Feature Extraction Algorithms for Short Preamplifier Transients

Holger Flemming

aGSI Helmholtzzentrum für Schwerionenforschung GmbH, Planckstraße 1, 64291 Darmstadt, Germany

Abstract

In this work the extraction of pulse time and amplitude from short pulse transients is analysed as they are recorded by a transient recorder ASIC called ATR16. Algorithms of different complexity are tested with simulated preamplifier data processed with a VHDL model of the ATR16 ASIC. It is shown that a time precision below 1 ns and a signal amplitude precision below 1 MeV energy equivalent is feasible.

Keywords: Simulation, Feature Extraction, Algorithms

1. Introduction

While the baseline approach for the PANDA EMC readout [1] foresees a transmission of the analogue signals produced by the preamplifier and shaper ASIC called APFEL [2] over long cables to outside sampling ADCs a new concept is based on digitiser ASICs placed close to the preamplifier. This ASIC called ATR16 is an analogue transient recorder [3].

Figure 1: Block diagram of the transient recorder unit of the ATR16.

Figure 1 shows a block diagram of the transient recorder unit of the ATR16. Each recorder block is connected to a four channel front end. Main component of the recorder unit is an analogue memory. The incoming analogue signal is sampled with up to 100 MS/s and written into the analogue memory in a cyclic manner. When a pulse is detected the write logic switches to the next row after a configurable delay. This way the complete pulse transient is stored in the memory and can be read out asynchronously by the read out circuit consisting of charge integrators and an analogue multiplexer. Afterwards the transient is digitised with a 33 MS/s, 12 bit pipeline ADC which is shared by the four channels.

In the past several groups studied the feature extraction of detector signals digitised with sampling ADCs in continuous operation [4–6]. Here elaborated digital filter and extraction algorithms can be utilised which is not possible with short finite traces as provided by the ATR16. So this work looks into the feature extraction of such short traces and the resulting contributions to energy and time precision which are required to be less than 1 MeV respectively 3 ns [1].

In section 2 the optimum sampling frequency and trance length is derived from the preamplifier signal characteristics. Then the examined algorithms for time and signal extraction are described in section 3. In section 4 the simulation environment is described and the results are presented with a final outcome concluded in section 6.

2. Optimisation of Sampling Parameter

Due to the architecture of the ATR16 which foresees an analogue transient recording each storage cell has to be realised by a capacitor which implementation is area consuming. So to minimise the number of samples in a single trace the sampling frequency should be close to the physical limit. From Shannon’s first theorem [7] it is known that the signal is completely defined when the sampling frequency is the doubled maximum frequency of the signal.

Following Sansen and Chang [8] the spectrum of the output signal of a charge sensitive amplifier is given by

$$H(s) = \frac{q}{sC_f} \left[ \frac{sT_0}{1 + sT_0} \right]^n \left[ \frac{A}{1 + sT_0} \right]^n$$

(1)

In figure 2 the corresponding energy spectral densities for a shaping time of 280 ns and shaper order of 1,2,3 and 1.75 are shown. The rational number 1.75 results from fits to measured pulse data of the APFEL ASIC.
As the signals have a continuous spectrum up to infinite frequencies one has to define an arbitrary limit given by the fraction of the spectral power below a certain frequency. So the spectral energy density was integrated and the relative missing energy is drawn in the lower plot of figure 2. One can find for the APFEL ASIC that 99.9 % of the spectral energy is in the frequency range below 3.62 MHz which leads to a minimum sampling frequency of approximately 7.2 MHz. For this work a sampling frequency of 8 MHz was chosen.

As the full pulse length is in the order of 1 µs 8 samples are required to cover the full pulse. To be able to extract the baseline from the transient a number of 16 samples was chosen for transient recording. Accordingly the analogue memory shown in figure 1 is organised in 16 columns.

3. Feature Extraction Algorithms

For this analysis several algorithms for extraction of pulse time and pulse amplitude are evaluated. A common requirement for all of these algorithms was that they might be implemented in hardware i.e. on the ATR16 ASIC itself or in an FPGA.

Figure 3 shows a typical trace. The recorded traces consists of 16 samples \(a_i\) with \(0 \leq i \leq 15\) which are shown in the upper plot of figure 3 as red marks. The solid red line is a fit of the analytical function of the signal. In the lower plot the differentiated signal \(d_i^1 = a_{i+1} - a_i\) with \(0 \leq i \leq 14\) is shown as blue marks and the twice differentiated signal \(d_i^2 = d_{i+1}^1 - d_i^1\) with \(0 \leq i \leq 13\) is shown as red marks.

