GlueX-doc-2776

Version 0.3

Original draft by A. Somov, July 14th, 2015

Updated on August 6th, 2015
(add draft of the data format)

Updated according to specs from Elton and the calorimeter group on August 28th, 2015
(negative N_{SB}, some modifications in the data format)

Addendum to FADC250 firmware specifications

(Hall D request)

The fadc250 firmware has been successfully used during the first commissioning runs in the spring of 2015. Some small additions and modifications to the fadc250 firmware algorithm are required in order to reduce the trigger processing time, provide additional monitoring tools, and reduce the data size reported by the fadc250 module.

Trigger

Specifications for the currently implemented trigger energy sum algorithm are described in GlueX-doc-2388. The following changes have to be made:

N_{SB} range

In order to reduce the trigger processing time and save FPGA resources the value of the parameter N_{SB} for the trigger should be fixed to 3 samples.

Trigger thresholds

Trigger and read out thresholds for both the hit-bit and energy sum algorithms should be set with respect to the fadc count 0, i.e., no baseline subtraction should be performed[1].

[1] This was originally implemented in the firmware version 90E, which was used during data taking. In later fadc versions thresholds were applied after pedestal subtraction.

Disabling channels for trigger and scalers

A 16-bit channel mask will be used to disable fadc channels for the trigger and hit-bit scalers. The asynchronous pedestal readout should not be disabled using this mask. Currently, the mask completely disables fadc channels. We would propose to use this mask and disable only inputs to the trigger and hit-bit scalers. Note, channels can still be excluded from the readout if needed by setting readout thresholds to large values.

Individual trigger thresholds (optional)

In the current firmware version, a common threshold for 16 fadc channels is used. In the future, we may be interested in applying channel dependent trigger thresholds, i.e., 16 individual (12 bit) thresholds can be implemented.

Asynchronous pedestal readout

Pedestals for various sub-detectors can be monitored by reading out pedestal sums for each fadc channel from fadc registers. During production runs, pedestals will be read out at SYNC_EVENT. Small baseline corrections (adjusting DAC values) can be applied during data taking if needed to account for the long term baseline drift. Computation of the pedestals shall be implemented as follows:

- Fadc amplitudes for each channel will be summed in a time window with a programmable size. The summation window size N_{ASYNC_PED} can be changed using a programmable register in the range between 4 and 16 samples.
- Fadc amplitudes in each sample are limited to the range 0 – 1023 (10 bits)[2]. The resulting energy sum can be coded using 10 + 4 = 14 bits.
- In each sample where the summation is performed a fadc amplitude has to be compared with the user defined threshold $A^{MAX}_{ASYNC\_PED}$ (10 bits). The threshold is defined with respect to the fadc count 0. If the amplitude is larger than $A^{MAX}_{ASYNC\_PED}$ or the fadc underflow occurs, the pedestal sum quality bit (bit 15) will be set to 1.
- Pedestals will be read out for all 16 fadc channels at the same time.

[2] Note, originally we were planning to limit the amplitude range to 255 fadc counts as the baseline for most sub-detectors is set to 100. Larger value for the baseline may be required in the future for the TPOL; studies are in process.

Readout

Pedestal sum

In order to allow for event-by-event pedestal subtraction, the 14-bit pedestal energy sum computed in the beginning of the readout window should be reported for every hit. The summation window size can be changed using a register $\boldsymbol{N_{PED}}$ between 4 and 16 samples. The same value of the pedestal threshold $\boldsymbol{A^{MAX}_{ASYNC\_PED}}$ can be used for the asynchronous and triggered readouts. The pedestal quality bit will be set in the same way as for the asynchronous readout and will be reported in data with the hit information, see the data format. The pedestal subtraction can be performed during offline analyses.

N_{SB} range

The maximum value of the programmable parameter N_{SB} should be reduced to N_{SB} (max) = 7 samples (3 bits), which will save FPGA resources. The readout window can start before or after (so called negative N_{SB}) the threshold sample. The negative N_{SB} can be enabled using a parameter N_{SB}^{SIGN} : 0 (before threshold sample), 1 (after threshold sample).

Threshold crossing

Taking data with small readout thresholds can result in multiple fake pulses in the readout window. The fpga pulse processing is triggered by pulse threshold crossing, which is currently checked for each individual time sample. In order to reduce the number of fake pulses and allow to take data at lower thresholds, the pulse-over-threshold should be checked for a couple of consequtive samples. The number of samples used in this check should be set using a programmable register $\boldsymbol{N_{SAMP}^{THR}}$ (2 bits). $\boldsymbol{N_{SAMP}^{THR}}$ = 0 and 3 correspond to 1 (pulse is above the threshold for 4 ns) and 4 samples, respectively.

Time-over-threshold counter

The pulse integral is computed in the range between NSB and NSA. The number of samples where the fadc amplitude is larger than the readout threshold in the integration window will be reported in data using 9 bits.

