FADC125 DATA FORMAT V2.0

DAVID LAWRENCE
CODY DICKOVER
NAOMI JARVIS
LUBOMIR PENTCHEV

1. Introduction

This document specifies the data types produced by the JLab 125 MHz Flash ADC module (FADC125). 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 "VME Data Format Standards for JLab Modules" produced and maintained by the JLab Data Acquisition group[1].

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.

[1]At the time of this writing, the document "VME Data Format Standards for JLab Modules" has been circulated via e-mail to select individuals and has not been officially published online. A copy of a draft version has been posted on the Hall-D wiki here: https://halldweb1.jlab.org/wiki/images/5/58/JlabModuleDataFormat.pdf . Contact a member of the JLab DAQ group to obtain a recent copy. Much of the critical information has been reproduced in this document so obtaining that document is not really required.

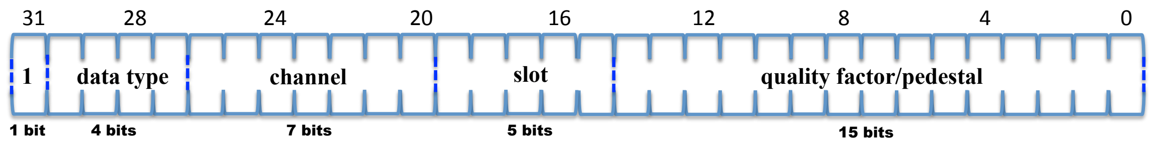
3. Data Type Tags

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:	– unused –
8:	– unused –
9:	pulse data (sum and time)
10:	pulse data and pulse samples
11:	– unused –
12:	scaler data
13:	event trailer (debug only)
14:	data not valid (empty module)
15:	filler (non-data) word

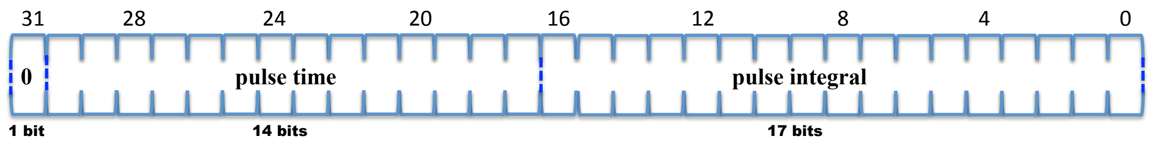
4. Pulse Data

The FADC125 has the ability to apply a pulse finding algorithm to the samples it obtains so that only the integral and time of the identified pulses are recorded, thus, reducing the data produced by the module. The results of the algorithm are stored in data type 9 with two 32-bit words per identified pulse (see details below). In addition, the module can output the full set of samples identified as a pulse using data type 6. Data type 10 basically combines the information from types 9 and 6 by first giving the pulse integral and time followed by the set of samples from which they were derived. For most applications, only data type 9 will be desired as it is the most compact representation of the relevant data. The pulse finding and time extraction algorithm may also provide information relating to the quality of the values it produces. These are stored in several bits reserved for *quality factor/pedestal.* The exact definition for these bits has not yet been specified. The following figure shows a diagram of the bit assignments for pulse data.

First word



Second word



5. Data Types

What follows are descriptions of the various data types produced by the FADC125:

Block Header (0): indicates the beginning of a block of events (High speed read-out 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 (used to specify how quality factor/pedestal bits are defined)
(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 from a trigger window pulse. Raw samples from an interval of the trigger window that starts with the first sample and includes a fixed number of samples. The first word indicates the channel number and number of samples in the 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 - 21)	= channel number (0-71)
(20 - 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): Raw ADC data samples from an identified pulse. The first sample number is relative to the beginning of the trigger window (sample 0). The *sample not valid* flag may be set for any sample including the last reported sample when the pulse integral consists of an odd number of samples.

Word 1:

(31)	= 1
(30 - 27)	= 6
(26 - 20)	= channel number (0-71)
(19 - 15)	= slot number (set by VME64x backplane)
(14 - 10)	= reserved (read as 0)
(9 - 0)	= first sample number for pulse

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 (9): Integral and time of an identified pulse within the trigger window.

Word 1:

(31)	= 1
(30 - 27)	= 9
(26 - 20)	= channel number (0-71)
(19 - 15)	= slot number (set by VME64x backplane)
(14 - 0)	= quality factor/pedestal

Word 2:

(31)	= 0
(30 - 17)	= pulse time (in 150ps time units)
(16 - 0)	= pulse integral

Pulse Data and Pulse Samples (10): Integral and time of an identified pulse within the trigger window followed by the samples from the trigger window from which the pulse was extracted. This essentially combines the data from types 9 and 6 by appending the samples as continuation words to the extracted pulse parameters. The one significant difference is that the first sample number of the pulse is not recorded here as it is for data type 6. However, one can use the extracted pulse time as a rough indication of where the pulse lies within the window. See notes for data types 9 and 6 for details.

Word 1:

(31)	= 1
(30 - 27)	= 10
(26 - 20)	= channel number (0-71)
(19 - 15)	= slot number (set by VME64x backplane)
(14 - 0)	= quality factor/pedestal

Word 2:

(31)	= 0
(30 - 17)	= pulse time (in 150ps time units)
(16 - 0)	= pulse integral

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