

FADC125 OPERATION AND DATA FORMAT REQUIREMENT V6.5

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 group¹. 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], modified where necessary for the drift chambers. The main differences are in the pulse data formats (data types 5, 6 and 9), which permit output of pulse time, amplitude, integral and pedestal in one pair of data words, and in the methods used to calculate the pulse time and integral. These are detailed later in this document.

The requirements for the FDC and CDC differ due to the nature of their signals. In the FDC, cathode strips are connected to the fADC modules, and the signal data are used to find the position of the hit. The signal from a charge avalanche is received by several adjacent strips and a weighted sum of the signals is used to find the track position. The central strip receives the largest signal but the smaller signals from the strips on either side are given larger weighting factors, and so the position resolution obtained is very sensitive to noise. The best results are obtained using either the signal integral or the maximum amplitude, depending on the characteristics of the noise present. We also need rough timing information for comparison with the hit time obtained from the TDCs connected to the FDC wires, in order to reduce the background of random hits. In the CDC, the fADC125 modules receive signal data from the anode wires which receive pulses with 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 is useful for particle identification. The requirements for the two detectors are broadly similar, but there are important differences in the algorithms and the data formats which are explained below.

¹An 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.

The following sections describe the methods used for pulse analysis, the configuration parameters, the data formats and operating modes. There are two versions of the firmware, one uses the operating modes for the CDC, and the other, the FDC.

2. PULSE ANALYSIS

The pulse analysis is illustrated in Figure 1 and described below.

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. This delay is specified as a number of samples by the pulse latency configuration parameter, PL. The trigger window contains NW consecutive samples located at the start of the data buffer which contain the pedestal window (NP samples), followed immediately by the hit search window, followed by an extra NE samples. The last sample in the hit search window is designated sample WE for the purpose of description only ($WE = NW - NE - 1$).

The NE samples at the end of the trigger window are included to permit upsampling at the end of the hit search 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, or in the pedestal window.

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 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 mean 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 algorithm used to determine the pulse time is described later. The range of samples which are summed to obtain the signal integral starts with the sample containing the pulse leading edge time and continues either for a total of IE samples, or until sample WE has been included. 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.

For the FDC, if a pulse is found, the samples in the hit window after the initial peak are searched for further peaks, until a total of NPK peaks have been found or the end of

the window is reached. The CDC firmware has NPK set equal to 1, so only one peak can be found.

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, which is described later. This returns the time of the pulse leading edge.
- (5) The integrated signal is calculated, starting with the sample containing the pulse leading edge and continuing either for a total of IE samples, or until sample WE has been included. 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 increasing samples followed by two subsequent consecutive samples which decrease in value. *** Check with Cody re pk ID ***
- (8) Up to $NPK-1$ later peaks in the hit window are identified.

