FIRMWARE for FADC250 Ver2 ADC FPGA

Table of Content:

ADC FPGA Functional Description

Revision:

Version 90D:

Add Status 2 to read back ADC counts. The ADC channel is select with Config 1 bit 11 down to 8.

Add Status 3 and 4 to read FPGA core temp and Vint.

Version 90E:

Trigger Path Processing Threshold (Config 3) is used for all Hit Bits.

Version 90F:

HitBit are now processed after Pedestal Subtraction

Version 910:

AUX_IO(3) <= TrigAcknowledge; TrigAcknowledge is issued after all process data from a trigger are written to external FIFO.

Version 911:

Add Sync Disable. When 1 disable Sync from resetting FIFO that receives data from ADC IC. Sync Disable is mapped to CONFIG3 bit 15

Overview:

The ADC FPGA receives 12-bit data words streaming at 250 MHz from 16 ADC. It performs **Channel Data Processing** for each ADC, computes **Energy Sum** of all ADC, and generates **Acceptance Pulse** for each ADC. The data selected in Channel Data Processing and results of Energy Sum are passed to CTRL FPGA to be sent to VME host and CTP respectively. The code is modular such that processing algorithms can easily be added or deleted.

Figure 1. Block Diagram (Input Mode):

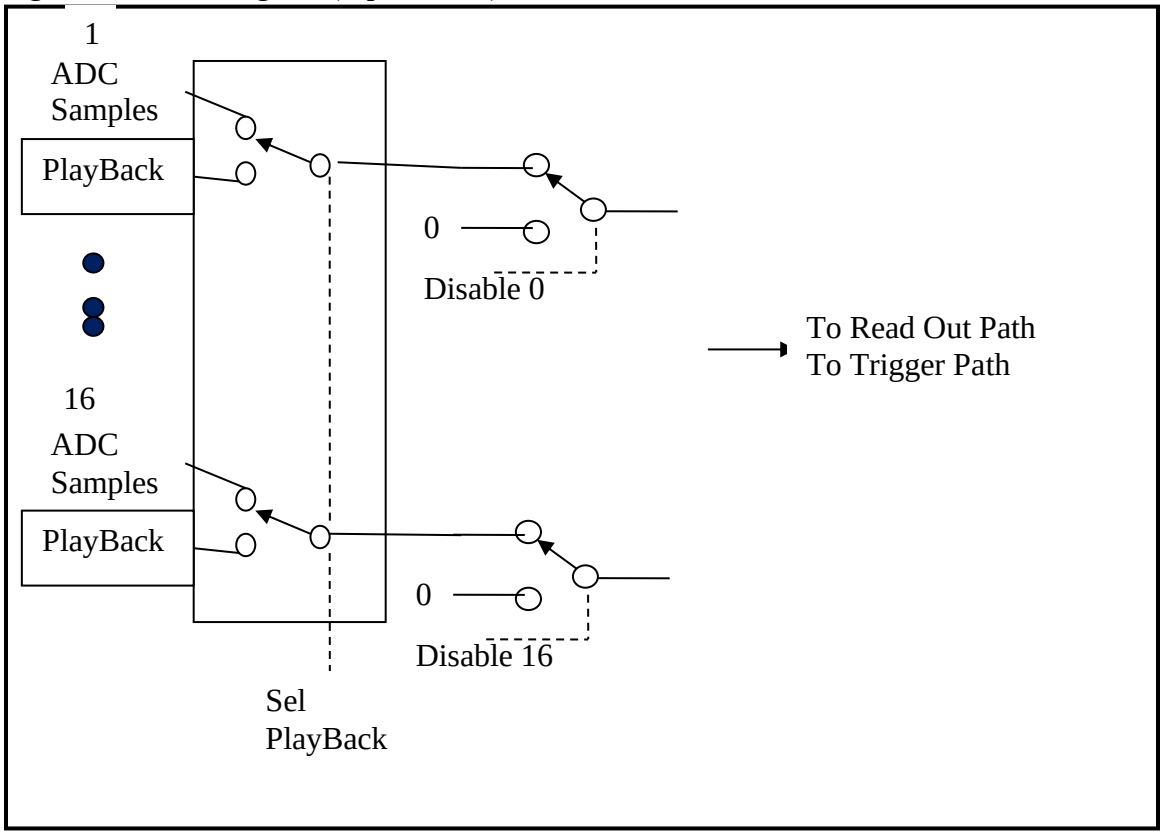


Figure 2. Block Diagram (Read Out Path):

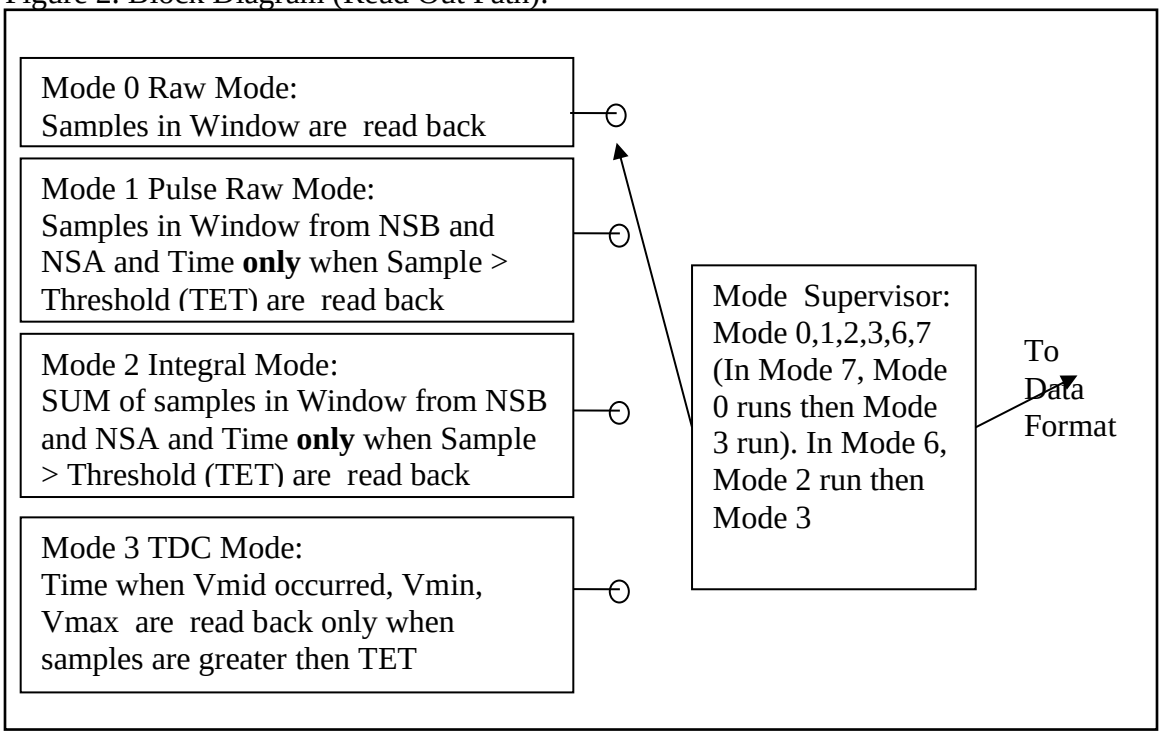


Figure 3. Trigger Path: Sample Processing Code

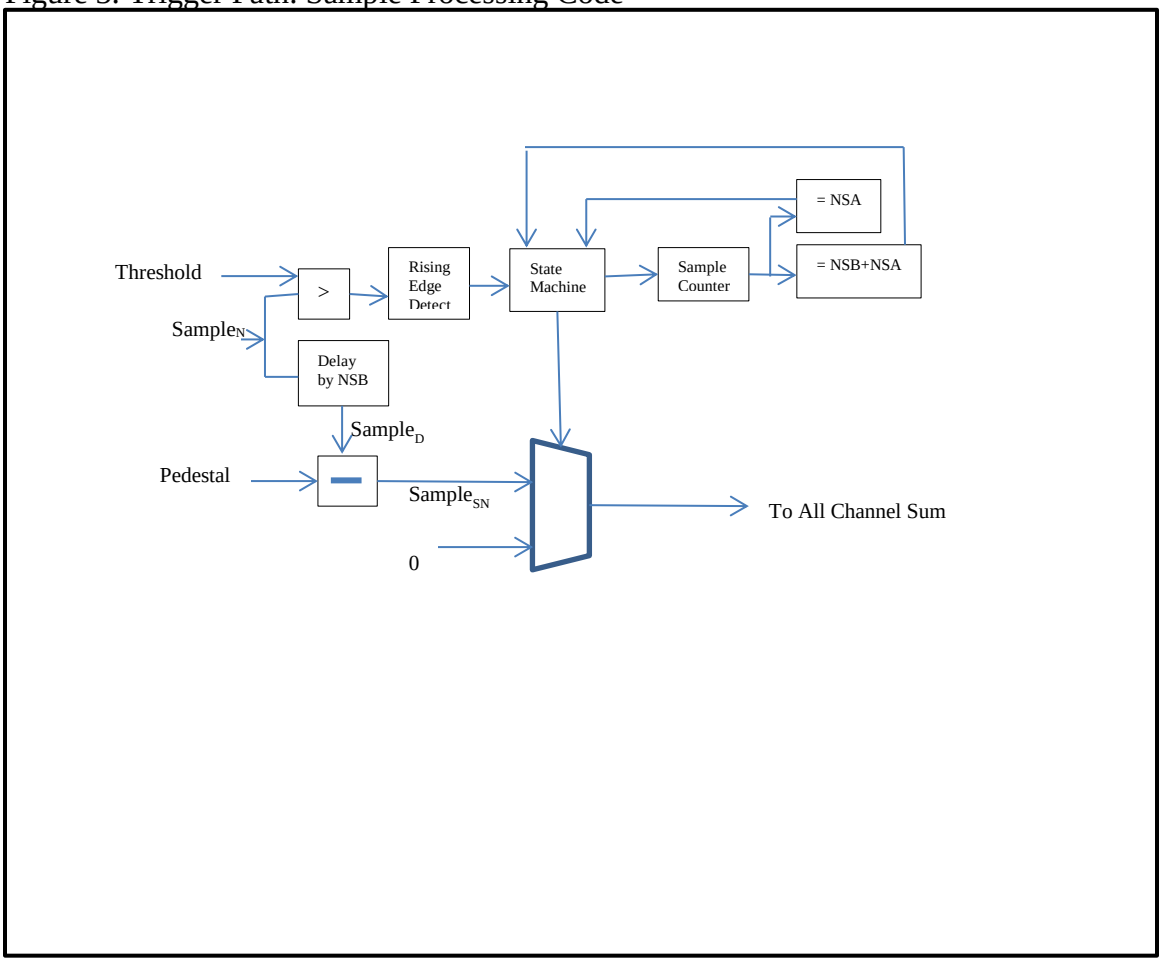
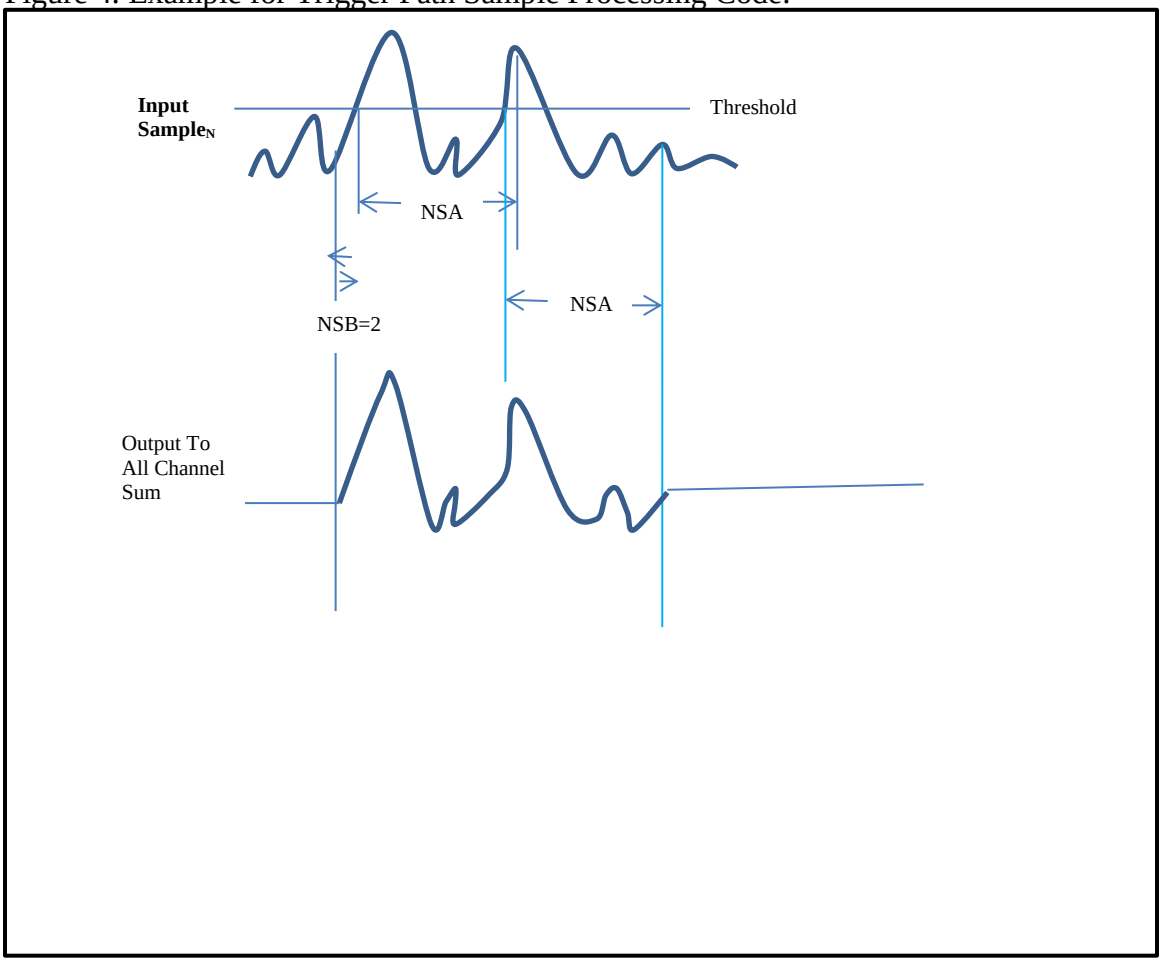


Figure 4. Example for Trigger Path Sample Processing Code:



Reset:

Hard Reset: Reset Everything except Time Stamp and ADC IC
Soft Reset: Reset Everything except Time Stamp, Registers, and ADC IC
Sync : Only reset Time Stamp.
ADC IC is reset through register bit.

Trigger Path Processing:

To implement FADC250 energy sum described in "Some specifications on the FADC250 and CTP trigger firmware for Hall D calorimeters" by A. Somov, this section of code has tow parts: Sample Processing Code and Summing Code.

Each ADC channel has Sample Processing Code that does the following:

1. When a sample is equaled or greater then a programmable threshold, it outputs a programmable (0 to 15) Number of Sample Before (NSAB) this sample and a programmable (0 to 63) Number of Sample After (NSA) this sample. NSA count includes this sample. The NSA + NSB has to be less than 63. The NSB count is one more then the value set.
2. If the pulse width is wider (greater) then NSA, NSA count will be repeated. This is repeated until at the end of NSA count, the sample is less then threshold. In other words, samples that are equaled or greater than threshold will be outputted.
3. If another pulse occurred while NSA count is in progressed, NSA count restarted. Another pulse occurred is defined as one sample less than threshold and another following sample is equaled or greater than threshold.