Data types

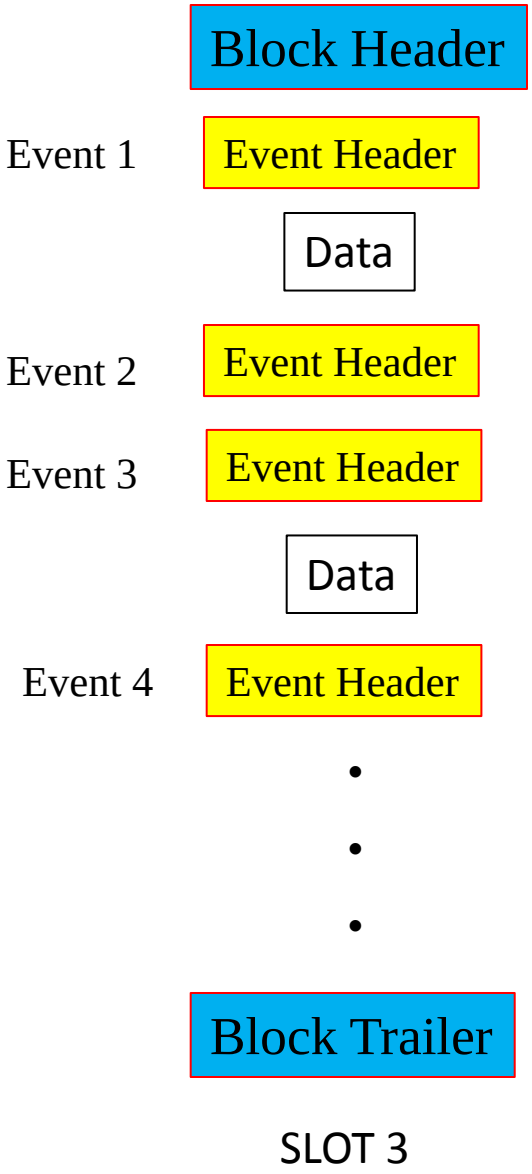
During spring and fall runs data were taken using two main modes: mode 7 and mode 8. Other modes such the ‘pulse raw data’ have never been used, though implementation of these modes requires FPGA resources. In the new firmware we propose to have only two main modes, which will be used for production runs and detector debugging:

- **Mode 9 (production) :** contains the pulse integral, time information (coarse and fine times), pulse peak amplitude, pedestal sum, and time over threshold. We propose the new data type (data type 9) which will be described below.
- **Mode 10 (debugging):** is the same as mode 9 but contains additional window raw data. Data will be reported using data types 9 and 4. The type 4 corresponds to the window raw data and is described in the fadc operating mode document provided by Ed and Hai.

