GlueX-doc-2274-v5 June 23, 2015

FADC125 OPERATION AND DATA FORMAT REQUIREMENT V5.06

NAOMI JARVIS
DAVID LAWRENCE
CODY DICKOVER
LUBOMIR PENTCHEV
BENEDIKT ZIHLMANN

1. Introduction

This document specifies the data format produced by the JLab 125 MHz Flash ADC module (fADC125). The module adheres to the overall format standard described in the document [1] produced and maintained by the JLab Data Acquisition group[1]. All data produced by the fADC125 is in the form of 32-bit words. The fADC125 operating modes and data format follow those of the fADC250 [2] as closely as possible, modified where necessary for the drift chambers. The main differences are in the pulse data formats (data types 7 to 9), which permit output of pulse time, amplitude, integral and pedestal in one pair of data words, the addition of combined data formats (data types 10 and 11), which concatenate output of calculated pulse time and integral data with raw ADC data, and in the methods used to calculate the pulse time and integral. These are detailed later in this document. Common data types are listed for completeness.

The requirements for the FDC and CDC differ due to the nature of the signals at the input of the fADC125. In the FDC, cathode strips are connected to the fADC modules, and the data are used to find the center-of-gravity on a cluster of strips, defining the avalanche position. For this procedure, the signals on the side strips that are just above the threshold have the highest weight. Therefore, the results depend strongly on the noise and the best resolution is obtained by using the sum over the first maximum, or just the maximum amplitude, depending on the noise type. To reduce the accidental background, one also needs rough timing information to be compared to the wire hit time obtained with F1TDCs. In case of the CDC, the fADC125 modules take signal data from the anode wires which receive a wide range of pulse amplitudes. The information is used both for precise timing, needed to find the hit position, and for total charge determination that serves as particle identification. The requirements for the two detectors are broadly similar, but

[1] An early version of the VME data format standard [1] can be found online as GlueX-doc-2365, and a later draft is presently (June 2015) at https://halldweb1.jlab.org/wiki/images/5/58/JlabModuleDataFormat.pdf. Contact a member of the JLab DAQ group to obtain the most recent copy. Much of the critical information has been reproduced in this document.

there are important differences in the algorithms and the data formats which are explained explicitly below.

The following sections describe the methods used for pulse analysis and the configuration constants required for each detector, and then list the data formats.

2. Pulse Analysis

The trigger signal is delayed so that the entire trigger window has already been transferred to the fADC's data buffer when the trigger arrives. The trigger window contains NW consecutive samples which contain the pedestal window (NP samples), followed immediately by the hit search window, followed by an extra 6 samples. The last sample in the hit search window is designated sample WE for the purpose of description only (WE = NW − 7).

The extra 6 samples which follow the hit window are included in the trigger window to permit upsampling at the end of the hit window, as the upsampling process requires knowledge of the 5 samples immediately before and after the region of interest. Similarly, the trigger window should be configured so that no pulses are expected with their leading edge in the first 6 of the NW samples. This requirement is satisfied if the earliest threshold crossing is in or after sample 12 (this is so if NP=16 or greater), assuming that there are no more than 6 samples between the leading edge and the threshold crossing.

The pedestal window starts with sample 0 and ends with sample NP−1. NP is chosen to be an integer power of 2 so that the initial mean pedestal (PINIT) can be calculated by summing the NP values and right-shifting the sum. The initial pedestal is used for hit identification.

The hit search window (samples NP to WE) is the region to be searched for a pulse, which is found when two or more consecutive ADC sample values meet or exceed PINIT+H, where H is the hit identification threshold. The threshold crossing sample is designated TC. The pulse analysis also uses two lower thresholds for timing, a high timing threshold TH, and a low timing threshold TL. Suitable values for all three thresholds are determined during calibration.

The local pedestal is obtained just before the pulse using a dynamic pedestal window of NP2 samples, which ends at PG samples before the hit threshold crossing sample, TC. NP2 is also chosen to be an integer power of 2 so that the pedestal can be calculated as the sum of samples TC−PG−NP2+1 to TC−PG, right-shifted appropriately. Ideally, the local pedestal would be very close to the initial pedestal, but this is not always the case, for example, when a pulse arrives before the signal from a previous pulse has returned to baseline. The local pedestal is the quantity returned by the firmware, to be subtracted from the peak amplitude and integral during later analysis.