As shown in Figure 3, Sample Processing Code does the following:

1) Delay $Sample_N$ from ADC by NSB = 2 samples
   a) $Sample_D$ = $Sample_N$ delay by NSB.
2) Subtract 12-bits Pedestal from 12-bits ADC samples. Each ADC has its own pedestal.
   a) $Sample_{SN}$ = $Sample_D$ - Pedestal
3) When $Sample_N$ crosses 12-bits threshold (one 12-bit threshold for all 16 ADC):
   a) Input $Sample_{SN}$ to "All Channel Sum" code
   b) Start Sample Counter.
   c) When Sample Counter counts to NSB+NSA, input Zero to Sum code.
   d) If $Sample_N$ crosses 12-bits threshold again while Sample Counter is counting.
      i) Restart Sample Counter
      ii) Continue inputting $Sample_{SN}$ to "All Channel Sum code".
      iii) When Sample Counter counts to NSA, input Zero to Sum code.
4) The output is NSB + NSA + 1. One extra sample after NSA is due to pipelining the code.
5) Figure 4 shows an example.

The outputs of all 16 , Sample Processing Code are added together and are transferred to the CTRL FPGA to be sent to the CTP on two full duplex gigabit transceiver ports. The transceivers are configured to operate at 2.5Gb/s per lane and will communicate directly to the VXS switch "A" slot.

Programmable Pulse Generator (PPG):

Input to Channel Data Processing can either come from ADC after pedestal subtraction or the Programmable Pulse Generator (PPG). Users can load simulated PMT data into the PPG via VME host. When a trigger occurs in test mode, the stored data is read and apply to Channel Data Processing. There are 16 PPG, one for each ADC channel and each PPG can hold 32 samples.

1. Channel Data Processing:

Overall View:
We briefly describe the current operating modes of the FADC250 and identify the data word types that each produces. Since several modes report pulse data, we begin by discussing how pulses are defined, and how times are assigned to pulses.

Pulse Definition

Several modes of operation of the FADC250 detect a pulse from the raw samples and report data associated with this pulse. We describe the pulse detection algorithm that is currently implemented in the firmware. Exceptions are noted below.

The trigger window consists of NW samples of the ADC. Samples are 4 ns apart in time. Samples within the trigger window are numbered from 1 to NW.

A pulse window width that would contain the entire expected detector pulse is specified by the user. The pulse window width (in number of samples) is defined from two programmed values (NSB, NSA): width is NSB + NSA.

Pulse identification is initiated if at least one sample in the trigger window is above the programmed threshold. The sample number of the first threshold crossing (TC) in the trigger window is determined. If the first sample of the trigger window is above threshold, a pulse is declared and TC = 1 for this pulse. Otherwise, the transition above threshold must be explicitly visible: at least one sample before TC must be at or below threshold.

The parameter NSB represents the number of samples before the threshold crossing sample (TC) that are included in the pulse data set. When TC is found and TC – NSB < 1, the expected pulse started before the trigger window. In this case, the first sample of the pulse data set is declared to be the first sample of the trigger window.

The parameter NSA represents the number of samples after the threshold crossing to include in the pulse data set. Sample number TC is included in this count. If the computed last sample number (TC + NSA – 1) of the pulse data set is greater than NW, the expected pulse extends past the trigger window. In this case, the last sample of the pulse data set is declared to be the last sample of the trigger window.

In summary, the pulse data set consists of sample numbers:

$$\text{MAX}((TC - NSB), 1) \quad \text{to} \quad \text{MIN}((TC + NSA - 1), NW).$$

Up to three distinct pulses may be detected within the trigger window. The maximum number of pulses is programmable. The earliest pulses found, up to the programmable limit, are reported. Any additional activity in the trigger window is ignored.
After a pulse is detected, a search for a subsequent pulse begins at the sample number after the pulse (TC + NSA). Another pulse is identified only if there is a clear transition above threshold after the previous pulse: at least one sample before TC must be at or below threshold. Consecutive pulses may overlap (i.e. share samples). The magnitude of possible overlap is determined by NSB.

Pulse Time

When operating in a pulse finding mode, a time is assigned to each pulse found. The simplest reportable time is TC, the number of the first sample above threshold for the pulse. ADC modes 2 and 3 report TC as pulse time.

A procedure for calculating a high-resolution time has also been implemented in the firmware. 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.

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.

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:

$$V(N1) \leq VMID < V(N1+1)$$

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.

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,

$$TF = 64*(VMID - V(N1)) / (V(N1+1) - V(N1)).$$

TF is reported as the fine time with values from 0 to 63. In addition to the time data word (type 8) consisting of the coarse and fine times, a pulse parameter data word (type 10) reports the values of VMIN and VPEAK used in the computation of the high resolution time.

A problem in the computation of the high resolution time will occur when VMIN is greater than VPEAK. In the current implementation of the algorithm, the simplest way to protect against this situation is to require that all 4 samples that determine the VMIN must be at or below threshold for the high resolution timing algorithm to be used. If this condition is not satisfied, the reported pulse time is TC, and both VMIN and VPEAK are reported as 0 in the pulse parameter data word (type 10) to identify the condition.

A problem with the algorithm occurs if VPEAK is not found within the trigger window. In this case, the reported pulse time is TC. To identify this condition, the pulse parameter data word (type 10) reports VPEAK = 0 and VMIN as measured.

In the current implementation of the algorithm, a technical difficulty arises when TC is near the end of the trigger window. If (NW – TC) < 5, the reported pulse time is TC, and the pulse parameter data word (type 10) reports VPEAK = 0 and VMIN as measured.

To standardize reporting of the pulse time, a 15-bit time value is always formed. The coarse time is reported in bits 14 through 6, while the fine time is reported in bits 5 through 0. Each count represents 62.5 ps. In situations where only the threshold crossing TC is reported, TC is set to be the coarse time and the fine time is 0.

ADC Modes

ALL operating modes produce data types 0 (block header), 1 (block trailer), 2 (event header), 3 (trigger time), 12 (scaler – optional), 14 (data not valid), 15 (filler) (see **Appendix 1**).

1 – Raw ADC data samples. If any sample in the trigger window is above the programmed threshold, all samples of the trigger window are reported. (Data type 4).

2 – Pulse Raw ADC data samples. A pulse is identified in the manner described above. Raw samples that constitute a detected pulse are reported. If the programmed width (NSB + NSA) is odd, an extra sample is reported. The threshold crossing sample number

(TC) is also reported so that the pulse samples can be referenced to the start of the trigger window. Up to 3 pulses can be identified within the trigger window. (Data type 6.)

3 – Pulse integral. A pulse is identified in the manner described above. The sum of raw samples that constitute the pulse data set is reported. The threshold crossing sample number TC is reported as the pulse time. Up to 3 pulses can be identified within the trigger window. (Data types 7, 8).

4 – High-resolution time. A high-resolution time for the pulse is reported. Parameters used to compute this time are also reported. Up to 3 pulses can be identified within the trigger window. (Data types 8, 10).

7 – Pulse integral + high-resolution time. Equivalent to the processes of mode 3 and 4. Note that the time in mode 3 is replaced by the high-resolution time of mode 4. (Data types 7, 8, 10).

8 – Raw ADC data samples + high-resolution time. Equivalent to the processes of mode 1 and 4. (Data types 4, 8, 10.) (This mode is used to validate the implementation of the high-resolution time algorithm.)

Recommended modes of operation are 1, 2, 3, 7.

Firmware Implementation

As in most complex projects, the FADC250 firmware was developed in stages. Pulse detection mode 3, reporting the pulse integral and threshold crossing time TC, was delivered early. The high resolution timing algorithm was developed later and added as an independent process so it would not disrupt the functioning modes of operation.

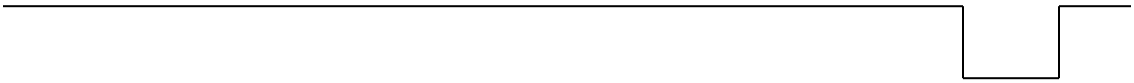
As a consequence of having separate processes for pulse identification and timing, we have observed rare occasions where there is a mismatch in the number of pulses identified (pulse integral) and the number of pulse times reported. To be safe, we suggest that the user eliminate this data from consideration.

A future version of the firmware is planned that combines the pulse finding and high resolution algorithms into a single process. This will eliminate the pulse and time mismatches described above, and soften some of the restrictions we have imposed on when the high resolution timing mode can be applied. It will also increase trigger rate capability of the module.

Mode Illustrations:

ADC Data

Trigger Input



Time Line

|←Programmable Trigger Window→|
-------- 100nS to 2uS ----------

|←----------Programmable Latency (100nS to 8uS ----------→ |

Data from ADC are stored continuously in circular buffer until Trigger input becomes active (low). The data that was stored from the time that the Trigger occurs back to the time specified by Programmable Latency within the Programmable Trigger Window are processed.

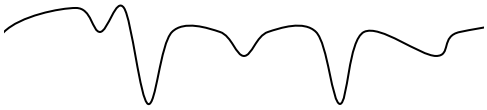
There are three main options to which these data are processed. The options are selectable by the user via VME register setting and two Trigger Inputs.

While data are being processed, ADC FPGA will continue storing incoming ADC data with no loss of data. Programmable Trigger Window (PTW) and Programmable Latency(PL) are common to all 8 ADC channels.

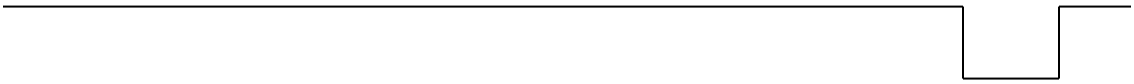
Mode 0 (Raw Mode):

Data within the Programmable Trigger Window [PTW] is passed with no further processing to the VME Host.

Option 1 Raw Mode Data to VME Host Illustration:



Trigger Input



Time Line

|←Programmable Trigger Window→|

|←----------Programmable Latency ----------→|

Mode 1 (Pulse Mode) :

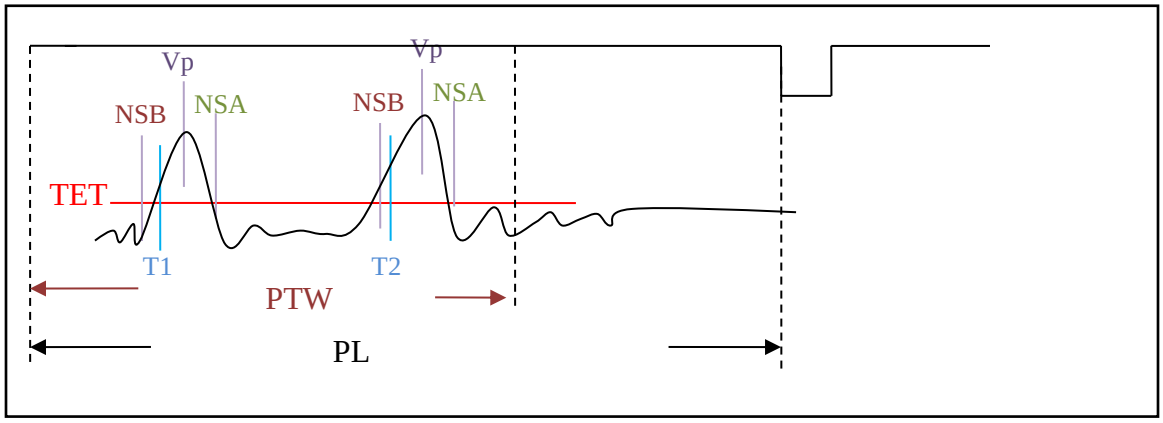
When an ADC sample has a value that is greater than Programmable Trigger Energy Threshold (TET), the number of samples before (NSB) the Maximum value (Vp) and the number of samples after (NSA) Vp are sent to VME Host. NSB and NSA are programmable. T1 and T2 are Time at which the first sample is greater than TET.

TET is 12 bits and unique to each ADC channel.

NSB has a maximum value of 512

NSA has a maximum value of 512

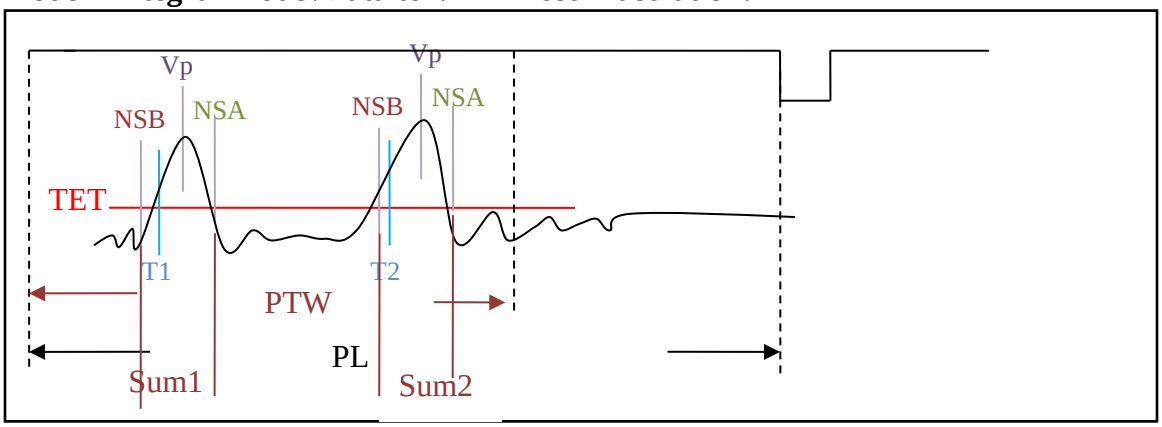
Mode 1 Pulse Mode Data to VME Host Illustration:



Mode 2 Integral Mode:

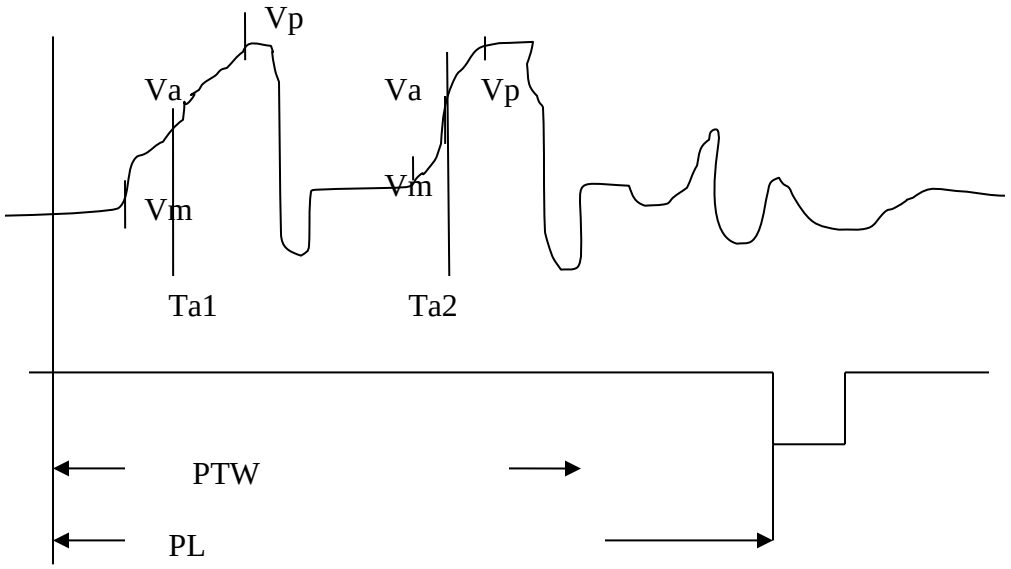
Data within NSB and NSA of Option 2 Raw Mode are summed around T1 and T2. PNS defines the number of samples before and after T1 and T2 include in Sum 1 and Sum 2 respectively. Only Sum 1, T1, Sum 2, and T2are passed to VME FPGA. T1 and T2 are Time at which the first sample is greater than TET.