For time extraction as well as amplitude extraction in common the baseline has to be known. Therefor a threshold \(D_{th}\) is used to define the beginning of the pulse. So the index of the first sample of the pulse \(i_0\) is given by the first difference \(d_i\) that fulfills \(d_{i_0}^1 > D_{th}\). All samples before this point are taken as baseline. So the baseline level \(\bar{b}\) is given by

\[
\bar{b} = \frac{1}{i_0} \sum_{i=0}^{i_0-1} a_i
\]

The baseline is shown as green line in the upper plot of figure 3. In the following the time and amplitude extraction algorithms are described in detail.

3.1. Time Extraction

The aim of the time extraction is to determine the pulse time with a better granularity than given by the sampling frequency. The fraction of a sampling intervall is called finetime. It is a measure for the pulse phase in relation to the sampling frequency.

3.1.1. Baseline Crossing

The idea of the baseline crossing algorithm is placing a tangent on the leading edge of the pulse and calculate the crossing point with the baseline. Therefore \(d_i^1\) with \(i_i > i_0\) and \(d_i^1 > d_{i+1}^1\) is searched.

\(a(t) = d_i^1 t + a_i\) describes the tangent at \(i_i\) shown as yellow line in figure 3. For \(t_0\) at the baseline crossing \(\bar{b} = a(t_0)\) one gets

\[
i_0 = \frac{\bar{b} - a_i}{d_i^1}
\]

3.1.2. Pulse Maximum

At the pulse maximum the first derivative has to be zero. Therefor the time of the pulse maximum can be determined by calculating the zero crossing point between \(d_i^1\) with \(d_{i_0}^1 \geq 0\) and \(d_{i_0+1}^1\) with \(d_{i_0+1}^1 < 0\). The connecting line between \(d_{i_0}^1\) and \(d_{i_0+1}^1\) (shown as blue line in figure 3) is given by \(d_i^1(t) = a t + d_{i_0}^1\). For \(d_i^1(t) = 0\) one gets

\[
i_0 = \frac{d_{i_0}^1}{d_{i_0}^1 - d_{i_0+1}^1}
\]
3.1.3. Inflexion Point

The same way the time of the pulse maximum can be calculated by the zero crossing of \(d_1\) the time of the inflexion point on the leading edge can be determined by the zero crossing of \(d_i\). When \(d_i^2\) with \(d_i^2 \geq 0\) and \(d_{i+1}^2\) with \(d_{i+1}^2 < 0\) one gets

\[ t_0 = \frac{d_i^2}{d_i^2 - d_{i+1}^2} \]  

(5)

This interpolation is shown as red line in the lower plot of figure 3.

3.2. Amplitude Extraction

As well as for the time extraction for the amplitude extraction a series of different algorithms has been tested. Beginning with a simple maximum value extraction, integral calculation of the whole pulse as well as for defined windows to a parabolic interpolation of the pulse maximum and a linear regression of the recorded transient and a standard pulse shape. Below the different algorithms are discussed in detail.

3.2.1. Pulse Maximum

This is the simplest possible algorithm. The highes sample \(a_m\) of the transient is determined. The signal amplitude is the difference between this sample and the baseline.

\[ S_{pm} = a_m - \bar{b} \]  

(6)

3.2.2. Pulse Integral

As explained above a threshold is used to define the beginning of pulse and to separate baseline from pulse. This algorithm accumulates the difference of all pulse samples and the baseline to calculate a pulse integral.

\[ S_{int} = \sum_{i=0}^{15} (a_i - \bar{b}) \]  

(7)

3.2.3. Window Integral

In difference to the previous algorithm the window integral algorithm only accumulates a defined number of samples around the pulse maximum. Therefor the position of the pulse maximum \(m\) is determined. Afterwards the window is defined by the number of samples before the maximum \(n_b\) and the number of samples after the maximum \(n_a\). Then the signal amplitude is given by

\[ S_{win} = \sum_{i=m-n_b}^{m+n_a} (a_i - \bar{b}) \]  

(8)

In this work two window definitions are analysed. The first one is \(n_b = 1\) and \(n_a = 2\) and the second one is \(n_b = 2\) and \(n_a = 4\). One should take note that the pulse maximum algorithm described in section 3.2.1 could be considered as a special case of the window integral algorithm with \(n_b = n_a = 0\).