The signal integral is calculated in two parts which are later summed: the first part of the pulse integral is calculated by the timing algorithm, which sums from the start of the pulse up to sample TC−1, and the remainder of the integral is calculated by a separate module, which sums the samples from sample TC up to and including the earlier of sample TC+IE and WE. For the CDC, the integration should run until WE, so IE should be configured to be larger than WE. The pedestal is not subtracted from the integral in the firmware.

Following the trigger arrival, the signal analysis proceeds as follows:

(1) The initial event pedestal, PINIT, is found, as the mean of the NP consecutive samples in the pedestal window.

(2) The samples within the hit search window are searched for a hit, which is found if two or more consecutive samples meet or exceed a hit threshold PINIT+H. If the hit is found, the first sample number of the pair at or over threshold is designated TC. If no hit is found, no further action is taken.

(3) The mean local pedestal is obtained as the mean of NP2 samples ending with TC−PG.

(4) A small number NU of consecutive samples surrounding sample TC are sent to the time-finding module, as described later.

(5) The integrated signal is calculated, starting with sample TC and ending with the earlier of samples WE and TC+IE, and added to the first part of the integral, which is returned from the timing module. Pedestal values are not subtracted from this.

(6) A count of the number of samples with data overflow during the integration period is made.

(7) The hit search window samples are scanned, starting with sample TC, for the first maximum in value. The maximum is found by looking for two subsequent consecutive samples which decrease in value.

3. Pulse Analysis Readout Quantities

The pulse analysis is broadly similar for the CDC and FDC, but different readout formats are used in order to meet the different requirements in range and precision of the two detectors. Pulse integration starts with the leading edge of the pulse and ends with the earlier of sample WE or sample TC+IE. For the CDC, the value of IE should be set to be greater than WE, so the pulse integrals will end with WE. For the FDC, the value of IE should be set to be smaller than WE, and so most pulse integrals will end with TC+IE. The peak time is returned for the FDC only, this is the sample number containing the first maximum after the threshold crossing.

If a pulse is found, the following quantities are returned:

Time: Leading edge time in units of sample/10, as described later

Time Quality Factor: Set to 0 if time is good, 1 if time is a rough estimate

Integral: Signal integral from the sample containing the leading edge of the pulse to the earlier of sample WE or sample TC+IE, right-shifted IBIT bits, all bits set if the integral exceeds the field maximum

Overflow count: Number of samples between TC and the earlier of WE or TC+IE with overflow bit set, all bits set if the field maximum is exceeded

Pedestal: Mean local pedestal (mean of samples from TC−PG−NP2+1 to TC−PG), right-shifted PBIT bits, all bits set if the field maximum is exceeded

Peak amplitude: The amplitude of the first maximum between TC and WE, right-shifted ABIT bits, all bits set if the field maximum is exceeded. This is determined by finding the first sample beyond the threshold crossing after which two consecutive samples decrease in value

Peak time: Sample number of the maximum (FDC only)

4. Leading Edge Time Algorithm

A small number of samples surrounding the hit threshold crossing sample TC are sent to the timing module. It calculates high and low timing threshold values, using the single sample TC−PG as pedestal, and searches first for the high timing threshold crossing and then back for the low timing threshold crossing. It upsamples the small region surrounding the low timing threshold crossing and searches through the upsampled data to find the low timing threshold crossing again. The low timing threshold crossing point is returned, with the quality factor set to 0. A number of validation tests are performed during this process; if any are failed, then a less accurate estimate of the hit time is returned, with the quality factor set to 1.

The details of the process are as follows:

(1) A small subset NU of consecutive samples surrounding the threshold crossing sample is sent to the time-finding module, with the local pedestal sample (previously TC−PG) in place PED and the threshold crossing sample (previously TC) in place PED+PG, which is now designated XTHR for convenience.

(2) The samples from 0 to PED are scanned. If any values are found to be equal to 0 or greater than PED_MAX, the algorithm returns a time of XTHR×10−RT, a quality factor of 1, and a 'rough integral' equal to the sum of samples from int(XTHR−RT×0.1) to XTHR−1. The value of RT is set in the configuration parameters.

(3) A constant offset is added to all the sample values so that the minimum sample value becomes equal to ADC_MIN. This decreases the likelihood of calculating any negative upsampled values.

(4) The local pedestal, P, is obtained as the ADC value at sample PED.

(5) The algorithm searches through the samples, starting with sample PED+1, to find sample TCH, where the ADC value first equals or exceeds the high timing threshold, P+TH. If this is not found, the algorithm returns the value of TCH×10−RT, a quality factor of 1, and the rough integral.

