.

vi. If any any of the first 5 samples is greater than TET or ~~underflow~~, the TDC will NOT proceed.
   1. pulse time is set to TC,
   2. Pulse Peak is set to zero.
   3. Time Quality bits 0 and 1 (appendix A) are set to 1.

vii. A problem with the algorithm occurs if VPEAK is not found within the trigger window. In this case, the reported parameters are as followed:
   1. Pulse time is set to TC.
   2. Pulse Peak is set to zero.
   3. Time Quality bit 1 (appendix A) is set to 1.

viii. Vpeak is found when the current sample is less than the previous sample. Moreover this condition has to be met 1 sample before end of window.

ix. If the pulse extended beyond the window, TC will be reported and Sum quality bit 2 is set.

x. When Vpeak is beyond NSA, bit 2 will be set. TDC time is reported if condition viii is met. If condition viii is not met, TC is report and Time quality bit 1 is also set. When a pulse occurred close to end of window and Vpeak cannot be found this bit will also set.

xi. Time reported is 15 bits and has a time resolution of 62 pico-seconds. The upper 9-bits are called the coarse time and the lower 6-bits are called the fine time. The fine time of TC time is always zero. Both the coarse time of TC and TDC are starting from one. In other word the coarse time of the first sample of the PTW is one.

xii. "If any of the first 5 samples is greater than TET the TDC time quality bit 1 will be set. If If any of the first 5 samples is greater than MaxPed or greater than TET or is overflow or underflow, the TDC time quality bit 0 will be set"

Time Quality Bit	Functions
0	Set when any of the first 5 samples is greater than

	MaxPed or TET or is underflow or overflow
1	When Vpeak cannot be found or when any of the first 5 samples is greater than TET.
2	Pulse Peak is beyond NSA or could be beyond end of window

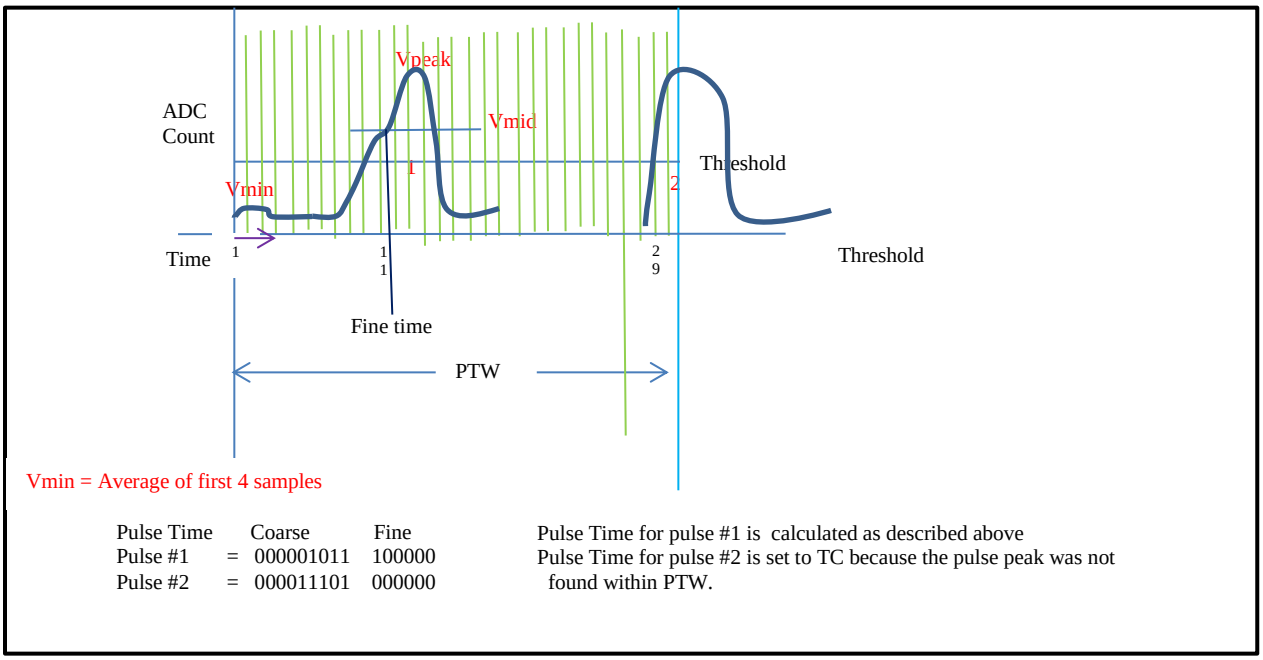


Figure 3: TDC

e. **To run Production Mode**
   i. Set up Programmable Parameters
   ii. Write Config 1 as follow:
      1. Bit 3 = 1 Accept and Process Trigger
      2. Bit 5-4 = Max Number of Pulse in PTW to process
      3. Bit 7 = 1 to process data from ADC IC; 0 to process data from Play Back (see below)
      4. Bit 9-8 = 01

f. **To change to Debug Mode**
   i. Write Config 1 as follow:
      1. Bit 3 = 0 Stop Accepting trigger
      2. Bit 5-4 = Max Number of Pulse in PTW to process
      3. Bit 7 = 1 to process data from ADC IC; 0 to process data from Play Back (see below)
      4. Bit 9-8 = 01 Do not change mode yet
   ii. Wait For Done All Trigger Received by polling bit 15 of Status 1
   iii. Write Config 1 as follow:
      1. Bit 3 = 1 Accept and process trigger
      2. Bit 5-4 = Max Number of Pulse in PTW to process
      3. Bit 7 = 1 to process data from ADC IC; 0 to process data from Play Back (see below)
      4. Bit 9-8 = 10 change to Debug Mode

2. Debug Mode (10).

In addition to calculating Pulse Sum, Pulse Pedestal, TDC, Pulse Peak as described in Production Mode, this mode also reports all the samples in PTW if there is at least one pulse (NSAT number of samples greater than threshold) in the PTW.

a. **To run Debug Mode**

i. Set up Programmable Parameters
ii. Write Config 1 as follow:
    1. Bit 3 = 1 Accept and Process Trigger
    2. Bits 5-4 = Max Number of Pulse in PTW to process
    3. Bit 7 = 1 to process data from ADC IC; 0 to process data from Play Back (see below)
    4. Bits 9-8 = 10

b. **To change to Production Mode**
i. Write Config 1 as follow:
    1. Bit 3 = 0 Stop Accepting trigger
    2. Bits 5-4 = Max Number of Pulse in PTW to process
    3. Bit 7 = 1 to process data from ADC IC; 0 to process data from Play Back (see below)
    4. Bits 9-8 = 10 Do not change mode yet
ii. Wait For Done All Trigger Received by polling bit 15 of Status 1
iii. Write Config 1 as follow:
    1. Bits 3 = 1 Accept and process trigger
    2. Bits 5-4 = Max Number of Pulse in PTW to process
    3. Bits 7 = 1 to process data from ADC IC; 0 to process data from Play Back (see below)
    4. Bits 9-8 = 01 change to Debug Mode

3. Read Out Programmable Parameters.

Name	Number Of Bits	Functions
PTW	9	**PTW+1** number of ADC samples to be processed per trigger. **Min is 6. Must be > NPED**
PL	10	Number of ADC samples back from trigger point to beginning of PTW
NSB	4	When bit 3 = 0, bits 2-0 is the number of ADC samples from Threshold Crossing (TC) to be included in Pulse Sum. When bit 3 = 1, bits 1-0 is the number of ADC samples from TC to be excluded in Pulse Sum.
NSA	9	Number of ADC samples after TC to be included in Pulse Sum. NSA includes TC Sample. **Min is 2**
TET	1	~~Trigger~~ Read Out Energy Threshold. A pulse is considered to be valid when a number of consecutive ADC samples (NSAT) are above TET
NSAT	2	Number of consecutive ADC samples that have to be above TET before a pulse is valid. 0-> 1 sample 1-> 2 samples 2->3samples 3->4samples
MNoP	2	Maximum number of pulses that will be processed. 0 → 1 1→2 2→3 3→4
NPED	4	**NPED + 1** number of sample to sum up for Pulse Pedestal. **Min is 4** Max is 15. **NPED has to be less than PTW.**
MaxPed	10	ADC Samples have to be below this Max Pedestal value to be valid to be included in read back pedestal sum.