Mode 2 Integral ModevData to VME Host Illustration:



Mode 3 TDC Algorithm:

The TDC algorithm calculates time of the mid value (Va) of a pulse relative to the beginning of the look back window. Va is the value between the smallest and the peak value (Vp) of the pulse. The smallest value (Vm) is the beginning of the pulse. The time consists of coarse time and fine time. The coarse time is the number of clock the sample value before the mid value and the fine time is the interpolating value of mid value away from next sample. The coarse value is 10 bits and the fine value is 6 bit. The resolution of LSB is 1/(CLK * 64). For a 250MHz the resolution is 62.5 pS. For example for a 20MHz clock, a pulse time (Ta) value of 110 means the mid-point of the pulse occurred at 6.875nS (62.5pS * 110) from the beginning of look back window.



Requirements for TDC Algorithm:

i) There must be at least 4 samples (background) before pulse. Four of these samples are used to determine the pedestal (Vnoise) floor. The minimum value of the pulse is the first value that is greater than Vnoise.

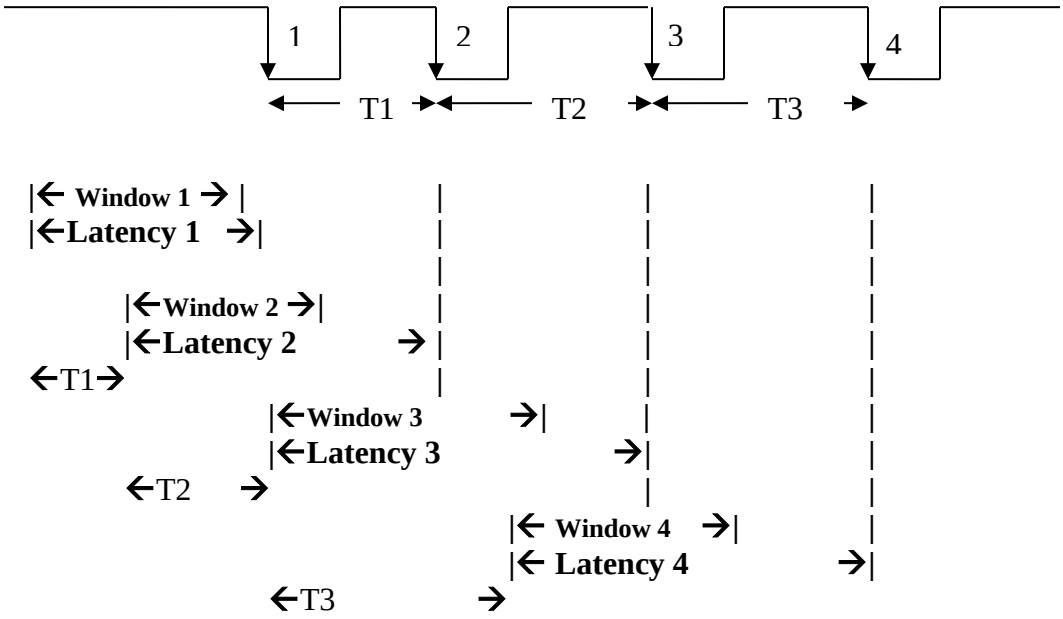
Mode 6 Algorithm:

Run Mode 2 follow by Mode 3 algorithms. Only the TDC time is reported back when the condition for TDC is met, else time from Mode 2 is reported back.

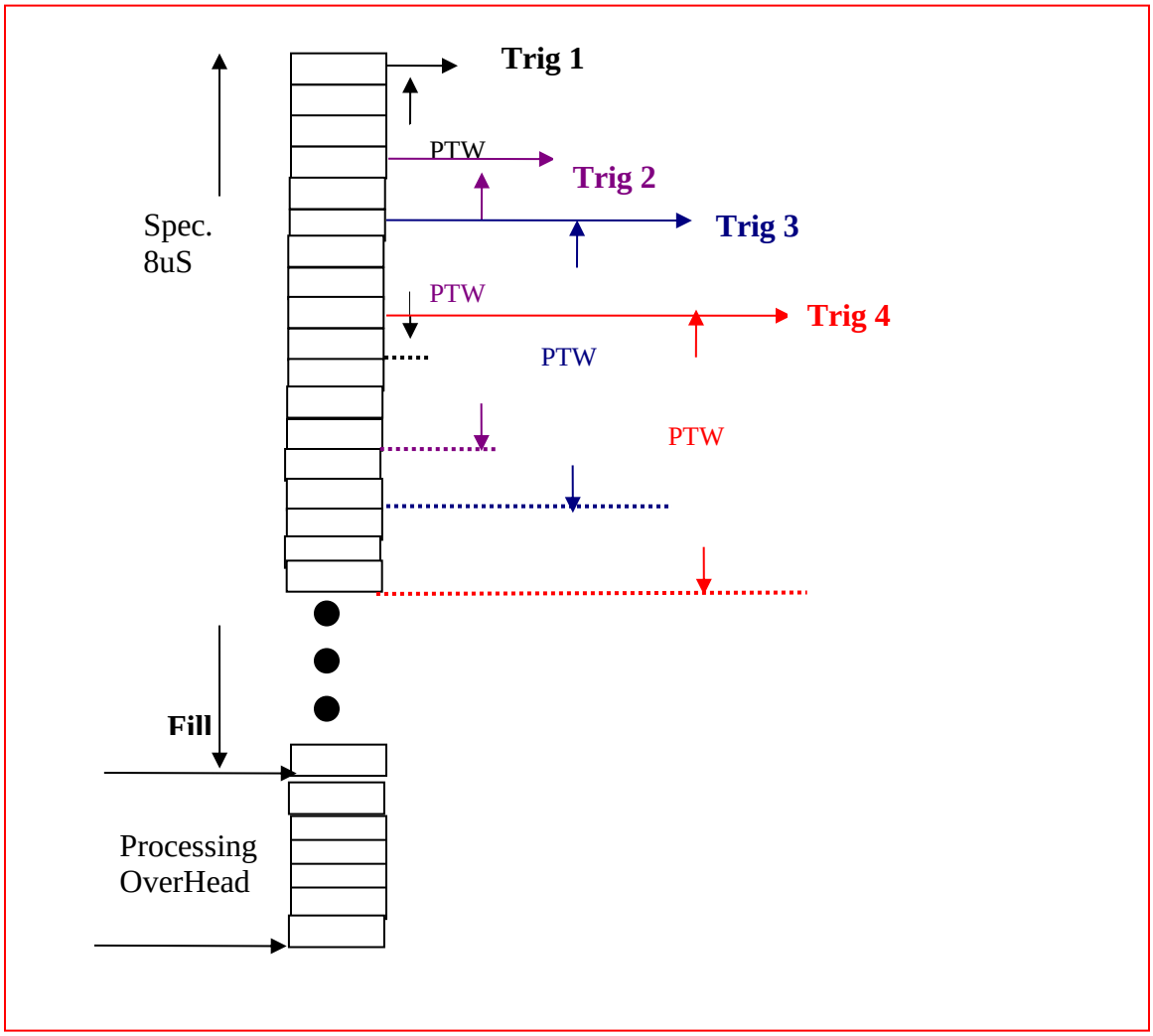
Trigger Input Buffer:

In the event that the Trigger Input rate is faster than the data processing time, the processing algorithm has to be able to process 100 consecutives triggers with no loss in time lines. If a trigger cannot be processed due to an overflow condition, the VME FPGA will be notified: "no data for trigger. If T1, T2, or T3 is less than 50 Ns, the trigger will not be recorded.

Successive Trigger Input Illustration:



Memory Model for Successive Trigger Input Illustration:

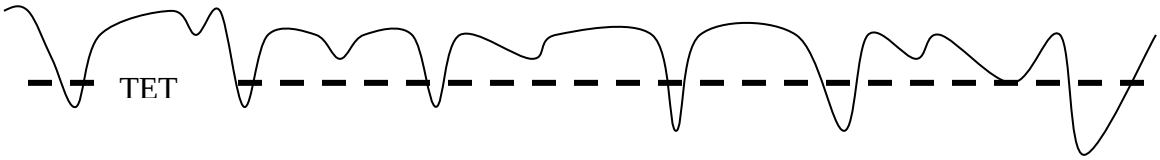


3. HITBITS:

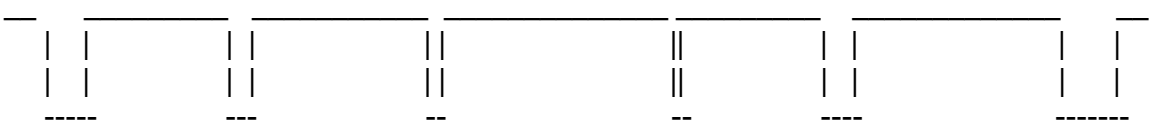
When counts from an ADC channel are greater than threshold, the corresponding Hit Bit for that channel is high. The HitBits are processed by TRIG_PROC_TOP to form coincident trigger.

HitBits Illustration:

ADC data

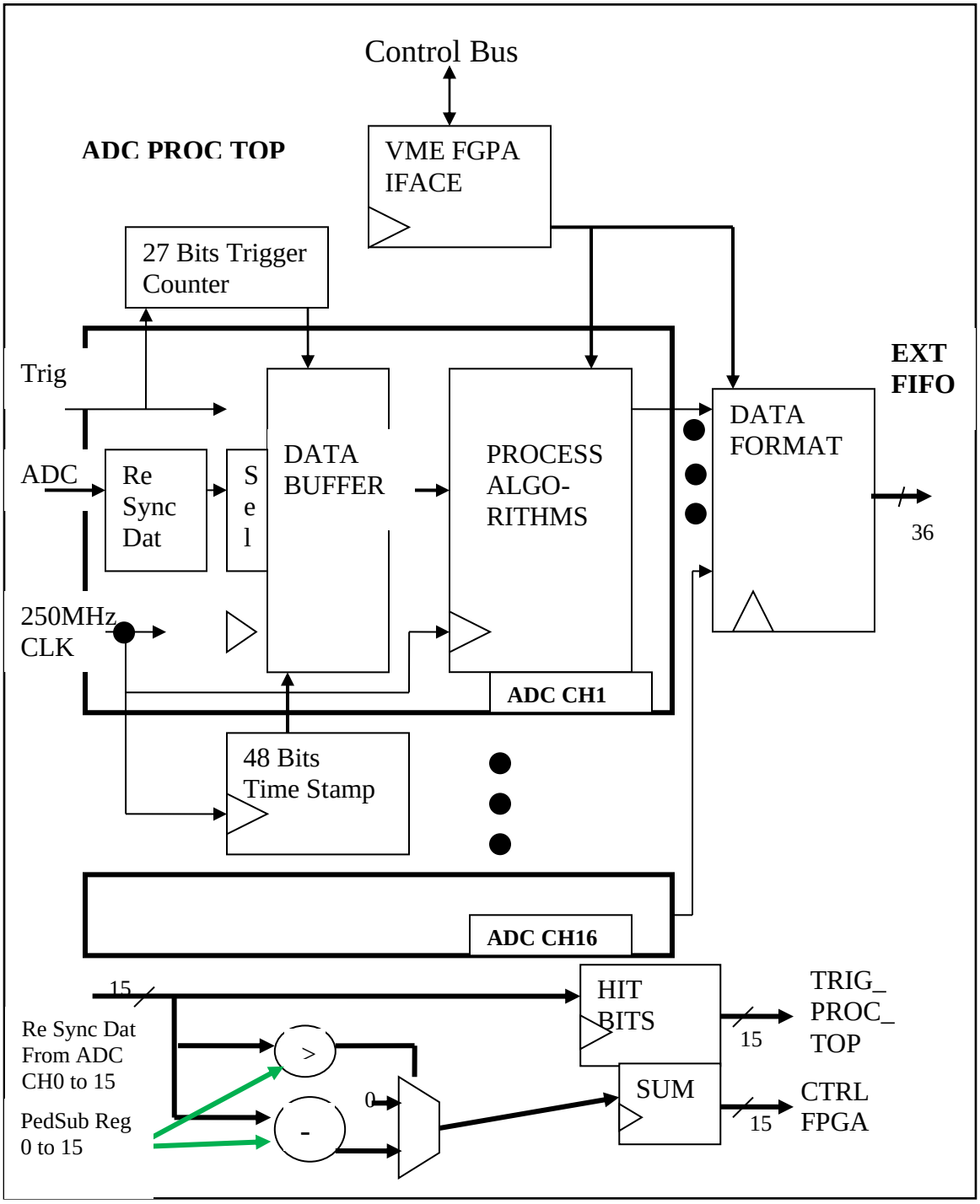


Hit Bit



Conceptual Architecture Diagram

Overview:


Control Bus
ADC PROC TOP
VME FGPA IFACE
27 Bits Trigger Counter
Trig
ADC
Re Sync Dat
S e l
DATA BUFFER
PROCESS ALGO-RITHMS
DATA FORMAT
EXT FIFO
36
250MHz CLK
ADC CH1
48 Bits Time Stamp
ADC CH16
15
Re Sync Dat From ADC CH0 to 15
PedSub Reg 0 to 15
>
-
0
HIT BITS
15
TRIG_ PROC_ TOP
SUM
15
CTRL FPGA

Data from each ADC is resynchronized with FPGA main CLK. The outputs of the Resync are inputs of Data Buffer, Pedestal Substraction, and Hit Bits circuits. Each ADC Channel has Resync, Data Buffer and Processing Circuits. The Data Buffer stores Resync Data, Trigger Number, and Time Stamp. Processing Circuit processes data from Data Buffer. Format read results from each ADC channels (0-15) Processing Circuit and mux it to external FIFO. For each of the ADC, Pedestal Subtraction Circuit subtracts a programmable constant from ADC sample if the sample is greater than the constant. If the sample is smaller than the constant, zero is output. The output of the Pedestal Subtraction Circuit is fed to the Sum circuit. Sum circuit adds the pedestal-subtracted-samples from each Pedestal Subtraction Circuit on a clock by clock basis. Bit Bits circuit compare Resync data to TET and produce a low active signal when Resync data is above TET.

The architecture supports Processing Modularity. Processing algorithms are independent of the other functions.

Sel block was added on March 3, 2008 to accommodate both 10 bits and 12 bits FADC boards This feature also allows individual ADC Channel values (counts) to be set to zero (effectively disable the ADC). CONF register (see below) configures these options.
On March 15, Sel block is expanded to include Programmable Pulse Generator.

