Sophisticated Online Analysis in ADC Boards

P. Wüstner, A. Erven, W. Erven, G. Kemmerling, H. Kleines, P. Kulessa, P. Marciniewski, H. Ohm, K. Pysz, V. Serdyuk, S. van Waasen and P. Wintz

***Abstract*—For the readout of the calorimeter of the WASA-at-COSY experiment a QDC board using flash ADCs and FPGAs to perform the pulse integration was developed. In the initial version only a simple pulse finding algorithm was implemented in order to avoid delay cables by storing the digitized signals in a pipeline of a few microseconds.**

Recently a new version with 16 ADCs at a sample rate of 240 MHz and 12 Bit resolution was developed for tests of cylindrical drift chambers (straw tubes) in the Straw Tube Tracker of the PANDA experiment. The goal was to measure the energy loss by charge readout in addition to drift time measurement. Due to the irregular cluster structure of the straw signals complex algorithms for pulse finding, pulse feature extraction and triggering were implemented.

The algorithm can detect pileup and find pulse groups. Integration is possible over single pulses and over complete groups. Different methods for the calculation of starting time, including constant fraction, are implemented. Internal trigger generation depends on various criteria, e.g. slew rate, amplitude or sum of consecutive samples.

I. Introduction

For the WASA-at-COSY experiment several years ago we developed a replacement for ”real QDCs” using flash ADCs and FPGAs to perform the pulse integration. In the initial version we only used a simple pulse finding algorithm which allowed us to refrain from using delay cables by storing the digitized signals in a pipeline for a few microseconds.

This combination of fast flash ADCs and large FPGAs is much more powerfull than just to simulate legacy QDCs.

In the last year we developed a new version especially for tests of cylindric drift chambers (straw tubes) for the PANDA Straw Tube Tracker. The goal was to measure the energy loss by charge readout in addition to drift time measurement. These boards use 16 flash ADCs with a sampling rate of 240 MHz and 12 bit resolution and five XILINX FPGAs (Spartan3). Four FPGAs perform the analysis, the fifth is used for the bus interface.

Once triggered, various analysis functions can be activated. The start time of the pulse is calculated using a fixed threshold, constant fraction method or the tangent of the slope. The accuracy of this time is in the order of 1/8 of the sampling interval, therefore eleminating the need for a TDC in less demanding applications. The algorithm can detect pileup and calculate start time, maximum and integral for each subpulse. It can also ignore pileups and calculate these numbers for the whole pulse group. It can calculate time over threshold for each subpulse.

The baseline of the input signal is permanently observed and used in each calculation.

A permanently running simple analysis of the ADC output can generate an internal trigger depending on various criteria: amplitude of the input signal, sum of a few consecutive samples, slew rate. These internal triggers can be derived from one single channel or a coincidence / majority logic for all or a subset of channels can be used. With the help of the crate controller this logic can include all modules of the crate.

In addition to all calculations the raw samples in a given intervall around the trigger or the start of the pulse can be written to the data stream. For permanent monitoring it is possible to do this only for the next detected pulse after a given number of triggers.

Independend of the accepted triggers all matched trigger conditions for single channels are counted. Thus the modules can be used as (of course frequency limited) scalers.

In addition the crate controller has an interesting feature for detector tests: it can ignore all triggers until a given high trigger rate is reached. In this case we can make high luminosity tests even if we have low luminosity most of the time: all uninteresting data is suppressed.

II. Baseline

Because of the capacitive coupling of the input signals a problem arises: after an input pulse (assume a high amplitude rectangle pulse) the input voltage of the ADCs is not zero but something below. It takes a long time until the input capacitor is discharged and the base line reaches a zero value again. Therefore all calculations for subsequent pulses (start time and integral) are not correct. It is also possible that subsequent pulses are not recognized because they do not fullfill the trigger conditions anymore.

With the known input characteristics (impedance and capacity) this baseline shift can be calculated. A digital simulation of an analog problem is never error free, therefore a (negative) feedback is needed to adjust the parameters of the correction code. The FPGA code permanently calculates this shift and corrects the ADC values. The subsequent code can then assume that all pulses (without pileup) start with an amplitude of zero.

III. Trivial Mode

The first mode implemeted just simulates legacy QDCs. It puts all raw ADC samples into a pipeline of 12 μs length (as all

Manuscript received June 11, 2012.

P. Wüstner, A. Erven, W. Erven, G. Kemmerling, H. Kleines and S. van Waasen are with the Zentralinstitut für Elektronik, Forschungszentrum Jülich, Germany.

H. Ohm, V. Serdyuk and P. Wintz are with the Istitut fuer Kernphysik, Forschungszentrum Jülich, Germany.

P. Kulessa and K. Pysz are with the Institute of Nuclear Physics PAN, Krakow, Poland.

P. Marciniewski is with the Department of Physics and Astronomy, Uppsala University, Sweden.

978-1-4673-1084-0/12/$31.00 ©2012 IEEE

other modes do). When a trigger arrives (via the backplane) the FPGA calculates the sum of all samples inside a given interval relative to the trigger. Becase of the pipeline this interval can start (and end) before the trigger time, threfore delay cables or other analog delay lines are not needed.