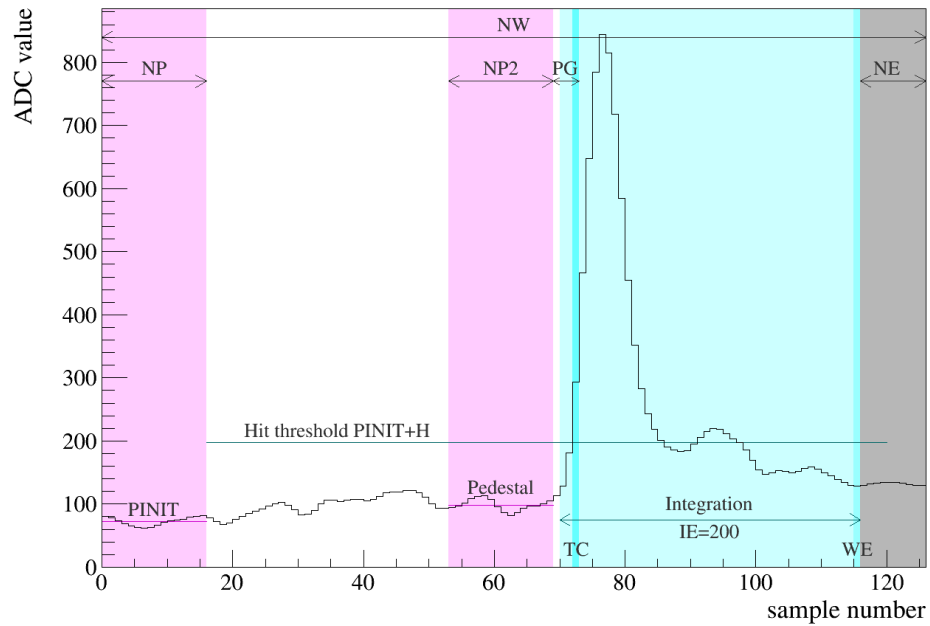
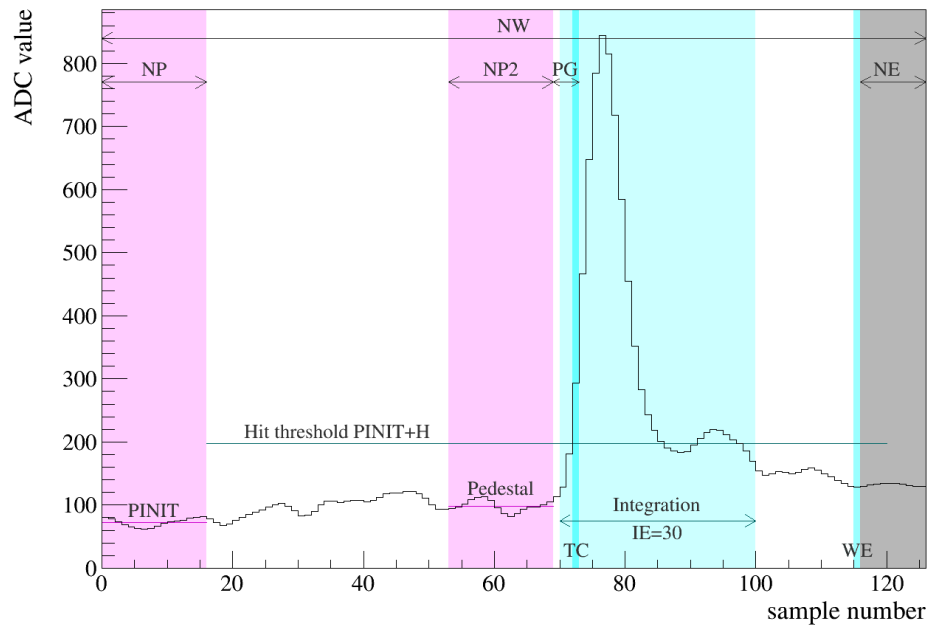


FIGURE 1. Examples of pulse analysis, showing the trigger, pedestal and hit search windows, and the integration region used with configuration parameter IE set to 30 (top) and 200 (bottom). Integration starts with the sample containing the leading edge of the pulse as determined by the timing algorithm.

3. PULSE ANALYSIS READOUT QUANTITIES

The pulse analysis is mostly similar for the CDC and FDC, but different readout formats are used in order to meet the different requirements in range and precision. Pulse integration starts with the leading edge of the pulse and ends with the earlier of IE samples later and sample WE. 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 $NPK > 1$, the integral or amplitude, peak time and pedestal are found for NPK peaks following the initial threshold crossing. The second and later peaks are identified by finding a sample with greater value than the previous sample and followed by two consecutive samples decreasing in value - only the first peak is required to meet or exceed the hit threshold. *** Check with Cody re pk ID ***

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

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

Time Quality Code: Set to 0 if time is good, 1 if time is less accurate

Integral: Signal integral starting with the sample containing the leading edge of the pulse and ending either after IE samples or with sample WE, right-shifted IBIT bits, all bits set if the integral exceeds the field maximum

Overflow count: Number of samples included in the integration with the 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. PULSE ANALYSIS CONFIGURATION PARAMETERS

The configuration parameters are described in Table 1. These will be programmable through the VME backplane. Unless otherwise noted, they are common for all channels. Maximum permissible parameter values are given in Table 2, together with typical values for the CDC and FDC.