Programmable Pulse Generator (PPG):

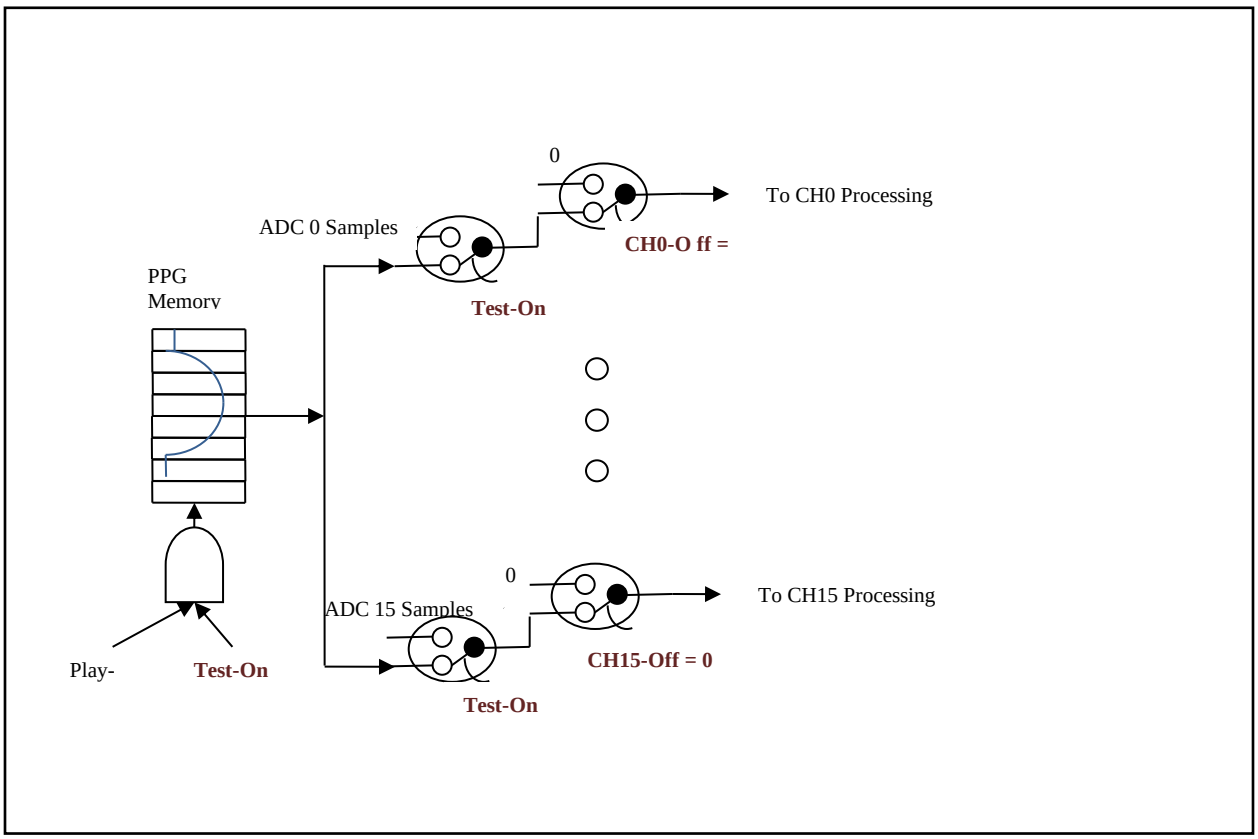
The PPG generates pulses by reading out digitized values of pulses (samples) stored in a memory. The memory has 32 locations. Each location has 16 bits and can hold one sample. Thirteen of the bits simulated the 12 data and 1 overflow bits output of the ADC implemented on the FAD250 board. The 16th bit facilitates writing and reading the memory. Samples are written to the PPG memory when VME write to address (0x0211 and 0x0011) and bit 16th is a one. The address automatically incremented after a sample is written. The last two samples written are required to have bit 16 zeroed (Sample value = 0x0000). After the last samples (0x0000) the address reset to the location of the first sample. Bits 14 and 15 are don't care bits. VME can verify the data is written by immediately read back the data (write follow by read).

Data are read out of PPG memory when Play-Back and Test-On are both logical one. The first location that will be read out is set by a register (Read-Out-Start-Address). Subsequent locations are read out at 4nS interval until Play-Back returns to logical zero. The read back cycles to Read-Out-Start-Address when bit 16th is a zero.

AUX_IO(1) is used as Play-Back.
Bit 7 of CONFIG register is used as Test-On
Bit 8-15 of CONFIG register is used as CHx-OFF

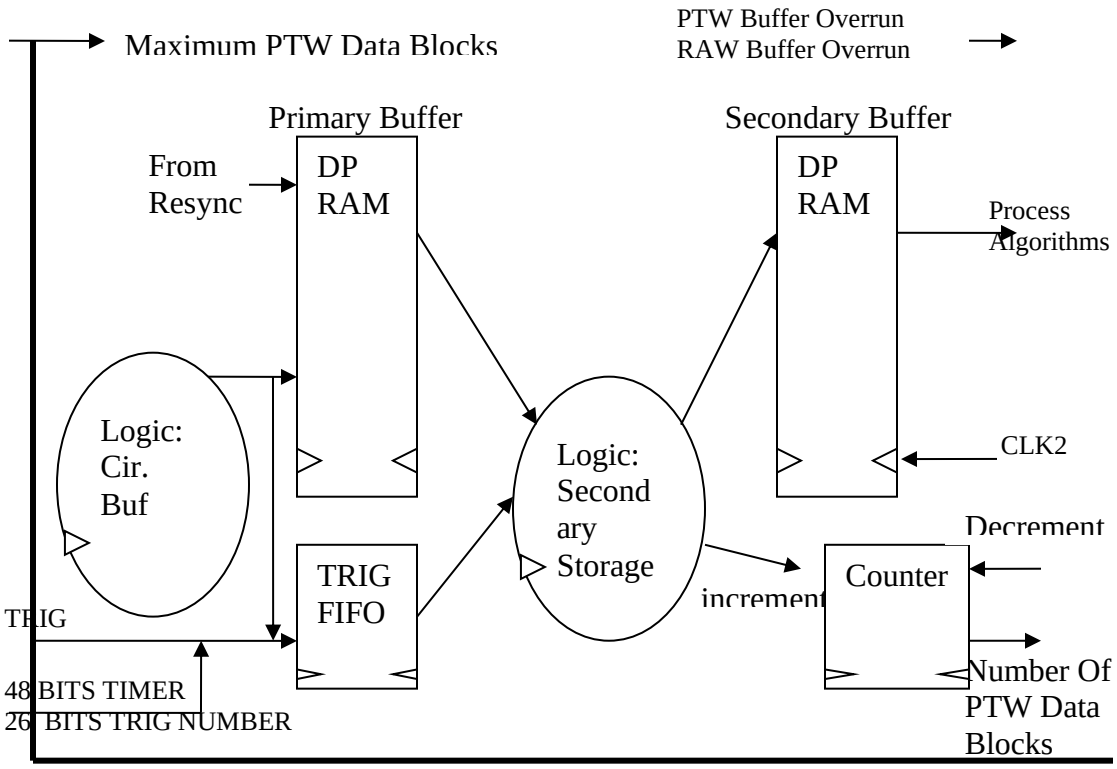
For the FADC250 Version 1 the PPG can only hold 16 samples. The last two samples have to be written with 0x8000.

PPG Mode:



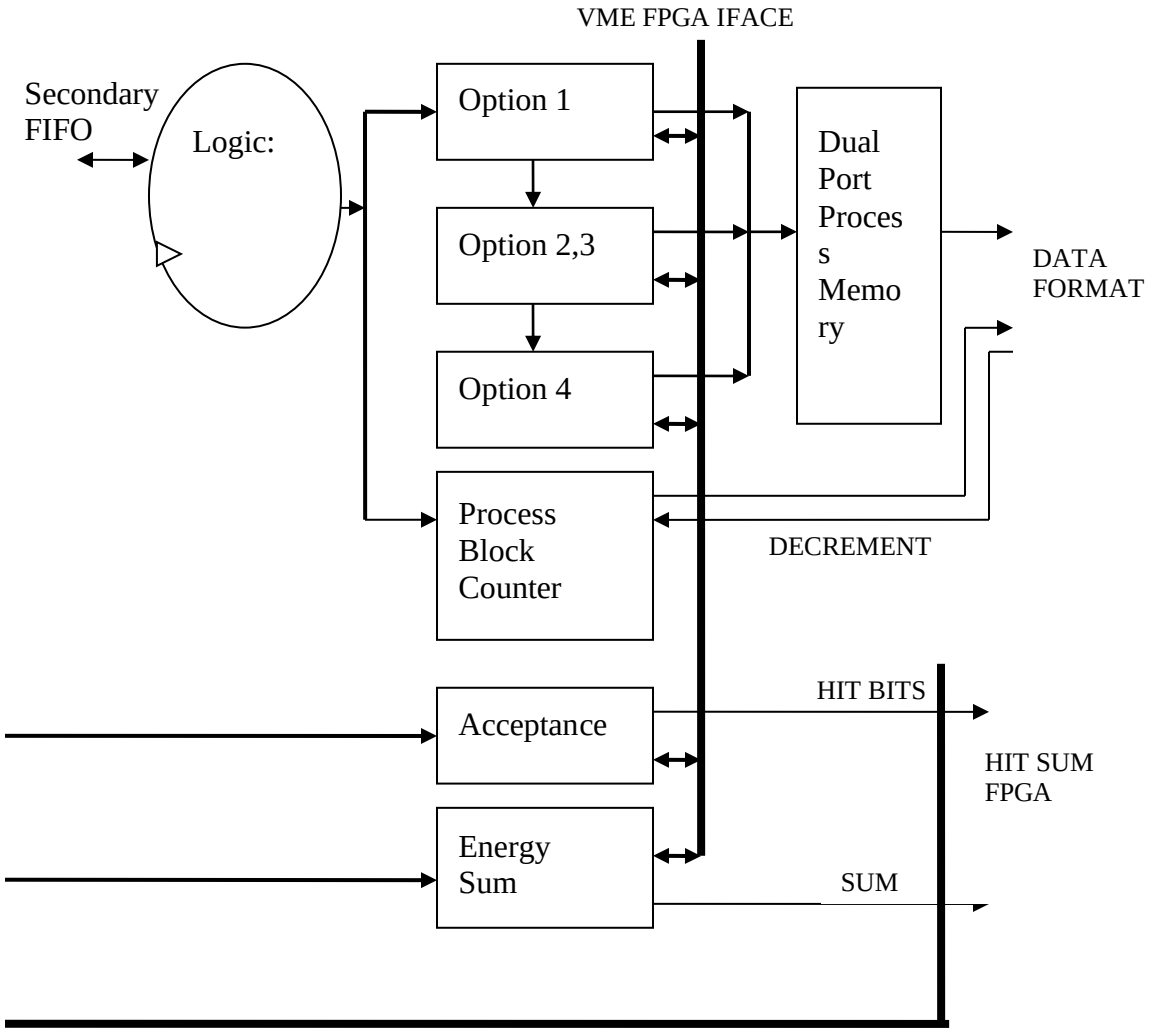
Test On is bit 6 of Configuration register. CHx-Off are bits 8-15 of configuration register.

Data Buffer:



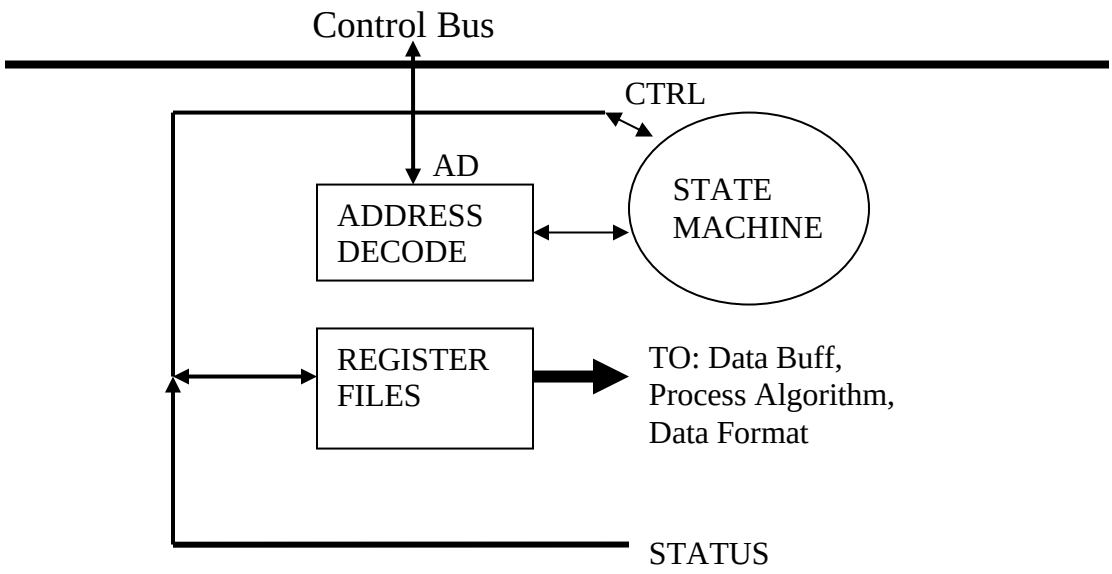
1. Synchronize data from ADC to 250 MHz FPGA CLK
2. Store ADC data to Primary Buffer. Implement Primary Buffer as ring (circular) buffer.
3. When a Trigger occurs, the trigger is stored along with the values of the 48 Bits Timer and the Pointer of the Primary Buffer in a FIFO.
4. For each trigger, data within Programmable Trigger Window are copied from Primary Buffer to Secondary Buffer with time stamps and markers necessary for further processing. After the block is copied, Number of PTW Data Blocks increments by one.
5. After a block is read and process, Decrement should be pulsed to decrease th Number of PTW Data Blocks by one.
6. Each ADC channel has its own Data Buffer.
7. When the Trigger Rate is faster then the time needed to copy ADC Data from Primary Buffer to Secondary Buffer, RAW BUFFER OVERRUN is set and remain set until RESET_N or SOFT_RESET_N goes low.
8. When Number of PTW Data Block is equaled Maximum PTW Data Blocks setted by the host, PTW Buffer Overrun sets and remains set until RESET_N or SOFT_RESET_N goes low.
9. Utilized 700 LUT, six 18000-bits RAM blocks. Max Clock is 252 MHz.