(6) Having found sample TCH, it then searches the samples from TCH back toward PED to find sample TCL, where the ADC value is less than or equal to the low timing threshold, P+TL. If TCL > NU−7, there are not enough later samples to calculate the upsampled values, and so the 'midpoint time' TCL×10+5 is returned, with a quality code of 1, and a 'good integral', equal to the sum of samples TCL to XTHR−1.

(7) The ADC data are upsampled by a factor of 5 to calculate values at 1.6 ns intervals, from TCL−0.2 to TCL+1.2.

(8) If any upsampled value is negative, the midpoint time is returned, with a quality code of 1, and the good integral.

(9) Two upsampling errors are calculated, the difference between the second upsampled point and the value of sample TCL, and the difference between the sixth upsampled point and the value of sample TCL+1. The upsampled values tend to be too high when the slope of the leading edge is extremely steep. The mean of the upsampling errors is obtained and added to TL to obtain an adjusted threshold value.

(10) The upsampled values are searched from TCL+1.2 down to TCL−0.2 to find the point where the values fall equal to or below the adjusted threshold. If this is not found, or found at TCL+1.2, the midpoint time is returned, with a quality code of 1 and the good integral.

(11) The adjusted threshold crossing time is found, to the nearest tenth of a sample, by interpolating between the upsampled points before (or at) and after the threshold crossing. This quantity is returned by the algorithm as an integer, with a quality factor of 0, and the good integral.

5. Timing algorithm configuration constants

The configuration constants for the timing algorithm are described in Table 1. These are hard-coded into the firmware, and are common for all channels. The sample period is 8ns and the fADC units have a range of 12 bits, 0-4095.

Table 1. List of configuration constants for the timing algorithm

Name	**Description**	**Units**	**Value**
NU	Number of samples in the subset to send to the time-finding module	samples	20
PED	Position in the subset of the sample which is to be used as local pedestal (the first sample is in position 0)	samples	6
RT	If ADC values are unsuitable for upsampling, return the hit time of TS×10−RT, and integrate from int(XTHR−0.1×RT) to XTHR−1	sample/10	34
PED_MAX	Maximum permissible value for samples 0 to PED	ADC units	511
ADC_MIN	Set lowest ADC value equal to this	ADC units	20

6. Pulse analysis configuration parameters

The configuration parameters are described in Table 2. These will be programmable through the VME backplane. Unless otherwise noted, they are common for all channels. Typical values for the CDC and FDC are given in Table 3.

Table 2. Pulse analysis configuration parameters

Name	**Description**	**Units**
NW	Trigger window length	samples
P1	Initial pedestal window length $NP = 2^{P1}$ samples	
P2	Local pedestal window length $NP2 = 2^{P2}$ samples	
IE	Integration ends with the earlier of samples NW−7 and TC+IE	samples
H 0-71	Hit threshold for each channel	ADC units
TH 0-71	High timing threshold for each channel	ADC units
TL 0-71	Low timing threshold for each channel	ADC units
IBIT	2^{IBIT} scale factor for integral	
ABIT	2^{ABIT} scale factor for peak amplitude	
PBIT	2^{PBIT} scale factor for pedestal	

Table 3. Typical values for the configuration parameters

Name	**CDC**	**FDC**
NW	106	44
P1	4	4
P2	4	4
IE	200	10
H 0-71	100	100
TH 0-71	80	80
TL 0-71	20	20
IBIT	4	
ABIT	3	0
PBIT	0	

Using the default scaling factor values given above, the available range before and after scaling of the readout quantities is given in Tables 4 and 5. The unscaled readout quantities are also listed for completeness.

Table 4. CDC readout quantity available range, data types 7 and 10

Quantity	**Field width (bits)**	**Field range**	**Scaling factor**	**Range before scaling**
Time	11	2047		2047
Quality	1	1		1
Integral	14	16383	16	262143
Overflows	3	7		7
Pedestal	8	255	1	255
Amplitude	9	511	8	4095

Table 5. FDC readout quantity available range, data types 8, 9 and 11

Quantity	**Field width (bits)**	**Field range**	**Scaling factor**	**Range before scaling**
Time	11	2047		2047
Quality	1	1		1
Integral	12	4095	?	?
Overflows	3	7		7
Pedestal	11	2047	?	?
Amplitude	12	4095	1	4095
Peak time	8	255		255

7. Restrictions on configuration parameters and constants

(1) NW > NP + 6
(2) NW > NU
(3) NP >= NP2
(4) H > TH > TL
(5) NU > 14
(6) PED > 4

8. FADC125 Data Format

8.1. **Data Type List.** Data types 7 to 11 are specifically for the fADC125.