3.2.4. Linear regression of Transient and Standard Pulse Shape

The last algorithm uses the fact that the pulse shape of the pulses generated by the APFEL ASIC is well known. In [8] the pulse shape of a charge sensitive amplifier with pulse shaper is deduced as

\[ V_{out}(t) = \frac{qA^n}{C_f n!} \left( \frac{t - t_0}{\tau_s} \right)^n e^{-n(T-t_0)/\tau_e} \]  

(9)

We get \(t_0\) from the time extraction and calculate the samples of a normalised pulse

\[ V_i = \frac{i(T - t_0)}{\tau_s} e^{-n(T-t_0)/\tau_e} \]  

(10)

with the sampling interval \(T\). The measured sample \(a_i\) and \(V_i\) are connected by

\[ a_i = S_{LR}V_i + b + \delta_i \]  

(11)

with the signal amplitude \(S_{LR}\), the baseline \(b\) and an error \(\delta_i\) which is caused by noise for example. To calculate \(S_{LR}\) and \(b\) from the samples \(a_i\) a linear regression is used.

4. Simulation

The described time and signal amplitude extraction algorithms were tested by simulations. For that a VHDL simulation of the ATR16 ASIC including modelling of the analogue components was used. The numerical stimulus is based on noise and signal analysis of the APFEL ASIC [6].

4.1. Stimulus Generation

In sec. 3.2.4 the analytical expression of an APFEL pulse was introduced in eq. 9 already. Using Stirlings approximation \(A^n e^n / C_f n!\) can be expressed as \(G_{PAS A} \cdot e^n\) with the preamplifier gain \(G_{PAS A}\). The input charge \(q\) is given by

\[ q = G_{PAS A} \cdot Y \cdot A_{eff} \cdot Q_{eff} \cdot e \]  

(12)

with the APD gain \(G_{APD}\), the light yield of the lead tungstate crystals \(Y\), the effective area coverage of the APDs \(A_{eff}\), the quantum efficiency of the APDs \(Q_{eff}\) and the elementary charge \(e\). The values of these parameters used in the simulation are given in table 1. The pulse shape characteristics \(n\) and \(\tau_e\) are fixed by a fit of measured APFEL pulses.

Corresponding to the PANDA EMC readout schema with two APDs for each scintillating crystal and two outputs of the APFEL per APD with gain factors of 1 and 16 the numerical stimulus consisting of APFEL pulses is split into two tribes and each tribe is split into two channels with gain factors corresponding to the APFEL ASIC. On each channel an individual noise signal is added with a noise spectrum that corresponds to measurements done with the APFEL ASIC. Figure 4 shows the noise spectra of the measured and the simulated APFEL noise.
The simulated output data were written into a file and analysed for calibration a stimulus with cyclic pulses with constant amplitude was used. The start times of the pulses were shifted in steps of 1/256 sampling periods to get an equally distributed finetime.

The numerical waveforms generated this way were used as stimulus for a full chip VHDL simulation of the ATR16 ASIC. The simulated output data were written into a file and analysed within the root frame work.

The first observation when analysing the data was that a comparison of the extracted times and amplitudes with fit parameters of fitted pulse functions shows a finetime dependency that has to be corrected.

### 4.2. Finetime Correction

Figure 5 shows the observed finetime characteristics of the pulse maximum extraction algorithm as an example for the time extraction algorithms. On the x axis the extracted finetime in steps of 1/64 sampling intervals is plotted and on the y axis the difference between the extracted finetime and finetime determined with a pulse fit. Obviously the difference depends on the phase of the pulse relative to the sampling frequency. A similar behavior is observable for the other algorithms.

For the second method the difference between the extracted finetime and the finetime determined with the pulse fit is calculated and filled in histograms for each finetime bin. The correction value for a finetime bin is given by the mean value of the corresponding histogram.

\[ L_j = \frac{1}{M} \sum_{i=0}^{M-1} \left( \frac{n_i}{N} - \frac{1}{M} \right) \]  

(13)

with the number of events \( n_i \) in the \( j \)th finetime bin and the total number of finetime bins \( M \).

#### 4.2.1. Non-Linearity Correction by Distribution Analysis

The first method is a statistical method frequently used to correct non linearities of TDCs which was described by Pelka et. al. in [10]. With a large number \( N \) of calibration events the correction vector is mainly given by the integral nonlinearity error of the \( j \)th time bin

\[ L_j = \frac{1}{N} \sum_{i=0}^{N-1} \left( \frac{n_i}{N} - \frac{1}{M} \right) \]

For calibration a stimulus with cyclic pulses with constant amplitude was used. The start times of the pulses were shifted in steps of 1/256 sampling intervals to get an equally distributed finetime.

The simulated output data were written into a file and analysed within the root frame work.

The first observation when analysing the data was that a comparison of the extracted times and amplitudes with fit parameters of fitted pulse functions shows a finetime dependency that has to be corrected.