Process Algorithms:



1. Read data from Secondary Buffer.
2. Parse data to Processing Algorithms
3. Process all three options of Data Channel Processing.
4. Create Acceptance (Hit bit) pulse
5. Compute Energy Sum

VME FPGA IFACE:

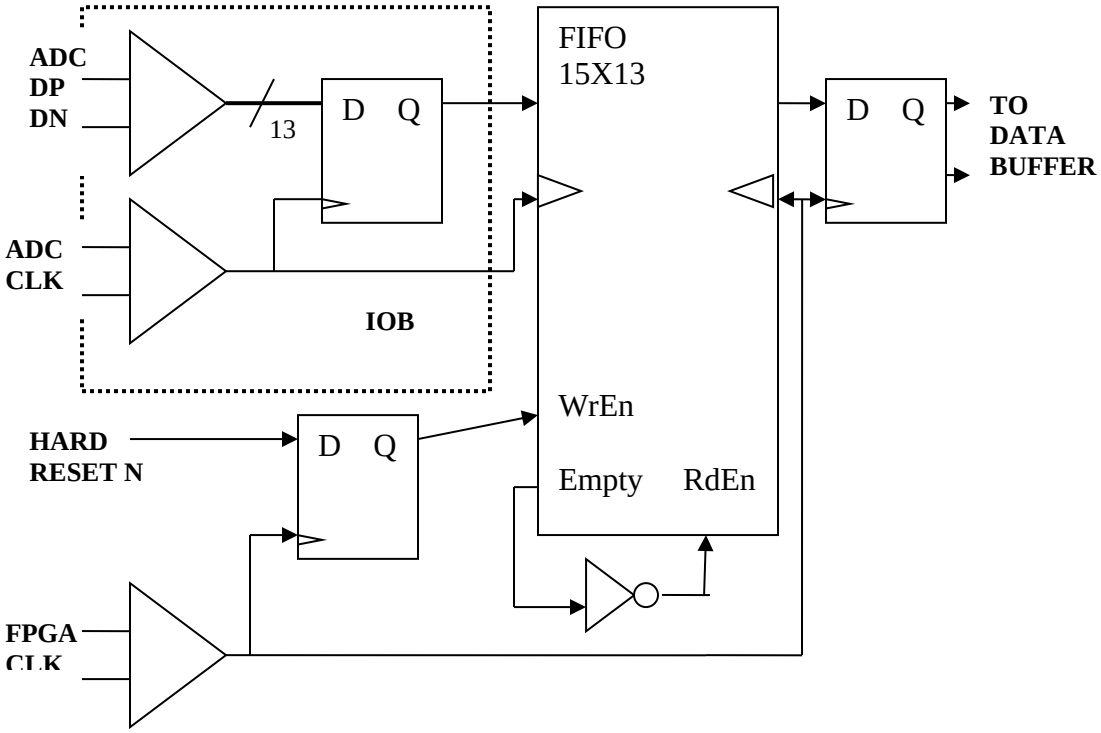


VHDL Hierarchy

1. **ADC_PROC_TOP**
    a. SYNC_ADC_IN_VER2
        i. IODELAY
    b. PlayBack_WV_Ver2
        i. DPRAM_16_1024 (UMEM)
    c. Data_Buffer_AllCh_Ver2_TOP
        i. Data_Buffer_Top (UADCx)
            1. DP_RAM1_TOP (2Kx13) (URAW_BUFFER_IN)
            2. FIFO_1 (UTrigger_Buffer)
            3. DP_RAM2_TOP(2Kx17) (UPTW_DATA_BUF)
            4. PTWCPSM
    d. TimeStamp_TOP
        i. Time_stamp (Xilinx core gen)
    e. Trigger_Number_TOP
        i. Trigger_number
    f. Processing_All_Ver2_Top
        i. PROCESSING_TOP (CHx_PROCESSING)
            1. DP_RAM3_TOP (2Kx18) (UPROCESS_BUF)
            2. PROCESSM
            3. fifo_12_64 (UProcAdrHist)
            4. TDC_TOP
                a. Linear_Interpolation (ULI)
                    i. Divide_18By12
                        1. DIVIDESM
                b. TDCSM
        ii. PROALLSM
    g. DataFormat_VER2_TOP
        i. DATFORSM
    h. TriggerProcessing_TOP
        i. TriggerProcessing
            1. Delay1_To_16
            2. Pedestal Sub
            3. TRPATHSM
        ii. LVDS_OUT_16Bits
    i. Hit_Bit_All_ver2_Top (UHIT_BITS_ALL_TOP)
        i. HIT_BITS_TOP (UHIT_x)
        ii. IOREG_8bits
2. **HOST_ADCFPGA_VER2_TOP**
    a. HSHOSTSM

VHDL Block Diagram

ADC Input ReSync



Each ADC has 12 bits data, an overflow, and an ADCCLK. The ADC Input Resync captures ADC's data and overflow bits with ADC's output clock to a 15 deep (smallest allow by ISE) by 13 bits FIFO. The FIFO allows the FPGA main CLK to be independent of ADC clock. The FPGA main CLK clocks the data out of FIFO and send to the Data Buffer Block.

The advantage of using ADC's own CLK to capture its data is the elimination of timing variations from ADC to ADC. Moreover, the FIFO Empty signal is used as FIFO Read Enable to allow variation in ADC start up time.

Data Buffer

Primary Memory Map

Address Location	Content
0	ADC Data 0
1	ADC Data 1
2	ADC Data 2
3	ADC Data 3
4	ADC Data 4
:	:
:	:
:	:
4078	ADC Data 4078
4079	ADC Data 4079
4080	ADC Data 4080
0	ADC Data 4081
1	ADC Data 4082
2	ADC Data 4083
3	ADC Data 4084
:	:
:	:
:	:

Primary Memory stores ADC data as it comes in. At the end of buffer, the storing re-circulates and overwrites previous data.

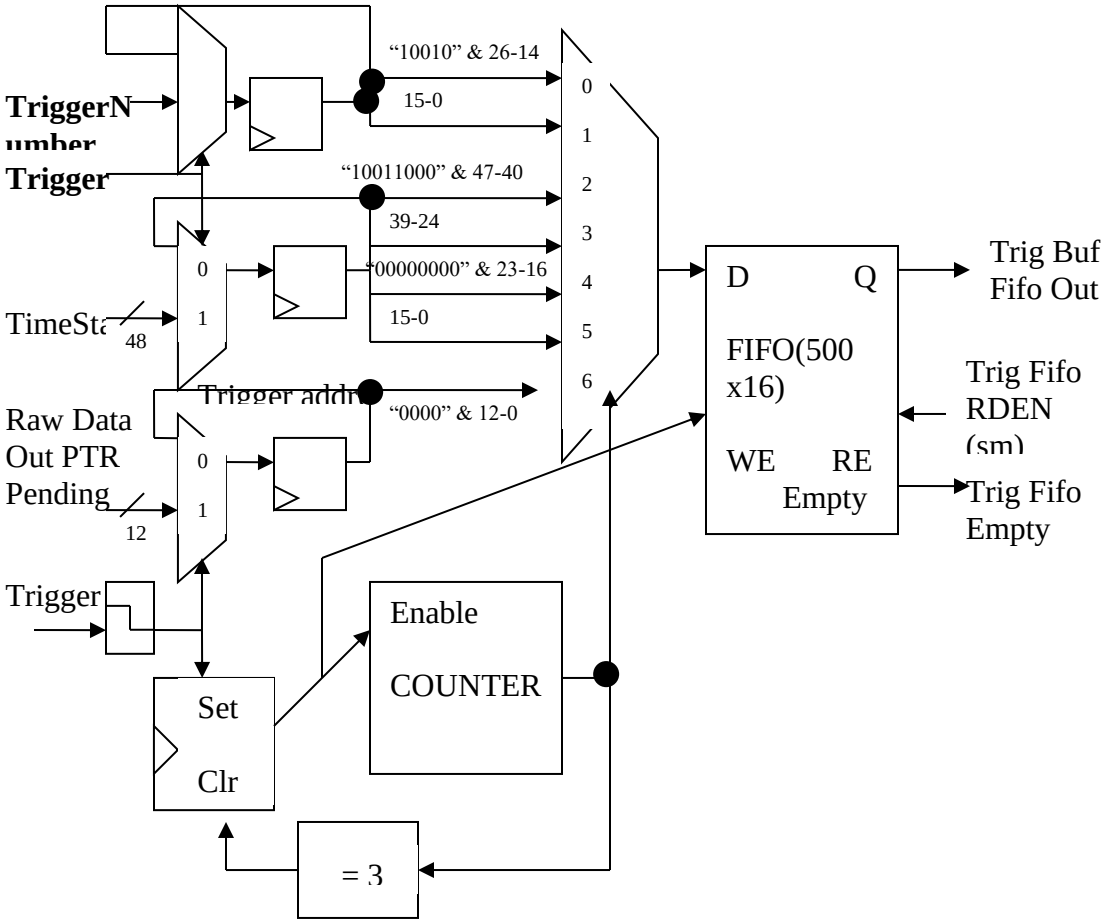
Data Buffer
Secondary Memory Map

Memory location from beginning of PTW	**Content**
0	PTW **0** “10010” Trigger Number bits 26-16
1	Trigger Number bits 15-0
2	“10011000” Time Stamp bits 47-40
3	Time Stamp bits 39-24
4	“00000000” Time Stamp bits 23-16
5	Time Stamp bits 15-0
6	PTW **0** data 0
:	PTW **0** data 1
:	:
N-4	“111” PTW **0** data N-4. “111” indicate almost last data
N-3	PTW **0** data N-3
N-2	PTW **0** data N-2
N-1	PTW **0** data N-1
N	PTW **0** last data
N+1	PTW **1** “10010” Trigger Number bits 26-16
N+2	Trigger Number bits 15-0
N+3	“10011000” Time Stamp bits 47-40
N+4	Time Stamp bits 39-24
N+5	“00000000” Time Stamp bits 23-16
N+6	Time Stamp bits 15-0
N+7	PTW **1** data 0
	PTW **1** data 1
	:
M-4	“111” PTW **1** data M-4. “111” indicate almost last data
M-3	PTW **1** data M-3
M-2	PTW **1** data M-2
M-1	PTW **1** data M-1
M	PTW **1** last data
M+1	PTW **2** “10010” Trigger Number bits 26-16
M+2	Trigger Number bits 15-0
M+3	“10011000” Time Stamp bits 47-40
M+4	Time Stamp bits 39-24
M+5	“00000000” Time Stamp bits 23-16
M+6	Time Stamp bits 15-0
M+7	PTW **2** data 0
	PTW **2** data 1
	:

O-4	“111” PTW **1** data O-4. “111” indicate almost last data
O-3	PTW **2** data O-3
O-2	PTW **2** data O-2
O-1	PTW **2** data O-1
O	PTW **2** last data

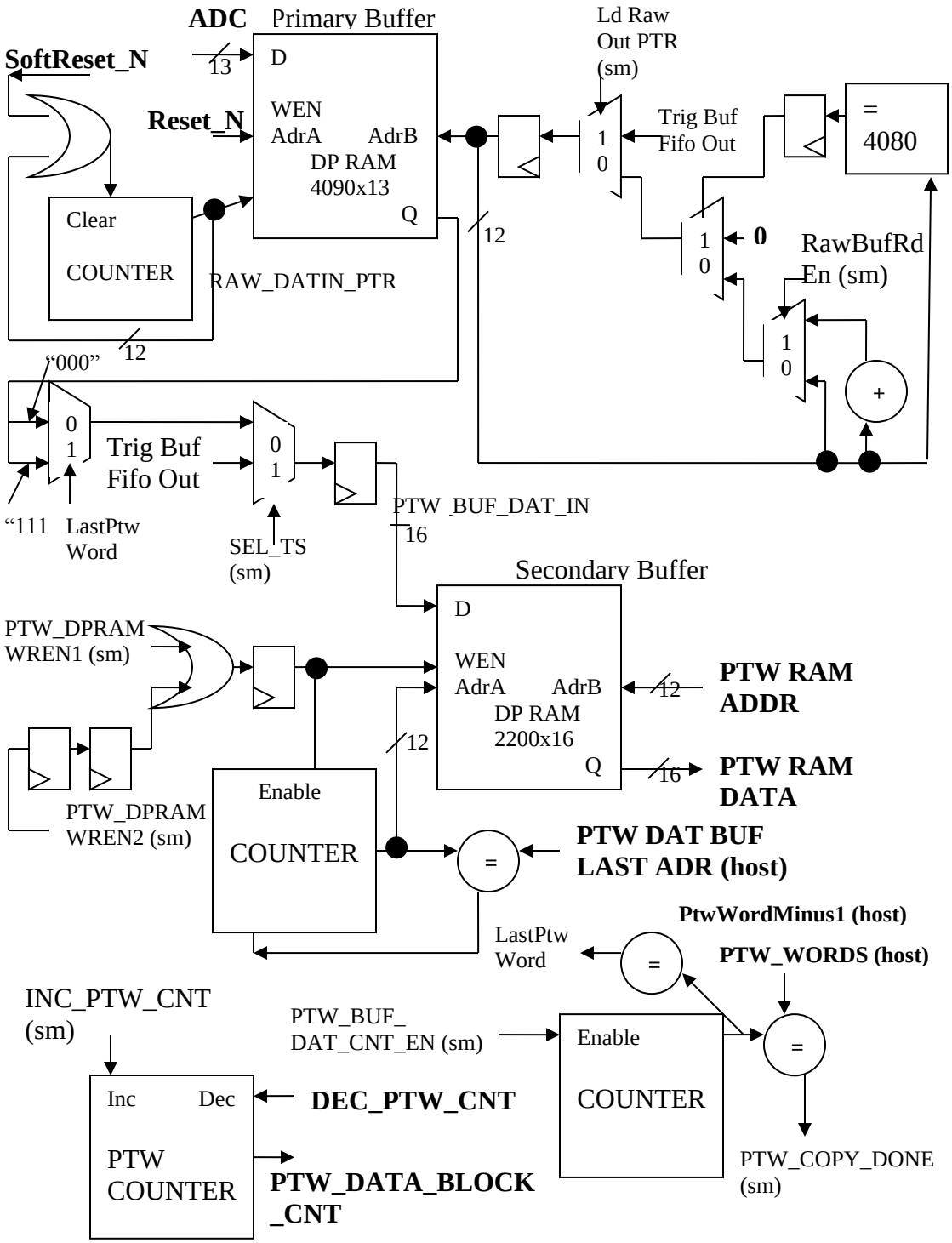
When a trigger occurs, a number of ADC data words (=PTW*25MHz) is copied from Primary to Secondary Buffer. The time at which the trigger occurred and the Trigger Number of Bits is included. Since the Number of ADC data words effects where the buffer ended and to minimize gate count, the location of the end of the buffers is provided by the Host Interface block. The Secondary Buffer Size is 2040 to accommodate 4 successive triggers of 2uS PTW (500 locations per trigger).

Trigger Buffer



When a trigger occurs, the time stamp and the pointer that points to beginning of Programmable Trigger Window (Raw Data Out PTR Pending) is store to 16 bit FIFO. The 48-bits time stamp is stored in 4 consecutive locations with LSB stored first. Bits 11-0 is padded with "1100" to signify the beginning of PTW window and Time Stamp Words. Bits 23-12, 35-24, and 47-36 are padded with "0100" to signify Time Stamp. After the first word is stored, TrigFifoEmpty goes high and kick off the State Machine to copy time stamp from FIFO to Secondary Dual-Port memory. Data in the PTW stored in the Primary Buffer starting at Trigger Address are copied to Secondary Buffer.

Data Buffer:
Primary and Secondary Buffer



After power up, data from ADC is stored in Ring Buffer continuously. When Trigger is in Trigger Buffer, the Time Stamp is copied from the Trigger Buffer to the Secondary Buffer. The Primary Address when the trigger occurred is retrieved from the Trigger Fifo to be used as the starting Primary address to copy ADC data over. A counter is keeping track of the number of ADC words copied. When the counter equaled the PTW words the copied process stop. Another counter that keeps track of the number of triggers that are in the Secondary Buffer ready for Process algorithm. When a block of trigger is process, this counter is decrement by the Process algorithm.

The Secondary Buffer storage is such that the starting address of each block of trigger data is determine by the PTW but it is fixed with PTW. For example, if PTW is 2uS, the starting address are 0, 504, 1008, 1512. The data formats from low to high address are

"1000" "TS bits 47-36"
"1000" "TS bits 35-24"
"1000" "TS bits 23-12"
"1000" "TS bits 11-0"
"010" " TriggerNumber bits 26-14"
"01" " TriggerNumber bits 13-0"
"000" "ADC data"
:
:
"001" "Last ADC data in PTW"

PTW Counter is coded such that when decrement commands and increment commands occurs exactly at the same time, decrement occurs before increment.