TABLE 1. Pulse analysis configuration parameters

Name	Description	Units
PL	Pulse latency	samples
NW	Trigger window length	samples
NPK	Maximum number of peaks to search for	
P1	Initial pedestal window length $NP = 2^{P1}$ samples	
P2	Local pedestal window length $NP2 = 2^{P2}$ samples	
PG	Time between local pedestal window and hit threshold crossing	samples
IE	Integral ends after IE samples, unless sample WE is reached	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 2. Typical and maximum values for the configuration parameters

* Check with Cody re PL *

Name	CDC	FDC	Maximum
PL	400	400	65535
NW	120	44	1024
NPK	1	4	15
P1	4	4	7
P2	4	4	7
PG	4	4	7
IE	200	30	1023
H0-71	100	100	511
TH0-71	80	80	511
TL0-71	20	20	63
IBIT	4	4	7
ABIT	3	0	3
PBIT	0	0	3

Using the typical scaling factor values given in Table 2, the available range before and after scaling of the readout quantities is listed in Tables 3 and 4. The unscaled readout quantities are also included for completeness.

TABLE 3. CDC readout quantity available range, data type 5

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 4. FDC readout quantity available range, data types 6 and 9

Quantity	Field width (bits)	Field range	Scaling factor	Range before scaling
Time	11	2047		2047
Quality	1	1		1
Integral	12	4095	16	65536
Overflows	3	7		7
Pedestal	11	2047	1	2047
Amplitude	12	4095	1	4095
Peak time	8	255		255

5. 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 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 code set to 0. A number of validation tests are performed during this process; if any are failed, then a less accurate estimate of the leading edge time is returned, including a code to indicate the error condition, with the quality code set to 1.

The details of the process are described below and illustrated in Figure 2. The quality codes and time values returned are summarized in Table 6.

- (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 X for convenience.
- (2) The NU samples are scanned. If any sample has a value of 0, the algorithm returns a time of $X \times 10 - 29$ and a quality code of 1. If any sample in the range 0 to PED is found to have a value greater than PED_MAX, the algorithm returns a time of $X \times 10 - 28$ and a quality code of 1.
- (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 a time of $X \times 10 - 27$ and a quality code of 1.
- (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 time $TCL \times 10 + 4$ is returned, with a quality code of 1. If the ADC value of sample TCL is equal to the low timing threshold, the time $TCL \times 10$ is returned, with a quality code of 0.
- (7) The ADC data are upsampled by a factor of 5 to calculate values at 1.6 ns intervals, from TCL to TCL+1.
- (8) If any upsampled value is negative, the time $TCL \times 10 + 5$ is returned, with a quality code of 1.
- (9) The upsampling error at TCL is calculated and added to P+TL to obtain an adjusted threshold value. The upsampling error is the difference between the original sample

and the corresponding upsampled value; the upsampled values tend to be too high when the slope of the leading edge is extremely steep.

- (10) The upsampled values are searched from $TCL+1$ down to TCL to find the point where the values fall equal to or below the adjusted threshold. If this is found at $TCL+1$, the time $TCL \times 10 + 9$ is returned with a quality code of 1.
- (11) The adjusted threshold crossing time is found, to the nearest tenth of a sample, by interpolating between the upsampled points before and after the threshold crossing. This quantity is returned by the algorithm as an integer, with a quality code of 0.