### 4.2. Finetime Correction

Figure 5 shows the observed finetime characteristics of the pulse maximum extraction algorithm as an example for the time extraction algorithms. On the x axis the extracted finetime in steps of 1/64 sampling intervals is plotted and on the y axis the difference between the extracted finetime and finetime determined with a pulse fit. Obviously the difference depends on the phase of the pulse relative to the sampling frequency. A similar behavior is observable for the other algorithms.
4.3. Amplitude Correction

Just as the extracted finetime the extracted amplitude shows a non linear finetime dependency. This is shown for the window integral algorithm in figure 7. To correct this dependency a vector of correction factors is used. Again two methods are used to generate these correction vectors.

4.3.1. Correction by Scaling to Maximum Value

In a first step the finetime with the maximum extracted amplitude is searched. In the second step for each finetime the ratio between the extracted amplitude for this finetime and the maximum extracted amplitude is calculated as correction factor for this finetime.

4.3.2. Correction by Fit

For each finetime the ratio between the extracted amplitude and the amplitude of the fitted pulse is calculated and stored as correction factor.

Figure 8 shows the amplitude spectrum of extracted amplitudes with the window integral algorithm without and with correction.

5. Analysis and Results

The test data contain cyclic pulses with an interval of $T_{\text{cyc}} = 4001.953$ ns and a geometric sequence of 15 pulse heights corresponding to energies from 10 MeV to 4800 MeV. Cyclic pulses have been chosen to enable an analysis in absence of pile up. An analysis of pile-up-effects will follow in sec. 5.3

5.1. Time Extraction

Figure 9 shows the time difference spectrum for pulses with an energy of 15.5 MeV in high gain channels extracted with the zero crossing of first derivative algorithm. A clear peak around zero is visible in the data which is well described by a Gauss function.

As the histogramed data are the difference of two channels $\sqrt{\frac{\sigma}{2}}$ of the fitted gauss function is used as a measure of the
time precision of a single channel. The time precision obtained this way is plotted for all extraction algorithms and pulse energies in low and high gain in figure 10. The time precision gets better with increasing pulse energy. As listed in tab. 2, a time precision in the order of a nano second is obtained for the highest energy.

Figure 10: Time precision in dependence of the pulse energy.

Table 2: Obtained time precisions.

| Extraction Algorithm      | High Gain 219 MeV | Low Gain 4800 MeV | Unit |
|---------------------------|-------------------|-------------------|------|
| Baseline crossing         | 2.69 ns           | 1.29 ns           |      |
| Time @ Maximum            | 1.32 ns           | 0.84 ns           |      |
| Time @ Inflection Point   | 2.52 ns           | 1.90 ns           |      |

5.2. Amplitude Extraction

Figure 11: Pulse height spectrum.

Figure 11 shows the pulse height spectrum of pulses analysed with the window integral method. Equivalent spectra have been generated for all extraction algorithms. One can resolve 11 peaks in the low gain mode and 8 peaks in the high gain mode. These peaks are fitted with Gauss functions to get the mean value and the width of the peaks. From the correlation of the corresponding energy of each peak which is well known from the stimulus generation and the mean value of the Gauss functions a calibration factor can be evolved. So the energy precision is determined as product of this calibration factor and the sigma value of the Gauss function. The obtained energy precision is plotted in figure 12.

Figure 12: Energy precision in dependency of pulse energy.

The energy precision by electronic noise and feature extraction shows no significant dependency from the pulse energy but the extraction performance of different algorithms varies by a factor of ≈ 2.5.

5.3. Effects of Pile up

The results of the previous sections are obtained by simulations with a cyclic stimulus that avoids pile-up events completely. To get a more realistic picture that includes the effects of pile-up the simulation and analysis was repeated with Poisson distributed stimuli with a geometric sequence of 16 event rates in the range from 1 kHz to 1 MHz. The energy spectrum was extended with additional discrete energies down to 500 keV and the incidences of the discrete energies are 1/E distributed.

To get sufficient statistics for highest energies at lowest rates in a reasonable time it was abstained from performing a full chip VHDL simulation but a chip model was developed in C++ rather.

The extraction algorithm checks the traces for a second increasing on the pulse tail to detect pile-up events. Those events are rejected and the following analysis is based on events with no or no obvious pile-up.

Figure 14 shows the same spectrum for Poisson distributed pulses as shown in figure 11 for cyclic pulses. The most obvious difference is that the spectra for Poisson distributed pulses becomes asymmetric with high energy tails. Comparing the spectra for 25.1 kHz rate and 251.2 kHz one gets the impression that...
the asymmetry increases with increasing rate. A similar behavior was also described for high rate detector measurements with a sampling ADC readout in [6].