Failure to adhere to Min values can result in unpredictable results.

4 Read Out Data Output.

Read Back data is written out to the External FiFo to be read by the Control FPGA. The data written out includes Trigger Number, Time at which the Trigger received, and Mode 9 or 10 data. The format is shown in Appendix A.

3.0 Trigger Processing

1) Pre-Processing (Processing FPGA)

1. Let P_i^J be the calorimeter PMT data for samples j and ADC input channel i, and let S_i^J be the scintillator data. The data for calorimeter would be summed either over 2 to 4 sample . The default at power up is 2 because the signals will be aligned in time sufficiently well that they should peak within adjacent 4 nsec windows. The data for scintillator is only two sample (j=1) because it isn't necessary to use 2 for the scintillator.
   - $CL_{i=5\text{-}8} = \sum_{j=1,n} P^J_I \geq$ threshold3 ; n =2,3,4 = CL1, CL2, CL3, CL4 respectively
   - $CL = \sum_{j=1,n}\sum_{i=5\text{-}8} P^J_I \geq$ threshold1 ; n =2,3,4
   -
   - $CR_{i=13\text{-}16} = \sum_{j=1,n} P^J_I \geq$ threshold3 ; n =2,3,4 = CR1, CR2, CR3, CR4 respectively
   - $CR = \sum_{j=1,n}\sum_{i=13\text{-}16} P^J_I \geq$ threshold4 ; n =2,3,4
   -
   - $SL_{i=1\text{-}4}$ = (Average of 2 consecutive sample $S^J_I \geq$ threshold2) . = SL1, SL2, SL3, SL4 respectively
   - $SR_{i=9\text{-}12}$ = Average of 2 consecutive sample $S^J_I \geq$ threshold2) . = SR1, SR2 SR3, SR4 respectively

2) CL, CR, SL, SR History Buffer (Control FPGA)

The history buffer, as shown in Figure F, provides the timing relationship of CL, CR, SL, SR signals **after one shot** to help aligning (by setting delay) these signals. On rising edge of any CL, CR, SL, SR bits that are selectable by the user, all the bits are store sequentially in a 32 deep memory location. The user can then read out the memory. The steps to use the history buffer are as follow:

- Select the signal(s) that trigger the storage by writing CL,CR,SL,SL History Buffer Bit Select.
- Set and Reset History Arm Bit, this also will reset the History read pointer to memory location 1 and will arm History Buffer
- Poll History Buffer Ready Bit for 1
- Read History Buffer at History Buffer Data Register until Buffer Ready Bit for 0 to indicate all data has been read out. It is not necessary to read out all data.

3) Trigger Generating (Control FPGA)

As shown in Figure B, CL, CR, digital pulse (time over threshold) go into programmable delays that delay the input by 4 to 100ns. The outputs of the delays go into one-shot circuits that produce pulses of 4 to 20 ns on rising edge of input. The outputs of one-shots go into Pre-Scaler circuits from 1 to 1000. When any outputs of pre-scaler goes high, a user programmable window is opened and the state of the helicity bit is latched. Any other outputs that go high during this time are captured. At the end of the window a trigger is generated and the trigger counter is incremented. At the end of the window, the captured outputs of coincident matrixes (trigger pattern) while the window is active and the trigger counter are stored in a FIFO to be read out by the user. The trigger counter can be reset by setting Reset Trigger Counter bit.

4) Static Equations Scalers (Counters) (Control FPGA)

As shown in Figure C, all CL, CR, SL and SR, digital pulses (time over threshold) go into programmable delays that delay the inputs by 4 to 100ns. The outputs of the delays go into one-shot circuits that produce pulses of 4 to 20 ns on rising edge of input. The outputs of SR one-shots are ORed (=SR). The outputs of SL one-shots are ORed (=SL). The rising edge of the ORed ouputs are counted individually or in coincident with one another. The counters are 32-bits and get read out at the helicity cycle of 30 to 2KHz. The 32-bits counters are

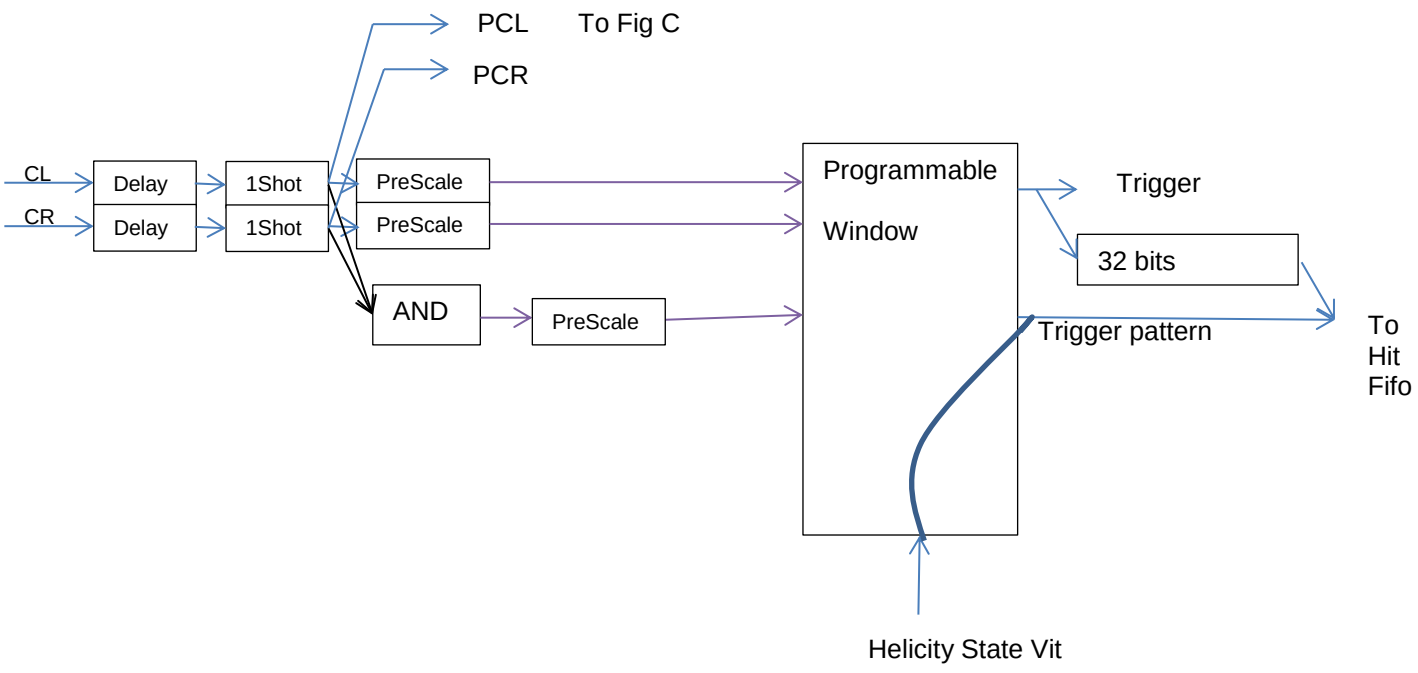
- CL
- CR
- SL
- SR
- CL and Cr
- CL and SL
- CR and SR
- CL and CR and SL and SR
- CL and CR and SL and SR delayed 0 to 152 nS
- Helicity State Bit

5) Dynamic Equations Scalers (Counters) (Control FPGA)

As shown in Figure D, $CL_{1,2,3,4}$; $CR_{1,2,3,4}$; SL; SR are input into 8 Scaler Coincident Matrixes (SCM) . Each SCM has one output. Each output is counted by 32-bits scalers (counters) on the rising edge and get read out at the helicity cycle of 30 to 2KHz.

