GlueX-doc-2274-v5 February 15, 2015

FADC125 OPERATION AND DATA FORMAT REQUIREMENT V5.01

DAVID LAWRENCE
CODY DICKOVER
NAOMI JARVIS
LUBOMIR PENTCHEV
BENEDIKT ZIHLMANN

1. Introduction

This document specifies the data format produced by the JLab 125 MHz Flash ADC module (fADC125) and those of its operating modes which differ from the Flash 250 MHz ADC module (fADC250) [2]. All data produced by the fADC125 will be in the form of 32-bit words. The module will adhere to the overall format standard described in the document [1] produced and maintained by the JLab Data Acquisition group[1].

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 combine 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 different 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 needs also a 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 exhibit a wide range of 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 (September 2014) 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 that 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. CDC Pulse Analysis

The arrival time of 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 comprise the pedestal window (NP samples) followed immediately by the hit search window (NH samples). NP is chosen to be an integer power of 2 so that the mean pedestal can be calculated easily by summing the NP values and right-shifting the sum by PSHIFT bits, where NP = 2^{PSHIFT}. The hit search window is the region to be searched for a hit.

In the following description, the hit search window occupies samples WS to WE in the buffer. The pedestal window starts with sample WS−NP and ends with sample WS−1; the hit search window starts with sample WS and ends with sample WE. Thus the trigger window, comprising both pedestal and hit search windows, starts with sample WS−NP and ends with sample WE.

The pulse analysis uses several thresholds which can be related to the pedestal standard deviation, σ. These are determined during calibration and uploaded into the fADC firmware as constants. Values optimized using the small prototype are available as a starting point: hit identification threshold H $\sim 5\sigma$, high timing threshold TH $\sim 4\sigma$, low timing threshold TL $\sim \sigma$. There is also a constant which defines the time difference (as number of samples) between the hit threshold crossing sample, TC, and the sample to be used for local pedestal, TP.

Following the trigger arrival, 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 the data meet or exceed a hit threshold PINIT+H. If the hit is found, the sample number 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 TP, from TP−NP2+1 to TP.
(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 sample WE, 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, until the count exceeds 6.

(7) The hit search window samples are scanned, starting with sample TC, for the first maximum in value.

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

Time: Leading edge time in units of sample/10

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 sample WE, right-shifted IBIT bits, all bits set if the integral exceeds $2^{IW} - 1$

Overflow count: Number of samples between TC and WE with overflow bit set, return 7 if more than 6 are found

Pedestal: Mean local pedestal (mean of samples from TP-NP2+1 to TP), right-shifted PBIT bits, all bits set if greater than $2^{PW} - 1$

First maximum amplitude: First maximum in signal value between TC and WE, right-shifted ABIT bits

3. FDC Pulse Analysis

The pulse detection is very similar to the algorithm described in [2] for fADC250.

The trigger window is defined by its beginning, BW, backwards with respect to the trigger time and width, NW, both in 8 ns samples. The samples within the window are numbered from 1 to NW.

The pulse identification is initiated if within the trigger window, there's at least one sample above the threshold, H, programmed individually for each channel. The pulse window starts NSB samples before the threshold crossing and ends at NSA samples after the threshold crossing, so that the total width is NSB+NSA samples. As in case of fADC250, if these limits are outside of the trigger window, the corresponding trigger window limit defines the pulse window.

We specify the extraction of the pulse parameters as follows:

Leading edge time: see the algorithm described below.

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

Integral: sum over the pulse window shifted right by IBIT bits (i.e. divided by 2^{IBIT}).

Overflows: number of samples with overflowed values within the pulse window, or 7 if bigger than 7.

Pedestal: calculated by summing PS samples (PS = 2^{PBIT}) starting from sample number PB within the trigger window. The sum is shifted right by PBIT bits and reported as a pedestal.

Peak amplitude: the amplitude of the first maximum within the pulse window, determined by finding the first sample beyond the threshold crossing for which the sample value decreases.

Peak time: sample number of the maximum.

4. Leading Edge Time Algorithm

The algorithm uses a small number of samples surrounding the hit threshold crossing. It checks that the sample values are suitable for upsampling - if not, it returns a rough estimate of the hit time - and, if they are suitable, it proceeds to upsample a small region surrounding the low timing threshold crossing. It tests the accuracy of the upsampling, and if successful, it returns the leading edge time as determined to be the time when the upsampled data (corrected if necessary) rise above the low timing threshold, otherwise it returns the midpoint of the samples before (or at) and after the low timing threshold crossing. The quality factor returned is 0, if the time returned is the leading edge time, or 1, if not.