To quantify the effect of the high energy tails in the energy spectrum the mean value as well as the standard deviation of the extracted energy of all events in defined ranges is calculated. The boundaries of these ranges are set to the geometric mean between two peaks. Figure 13 shows the standard deviation and the relative shift of the mean value calculated for the energy range around 219 MeV in the high gain mode.

The standard deviation increases for event rates above 100 kHz. Up to 500 kHz a precision of 1 MeV can be obtained by the window integral (1,2), maximum value and parabolic approximation algorithms. The relative shift of the mean value for these algorithms is below ±1%. The irregular behavior of the standard deviation of the linear regression extraction algorithm is not yet understood, but as only a minor benefit in precision could be obtained with this algorithm while it requires a large numerical effort it is not considered for practical use anyway.

The time precision is not affected by pile-up as in figure 15 no rate dependency is observable.

6. Summary and Conclusion

For this work simulated pulse transients of 16 samples have been analysed with four algorithms for time extraction and five algorithms for amplitude extraction. The algorithms and corrections have been shown. For amplitude extraction the most elaborate algorithm (linear regression, see sec. 3.2.4) provides the best energy precision as expected but the precision of the very simple maximum value algorithm (sec. 3.2.1) is slightly worse only. Surprisingly the energy precision does not benefit from integration across several sampling bins. A reason for this might be that the noise is dominated by low frequency components while integration suppresses high frequency noise only.

The time precision gets better with increasing pulse amplitude. Even though the sampling period of the transients is 125 ns time precisions in the order of 1 ns can be obtained.
In addition the rate dependency of realistic Poisson distributed events was analysed. With increasing event rate pile-up events lead to a high energy tail and a slightly degrading energy precision that still fulfills the requirements while the time precision does not depend on the event rate.

The results finally show that feature extraction of short event transients is feasible with a sufficient energy and time precision. Based on this analysis the development of the ATR16 was continued and an on-chip feature extraction unit was implemented realising the algorithms investigated in this work.

Acknowledgments

The author would like to thank Oliver Noll for many fruitful discussions concerning generation of the simulation stimulus and feature extraction algorithms.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

[1] The PANDA Collaboration, Technical Design Report for PANDA Electromagnetic Calorimeter (EMC), Tech. rep., Facility for Antiproton and Ion Research (October 2008) [cited 06/15/2020].
URL https://arxiv.org/abs/0810.1216

[2] H. Flemming, P. Wieczorek, Low noise preamplifier ASIC for the PANDA experiment. Journal of Instrumentation 6 (12) (2011) C12055–C12055. doi:10.1088/1748-0221/6/12/c12055
URL https://iopscience.iop.org/article/10.1088/1748-0221/6/12/C12055/meta

[3] H. Flemming, H. Deppe, P. Wieczorek, A family of transient recorder ASICs for detector readout, accepted by JINST (10/2021).

[4] E. Guliyev, M. Kavatsyuk, P. Lemmens, G. Tambave, H. Löhner, VHDL implementation of feature-extraction algorithm for the PANDA electromagnetic calorimeter. Nuclear Instruments and Methods in Physics Research A 664 (2012) 22–28.
URL https://www.sciencedirect.com/science/article/pii/S0168900211019267

[5] M. Preston, P. Marciniewski, P.-E. Tegner, Fpga-based algorithms for feature extraction in the panda shashlyk calorimeter, International Conference on Instrumentation For Colliding Beam Physics (2020) 1–10.

[6] E. J. O. Noll, Digital Signal Processing for the Measurement of Particle Properties with the PANDA Electromagnetic Calorimeter, Ph.D. thesis, Johannes Gutenberg-Universität Mainz (2020). doi:http://doi.org/10.25358/openscience-5078

[7] C. E. Shannon, Communication in the Presence of Noise (1949) 10–21.
URL https://ieeexplore.ieee.org/abstract/document/1697831

[8] W. M. Sansen, Z. Y. Chang, Limits of Low-Noise Performace of Detector Readout Front Ends in CMOS Technology, IEEE Transactions on Circuits and Systems 37 (1990) 1375–1382.

[9] Datasheet of apd type s11048(x2).
URL https://jazz.physik.unibas.ch/panda/uploads/2012-11-13--12-43-06_Specification_Sheet_S11048_X2_200101.pdf

[10] R. Polka, J. Kalisz, R. Szplet, Nonlinearity Correction of the Integrated Time-to-Digital Converter with Direct Coding. IEEE Transactions on Instrumentation and Measurement 46 (1997) 449–453.
URL https://ieeexplore.ieee.org/abstract/document/571882