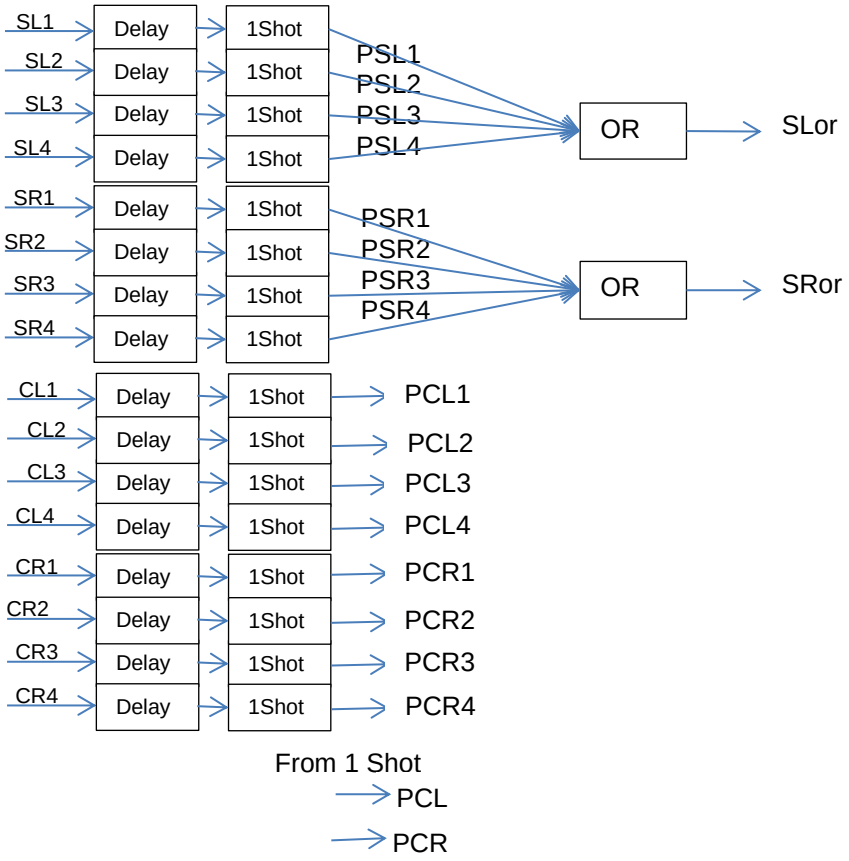
As shown in Figure E, the coincident matrix performs AND, OR functions of the 10 inputs. The active inputs for the AND, OR functions are selectable by the user. The input can also be inverted.

Figure B: Trigger Processing



Triggerattern <= Counter & Helicity_Q & Concident_CL & Concident_CR & Concident_CLandCR,

Figure C: 32 bits Scalers With Static Equations



32 bits Scalers Counting:
PCL
PCR
SLor
SRor
PCL and PCR
PCL and PSL
PCR and SR
PCL and PCR and SLor and SRor
(PCL and PCR and SLor and SRor) delayed 0 to 152 ns
Helicity State Bit

Figure D: 32 bits Scalers With Dynamic Equations

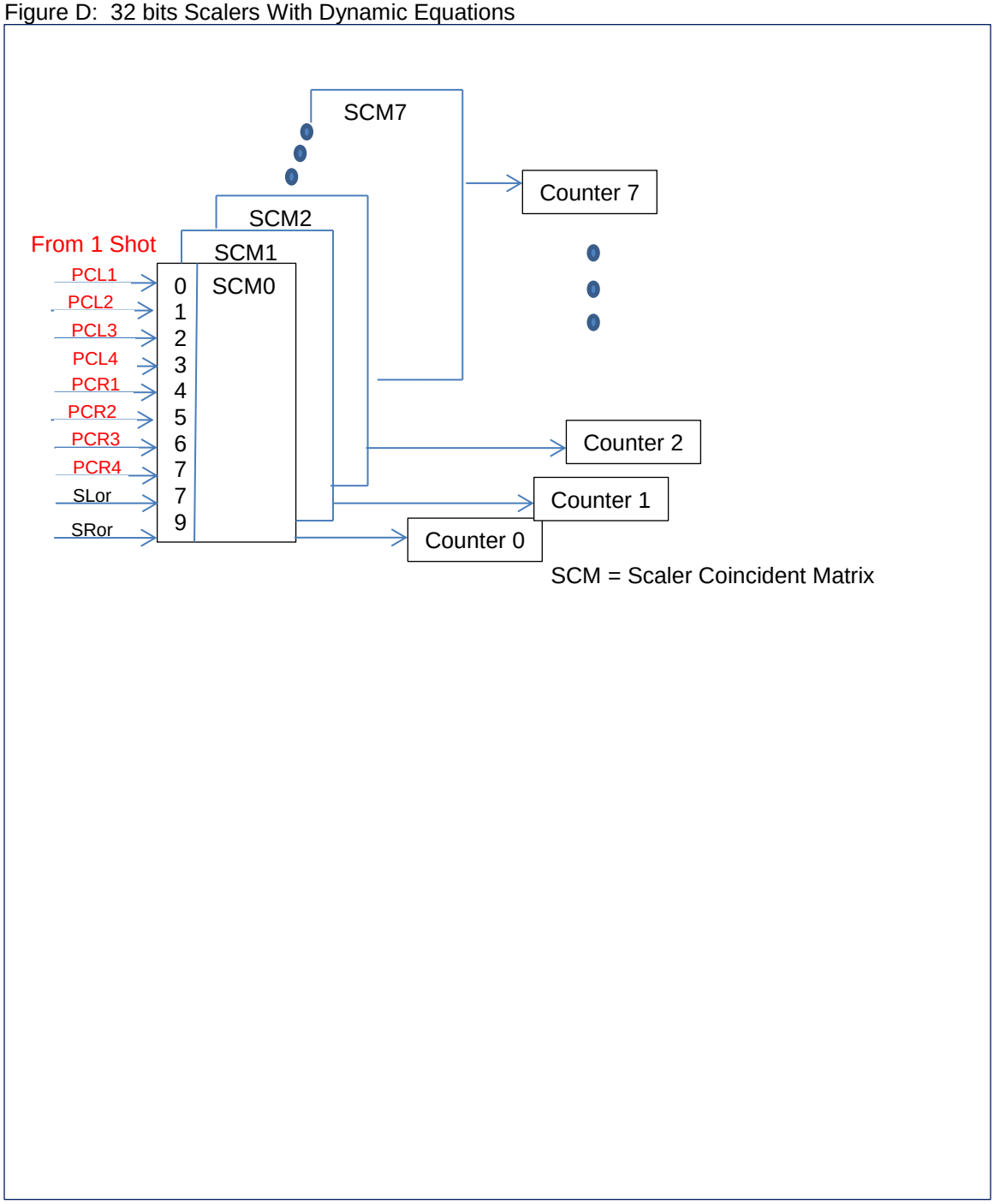


Figure E: Scaler Coincident Matrix (SCM)