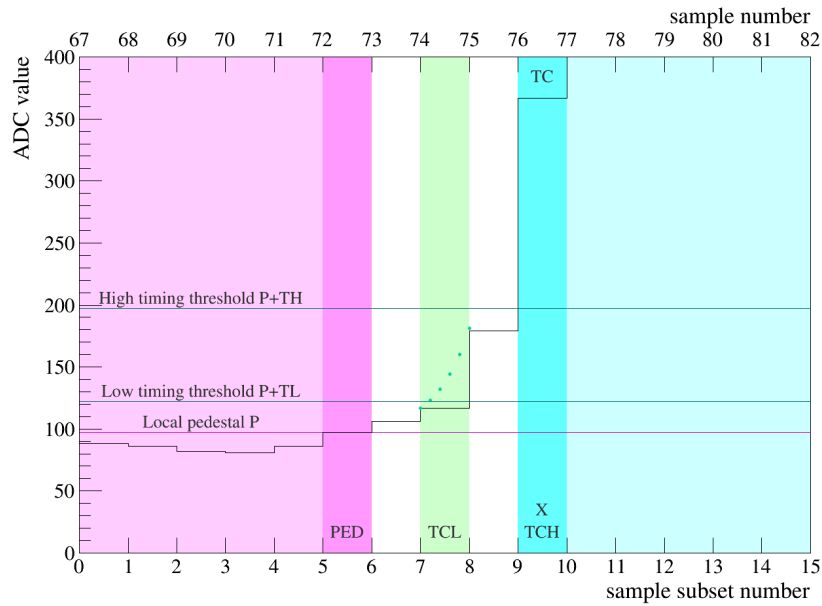


FIGURE 2. Example of pulse analysis, with timing thresholds and upsampled data

6. TIMING ALGORITHM CONFIGURATION CONSTANTS

The configuration constants for the timing algorithm are described in Table 5. 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 5. 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
NE	Number of extra samples following the hit search window	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	5
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

7. TIMING ALGORITHM QUALITY CODES

The quality codes and time values returned by the timing algorithm are summarized in Table 6. In the first case (a) the algorithm completes successfully and the pulse leading edge time is returned, in units of tenth-samples, together with a quality code of 0. In the remaining six cases (b-g), an estimate of the pulse time and a quality code of 1 are returned; the last digit of the pulse time indicates which error condition was encountered by the timing algorithm. The time estimate is more accurate for cases e-g than for cases b-d.

In cases b, c and d, problems were identified with one or more values in the subset of data samples sent to the algorithm, and the time returned is the hit threshold crossing time (X), converted to tenth-samples, minus a constant. This gives a rough estimate of the pulse time. In cases e, f and g, the sample containing the leading edge has been found, and the time returned is the low threshold crossing time, converted to tenth-samples, plus a constant.

Cases d and e are most likely to be encountered when there is a large change in the pedestal between the start of the hit search window and the arrival of the pulse. In case d, the high timing threshold crossing was not found, however in case e, both the high and low timing threshold crossings were found in the sample data, but the low timing threshold crossing occurred too late in the dataset for upsampling to be possible. In cases f and g, the low timing threshold crossing was found in the sample data but a problem was found with the upsampled data; the most probable cause of this is an extremely steep gradient at the start of the pulse.

TABLE 6. Quality factor codes returned by the timing algorithm

Case	Code	Time returned	Condition
a	0	Leading edge time	Good
b	1	$X \times 10 - 29$	Sample value of 0 found
c	1	$X \times 10 - 28$	Sample value greater than PED_MAX found
d	1	$X \times 10 - 27$	Sample values lie below the high timing threshold
e	1	$TCL \times 10 + 4$	Low timing threshold crossing occurs too late
f	1	$TCL \times 10 + 5$	One or more upsampled values are negative
g	1	$TCL \times 10 + 9$	The upsampled values are too high

8. RESTRICTIONS ON CONFIGURATION PARAMETERS AND CONSTANTS

- (1) $PL \geq NW$
- (2) $NW > NP + 6$
- (3) $NW > NU$
- (4) $NU > 14$
- (5) $NE \geq NU - PG - PED$
- (6) $NE \geq 6$
- (7) $NPK > 0$
- (8) $NP \geq NP2$
- (9) $NP2 > 0$
- (10) $H > TH > TL$
- (11) $PED > 4$
- (12) $PG > 0$
- (13) $PED + PG < NU$

