An Draft Report Not for Distribution

High Level DAQ and interactions with Device Firmware

File: /users/atlas/may/private/LaTeX/highLevelDaq.tex

Date: December 2, 2011 Subject: From: Ed May HEP Argonne, Henry Frisch HEP University of Chicago

1 Introduction

Some notes on High Level DAQ for 160 channel PSEC4 based system. Use as a basis for discussions with hardware/firmware developers. The viewpoint of one of the authors (Ed May) is from the perspective of the DAQ computer and the offline analysis.

2 General Considerations

Any and all initialization and configuration data transferred from PC to the FPGA firmware should be back readable.

A general status register should be available and readable from the PC. This provides state, status information on firmware and hardware.

On device power up the FPGA firmware should be place in a default state which can be determined from reading the status register.

Any device (meta data) information setup up by the FPGA firmware at compile time and/or initialization should be readable.

The FPGA firmware version number should be readable.

It may be desirable to have a mechanism to both download and upload the firmware program from the DAQ PC.

If the FPGA contains assigned memory blocks for general purpose flat or circular buffers is should be read/writable from the DAQ PC. This is most useful for testing the memory and exercising large data transfers. Any hardware switches on the DAQ device should be readable.

Any hardware serial or other identification numbers should be readable.

Pedestals and/or calibration data downloaded should also be readable as an upload.

The device to DAQ pc communications protocol must support programmed data transfer read and write operations. Broadcast operations if available would be useful for initializing operations in system of multiple devices. DMA or other bulk stream data transfers should be supported for large (possibly variable length) transfers to provide the most efficient data collection.

A polling mode of operation based on contents of the status register must be available.

A data transfer on interrupt mode based operation may be provided. Checksums should be appended to long data transfers.

3 Guidelines specific to LAPPD PSEC3/4 based DAQ device

(based on May 20, 2011 presentation by Mircea Bogden) See Figure 1. Physical Archetecture:

- AC Analog Card contains 8 physical PSEC4s as 2 logical groups of 4
- DC Digital Card contains 1 LFPGA
- CC Center Card contains 1 CFPGA
- PC DAQ personal computer talks to CC via USB2
- For test systems PC will talk to DC directly (no CC)

Multiplicities between communicating Devices

Device Hierachy	PSEC4	AC	DC	CC
controler			LFPGA	CFPGA
Multiplicity	4	8	8	2

Table 1:

In normal (large system operation 160 channels) the DAQ PC only communicates to the CC via USB 2.0 of which there are 2 I/Fs For testing, debugging, and small multichannel systems a DC can be controlled via its own USB.

What are the firmware operation modes? Four possible modes may be:

1. delivers just raw data

2. delivers pedestal subtracted data

3. delivers calibrated data

4. delivers features extracted from data

These modes may in fact be different versions of the firmware! When operating in a less capable firmware mode the DAQ pc program or Offine program will provide higher functionality. This is currently the case with the 4-chan PSEC3 evaluation card.

Will the FPGAs store more than 1 event per readout cycle to the DAQ PC?

How will the synchronization of a 2 CC based system be handled. This is likely complicated when the CCs buffer multiple events!

How different will the register sets and commands be for the DC firmware compared to the CC firmware?

Can the hierarchical hardware nature be completely hidden so that the logical structure presented to the DAQ PC is the same whether talking to a the DC firmware or the CC firmware?

Can the hierarchical hardware nature be completely hidden so that the logical structure presented to the DAQ PC is the same whether talking to a the DC firmware or the CC firmware?

What is the data transfer model?

- Push: CC decides what and how much to send.
- Pull: PC decides what data and how much to request.

Is a calibrated adc cell value an integer or a float? Signed integer is preferable.

Types of read data:

1. all channels all adc cells no pedestals no calibrations

- 2. all channels all adc cells with pedestal corrections but no calib
- 3. all channels all adc cells with both

Where "all channels" is modifiable (on a run basis) by a MASK of channels on/off which is downloaded at initialization of a run.