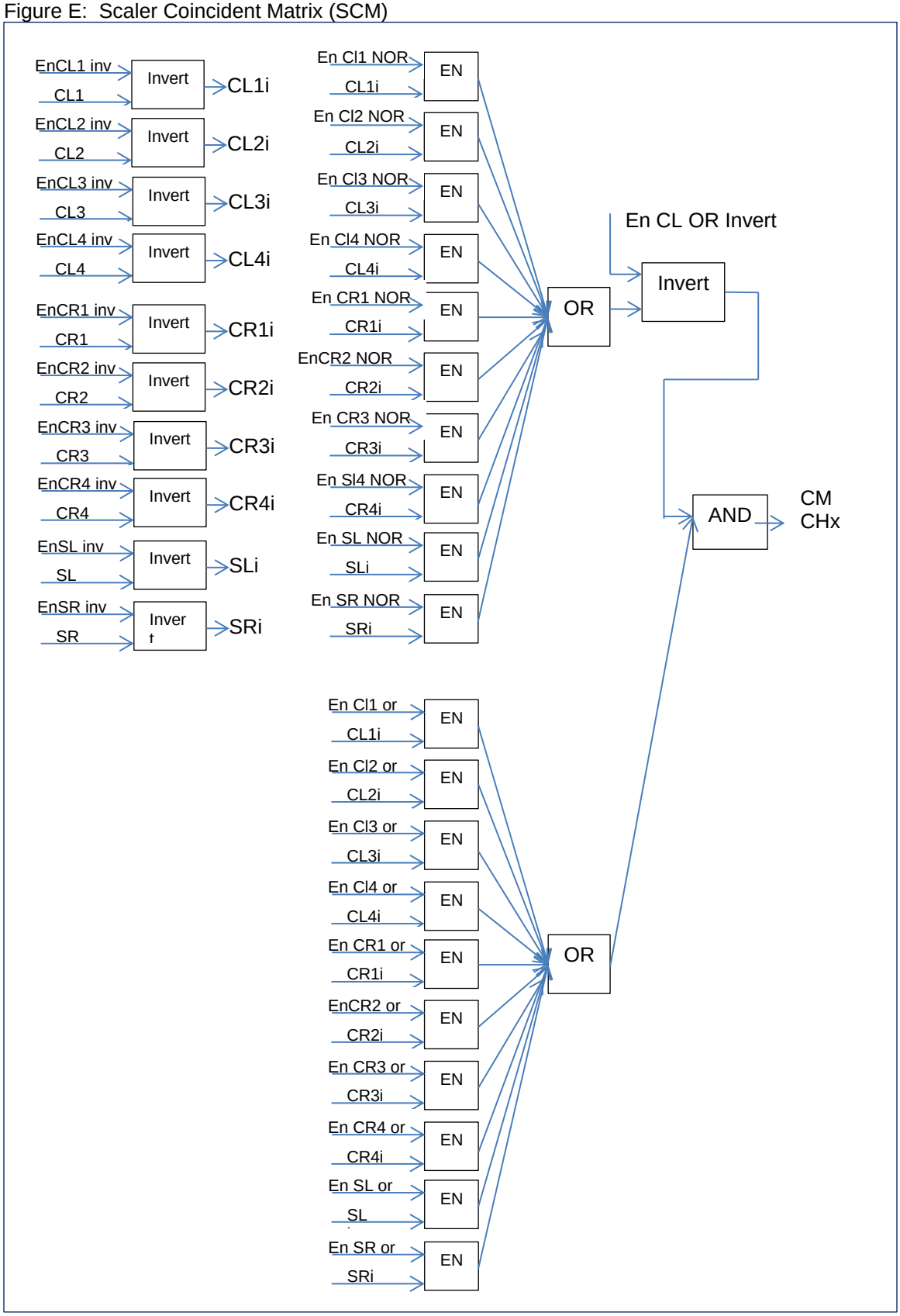
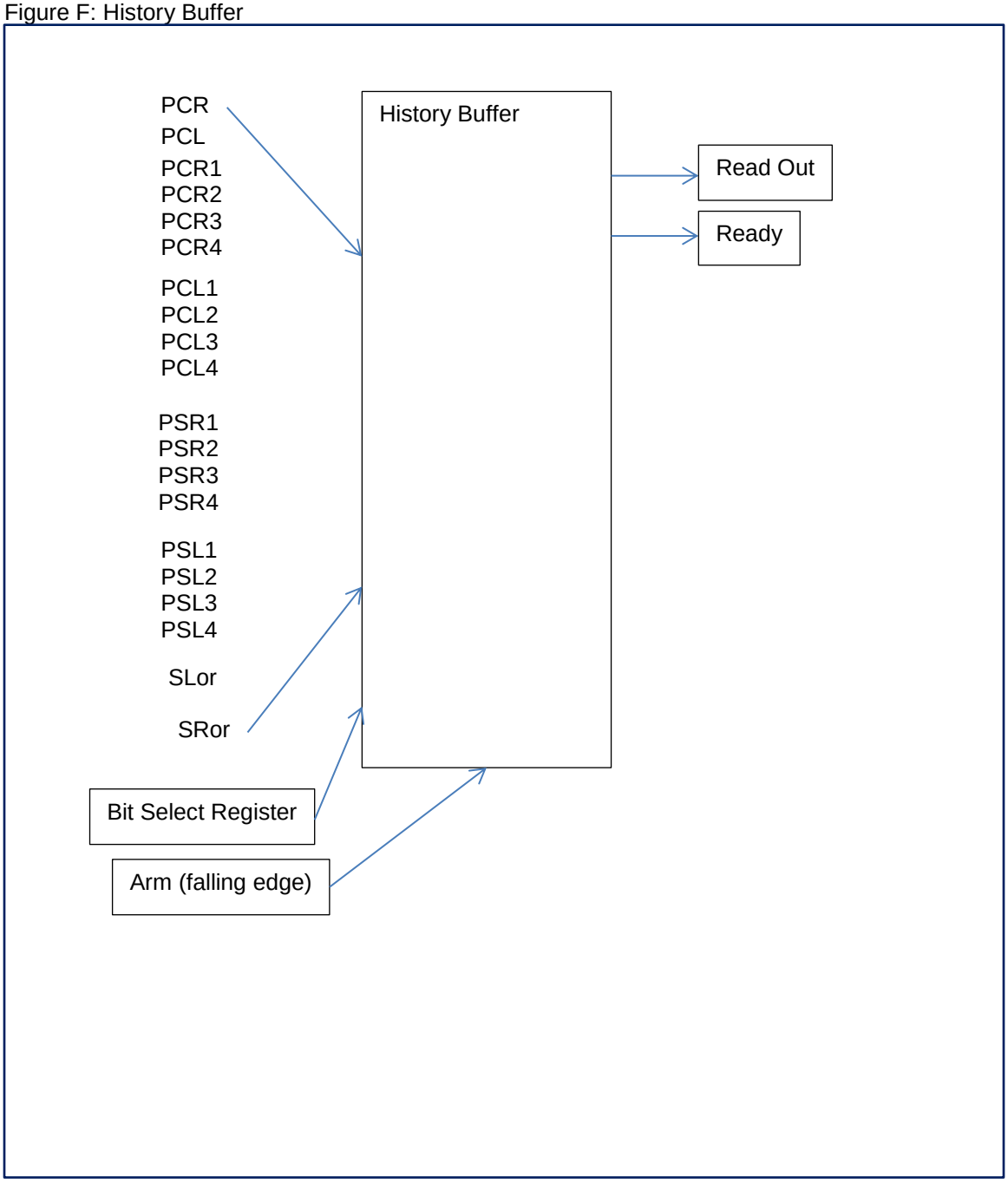