In the simplest case the output data consist of the sum for each channel, but the time and value of the maximum amplitude can be added.

Zero supression with individual thresholds for each channel is of course implemented.

IV. Pulse Finding Algorithm

In most cases the input pulses don't arrive at a constant time before or after the trigger. Legacy QDCs just integrate over a large fixed interval around the expected position of the pulse, but the real pulse is much shorter. Before and after the real pulse only noise is integrated.

Our pulse finding algorithm defines a fixed interval relative to the trigger. Inside this interval the algorithm is looking for the start of a pulse. It adds the values of some adjacent ADC samples. If the sum exceeds some given threshold the start of a pulse is assumed and the data can be analyzed further.

V. Pulse Analysis

If a pulse is found it is analysed with verious algorithms which can be selected and parametrized via setup registers. Most algorithms can be executed by the FPGA code in parallel without increasing the time needed.

Following calculations can be done:

- Integral: This is just the sum of a given number of samples after the start of the pulse. If a pileup is detected (see next section) the summing is stopped at the beginning of the pileup. The calculated sum is then too small, but can be corrected in some cases (known pulse shape) by the offline analysis later.
- Starttime: The start time of the pulse can be interpolated much better than the distance between adjacent samples. The resolution lies in the order of 1/8 of the sample distance. Two algorithms are available:
  - Tangent method: The tangent of the rising edge is calculated using two or more samples near the steepest part. The crossing of this tangent with the base line is used as start time.
  - Constant fraction: The behavior of a constant fraction discriminator is simulated. First the maximum of the amplitude is determined. Then the point in the rising edge corresponding to a fraction of the maximum is calculated. This point will normally be somewhere between two samples, therefore the corresponding time is interpolated.
- Time over threshold (TOT): In order to compare possible DAQ concepts for the straw detectors a TOT algorithm has been implemented: The number of samples above a threshold is counted.
- Overflows: The number of samples with ADC overflow is counted. This looks similar to the TOT with a maximum threshold, but because the baseline is not constant the threshold (difference between the fixed ADC maximum and the variable baseline) is also not constant. This is mainly a debugging feature.

VI. Pileup Detection

Inside a pulse pileup can be detected and handled. If we found a maximun and later a minimum but the amplitude has not yet vanished into the noise a pileup is assumed. Of course for the maximum and minimum detection a small number of samples is used instead of only one to avoid problems with noise.

When pileup is detected the analysys of the current pulse is finished and the next pulse is analysed. The minimum between both pulses is used as new baseline. Of course the calculated values of the integrals and the start time of the second pulse are not correct but can sometimes, if the shape of the pulses are known, corrected during offline analysis.

The response of the straw detectors to a single partical is not a single pulse but rather a bunch of overlapping pulses, called cluster. Therefore the pileup detection can be disabled and then a cluster is analysed like a single pulse.

VII. Trigger Capability

Each channel can generate an internal trigger

- if a pulse is detected
- if an amplitude threshold is reached
- if an amplitude threshold is reached for more than a given number of samples
- if the slew rate reaches a given value

With a table lookup the individual channel triggers can be combined to generate a trigger for the whole module. This trigger can be propagated to all other modules of the crate via the crate controller.

VIII. Scaler Capability

The individual channel triggers are counted, thus the module can work like a scaler. This scaler data can be inserted in the 'normal' data stream, mixed with the other data from the pulse analysis. In addition it is possible to read out these data at any time.

IX. Conclusion and Outlook

Starting with a simple simulation of a legacy QDC using flash ADCs and FPGAs we developed a module with much more capabilities. It can find the pulses anywhere in a large interval relative to the trigger, it handles pileups, it works as a (low resolution) TDC, it works as a scaler and it can generate its own triggers, replacing the typical combination of discriminators and coincidence logic.

All these capabilities are used to test straw detectors and to compare different methods for the online analysis.

We think that this module and FPGA code can be used as a good starting point for the development of the readout of the straw detectors of the PANDA experiment at FAIR.

References

[1] M. Drochner, W. Erven, P. Wüstner and K. Zwoll, *The Second Generation of DAQ-Systems at COSY*, IEEE RT97, Realtime 97 Conference records, Beaune 1997.

[2] A. Ackens, W. Erven, M. Drochner, H. W. Loevenich, D. Maeckelburg, M. Ramm, P. Wüstner and K. Zwoll, *System Development for COSY Experiments*, IEEE RT2003, Realtime 2003 Conference records, Montreal 2003.

[3] P. Wüstner et al., System development for COSY experiments, 13th IEEE Realtime conference 2003, Montreal.

[4] P. Wüstner, A. Ackens, U. Clemens, M. Drochner, W. Erven, P. Kämmerling, G. Kemmerling, H. Kleines,
W. Loevenich, and K. Zwoll, *New Synchronisation System for Experiments at COSY*, IEEE RT2005, Realtime 2005 Conference records, Stockholm 2005.