From Eric Oberla's 4 Feb 2011 note: 'PSEC-3 Evaluation Documentation' the follow information needs to be supplied from the DAQ PC to configure the PSEC3/4 chips:

- 1. Pedestal voltage via integer to a 12 bit DAC
- 2. Internal Trigger threshold via int to 12 DAC
- 3. 1 bit to specify trigger sign
- 4. 1 bit to specify trigger mode
- 5. 2 bit integer to set sampling rate

Need to have a complete list of control and data transfer messages.

Need to have a complete list of firmware registers with functions controlled and monitored.

4 A note on addressing

With respect to May 20, 2011 'PSEC Beam Test Readout System' from Micea Bogdan. This presentation show the hardware proposed for 160 channel readout system.

Simplest data delivered to DAQ PC is an ordered stream of 256 adc values from all the 160 channels. This is the approach used in the 4 channel evaluation board DAQ PC program

In order to have more flexibility and variable length data transfers an addressing scheme needs to be implemented. For example, suppressing unwanted, dead or noisy channels. Suppress raw data, send only extracted feature data.

A fundamental decision is whether the addresses begin at 0 or 1. The use of 0 as an address usually means the size of the address fields is smaller.

My preference is to start addressing from 0. One slight issue is all array addressing in Matlab/Octave starts at 1.

There are two extreme possible addressing schemes: flat (ie just channel numbers)

field name	channel-number	cell-number	adc-value
range	0-159	0-255	0-1023
width(bits)	8	8	10

Table	2:	Flat	addressing
-------	----	------	------------

hierarchical (based on cards or subsystem IDs)

field	center	digital	analog	psec	chan	cell	adc
name	card	card	card	chip	number	number	value
logical	CFPGA	LFPGA					
name							
range	0-1	0-7	0-7	0-4	0-6	0-255	0-1023
width(bits)	1	3	3	4	4	8	10

Table 3: Hierarchical Addressing

It is desirable in the hierarchical scheme to have the address set by switches and pinnings on the physical cards. Alternatively they must be set by the code loaded into the digital and center card FPGAs.

Converting the channel signals and positions into particle deposited charge, time and location likely will be done in the high level DAQ PC and in the offline. There is some motivation (based on parallel processing) to do some (or all) of this in the LFPGA or the CFPGAs. Some of this will require geometry information. Depending on where the geometry of the detector strips is required a scheme which include geometry metadata would be required.

System Metadata (can be whole words- space doesn't matter here, as these aren't transmitted each event): Some elements of a scheme to provide this are:

1. Number of panels (a panel is a 1 by N row of tiles with common anode strips. Note 1 CFPGA per panel is assumed by definition).

- 2. Number of strips per panel
- 3. Number of channels per chip
- 4. Number of chips per panel
- 5. Bit map of strips onto chip channels or equiv (e.g. to cover 40 strips with 6-channel chips might want to use only 5 channels per chip and 8 chips rather than 7 chips with 6 channels for 6 chips and 4 for one, or some other scheme- but do need to know
- 6. Number of LFPGA's per end
- 7. Number of chips per LFPGA
- 8. Bit mapping of chips to LFPGAs
- 9. Number of LFPGA's per CFPGA per end

Addressing:

- 1. All system addresses run from 0 to N-1
- 2. Channel address on chip: 8 bits (need 3 for PSEC4)
- 3. Chip address in LFPGA: 8 bits (need 3 for PSEC4)
- 4. LFPGA address on CFPGA: 8 bits (need 2 for 40-strip scheme, for 30-strip scheme)
- 5. Panel address in DAQ system (CFPGA address) : 8 bits (should handle anything we're doing anytime soon);

Note USB 2.0 standard says maximum of 60 MH/sec for realistic devices 1MB/sec. Basic data packet is PID + 0-1024 bytes + 16bitCRC Fundamental word size is unsigned short (ie 2 bytes)

Need to have a complete description of how any data is mapped into USB buffers.