Figure F: History Buffer



4.0 Monitoring

1. Pedestal Sum

For each ADC channel, the sum of a programmable number unprocessed ADC samples can be read. When a read pedestal request is received from Control FPGA, it does the following for each ADC channel:

1. Capture a programmable number of unprocessed ADC samples (MNPED). These are consecutive samples.
2. Sum up these samples. When any of MNPED samples is greater than a programmable max pedestal threshold (PMaxPed) or less than zero (ADC bit 13 is 1 and bit 0-12 are zeroes), bit 14 of the 14 bits (13-0) Sum will be set.
3. When sum is done, bit Monitored Pedestal Sum Done is set to 1. The time taken for summing is (MNPED + 3 + 2) * 1/Clock.
4. Host reads all 16 Sums at register STATUS2. All 16 sums are mapped to only one register. The first read will be sum of ADC channel 1, follow by sums of channels 2,3,...,16. Monitored Pedestal Sum Done bit will ReSet to 0.
5. The Pedestal Sum only monitors ADC Sample. It does not monitor PlayBack value.
6. **To Read Monitored Pedestal Sums**
    a. **Reset Monitored Pedestal Sum Request bit (Config1 Bit 15)**
    b. **Set Monitored Pedestal Sum Request bit**
    c. **Poll** Monitored Pedestal Sum Done **Bit (STATUS2 Bit 15) for a one**
    d. **Read STATUS2 16 times for Pedestal Sums of ADC channel 1,…, 16**

2. Monitoring Pedestal Sum Programmable Parameters

Names	Number Of Bits	Functions
MNPED	4	**MNPED + 1** number of sample to sum up for Pulse Pedestal for Monitoring. **Min is 4** Max is 15.
PMaxPed	10	When an ADC Samples is greater than this, **bit 14 of the 14 bits (13-0) Sum will be set.**

3. FPGA Die Temperature

The temperature of the FPGA die can be read at register STATUS3 (Die Temp). The Celsius temperature is calculated as follow:

```
DieTemp_C = ((float)(STATUS3 >> 6) * 503.975/1024) - 273.15;
```

4. Firmware Version Number

Firmware version number can be read at register STATUS1 bits 14 to 0. Bits 14-8 is 0x0C and bits 7-0 indicates the code revision

5. Trigger Number 12 bits

The present Trigger Number is also available at register STATUS1.

6. STATUS Register

Names	Number Of Bits	Functions

STATUS 0	16	Bits 14 to 0: Code Version Bit 15: 1= Command can be sent to AD9230
STATUS 1	16	15 → 1 Done process all Trigger received 11-0 →Trigger Number
STATUS 2	16	Monitored Pedestal Sum. Bit 15 → 1 Done computing Pedestal Sums. Bit 14 → 0 Pedestal Sum O 1. 1 one or more ADC Sample is/are out of bound Bit 13-0 → Sum of ADC Samples
STATUS 3	16	FPGA Die Temperature
STATUS 4	16	7-0 Content of ADC register read back

5.0 IC Configuration

1. ADC IC AD9230

The ADC AD9230 ICs are configured to handle negative going pulses. If there is a need to configure the ADC for other modes (for example positive going pulses, or ADC IC to send test patterns).

a. To configure all ADC ICs at one time
   i. Poll bit 15 of Status 0 for a one. This indicates firmware is ready to accept command.
   ii. Write 0 to bit 7 of Config 4. Rising edge firmware sends data to AD9230
   iii. Select register of AD9230 to write to by writing to bits 15-8 of Config 5. Write data to be written to register of AD9230 by writing bits 7-0 of Config 5.
   iv. Set bits 7,6 and reset Bit 5 of Config 4. Bit 6 tells firmware to write to AD230. Bit 5 tells firmware to write to all AD9230

b. For Example to configure all ADC to convert negative going signal:
   i. Configure AD9230 delay clock
      1. Poll bit 15 of Status 0 for a one.
      2. Reset bit 7 of Config 4
      3. 0x17 to bits 15-8 of Config 5 to select AD9230 ADC_CLK_OUT_DELAY_REG
      4. 0x9E to bits 7-0 of Config 5. Data to write to ADC_CLK_OUT_DELAY_REG 0x9E. Delay clock b
      5. Set bit 7 and 6, reset bit 5 of Config 4. This tells firmware to write to AD9230
      6. Poll bit 15 of Status 0 for a one
      7. Reset bit 7 of Config 4
      8. 0xFF to bit 15-8 of Config 5 to select ADC_MASTER_TO_SLAVE_REG
      9. 0x01 to bit 7-0 of Config 5. Data to write to ADC_MASTER_TO_SLAVE_REG. Tell AD9230 to execute delay clock setting.
      10. Set bit 7 and 6, reset bit 5 of Config 4
   ii. Configure AD9230 to run in CML mode
      1. Poll bit 15 of Status 0 for a one.
      2. Reset bit 7 of Config 4
      3. 0x0F to bits 15-8 of Config 5 to select AD9230 ADC_AIN_CONFIG_REG
      4. 0x02 to bits 7-0 of Config 5. Data to write to ADC_AIN_CONFIG_REG. Run in CML mode
      5. Set bit 7 and 6, reset bit 5 of Config 4. This tells firmware to write to AD9230
      6. Poll bit 15 of Status 0 for a one
      7. Reset bit 7 of Config 4
      8. 0xFF to bit 15-8 of Config 5 to select ADC_MASTER_TO_SLAVE_REG
      9. 0x01 to bit 7-0 of Config 5. Data to write to ADC_MASTER_TO_SLAVE_REG. Tell AD9230 to execute delay clock setting.
      10. Set bit 7 and 6, reset bit 5 of Config 4
   iii. Tell AD9230 to turn off test mode
      1. Poll bit 15 of Status 0 for a one.
      2. Reset bit 7 of Config 4
      3. 0x0D to bits 15-8 of Config 5 to select AD9230 ADC_TEST_REG

4. 0x00 to bits 7-0 of Config 5. Data to write to ADC_TEST_REG. Turn off test mode
5. Set bit 7 and 6, reset bit 5 of Config 4. This tells firmware to write to AD9230
6. Poll bit 15 of Status 0 for a one
7. Reset bit 7 of Config 4
8. 0xFF to bit 15-8 of Config 5 to select ADC_MASTER_TO_SLAVE_REG
9. 0x01 to bit 7-0 of Config 5. Data to write to ADC_MASTER_TO_SLAVE_REG. Tell AD9230 to execute delay clock setting.
10. Set bits 7 and 6, reset bit 5 of Config 4

6.0 Miscellaneous Functions:

1. ADC Channel Disable

Each ADC channel has a bit that when set will zero the input to Trigger Sum and Trigger Hit Bits. Note, channels can be excluded from the readout if needed by setting readout thresholds (TET) to 4095. To disable (zeroes input to Trigger Sum and Hit Bit) an ADC channel, set the corresponding bit of Config 2 to 1.

2. Sync_reset

When Sync_reset signal from Control FPGA is high and bit 15 of Config 3 is 0, the following signals are in RESET (zeroes):

1. Time Stamp
2. Trigger Number
3. Not accepting Trigger signal from Control FPGA

3. Soft Reset

When Soft Reset signal from Control FPGA is asserted low, everything EXCEPT Configuration Registers and AD9230 is reset.