Data Buffer:

STATUS

Data Processing:
Memory Map

Data Processing Memory Assignment for Mode 0

Memory location from beginning of PTW	Content (WITH EVENT)
0	"00" "10010" Trigger Number bits 26-16
1	"00" Trigger Number bits 15-0
2	"00" "10011000" Time Stamp bits 47-40
3	"00" Time Stamp bits 39-24
4	"00" "00000000" Time Stamp bits 23-16
5	"00" Time Stamp bits 15-0
6	"00" PTW data 0
7	"00" PTW data 1
8	"00" PTW data 2
9	"00" PTW data 3
etc	etc
N+7	*"11" "FFFF" : end of PTW
Memory location from beginning of PTW	**Content (WITHOUT EVENT)**
0	"00" "10010" Trigger Number bits 26-16
1	"00" Trigger Number bits 15-0
2	"00" "10011000" Time Stamp bits 47-40
3	"00" Time Stamp bits 39-24
4	"01 "00000000" Time Stamp bits 23-16
5	"01" Time Stamp bits 15-0
6	"01" "0000"
7	"01" "0000"
:	:
:	:
N+6	N
N+7	"11" "0000" : end of PTW

N = PTW

Data Processing Memory Assignment for Mode 1:

Memory location from beginning of PTW	Content (WITH EVENT)
0	"00" "10010" Trigger Number bits 26-16
1	"00" Trigger Number bits 15-0
2	"00" "10011000" Time Stamp bits 47-40
3	"00" Time Stamp bits 39-24
4	"00" "00000000" Time Stamp bits 23-16
5	"00" Time Stamp bits 15-0
6	"10" "0000" Pulse Number "00" SampleNumber from Thredhold bits 9-0
7	"00" PTW pulse 0 data 0
8	"00" PTW pulse 0 data 1
N	"00" PTW pulse 0 data last
N+1	"10" "0000" Pulse Number "01" SampleNumber from Thredhold bits 9-0
N+2	PTW pulse 1 data 0
N+3	PTW pulse 1 data 1
M	PTW pulse 1 data last
M+1	"10" "0000" Pulse Number "10" SampleNumber from Thredhold bits 9-0
M+2	PTW pulse 2 data 0
M+3	PTW pulse 2 data 1
O	PTW pulse 2 data last
O+1	"10" "0000 Pulse Number "11" SampleNumber from Thredhold bits 9-0
O+2	PTW pulse 3 data 0
O+3	PTW pulse 3 data 1
P	PTW pulse 3 data last
P+1+7	"11" "0000" : end of PTW
Memory location from beginning of PTW	**Content (WITHOUT EVENT)**
0	"00" "10010" Trigger Number bits 26-16
1	"00" Trigger Number bits 15-0
2	"00" "10011000" Time Stamp bits 47-40
3	"00" Time Stamp bits 39-24
4	"01" "00000000" Time Stamp bits 23-16
5	"01" Time Stamp bits 15-0
6	x"10000"
7	x"10000"
8	"11" "0000" : end of PTW

Data Processing Memory Assignment for Mode 2:

Memory location from beginning of PTW	**Content (WITH EVENT)**
0	**“00” “10010” Trigger Number bits 26-16**
1	**“00” Trigger Number bits 15-0**
2	**“00” “10011000” Time Stamp bits 47-40**
3	**“00” Time Stamp bits 39-24**
4	**“00” “00000000” Time Stamp bits 23-16**
5	**“00” Time Stamp bits 15-0**
6	**“10” “0000” Pulse Number “00” SampleNumber from Thredhold bits 9-0**
7	**“00” Pulse 0 Sum bits 18-3**
8	**“00” “0000000000000” Pulse 0 Sum bits 2-0**
9	**“10” “0000” Pulse Number “01” SampleNumber from Thredhold bits 9-0**
10	**“00” Pulse 1 Sum bits 18-3**
11	**“00” “0000000000000” Pulse 1 Sum bits 2-0**
12	**“10” “0000” Pulse Number “10” SampleNumber from Thredhold bits 9-0**
13	**“00” Pulse 2 Sum bits ~~18-3~~20-5**
14	**“00” “0000000000000” Pulse 2 Sum bits ~~2-0~~ 4-0**
15	**“10” “0000” Pulse Number “11” SampleNumber from Thredhold bits 9-0**
16	**“00” Pulse 3 Sum bits 18-3**
17	**“00” “0000000000000” Pulse 3 Sum bits 2-0**
18	**“11” “0000” : end of PTW**
Memory location from beginning of PTW	**Content (WITHOUT EVENT)**
0	**“00” “10010” Trigger Number bits 26-16**
1	**“00” Trigger Number bits 15-0**
2	**“00” “10011000” Time Stamp bits 47-40**
3	**“00” Time Stamp bits 39-24**
4	**“01” “00000000” Time Stamp bits 23-16**
5	**“01” Time Stamp bits 15-0**
6	**x”10000”**
7	**x”10000”**
8	**“11” “0000” : end of PTW**

Data Processing Memory Assignment for Mode 3:

Memory location from beginning of PTW	Content (WITH EVENT)
0	"00" "10010" Trigger Number bits 26-16
1	"00" Trigger Number bits 15-0
2	"00" "10011000" Time Stamp bits 47-40
3	"00" Time Stamp bits 39-24
4	"00" "00000000" Time Stamp bits 23-16
5	"00" Time Stamp bits 15-0
6	"10" "0000" Pulse Number "00" SampleNumber from Thredhold bits 9-0
7	"00" Tfine(5..0) Vmin (11..4)
8	Vmin(3..0) Vp (11..0)
9	"10" "0000" Pulse Number "01" SampleNumber from Thredhold bits 9-0
10	"00" Tfine(5..0) Vmin (11..4)
11	"00" Vmin(3..0) Vp (11..0)
12	"10" "0000" Pulse Number "10" SampleNumber from Thredhold bits 9-0
13	"00" Tfine(5..0) Vmin (11..4)
14	"00" Vmin(3..0) Vp (11..0)
15	"10" "0000" Pulse Number "11" SampleNumber from Thredhold bits 9-0
16	"00" Tfine(5..0) Vmin (11..4)
17	"00" Vmin(3..0) Vp (11..0)
18	"11" "0000" : end of PTW
Memory location from beginning of PTW	Content (WITHOUT EVENT)
0	"00" "10010" Trigger Number bits 26-16
1	"00" Trigger Number bits 15-0
2	"00" "10011000" Time Stamp bits 47-40
3	"00" Time Stamp bits 39-24
4	"01" "00000000" Time Stamp bits 23-16
5	"01" Time Stamp bits 15-0
6	x"10000"
7	x"10000"
8	"11" "0000" : end of PTW

Data Processing:

Data Processing for all mode involves scanning the entire secondary buffer. If there is no pulses (data that cross thredshold), x"FFF0" is written to processing memory (PTW) data locations. Trigger Number and Time Stamp info are copied from secondary buffer to processing memory. X"FFF0" signal DataFormat block to prevent data from written to external FIFO. This feature only writes ADC channel that has data that cross thresdhold (TET).

Data Processing consists of 4 state machines, counters, and pointers. The 4 State Machines include Main and one for each of the 3 Processing Options. When there is ADC data to process, Main State Machine read Time Stamps and Trigger Number from Secondary Buffer and write to Data Processing Buffer. It then calls on one of the other three state machines to process the Option that is in effect.

The state machine for option 1 does the following:

1. Copies PTW * 20MHz number of words from Secondary Data Buffer to Data Processing.
2. Increment number of process counter by one

The state machine for option 2 does the following:

1. Read ADC data from Secondary Data Buffer. Start PULSE_TIMER to tick mark the data read.
2. If ADC data is above Trigger Threshold, it writes PULSE_TIMER to Process Buffer. Then it copies NSB and NSA number of words from Secondary Data Buffer to Data Processing Buffer as follow:
    a. If the number of words read before threshold is greater than NSB load RD_PTW_PTR with address that is NSB before threshold. If the number of words read (WORD_AFTER_TS_CNT) is less than NSB, load the RD_PTW_PTR with address of WORD_AFTER_TS_CNT word back from threshold.
    b. Start NSB_CNT.
    c. When NSB_CNT = NSB if WORD_AFTER_TS_CNT > NSB **or** NSB_CNT = WORD_AFTER_TS_CNT if WORD_AFTER_TS_CNT < NSB start NSA_CNT.
    d. When NSA_CNT = NSA, it stop reading Secondary Buffers.
3. Repeat Step 1 and 2 until number (PTW * 250MHz) numbers of words have been read.
4. Write "FFFF" to signal the end of PTW.
5. Increment number of process counter by one

The state machine for option 2 does the following:

1. Read ADC data from Secondary Data Buffer.
2. If ADC data is above Trigger Threshold, it unable accumulated sum circuit to add ADC value from NSB to NSA ADC words.
3. Write accumulated sum to Secondary Data Buffer
4. Repeat Step 1, 2, and 3 until number number PTW * 20MHz numbers of words have been read.
5. Write "FFFF" to signal the end of PTW.

6. Increment number of process counter by one

In mode 2 and 3, when the number of words read before the ADC value exceeds the Trigger Threshold is less then NSB, only that many word are processed.

Each state machine is responsible to change and reset the counters that pertained to the option.

The counters and their functions are listed below.

1. WORD_AFTER_TS_CNT: keep track of words read from beginning of PTW to the ADC sample that exceeds the Trigger Threshold. If WORD_AFTER_TS_CNT is less then NSB when this Threshold exceeded occurs, the NSB_PTR_ENOUGH pointer is used as starting address. Only WORD_AFTER_TS_CNT number of word before Threshold is processed.
2. TS_CNT: keep track of the number of time stamp and trigger number words read from the Secondary Buffer. Main state machine uses this to stop copying time stamp and trigger number words.
3. PTW_WORDS_CNT: keep track of the number of word in PTW has been read out. It is cleared when it is equaled to number of "PTW words + 4 Time Stamp words + 2 Trigger Number words".
4. NSB_CNT: Keep track of the number of words before Threshold has read and process.
5. NSA_CNT: Keep track of the number of words after Threshold has read and processed.
6. PULSE_TIMER: Tick mark ADC data read from Secondary Buffer from beginning of PTW.
7. PULSE_NUMBER: Keep track of the number of pulses in PTW.
8. HOST_BLOCK_CNT: Keep track of the number of PTW ready to transfer to host. The host decrement this counter after the host read one PTW.

The pointers and theirs functions are listed below:

1. NSB_PTR_ENOUGH: This pointer is used as starting address if the number of words read from PTW beginning to Threshold is greater than NSB value. A number of NSB words is processed.
2. NSB_PTR_NOT_ENOUGH: This pointer is used as starting address if the number of words read from PTW beginning to Threshold is less than NSB value. Only WORD_AFTER_TS_CNT number of word is processed.

Counters that also serve as pointers are listed below:

1. RD_PTW_PTR: This is the address to the Secondary Buffer. It is load with either NSB_PTR_ENOUGH or NSB_PTR_NOT_ENOUGH and increment under state machine control. It is cleared (restart at address 0) when PTW_WORDS_CNT is equaled to "number of PTW words + 4 Time Stamp words + 2 Trigger Number words".

TDC Algorithm Overview:

The TDC algorithm calculates time of the mid value of a pulse relative to the beginning of the look back window. The mid value is the value between the smallest and the peak value of the pulse. The smallest value is the beginning of the pulse. The time consists of coarse time and fine time. The coarse time is the number of clock the sample value before the mid value and the fine time is the interpolating value of mid value away from next sample. The coarse value is 10 bits and the fine value is 6 bit. The resolution of LSB is 1/(CLK * 64). For a 250MHz the resolution is 62.5 pS. For example for a 20MHz clock, a pulse time value of 110 means the mid-point of the pulse occurred at 6.875nS (62.5pS * 110) from the beginning of look back window.

Requirements for TDC Algorithm:

ii) There must be at least 5 samples (background) before pulse. Four of these samples are used to determine the pedestal (Vnoise) floor. The minimum value of the pulse is the first value that is Vnoise.

TDC Algorithm for Mode 3:

1) Search for Vaverage
   - **a)** Latch starting PTW_RAM ADR
   - **b)** Read four samples. Vnoise = Average of 4 samples. Increment sample count by 4.
   - **c)** Vmin = Vnoise. Increment sample count.
   - **d)** Read until Vram < Vram_delay. Vpeak = Vram if Vram is greater than TET. Increment sample count.
   - **e)** Store PTW_RAM ADR for Vpeak.
   - **f)** Vaverage = (Vpeak – Vmin) / 2

2) Search for sample before (Vba) and sample after (Vaa) Vaverage
   - **a)** Restore starting PTW_RAM ADR. Increment Pulse Timer whenever the address is incremented.
   - **b)** Read until Vram > Vmin. Vba = Vram
   - **c)** Read one more for Vaa
   - **d)** Calculated Tfine

3) Write Pulse Number and Pulse Timer to Processing RAM.

4) Increment Pulse Number

5) Restore PTW_RAM ADR for Vpeak. Load sample count to Pulse Timer.

6) Read until Vram < Vmin. End of first pulse. Increment Pulse Timer whenever the address is increment. End processing whenever PT_RAM data is ended.

7) Go to step 1.

Data Format :

Data format read data from Data Processing Memory, put the data in proper format as described in FADC Data Format, and write to external FIFO to host. The data format falls into 5 categories: Event_Header, Time_Stamp, Window_Raw_Word1, Pulse_Raw_Word1, Window_Pulse_Raw_Words_2_to_N, Pulse_Integral and Event_Trailer. The words are 36 bits wide.

Event_Header indicates the start of an event and bits are assigned as follow:
(35-34) = 0
(33-32) = 1
(31) = 1
(30-27) = 2
(26-0) = trigger number

➔ x"19 trigger number"

Trigger Time (Time_Stamp) indicates time of trigger occurrence relative to the most recent global reset. The six bytes (48 bits) of trigger time Ta Tb Tc Td Te Tf are format in two 32-bits words:

Word1:
(35-34) = 0
(33-32) = 0
(31) = 1
(30-27) = 3
(26-24) = 0
(23-16) = Ta
(15-8) = Tb
(7-0) = Tc
➔ x"0980 time stamp hi

Word2:
(35-34) = 0
(33-32) = 0
(31) = 0
(30-24) = 0
(23-16) = Td
(15-8) = Te
(7-0) = Tf
➔ x"0000 time stamp lo

Window Raw Word1 indicates the beginning of Window Raw Data.
(35-34) = 0
(33-32) = 0
(31) = 1
(30-27) = 4
(26-23) = Channel number (0-7)

(22-12) = 0
(11-0) = Window Width (PTW) (in number of samples).
➔ x"0A ChannelNumber 00 numberOfSamples"

3322 2222 2222 1111 1111 1198 7654 3210
1098 7654 3210 9876 5432 10

1010 0Cha n000 0000 0000 Ptw- ---- ----

Pulse Raw Word1 indicates the beginning of Pulse Raw Data.
(35-34) = 0
(33-32) = 0
(31) = 1
(30-27) = 6
(26-23) = Channel number (0-7)
(22-21) = pulse number (0-3)
(20-10) = 0
(9-0) = time from beginning of PTW that the pulse crossed thredshold
➔ x"0B ChannelNumber 00 TIME"

3322 2222 2222 1111 1111 1198 7654 3210
1098 7654 3210 9876 5432 10

1011 0Cha nP#0 0000 0000 00Ti me-- ----

Remaining words for Pulse Raw Data and Window Raw Data have the same format.