0: block header
1: block trailer
2: event header
3: trigger time
4: window raw data (samples)
5: – unused –
6: ~~pulse raw data (samples)~~
7: pulse data (integral and time, CDC format)
8: pulse data (integral and time, FDC format)
9: pulse data (peak amplitude and time, FDC format)
10: pulse data and pulse samples, CDC format
11: pulse data and pulse samples, FDC format
12: ~~scaler data~~
13: event trailer (debug only)
14: data not valid (empty module)
15: filler (non-data) word

8.2. **Data Word Categories.** Data words from the module are divided into two categories: Data Type Defining (bit 31 = 1) and Data Type Continuation (bit 31=0). Data Type Defining words contain a 4-bit data type tag (bits 30-27) along with a type dependent payload (bits 26-0). Data Type Continuation words provide additional data payload (bits 30-0) for the *last defined data type.* Continuation words permit data payloads to span multiple words and allow for efficient packing of raw ADC samples. Any number of Data Type Continuation words may follow a Data Type Defining word.

8.3. **Data Types.** Descriptions of the data types produced by the FADC125 follow:

Block Header (0): indicates the beginning of a block of events (High speed readout of the board or set of boards is done in blocks of events.)
(31) = 1
(30 - 27) = 0
(26 - 22) = slot number (set by VME64x backplane)
(21 - 18) = module ID (=2 for FADC125)
(17 - 15) = data format (for future use in format specification)
(14 - 8) = block number (incrementing scalar counting completed blocks)
(7 - 0) = number of events in block (1-255)

Block Trailer (1): indicates the end of a block of events. The data words in a block are bracketed by the block header and trailer.

(31)	= 1
(30 - 27)	= 1
(26 - 22)	= slot number (set by VME64x backplane)
(21 - 0)	= total number of events in event block

Event Header (2): indicates the start of event specific data. The included event number is useful to ensure the proper alignment of event fragments when building events. The 22-bit trigger number will roll over, but (4 M count) is not a limitation as it will be used to distinguish events within blocks, or among other events that are currently being built or transported.

(31)	= 1
(30 - 27)	= 2
(26 - 22)	= slot number (set by VME64x backplane)
(21 - 0)	= event number (trigger number)

Trigger Time (3): Time of trigger occurrence relative to the most recent global reset. Time is measured by a local counter/scaler that is clocked by the system clock or by a local module clock that may or may not have been synchronized with the system clock. In principle, a global reset signal is distributed to every module. The assertion of the global reset will clear the counters and inhibits counting. The de-assertion of the global reset enables counting and thus, sets t=0 for the module. The trigger time is necessary to ensure system synchronization and is useful for aligning event fragments when building events. For example, in the FADC250 there is a 48 bit counter (1 count = 4ns). 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). However, the module may be configured to only emit the first word (Type Defining) which contains the lower 24 bits of the trigger time. This should be sufficient for most cases and reduces the overall data size slightly.

Word 1:

(31)	= 1
(30 - 27)	= 3
(26 - 24)	= reserved (read as 0)
(23 - 16)	= T_D
(15 - 8)	= T_E
(7 - 0)	= T_F

Word 2:

(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 of the whole trigger window are reported if at least one value is above the threshold H for the particular channel. The first word indicates the channel and slot number and number of samples in the trigger window. Multiple continuation words contain two raw samples each. The earlier sample is stored in the most significant half of the continuation word. Strict time ordering is maintained in the order of the continuation words. A *sample not valid* flag may be set for any sample; for example the last reported sample is tagged *not valid* when the pulse interval consists of an odd number of samples.

Word 1:

(31)	= 1
(30 - 27)	= 4
(26 - 20)	= channel number (0-71)
(19 - 15)	= slot number (set by VME64x backplane)
(14 - 12)	= reserved (read as 0)
(11 - 0)	= window width (in number of samples)

Word 2-N:

(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 Raw Data (6):** All the raw ADC data samples for an identified pulse are reported. The first word contains the channel number, the slot number and the sample number of threshold crossing (first sample above the threshold). For the rest of the words the same rules apply as for the Window Raw Data (4).~~

Word 1:

(31)	= 1
(30 - 27)	= 6
(26 - 20)	= channel number (0-71)
(19 - 15)	= slot number (set by VME64x backplane)
(14 - 12)	= reserved (read as 0)
(11 - 0)	= sample number of threshold crossing

Word 2-N:

(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 Data (integral and time in CDC format) (7): Integral, time, pedestal and amplitude of an identified pulse within the trigger window.