4. Hard Reset

When Hard Reset signal from Control FPGA is asserted low, everything is Set to values shown in Appendix C. All AD9230 ICs are reset to power state.

5. Play Back

User defines pulses maybe injected into the processing pipeline using a playback feature. Play Back stores 32, 13-bit ADC values in RAM and cycles through 32 ADC values when a Trigger_2 signal from Control FPGA goes from low to high. There are 16 Play Back, one per ADC Channel. All 512 ADC values are written into memory via Control Bus. Since all 512 ADC values occupy only one Control Bus address, all 512 values have to be written sequentially all at once The first 32 values are associated with ADC channel 0 and last 32 values are associated with ADC channel 15.

When **bit 7 of Config 1** is set, Play Back outputs (instead of ADC IC outputs) are applied to all processing functions of all ADC channels.

a. To write 510 ADC values to RAM
   i. Put ADC value on bits 12-0 of Test Wave Form register. Set Bit 15.
   ii. Write Test Wave Form register
   iii. Read Test Wave Form register to verify that value is stored to RAM
   iv. Repeat above three steps for 510 values
b. To write 511th and 512th ADC values to RAM
   i. Put 511th ADC value on bits 12-0 of Test Wave Form register. ReSet Bit 15.
   ii. Write Test Wave Form register
   iii. Read Test Wave Form register to verify that value is stored to RAM
   iv. Repeat above three steps for 512th values

7.0 House Keeping (Transparent to user):

1. Control Bus

Registers and Status are accessed through Control Bus connected to FPGA. Control Bus is asynchronous 16 bits data bus. In the basic mode, Control Bus can access 65535 addresses. In

extended mode the Control Bus can access up to 65,535 x 65,535 addresses by mean of secondary address feature (See FADC250 Program Manual)

a. The signals of Control Bus are:
   i. ADR_DAT = Address and Data
   ii. AS_N = Address strobe
   iii. AK_N = Address Acknowledge
   iv. DS_N = Data strobe
   v. DK_N = Data Acknowledge
   vi. RD_N = Low indicate Read cycle
   vii. MS = Mode select
   viii. SS = Slave Status.

b. To write to Register
   i. Control FPGA puts register's address on ADR_DAT bus. Drives MS low
   ii. Control FPGA brings AS_N low
   iii. Processing FPGA accept register's address. Drive SS low
   iv. Processing FPGA brings AK_N low.
   v. Control FPGA puts register's value on ADR_DAT bus
   vi. Control FPGA brings DS_N low
   vii. Processing FPGA accept register's value
   viii. Processing FPGA brings DK_N low.
   ix. Control FPGA drives DS_N high
   x. Processing FPGA drives DK_N high,
   xi. Control FPGA drives AS_N high
   xii. Processing FPGA drives AK_N high

c. To read Register **(Note Unused bits are read back as zeroes)**
   i. Control FPGA puts register's address on ADR_DAT bus. Drives MS low
   ii. Control FPGA brings AS_N low
   iii. Processing FPGA accept register's address. Drive SS low
   iv. Processing FPGA brings AK_N low.
   v. Control FPGA stops driving ADR_DAT bus
   vi. Control FPGA brings DS_N low.
   vii. Processing FPGA puts register's value on ADR_DAT bus
   viii. Processing FPGA brings DK_N low
   ix. Control FPGA accepts register's value.
   x. Control FPGA drives DS_N high
   xi. Processing FPGA drives DK_N high, Stop driving ADR_DAT bus
   xii. Control FPGA drives AS_N high
   xiii. Processing FPGA drives AK_N high

Appendix A: Data Format of FADC Processing

Event Header (2) – indicates the start an event.

(35 – 32) = 0001
(31) = 1
(30 – 27) = 2
(26 – 22) = 00000
(21 – 12) = trigger time (bits 9 – 0 (see below))
(11 – 0) = trigger number

Trigger Time (3) – time of trigger occurrence relative to the most recent global reset. Time in the ADC data processing chip is measured by a 48-bit counter that is clocked by the 250 MHz system clock. The six bytes of the trigger time

$$\text{Time} = T_A\ T_B\ T_C\ T_D\ T_E\ T_F$$

are reported in two words (Type Defining + Type Continuation).

Word 1:

(35 – 32) = 0000
(31) = 1
(30 – 27) = 3
(26 – 24) = T_C bits 2 – 0 (duplicated in Word 2)
(23 – 16) = T_D
(15 – 8) = T_E
(7 – 0) = T_F

Word 2:

(35 – 32) = 0000
(31) = 0
(30 – 24) = reserved (read as 0)
(23 – 16) = T_A
(15 – 8) = T_B
(7 – 0) = T_C

Window Raw Data (4) – raw ADC data samples for the trigger window. The first word identifies the channel number and window width. Multiple continuation words contain two samples each. The earlier sample is stored in the most significant half of the continuation word. Strict time ordering of the samples is maintained in the order of the continuation words. A *sample not valid* flag bit 13 will be set when PTW+1 is odd.

Word 1:

(35 – 32) = 0000
(31) = 1
(30 – 27) = 4
(26 – 23) = channel number (0 – 15)
(22 – 12) = reserved (read as 0)
(8 – 0) = PTW + 1 (window width (in number of samples))

Words 2 - N:

(35 – 32) = 0000
(31) = 0
(30) = reserved (read as 0)
(29) = sample x not valid
(28 – 16) = ADC sample x (includes overflow bit)
(15 – 14) = reserved (read as 0)
(13) = sample x + 1 not valid
(12 – 0) = ADC sample x + 1 (includes overflow bit)

Pulse Parameters (9) – computed pulse parameters for detected pulses in a channel. The first word identifies the channel number, event number within the block, and pedestal information for the window. Multiple continuation word *pairs* contain information about the pulses detected. For a channel with hits detected:

Word 1: Channel ID and Pedestal information (reported *once* for a channel with hits)
- (35 – 32) = 0000
- (31) = 1
- (30 – 27) = 9
- (26 – 19) = event number within block (1 – 255)
- (18 – 15) = channel number (0 – 15)
- (14) = pedestal quality
- (13 – 0) = pedestal sum

Word 2 : Integral of first pulse in window
- (35 – 32) = 0000
- (31) = 0
- (30) = 1
- (29 – 12) = 18-bit sum of raw samples that constitute the pulse data set
- (11) = NSA extended beyond PTW
- (10) = One or more samples is overflow = 0x1FFF
- (9) = One or more sample is underflow = 0x1000
- (8 – 0) = number of samples within NSA that the pulse is above threshold

Word 3 : Time of first pulse in window
- (35 – 32) = 0000
- (31) = 0
- (30) = 0
- (29 – 21) = coarse time (4 ns/count)
- (20 – 15) = fine time (0.0625 ns/count)
- (14 – 3) = pulse peak
- (2) = Vpeak is beyond NSA or could be beyond window end
- (1) = Vpeak cannot be found
- (0) = 1 or more of first 4 samples is above either MaxPed or TET

Words 2 and 3 are repeated for *each additional pulse* found in the window for the channel.