(35-34) = 0
(33-32) = 0
(31) = 0
(30) = 0
(29) = 1 indicates sample x not valid
(28-16) = ADC sample x (includes overflow bit)
(15-14) = 0
(13) = 1 indicates sample x+1 not valid.
(12-0) = ADC sample x+1 (includes overflow bits).

3322 2222 2222 1111 1111 1198 7654 3210
1098 7654 3210 9876 5432 10

00xA dcSa mple ---- 00xA dcSa mple ----

Pulse Time (8) – time associated with an identified pulse within the trigger window.
(31) = 1
(30 – 27) = 8
(26 – 23) = channel number (0 – 15)

(22 – 21) = pulse number (0 – 3)
(20 – 19) = measurement quality factor (0 – 3)
(18 - 16) = reserved (read as 0)
(15 – 6) = coarse pulse time
(5 – 0) = fine pulse time

3322 2222 2222 1111 1111 1198 7654 3210
1098 7654 3210 9876 5432 10

1100 0Cha nP#0 0000 Puls eTim e

Pulse Integral (7) – integral of an identified pulse within the trigger window. The pulse integral may be a simple sum of raw data samples over the pulse duration, or the result of a complex fit to pulse shape. Pedestal subtraction may be included.

(31) = 1
(30 – 27) = 7
(26 – 23) = channel number (0 – 15)
(22 – 21) = pulse number (0 – 3)
~~(20 – 19) = measurement quality factor (0 – 3)~~
~~(18 – 0) = pulse integral~~
(20-0) = pulse integral

3322 2222 2222 1111 1111 1198 7654 3210
1098 7654 3210 9876 5432 10

1011 1Cha nP#0 0Pul seIn tegr al

Pulse Vmin Vpeak (10) – ADC count for minimum and peak value of a pulse. This is too be used off line to apply correction to Pulse Time in TDC mode.

(31) = 1
(30 – 27) = 10
(26 – 23) = channel number (0 – 15)
(22 – 21) = pulse number (0 – 3)
(20 – 12) = Vmin
(11 – 0) = Vpeak

3322 2222 2222 1111 1111 1198 7654 3210
1098 7654 3210 9876 5432 10

1101 0Cha nP#v minn nnn vpea kkkk kkkk
D

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

Example:

Raw Data (mode0) :

x”19_____” Event Header
x”98_____“ Time Stamp upper 24 bits.
x”_______” Time Stamp lower 24 bits.
x”A_____” Channel Number, Window Width (PTW)
x”_____” Raw Data
x”2E8000000” End of Event

Pulse Data (mode 1):

x”19_____” Event Header
x”98_____“ Time Stamp upper 24 bits.
x”_______” Time Stamp lower 24 bits.
x”B_____” ChanNum(26-23), PulseNumb(22-21),Time from beginning of PTW that the pulse crossed thredshold(9-0).
x”_______” 2 pulses (12-0) (28-16) per 36 bits words.
x”2E8000000” End of Event

Pulse Sum (mode 2):

x”19_____” Event Header
x”98_____“ Time Stamp upper 24 bits.
x”_______” Time Stamp lower 24 bits.
x”C_____” Pulse time, ChanNum(26-23), PulseNumb(22-21),Time(15-0)
x”B8_____” Channel Numbe(26-23)r, Pulse Number(22-21), Pulse Integral (18-0)
x”2E8000000” End of Event

TDC (mode 3):

x”19_____” Event Header
x”98_____“ Time Stamp upper 24 bits.
x”_______” Time Stamp lower 24 bits.
x”C_____” Pulse time, ChanNum(26-23), PulseNumb(22-21),Time(15-0)
x”D_____” ChanNum(26-23), PulseNumb(22-21),Vm(20-12),Vp(11-0)
x”2E8000000” End of Event

Raw Data and TDC (mode 7)

x”19_____” Event Header
x”98_____“ Time Stamp upper 24 bits.
x”_______” Time Stamp lower 24 bits.
x”A_____” Channel Number, Window Width (PTW)
x”_____” Raw Data
x”C_____” Pulse time
x”D_____” VminVpeak
x”2E8000000” End of Event

Data Format VHDL:

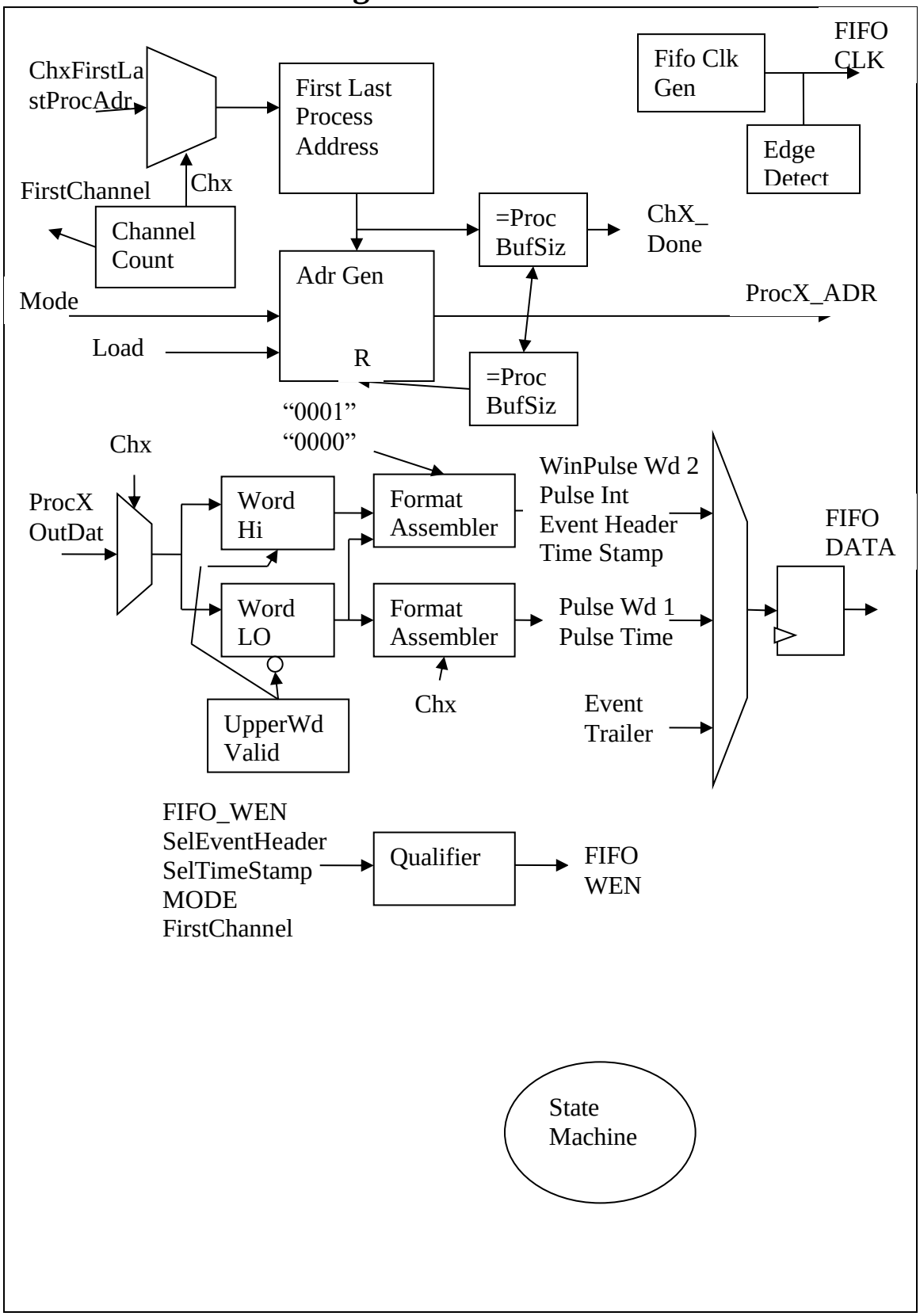
The VHDL code read data streams from processing block, format them per document "FADC Data Format" by Ed Jastrzembski.
When all HOST_BLOCKx_CNT is greater then one, DATFORSM begins the write out algorithm. The algorithm is as follow:

1) Pop the starting and last address of the data in the processing buffer.
2) Load the starting adddress of Channel 0 to Address counter. Start FIFO clock. Inc Address counter on rising edge of FIFO clock.
   Output Address to PROCx_ADR
3) Read data from PROCx_OUTDAT. Assemble them into Event Header, TimeStamp1, and TimeStamp2 and writes to FIFO.
4) In mode 0, the Address is stop after TimeStamp2 address (5) to allow time to insert Window Raw Data Word 1 which contains Channel Number and Window Width.
5) In mode 1 and mode 2, the Address is stop after Pulse Number and SampleNumber from Threshold Address (6), to allow time to assemble Pulse Raw Data Word 1 which contains Channel Number and
   first sample number for pulse or Pulse Time which contains Channel Number and pulse time.
6) The data are read and write in pairs until the Address counter equal last address of the processing buffer. The channel are incremnent and repeats step 1 through 6.
7) After the last channel is finish, Event Trailer is written to FIFO.

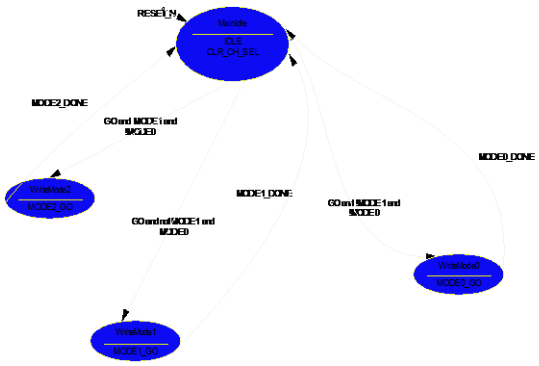
Because of the different in the data format between the modes: 0,1,and 2, each mode has its own state machine.
In mode 2, there might be extra words (for some setting of NSA and NSB) in the processing buffer after the last integral, the statemachine does not write this to FIFO.
In mode 0 and 1, for even number of data, the number of data written to FIFO is 2 more, for odd number of data, the number of data written to FIFO is one more.

Data Streams from Processing for diferent modes:
In mode 0: EventHeader, TimeStamp1, TimeStamp2, WindowRaw(not from processing), Deven Dodd,..., TimeEnd
In mode 1: EventHeader, TimeStamp1, TimeStamp2, PulseRaw(upper 16 from processing, not lower 16), Deven Dodd,..., TimeEnd
In mode 1: EventHeader, TimeStamp1, TimeStamp2, PulseRaw(upper 16 from processing, not lower 16), Integral, TimeEnd