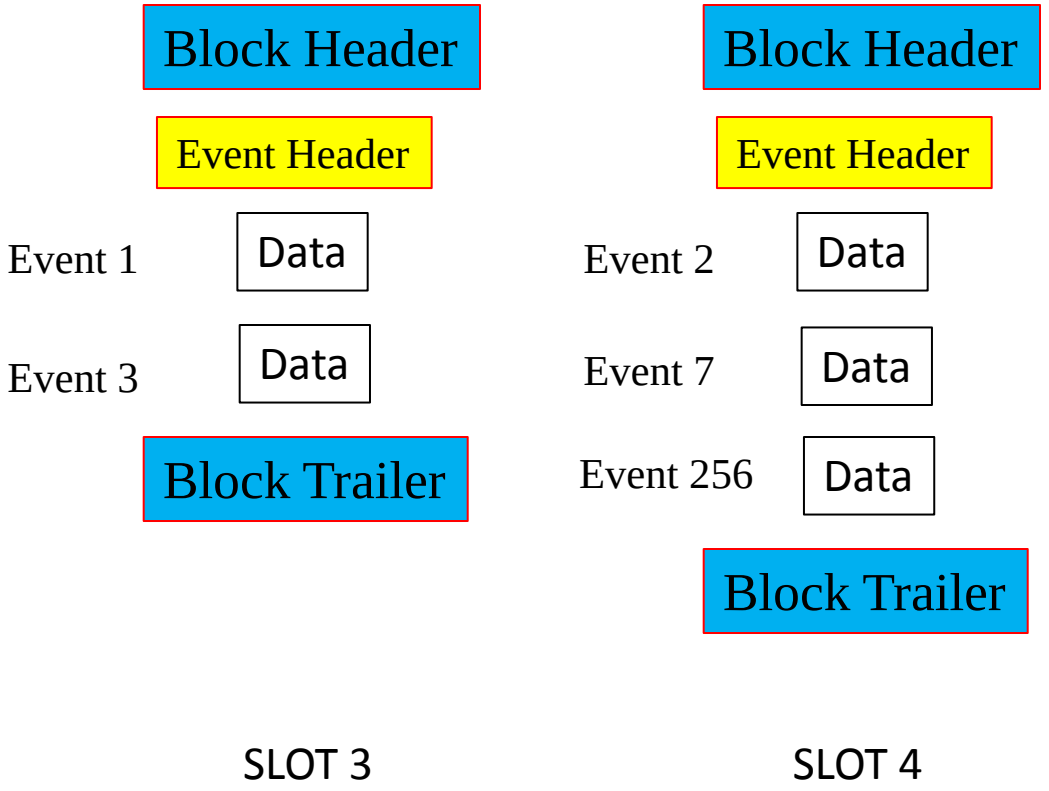
Data format

The major contribution to the size of data reported by a fadc comes from event headers reported for every event (trigger) in the block. We should be able to suppress reporting headers (using a programmable register), except for the first event in the block. The first header will contain several bits of the trigger time and trigger number, and will be used to check synchronization of data reported from fadc modules from different slots in a crate. This header can be subsequently removed after the data integrity check in the second readout list. The relative event number in a read out block will be coded in the data (hit header) of each hit. The proposed format for the data type 9 is presented below. We also propose to use data type 11 for the possible future format extension, i.e., type 11 should indicate that the format is defined using other format extension bits (will be details in the future).

Currently used (use as debug mode)



New (production mode)



Use event headers in the beginning of the block to check synchronization between fadcs in different slots

Switch between the production (event headers suppressed) and the debug mode (report all headers) using a programmable register

Event header

- Combine event number and trigger time in the event header **(new)**
- The data type can be reported in the event header, that will allow for the type/format extension in the future **(new optional)**

Current

(31) = 1
(30 - 27) = 2
(26 - 22) = Slot number
(21 - 0) = Event number (trigger number)

New

(31) = 1
(30 - 27) = 2
(26 - 22) = Slot number
(21 - 12) = Trigger time (10 bits)
(11 - 0) = Event number (12 bits)
(trigger number)

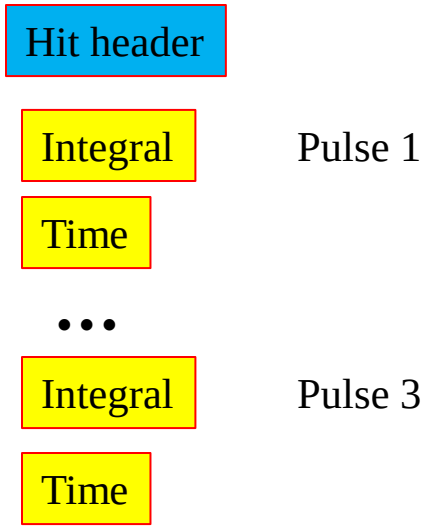
New optional

(31) = 1
(30 - 27) = 2
(26) = Format extension bit
0 - default data type
1 - extended data type
(use next 4 bits for the type extension)
(25 - 22) = Data type
(21 - 12) = Trigger time (10 bits)
(11 - 0) = Event number (12 bits)
(trigger number)

Format for Data Type 9

Reported once per hit

(31) = 1
(30 - 27) = data type 9 (4 bits)
(26 - 19) = event number (trigger number) (8 bits)
(18 - 15) = channel (4 bits)
(14) = pedestal quality (1 bit)
(13 - 0) = pedestal sum (14 bits)



Reported for N pulses in the window

(31) = 0
(30) = 1 - integral
(29 - 12) = integral (18 bits)
(11 - 9) = quality integral (3 bits)
bit 0 - underflow used,
bit 1 - overflow used,
bit 2 – energy sum overflow
(8 – 0) = time-over-threshold (9 bits)

(31) = 0
(30) = 0 - time
(29 - 21) = coarse time (9 bits)
(20 - 15) = fine time (6 bits)
(14 - 3) = peak (12 bits)
(2 - 0) = quality (3 bits)

Summary of requested changes

Description	Parameter	Range
Trigger N_{SB}	N_{SB}^{TRIG}	3 samples (fixed)
Readout N_{SB}	N_{SB}^{READ} N_{SB}^{SIGN}	3 bits [0 – 7] samples 1 bit 0 - start before thr sample 1 - start after thr sample
Trigger mask (disable channels for trigger and hit-bit scalers)	$TRIG^{MASK}$ (use channel disable mask)	16 bits 1 – channel disabled
Asynchronous pedestal sum readout	N_{ASYNC_PED} (number of samples to sum) $A^{MAX}_{ASYNC\_PED}$ (amplitude threshold)	4 bits [4 – 16] samples 10 bits [0 – 1024] adc counts
Pedestal sum readout	N_{PED} (number of samples to sum) $A^{MAX}_{ASYNC\_PED}$ (amplitude threshold, the same as for async readout)	4 bits [4 - 16] samples 10 bits [0 – 1024] adc counts
Threshold crossing (readout)	N_{SAMP}^{THR}	2 bits 0 - 1 sample 3 - 4 samples
FADC operating modes	***Mode 9 (production)*** ***Mode 10 (debug)***	Data type 9 (Integral + Time + Pedestal + time-over-threshold) Data type 9 + Raw samples
Data type	***Type 9*** ***Type 11 (format extension)***	See format definition