Word 1:

(31)	= 1
(30 - 27)	= 7
(26 - 20)	= channel number (0-71)
(19 - 15)	= slot number (set by VME64x backplane)
(14 - 4)	= leading edge time
(3)	= time quality bit
(2 - 0)	= overflow count

Word 2:

(31)	= 0
(30 - 23)	= pedestal
(22 - 9)	= integral
(8 - 0)	= first max amplitude

Pulse Data (integral and time in FDC format) (8): Integral, time, pedestal and peak time of an identified pulse within the trigger window.

Word 1:

(31)	= 1
(30 - 27)	= 8
(26 - 20)	= channel number (0-71)
(19 - 15)	= slot number (set by VME64x backplane)
(14 - 4)	= leading edge time
(3)	= time quality bit
(2 - 0)	= overflow count

Word 2:

(31)	= 0
(30 - 19)	= integral
(18 - 11)	= peak time (in samples)
(10 - 0)	= pedestal

Pulse Data (peak amplitude and time in FDC format) (9): Amplitude, time, pedestal and peak time of an identified pulse within the trigger window.
The only difference between data types 8 and 9 is that in type 9 we have the peak amplitude instead of the integral.

Word 1:

(31)	= 1
(30 - 27)	= 9
(26 - 20)	= channel number (0-71)
(19 - 15)	= slot number (set by VME64x backplane)
(14 - 4)	= leading edge time
(3)	= time quality bit
(2 - 0)	= overflow count

Word 2:

(31)	= 0
(30 - 19)	= peak amplitude
(18 - 11)	= peak time (in samples)
(10 - 0)	= pedestal

Pulse Data and Pulse Samples, CDC format (10): The samples within the pulse window are appended as continuation words to the extracted pulse parameters formatted as in data type 7.

Word 1:

(31)	= 1
(30 - 27)	= 10
(26 - 20)	= channel number (0-71)
(19 - 15)	= slot number (set by VME64x backplane)
(14 - 4)	= leading edge time
(3)	= time quality bit
(2 - 0)	= overflow count

Word 2:

(31)	= 0
(30 - 23)	= pedestal
(22 - 9)	= integral
(8 - 0)	= first max amplitude

Word 3-N:

(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 Data and Pulse Samples, FDC format (11): The samples within the pulse window are appended as continuation words to the extracted pulse parameters formatted as in data type 8.

Word 1:

(31)	= 1
(30 - 27)	= 11
(26 - 20)	= channel number (0-71)
(19 - 15)	= slot number (set by VME64x backplane)
(14 - 4)	= leading edge time
(3)	= time quality bit
(2 - 0)	= overflow count

Word 2:

(31)	= 0
(30 - 19)	= integral
(18 - 11)	= peak time (in samples)
(10 - 0)	= pedestal

Word 3-N:

(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)

~~**Scaler Header (12):** Indicates the beginning of a block of scaler data words. The number of scaler words to follow is given in the header.~~

(31)	= 1
(30 - 27)	= 12
(26 - 10)	= reserved (read as 0)
(9 - 0)	= number of scaler words to follow

Word 2-N:

(31)	= 0
(30 - 0)	= scaler counts

Event Trailer (13): (**suppressed for normal readout - debug mode only**) last word from ADC processing chip for event

(31)	= 1
(30 - 27)	= 13
(26 - 22)	= slot number (set by VME64x backplane)
(21 - 0)	= undefined

Data Not Valid (14): module has no data available for read out, or there is an error condition in the module that will not allow it to transfer data.

(31)	= 1
(30 - 27)	= 14
(26 - 22)	= slot number (set by VME64x backplane)
(21 - 0)	= undefined

Filler Word (15): non-data word appended to the block of events. Forces the total number of 32-bit words read out of the module to be a multiple of 2 or 4 when 64-bit or 2e VME block transfers are used.

(31)	= 1
(30 - 27)	= 15
(26 - 22)	= slot number (set by VME64x backplane)
(21 - 0)	= undefined

References

[1] "VME Data Format Standards for JLab Modules" GlueX-doc-2365 http://argus.phys.uregina.ca/cgi-bin/public/DocDB/ShowDocument?docid=2365 and also a more recent version at https://halldweb1.jlab.org/wiki/images/5/58/JlabModuleDataFormat.pdf

[2] E.Jastrzembski, H.Dong "Summary of the FADC250 Operating Modes" 2/18/2014