9. FADC125 DATA FORMAT

9.1. Data Type List. Data types 5, 6 and 9 are the new types defined specifically for the fADC125. Data types 7, 8 and 10 were used by the earlier fADC125 firmware for pulse integral, pulse time, and pulse pedestal; we are leaving them unused here so that the same software can be used to decode old and new data. Data types 5, 6 and 9 were not used previously and the remaining data types 0-4 and 13-15 are unchanged. Types 11 and 12 remain unused and available for future developments.

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

9.2. 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.

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

0: Block Header: 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)

1: Block Trailer: 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

2: Event Header: 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)

3: Trigger Time: 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. 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

4: Window Raw Data: 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. * **Check with Cody re flag** *

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) = NW (window width, 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)

5: Pulse Data, CDC format: integral, time, pedestal and amplitude of an identified pulse within the trigger window.

Word 1:

(31) = 1
 (30 - 27) = 5
 (26 - 20) = channel number (0-71)
 (19 - 15) = NPK = 1
 (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

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

Word 1:

(31) = 1
 (30 - 27) = 6
 (26 - 20) = channel number (0-71)
 (19 - 15) = NPK
 (14 - 4) = leading edge time
 (3) = time quality bit
 (2 - 0) = overflow count

Word 2 to NPK+1:

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

9: Pulse Data, FDC format (peak amplitude and time): amplitude, time, pedestal and peak time of an identified pulse within the trigger window. The only difference between data types 6 and 9 is that in type 6 we have the peak amplitude instead of the integral.

Word 1:

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

Word 2 to NPK+1:

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

13: Event Trailer: (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

14: Data Not Valid: 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

15: Filler Word: 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

9.4. ADC Modes. All of the fADC operating modes produce the header and trailer words: block header (type 0), block trailer (type 1), event header (type 2), trigger time (type 3), data not valid (type 14) and filler (type 15). The modes and data types included are listed below. The data types in parentheses are output when necessary; if 'data not valid' (type 14) appears, then it will be alone, as it is caused by attempted reads when the fADC buffer is empty.

CDC short:

(block header)	type 0	1 word	0x8
event header	type 2	1 word	0x8
trigger time	type 3	2 words	0x98 00
CDC pulse data	type 5	2 words	0xA
(block trailer)	type 1	1 word	0x8
(filler)	type 15	1 word	0xF

FDC short:

(block header)	type 0	1 word	0x8
event header	type 2	1 word	0x8
trigger time	type 3	2 words	0x98 00
FDC pulse data	type 6	1+NPK words	0xB etc
(block trailer)	type 1	1 word	0x8
(filler)	type 15	1 word	0xF

FDC short, with amplitude:

(block header)	type 0	1 word	0x8
event header	type 2	1 word	0x8
trigger time	type 3	2 words	0x98 00
FDC pulse data	type 9	1+NPK words	0xC etc
(block trailer)	type 1	1 word	0x8
(filler)	type 15	1 word	0xF

CDC long:

(block header)	type 0	1 word	0x8
event header	type 2	1 word	0x8
trigger time	type 3	2 words	0x98 00
CDC pulse data	type 5	2 words	0xA
window raw data	type 4	1+NW/2 words	0xA etc
(block trailer)	type 1	1 word	0x8
(filler)	type 15	1 word	0xF

FDC long:

(block header)	type 0	1 word	0x8
event header	type 2	1 word	0x8
trigger time	type 3	2 words	0x98 00
FDC pulse data	type 6	1+NPK words	0xA etc
window raw data	type 4	1+NW/2 words	0xA etc
(block trailer)	type 1	1 word	0x8
(filler)	type 15	1 word	0xF

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