Event Trailer: Indicate the end of an event.
EVENT_TRAILER = "0010" & X"E8000000";

Note: For maximum compression of data the Event Header (2) and Trigger Time (3) words may be suppressed from module readout in the Control FPGA (See Programming the FADC250)

Appendix B: Example of Data Format for Production Mode 9

1st Trigger Occurred at Time 0x123456
Channel 1 and 15 has 1 good pulse each

Event Header (2) – indicates the start an event.
Word 1:
(35 – 32) = 0001
(31) = 1
(30 – 27) = 2
(26 – 22) = 00000
(21 – 12) = "00" & x"56" (trigger time (bits 9 –))
(11 – 0) = x"0001"

Trigger Time
Word 2:
(35 – 32) = 0000
(31) = 1
(30 – 27) = 3
(26 – 24) = "011" T_C bits 2 – 0 (duplicated in Word 2)
(23 – 16) = x"4"
(15 – 8) = x"5"
(7 – 0) = x"6"

Word 3:
(35 – 32) = 0000
(31) = 0
(30 – 24) = 0
(23 – 16) = x"1"
(15 – 8) = x"2"
(7 – 0) = x"3"

Pulse Parameters

Channel 1 data
Word 4: Channel ID and Pedestal information (reported *once* for a channel with hits)
(35 – 32) = 0000
(31) = 1
(30 – 27) = 9
(26 – 19) = 00000000 event number within block (0 – 255)
(18 – 15) = 0001
(14) = 0
(13 – 0) = pedestal sum
0xC800----

Word 5 : Integral of first pulse in window
(35 – 32) = 0000
(35 – 32) = 0000
(31) = 0
(30) = 1
(29 – 12) = 18-bit sum of raw samples that constitute the pulse data set
(11 – 9) = 000
(8 – 0) = number of samples within NSA that the pulse is above threshold

Word 6 : Time of first pulse in window
(35 – 32) = 0000

(35 – 32) = 0000
(31) = 0
(30) = 0
(29 – 21) = coarse time (4 ns/count)
(20 – 15) = fine time (0.0625 ns/count)
(14 – 3) = pulse peak
(2 – 0) = time quality

Channel 15 data

Word 7: Channel ID and Pedestal information (reported *once* for a channel with hits)
(35 – 32) = 0000
(31) = 1
(30 – 27) = 9
(26 – 19) = 00000000 event number within block (0 – 255)
(18 – 15) = 1111
(14) = 0
(13 – 0) = pedestal sum

Word 8 : Integral of first pulse in window
(35 – 32) = 0000
(31) = 0
(30) = 1
(29 – 12) = 18-bit sum of raw samples that constitute the pulse data set
(11 – 9) = 000
(8 – 0) = number of samples within NSA that the pulse is above threshold

Word 9 Time of first pulse in window
(35 – 32) = 0000
(31) = 0
(30) = 0
(29 – 21) = coarse time (4 ns/count)
(20 – 15) = fine time (0.0625 ns/count)
(14 – 3) = pulse peak
(2 – 0) = time quality

Event Trailer: Indicate the end of an event.
Word 10
EVENT_TRAILER = "0010" & X"E8000000";

Appendix C: Control Bus Memory Map for Registers

Unused Bits are read back as zeroes

Failure to adhere to Min values can result in unpredictable results

Name	Width (Bits)	Quantity	Access	Primary Address (Secondary Address)	Power Up Values (hex)	Function
STATUS0	16	1	R	0x0000 (---)		Bits 14 to 0: Code Version Bit 15: 1= Command can be sent to AD9230
STATUS1	16	1	R	0x0001 (---)		15 : 1 Done all Trig received 11-0 : TRIGGER NUMBER
STATUS2	16	1	R	0x0002 (---)		Monitored Pedestal 15 → Sum is valid 14 → 0 Sum OK. 1 One or more is out of bound 13-0 → Sum
CONFIG1	16	1	R/W	0x0003 (---)	0000	Bit 0-2 (old code process mode): Bit 3: 1:Run Bit 5-4 : Max Number of Pulses in Mode 10 and 9 Bit 7: Test Mode (play Back). Bit 8: 0 → mode 9 1 → mode 10 11-10 NSAT 15→ Request Sum of Pedestal for monitoring purpose
CONFIG2			R/W	0x0004 (---)	0000	When 1 ADC values = 0 Bit 0 → ADC 0 Bit 1 → ADC 1 Bit 2 → ADC 2 Bit 3 → ADC 3 Bit 4 → ADC 4 Bit 5 → ADC 5 Bit 6 → ADC 6 Bit 7 → ADC 7 Bit 8 → ADC 8 Bit 9 → ADC 9 Bit 10→ ADC 10 Bit 11→ ADC 11 Bit 12→ ADC 12 Bit 13→ ADC 13 Bit 14→ ADC 14 Bit 15→ ADC 15
CONFIG4	16	1	R/W	0x0005	0000	15 : Sync Disable. 1 not allow SYNC signal to Reset ADC

						7 => rising edge write to AD9230 ADC 6 => 1 write to all ADC 5 => 0 write to AD9230 1 read from AD9230 . Data is at Stat 4 => 1 Reset ADC 3..0 => Select ADC to write to
CONFIG5	16	1		0x0006	0000	15..8 => Registers inside AD9230 7..0 => Data to write to register.
PTW	9	1	R/W	0x0007 (---)	0000	**PTW + 1** number of ADC sample to include in trigger window. PTW = Trigger Window (ns) * 250 MHz. **Minimum is 6**.
PL	11	1	R/W	0x0008 (---)	0000	Number of sample back from trigger point. PL = Trigger Window(ns) * 250MHz
NSB	4	1	R/W	0x0009 (---)	0000	3..0: Read Back Path NSB Number of sample before trigger point to include in data processing. This include the trigger Point. When NSB bit 3 is 1: **NSA has to be > NSB bits 1,0 by at least 4 => NSA – (NSB bits 1,0) ≥ 3**
NSA	15	1	R/W	0x000A (---)	0000	8..0: Read Back Path NSA Number of sample after trigger point to include in data processing. **Minimum is 2**
TET	12	16	R/W	0x000B - 0x001A	0000	Read Out Energy Threshold.
CONFIG6 (Monitored Pedestal Sum)	16	1	R/W	0x001B	0000	13-10 MNPED : The number of ADC sample to sum up is MNPED + 1. **Min is 4** 9-0 PMaxPed : When an ADC Samples is greater than this, bit 14 of the 14 bits (13-0) Sum will be set.
CONFIG7 (Read Back Pedestal Sum)	16	1	R/W	0x001C	0000	13-10 NPED : The number of ADC sample is NPED + 1 9-0 MaxPed
Test Wave Form	16	1	R/W	0x001D	0000	Write to PPG. Read should immediately follow write.
Reserve			R/W	0x001E- 0x002D	0000	
Config 3 Trigger Path Threshold 1	16	1	R/W	0x002E	0000	CL1-4, CR1-4 Trigger Threshold
Config 8 Trigger Path Threshold 2	13	1	R/W	0x002F	0000	15-13 CL, CR Added Length. Added Length is 2 more than bits15-13. 0=2, 1=3, 2=4 0-12 SL1-4, SR1-4 Trigger Threshold
Config 9 Trigger Path Threshold 3	16	1	R/W	0x0030	0000	CL Trigger Threshold
STATUS 3	16	1	R	0x0031		FPGA core temp (DieTemp)
STATUS 4	16	1	R	0x0032		7-0 Result of ADC register readback
Config 10	16	1	R/W	0x0033	0000	CR Trigger Threshold