Need to estimate the total transfer operation count and rate for reading a system with N channels for single events and buffered multievents. "With bulk transfers, typical transfer rates are around 900kb/s to a single endpoint"

Some examples of data mapping

For variable message type need Identifiers and Data length

Below is an example of a fixed length flat address space. Clearly there would be many more 'Data Type Identifiers' and the list would grow over time as new functionality is added.

Data Type Identifier	Data Length	Description
10	259	Raw 256 cell data for 1 Channel
99		Can not satisfy request

Table -	4:
---------	----

Addr In Buffer	Description
0	Data Type Identifier $= 10$
1	Channel Address
2	cell 0 ADC
257	cell 255 ADC
258	Check Sum

Table 5:

Addr In Buffer	Description
0	Data Type Identifier $= 99$
1	Channel Address
2	Error Code
3	Data Type Identifier $= 10$
4	Check Sum

Table 6:

5 Calibration

Issues regarding Calibration of electronics.

A comment: Since there is significant cross-talk between channels, I suspect only a small number of channels can be pulsed simultaneously from a pulser, so that we get an accurate measure of the responce. This could significantly raise the time to acquire calibration and monitoring data.

- 1. What are any one time calibration studies to be done?
- 2. Are there any routine calibration tasks? How many? How often?
- 3. Can test beam and/or experiment data be used for the calibration task?
- 4. How is a known source of calibration signals produced and injected into the electronics channels.
- 5. Said differently is there a significant difference between time scale for the electronics compared to the photocathode signals, especially as a function of position on the photocathode.
- 6. What are calibration tasks with repect to timing compared to amplitude?
- 7. Are there issues with respect to Linearity for time and/slash or amplitude?
- 8. Are the calibrations done at the channel or cell level?
- 9. How much data per channel is required?
- 10. How long will a full calibration take?
- 11. Is it required to keep the raw data from calibration runs.
- 12. Are there routine calibration tasks which are run periodically?
- 13. Are these the same of different from "Monitoring" as described in the next section.

6 Detector and Electronics Monitoring

A comment: for the purposes of monitoring, one can likely tolerate distortions due to crosstalk, so multiple channels can be flashed or pulsed. However, the ammount of data to handle could be very large!

- 1. Issues regarding short and long term monitoring of the electronics health.
- 2. Issues regarding short and long term monitoring of the detector performance.
- 3. How often should dedicated monitoring data be acquired?
- 4. Should there be special runs with special triggers or calibration pulses?
- 5. How much data should be acquired during these runs.
- 6. Detection of channels which are dead/alive. Detection of cells which dead or alive. Detection of noisy channels.

7 Error Notification

If the DC and/or CC detectes an error condition, how is this communicated to the PC program.

- 1. How many error classes?
- 2. How many error codes within a classes?
- 3. Should there be warnings?

8 No Name

- 1. Codes for the various devices are stored on Code Web page.
- 2. Documentation is stored on Code Web page.
- 3. Configuration, Calibration and Database files are stored on Data Web page.
- 4. Archived raw (and processed) data files are stored on Data Web page.

9 Projects Deliverables Schedule

Items for a table:

- 1. "Project Name" 'DC firmware' 'CC firmware' 'DAQ PC code' 'Near-line/Offline'
- 2. "Developer" 'a name'
- 3. "Date Version 1" 'a date'

Project Name	Developer	Date Version 1 available
AC psec4	Eric Oberla	now
DC card	raig Harabedian	soon
CC card	Mircea Bogdan	soon

Table 7: Hardware projects

Project Name	Developer	Date Version 1 available
DC Firmware	Aaron Meyer	
CC firmware	Maxwell Hutchinson	
DAQ PC code	Andrew Wong	
Nearline/Offline code	Matt Wetstein	

Table 8: Software projects

Beam Test Configuration



Figure 1: Mircia presentation May 2011.