In more detail, the process is as follows:

(1) A small subset NU of consecutive samples surrounding the threshold crossing sample are sent to the time-finding module, with the threshold crossing sample (previously TC) in place XTHR.
(2) The samples from 0 to TP are scanned. If any values are found to be equal to 0 or greater than LIMIT_PED_MAX, the algorithm returns a time of XTHR$\times$10$-$RT and a quality factor of 1. The value of RT is set in configuration parameters.
(3) The samples from TP+1 to NU$-$1 are scanned. If any sample values equal to 0 or greater than LIMIT_ADC_MAX are found, the algorithm returns a time of XTHR$\times$10$-$RT and a quality factor of 1.
(4) A constant offset added to all samples such that the minimum sample value is equal to SET_ADC_MIN. This decreases the likelihood of calculating any negative upsampled values.
(5) The local pedestal, P, is obtained as the ADC value at sample TP.
(6) The algorithm searches through the samples, starting with sample TP+1, to find sample TCH, where the ADC value first exceeds or equals the high timing threshold, P+TH. If this is not found, the algorithm returns and the value of TCH$\times$10$-$RT and a quality factor of 1.
(7) Having found sample TCH, it then searches the samples from TCH back toward TP to find sample TCL, where the ADC value is less than or equal to the low timing threshold, P+TL. If TCL $>$ NU$-$7, the time of XTHR$\times$10$-$RT is returned with a quality factor of 1, as in this case there are not enough later samples to calculate the upsampled values.

(8) 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.

(9) If any upsampled value is negative, the midpoint time TCL×10+5 is returned, with a quality code of 1.

(10) 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 errors are summed and compared with LIMIT_UPS_ERR; if this is exceeded, the midpoint time TCL×10+5 is returned, with a quality code of 1.

(11) The mean of the upsampling errors is obtained and added to TL to obtain an adjusted threshold value.

(12) 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 TCL×10+5 is returned, with a quality code of 1.

(13) 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.

5. Timing algorithm configuration constants

The configuration constants for the timing algorithm are described in Table 1. These will be programmable through the VME backplane. They are common for all channels.

6. CDC 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. Note that the sample period is 8 ns and the ADC units are 12 bits, 0-4095.

7. FDC pulse analysis configuration parameters

The configuration parameters are described in Table 3. They are programmable through the VME backplane.

All the quantities in the text abbreviated with capital letters are assumed to be programmable parameters. Unless otherwise noted, these parameters are common for all the channels.

Table 1. List of configuration constants for the timing algorithm

Name	Description	Units	Default
NU	Number of samples in the subset to send to the time-finding module	samples	15
TS	Position in the subset of the earliest sample which meets or exceeds the hit-threshold H (the first sample is in position 0)	samples	9
TP	Position in the subset of the sample which is to be used as local pedestal (the first sample is in position 0)	samples	5
RT	If ADC values are unsuitable for upsampling, return the hit time of $TS \times 10 - RT$	sample/10	24
LIMIT_PED_MAX	Maximum permissible value for samples 0 to TP	ADC units	511
LIMIT_ADC_MAX	Maximum permissible value for samples TP+1 to NU−1	ADC units	4095

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

Table 2. CDC pulse analysis configuration parameters

Name	Description	Units	Default
NW	Trigger window length NW = NP+NH	samples	116
NP	Pedestal window length	samples	16
NP2	Local pedestal window length	samples	16
NH	Hit search window length	samples	100
WS	Position in buffer of start of hit search window length	samples	
WE	Position in buffer of end of hit search window length WE = WS + NH−1	samples	
H 0-71	Hit threshold for each channel	ADC units	80
TH 0-71	High timing threshold for each channel	ADC units	64
TL 0-71	Low timing threshold for each channel	ADC units	16
IS	If the ADC values are unsuitable for upsampling, integration starts IS samples before the hit threshold crossing	samples	3
IBIT	2^{IBIT} scale factor for integral		4
ABIT	2^{ABIT} scale factor for first signal amplitude		3
PBIT	2^{PBIT} scale factor for pedestal		2
IW	width of field for integral	bits	14
AW	width of field for signal amplitude	bits	9
PW	width of field for pedestal	bits	8

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:

Table 3. List of Configuration parameters, FDC

Name	**Description**	**Units**
BW	trigger window beginning	8 ns samples
NW	trigger window width	8 ns samples
H 0-71	72 thresholds, one for each channel	12-bit ADC units
TH 0-71	High timing threshold for each channel	12-bit ADC units
TL 0-71	Low timing threshold for each channel	12-bit ADC units
NSB	beginning of pulse window w.r.t. threshold crossing	8 ns samples
NSA	end of pulse window w.r.t. threshold crossing	8 ns samples
PS	number of samples for pedestal determination	8 ns samples
PB	first sample for pedestal determination	8 ns samples
IBIT	2^{IBIT} scale factor for integral	bits
PBIT	2^{PBIT} scale factor for pedestal	bits

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 TH 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 and time 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 and 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): The only difference compared to the previous type is that 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 integral and time of an identified pulse are reported, followed by all the samples within the pulse window. This essentially combines the data from types 6 and 7 by appending the samples as continuation words to the extracted pulse parameters. The one difference is that the first sample number (identifying the threshold crossing) is not recorded here as it is for data type 6. However, one can use the extracted pulse time and peak time to find where the pulse lies within the window. See notes for data types 6 and 7 for details. This data type is needed only for diagnostics.

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

<u>Word 3-N:</u>

(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 integral and time of an identified pulse are reported, followed by all the samples within the pulse window. This essentially combines the data from types 6 and 7 by appending the samples as continuation words to the extracted pulse parameters. The one difference is that the first sample number (identifying the threshold crossing) is not recorded here as it is for data type 6. However, one can use the extracted pulse time and peak time to find where the pulse lies within the window. See notes for data types 6 and 7 for details. This data type is needed only for diagnostics.

<u>Word 1:</u>

(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

<u>Word 2:</u>

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

<u>Word 3-N:</u>

(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