Trigger Path Threshold 4						

Appendix D: VME Bus Memory Map for Registers

Unused Bits are read back as zeroes

Name	Width (Bits)	Access	Power Up Values (hex)	Function
CONFIG1	32	R/W	0000	4-0 SL1 Pulse Delay 7-5 SL1 1 Shot Width 12-8 SL2 Pulse Delay 15-13 SL2 1 Shot Width 20-16 SL3 Pulse Delay 23-21 SL3 1 Shot Width 28-24 SL4 Pulse Delay 31-29 SL4 1 Shot Width
CONFIG2	32	R/W	0000	4-0 SR1 Pulse Delay 7-5 SR1 1 Shot Width 12-8 SR2 Pulse Delay 15-13 SR2 1 Shot Width 20-16 SR3 Pulse Delay 23-21 SR3 1 Shot Width 28-24 SR4 Pulse Delay 31-29 SR4 1 Shot Width
CONFIG3	32	R/W	0000	4-0 CL1 Pulse Delay 7-5 CL1 1 Shot Width 12-8 CL2 Pulse Delay 15-13 CL2 1 Shot Width 20-16 CL3 Pulse Delay 23-21 CL3 1 Shot Width 28-24 CL4 Pulse Delay 31-29 CL4 1 Shot Width
CONFIG4	32	R/W	0000	4-0 CR1 Pulse Delay 7-5 CR1 1 Shot Width 12-8 CR2 Pulse Delay 15-13 CR2 1 Shot Width 20-16 CR3 Pulse Delay 23-21 CR3 1 Shot Width 28-24 CR4 Pulse Delay 31-29 CL4 1 Shot Width
CONFIG5	32	R/W	0000	4-0 CL Pulse Delay 7-5 CL 1 Shot Width 12-8 CR Pulse Delay 15-13 CR 1 Shot Width
CONFIG6	32	R/W	0000	10-0 CL PreScaler 21-11 CR PreScaler 30 — 1 set CL PreScalar. 0 to run 31 — 1 set CR PreScalar. 0 to run
CONFIG7	32	R/W	0000	10-0 CL and CR PreScaler 18-11 Coincident Window Width 23-19 30 ---- 1 Reset Trigger Counter 31— 1 set CL and CR PreScalar. 0 to run

CONFIG8	32	R/W	0000	Select History Buffer Trigger Bits 0 CR 1 CL 2 CR1 3 CR2 4 CR3 5 CR4 6 SR1 7 SR2 8 SR3 9 SR4 10 CL1 11 CL2 12 CL3 13 CL4 14 SL1 15 SL2 16 SL3 17 SL4 18 SR_or 19 SL_or 31 Falling edge Arm History Buffer
----------	--------	----------	----------	
----------	--------	----------	----------	Bit Mapping for Config 9-16 0 Invert Select CR1 1 Invert Select CR2 2 Invert Select CR3 3 Invert Select CR4 4 Invert Select CL1 5 Invert Select CL2 6 Invert Select CL3 7 Invert Select CL4 8 Invert Select SRor 9 Invert Select SLor ---------- 10 AND Select CR1 11 AND Select CR2 12 AND Select CR3 13 AND Select CR4 14 AND Select CL1 15 AND Select CL2 16 AND Select CL3 17 AND Select CL4 18 AND Select SRor 19 AND Select SLor ---------- 20 OR Select CR1 21 OR Select CR2 22 OR Select CR3 23 OR Select CR4 24 OR Select CL1

				25 OR Select CL2 26 OR Select CL3 27 OR Select CL4 28 OR Select SRor 29 OR Select SLor 30 En CL* OR Invert
CONFIG9	32	R/W	0000	SCM 7 Config 9-0 Invert Select 19-10 AND Select 29-20 OR Select 30 En CL OR Invert
CONFIG10	32	R/W	0000	SCM 6 Config 9-0 Invert Select 19-10 AND Select 29-20 OR Select 30 En CL OR Invert
CONFIG11	32	R/W	0000	SCM 5 Config 9-0 Invert Select 19-10 AND Select 29-20 OR Select 30 En CL OR Invert
CONFIG12	32	R/W	0000	SCM 4 Config 9-0 Invert Select 19-10 AND Select 29-20 OR Select 30 En CL OR Invert
CONFIG13	32	R/W	0000	SCM 3 Config 9-0 Invert Select 19-10 AND Select 29-20 OR Select 30 En CL OR Invert
CONFIG14	32	R/W	0000	SCM 2 Config 9-0 Invert Select 19-10 AND Select 29-20 OR Select 30 En CL OR Invert
CONFIG15	32	R/W	0000	SCM 1 Config 9-0 Invert Select 19-10 AND Select 29-20 OR Select 30 En CL OR Invert
CONFIG16	32	R/W	0000	SCM 0 Config 9-0 Invert Select 19-10 AND Select 29-20 OR Select 30 En CL OR Invert
Will be mapped at Ctrl FPGA register				History Buffer Ready for read out
STATUS 0 History Buffer Data Register	32	R		0 CR 1 CL 2 CR1 3 CR2

				4 CR3 5 CR4 6 CL1 7 CL2 8 CL3 9 CL4 10 CL1 11 CL2 12 CL3 13 CL4 14 SL1 15 SL2 16 SL3 17 SL4 18 SLor 19 SRor
STATUS 2	32	R		CL SCALAR (Counter)
STATUS 3	32	R		CR SCALAR (Counter)
STATUS 4	32	R		SL SCALAR (Counter)
STATUS 5	32	R		SR SCALAR (Counter)
STATUS 6	32	R		CL and CR SCALAR (Counter)
STATUS 7	32	R		CL and SL SCALAR (Counter)
STATUS 8	32	R		CR and SR SCALAR (Counter)
STATUS 9	32	R		CL and CR and SL and SR SCALAR (Counter)
STATUS 10	32	R		CL and CR and SL and SR delayed 100 to 150 nS SCALAR (Counter)
STATUS 11	32	R		Helicity State Bit SCALAR (Counter)
STATUS 12	32	R		SCM 0 SCALAR (Counter)
STATUS 13	32	R		SCM 1 SCALAR (Counter)
STATUS 14	32	R		SCM 2 SCALAR (Counter)
STATUS 15	32	R		SCM 3 SCALAR (Counter)
STATUS 16	32	R		SCM 4 SCALAR (Counter)
STATUS 17	32	R		SCM 5 SCALAR (Counter)
STATUS 18	32	R		SCM 6 SCALAR (Counter)
STATUS 19	32	R		SCM 7 SCALAR (Counter)

Appendix B

Bias DAC Count	**ADC Count With No Input (in hex)**
4096	1000
4000	1000
3500	1000
3300	66
3100	173
3000	1FB
2900	281
2800	30A
2700	393
2600	419
2500	49E
2400	526
2300	5ae
2200	632
2100	6BA
2000	741
1900	7C6
1800	850
1700	8D5
1600	95B
1500	9E1
1400	A6B
1300	AF0
1200	B7B
1100	C00
1000	C86
900	D0E
800	D91
700	E19
600	EA1
500	F29
400	FAC
300	1FFF

Appendix C:

Setting ADC and DAC For Positve Going Pulse

Set the DAC value to 700.

Sample C Code to configure ADC IC to invert output.

```
#define  OUTPUT_MODE_REG 0x1400  // this is the register inside the ADC to invert output
#define ADC_MASTER_TO_SLAVE_REG  0xFF00  // write 1 to this register to transfer data from
Master Shift Reg to Salve
#define ADC_TX_MASTER_TO_SLAVE_CMD  0x01
#define ADC_INVERTOUTPUT 0x0004 // Invert Output

    AdcConfigValue =  OUTPUT_MODE_REG | ADC_INVERTOUTPUT; //
    RegFile[20] = AdcConfigValue;
    printf("Sent ADC ADC_INVERTOUTPUT Config Word 3\n");
    SendSetRegisters();
    Sleep(10);

  AdcConfigValue =  ADC_MASTER_TO_SLAVE_REG | ADC_TX_MASTER_TO_SLAVE_CMD; //
Tell ADC to exec the commands sent
    RegFile[20] = AdcConfigValue;
    printf("Sent ADC_TX_MASTER_TO_SLAVE_CMD ADC Config Word 3\n");
    SendSetRegisters();
    Sleep(10);
```