Data Format VHDL Diagram



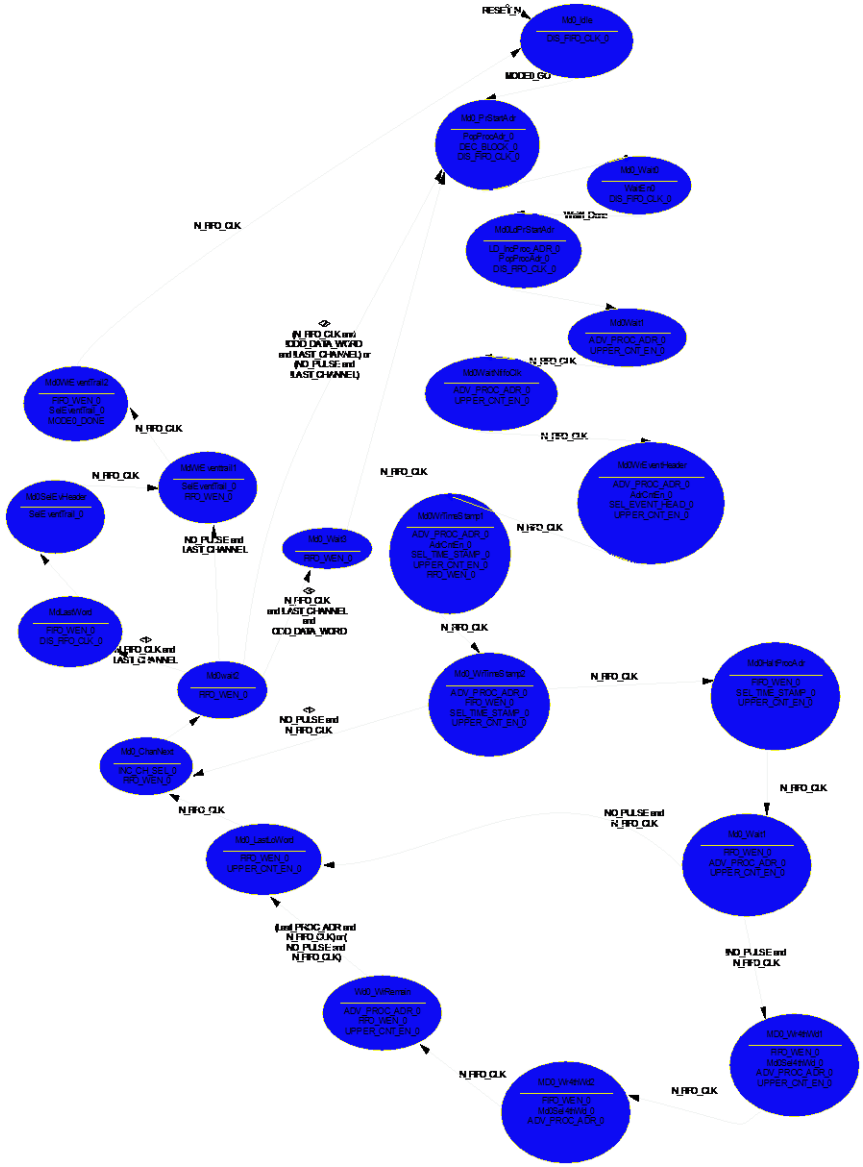
Data Format State Machine Main



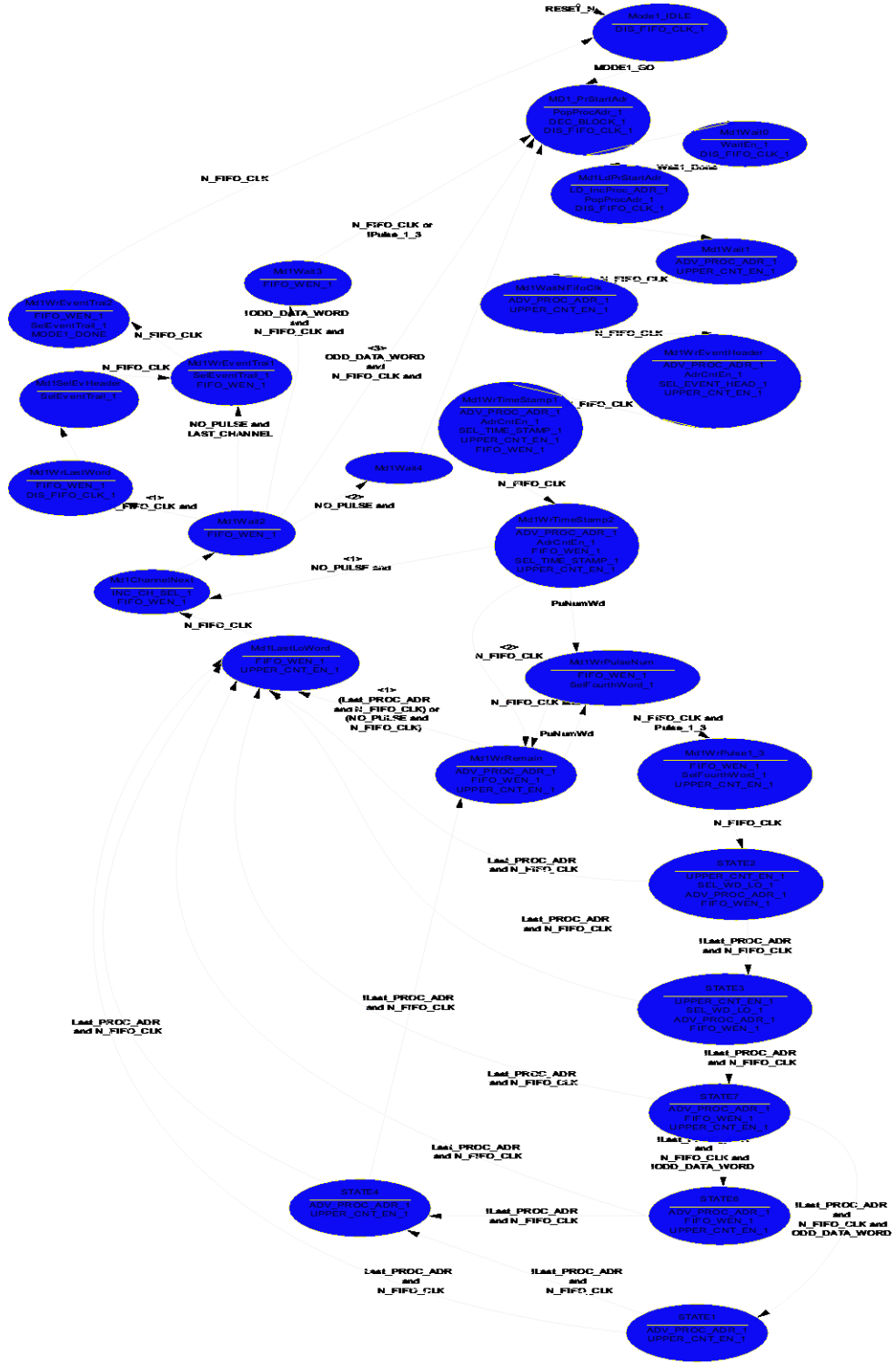
Main State Machine does the following:

1. Call State Machine for Mode 0,1,or 2.

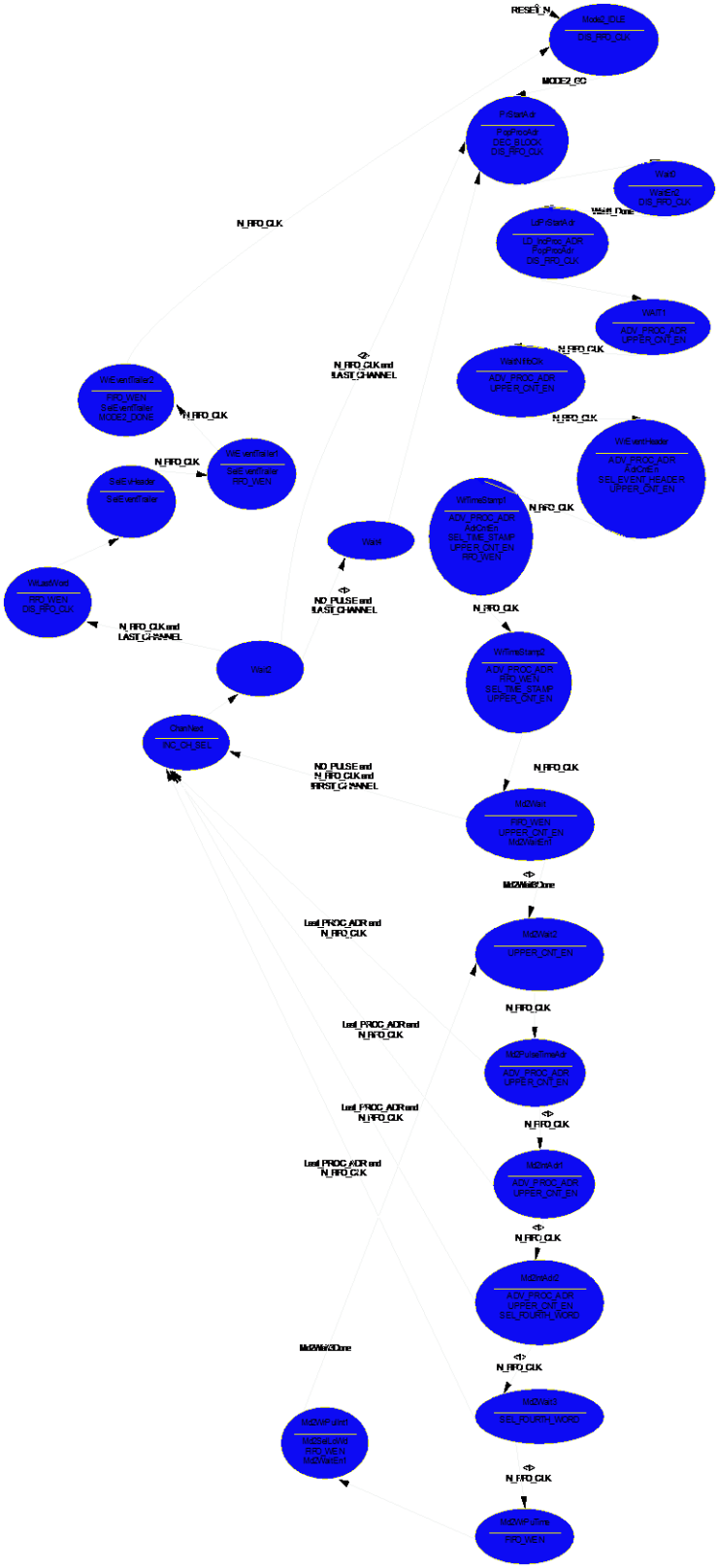
Data Format State Machine For Mode 0



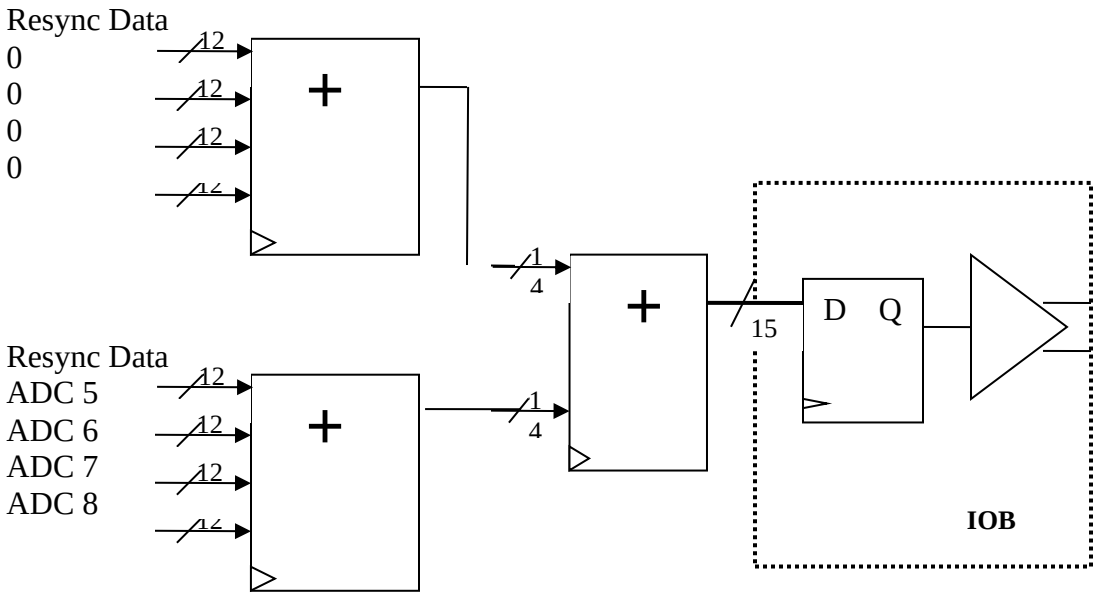
Data Format State Machine RD For Mode 1



Data Format State Machine RD For Mode 2

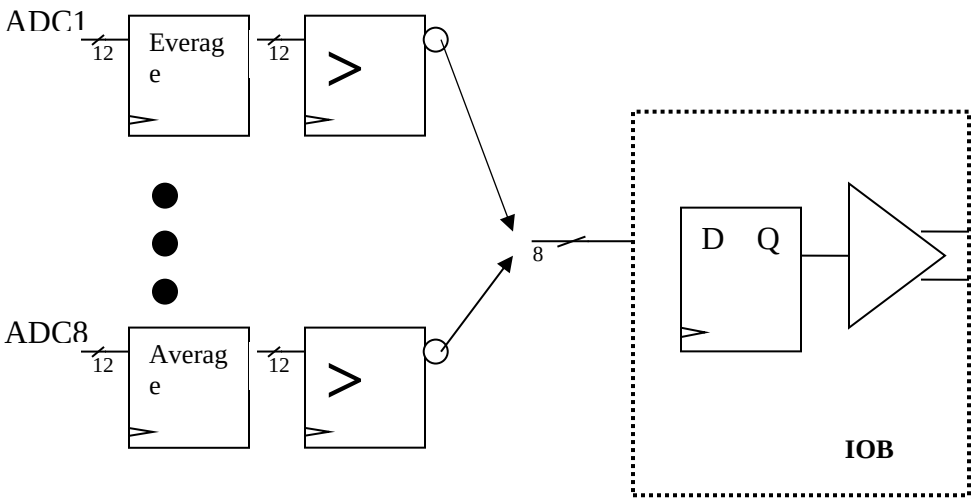


SUM


Resync Data
0
0
0
0
12
12
12
12
Resync Data
ADC 5
ADC 6
ADC 7
ADC 8
12
12
12
12
+
+
+
14
14
15
D
Q
IOB

HIT BITS

-



VME FPGA IFACE:

Control Bus Memory Map for FADC FPGA

Name	Width (Bits)	Quantity	Access	Primary Address (Secondary Address)	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 (---)	TRIGGER NUMBER BIT 15 to 0
STATUS2	16	1	R	0x0002 (---)	Monitor Data For Debugging Purpose
CONFIG 1	16	1	R/W	0x0003 (---)	Bit 0-2 (process mode): 000 → Select Mode 0 001 → Select Mode 1 010 → Select Mode 2 011 → Select Mode 3 111 → Run Mode 0 then Mode 3 for each trigger Bit 3: 1:Run Bit 5-4 : Number of Pulses in Mode 1 and 2 Bit 7: Test Mode (play Back). Bit 11 to 8: Select which ADC Data channel to be read at Status 2.
CONFIG 2			R/W	0x0004 (---)	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
CONFIG 4	16	1		0x0005	7 => rising edge write to AD9230 ADC 6 => 1 write to all ADC 5 => 0 write to AD9230 1 read from AD9230 4 => 1 Reset ADC 3..0 => Select ADC to write to
CONFIG 5	16	1		0x0006	15..8 => Registers inside AD9230 7..0 => Data to write to register.
PTW	9	1	R/W	0x0007 (---)	Number of ADC sample to include in trigger window. PTW = Trigger Window (ns) * 250 MHz. **Minimum is 6**. **Always report Even Number. For odd PTW number, discard the last sample reported.**
PL	11	1		0x0008 (---)	Number of sample back from trigger point. PL = Trigger Window(ns) * 250MHz
NSB	12	1		0x0009 (---)	8..0: Read Back Path NSB Number of sample before trigger point to include in data processing. This include the trigger Point. **Minimum is 2 in all mode.**

					12..9: Trigger Path NSB
NSA	13	1		0x000A (---)	8..0: Read Back Path NSA Number of sample after trigger point to include in data processing. **Minimum is (6 in mode 2)and (3 in mode 0 and 1). Number of sample report is 1 more for odd and 2 more for even NSA number.** 14..9: Trigger Path NSA
TET	12	16		0x000B - 0x001A	Trigger Energy Thredhold.
PTW DAT BUF LAST ADR	12	1		0x001B	Last Address of the Secondary Buffer. See calculation below
PTW MAX BUF	8	1		0x001C	The maximum number of unprocessed PTW blocks that can be stored in Secondary Buffer. See Calculation below.
Test Wave Form	16	1		0x001D	Write to PPG. Read should immediately follow write.
ADC0 Pedestal Subtract	16	1	R/W	0x001E	Subtract from ADC0 Count before Summing
ADC1 Pedestal Subtract	16	1	R/W	0x001F	Subtract from ADC1 Count before Summing
ADC2 Pedestal Subtract	16	1	R/W	0x0020	Subtract from ADC2 Count before Summing
ADC3 Pedestal Subtract	16	1	R/W	0x0021	Subtract from ADC3 Count before Summing
ADC4 Pedestal Subtract	16	1	R/W	0x0022	Subtract from ADC4 Count before Summing
ADC5 Pedestal Subtract	16	1	R/W	0x0023	Subtract from ADC5 Count before Summing
ADC6 Pedestal Subtract	16	1	R/W	0x0024	Subtract from ADC6 Count before Summing
ADC7 Pedestal Subtract	16	1	R/W	0x0025	Subtract from ADC7 Count before Summing

ADC8 Pedestal Subtract	16	1	R/W	0x0026	Subtract from ADC8 Count before Summing
ADC9 Pedestal Subtract	16	1	R/W	0x0027	Subtract from ADC9 Count before Summing
ADC10 Pedestal Subtract	16	1	R/W	0x0028	Subtract from ADC10 Count before Summing
ADC11 Pedestal Subtract	16	1	R/W	0x0029	Subtract from ADC11 Count before Summing
ADC12 Pedestal Subtract	16	1	R/W	0x002A	Subtract from ADC12 Count before Summing
ADC13 Pedestal Subtract	16	1	R/W	0x002B	Subtract from ADC13 Count before Summing
ADC14 Pedestal Subtract	16	1	R/W	0x002C	Subtract from ADC14 Count before Summing
ADC15 Pedestal Subtract	16	1	R/W	0x002D	Subtract from ADC15 Count before Summing
Config 3	16	1	R/W	0x002E	15 : Sync Disable (11..0) Trigger Path Processing Threshold
STATUS 3	16	1	R	0x002F	FPGA core temp (DieTemp)
STATUS 4	16	1	R	0x0030	FPGA V Internal (Vint)

PTW MAX BUF = INT(2016 / (PTW + 8))

Where:

2016 → Number of address of Secondary Buffer

PTW → Trigger Window width in nano-second

PTW DAT BUF LAST ADR = PTW MAX BUF * (PTW + 8)- 1;

DieTempDieTemp 6 → 4 address for Time Stamp and 2 address for Trigger Number

NumberOfBytePerTrigger → PTW * 250 MHz.

TrIgger Path NSB + Trigger Path NSA has to be less then 63.

Temperature_C = ((float)(DieTemp >> 6) * 503.975/1024) - 273.15;
VCC_INT (Volts) = ((Vint >> 6) / 1024) x 3V