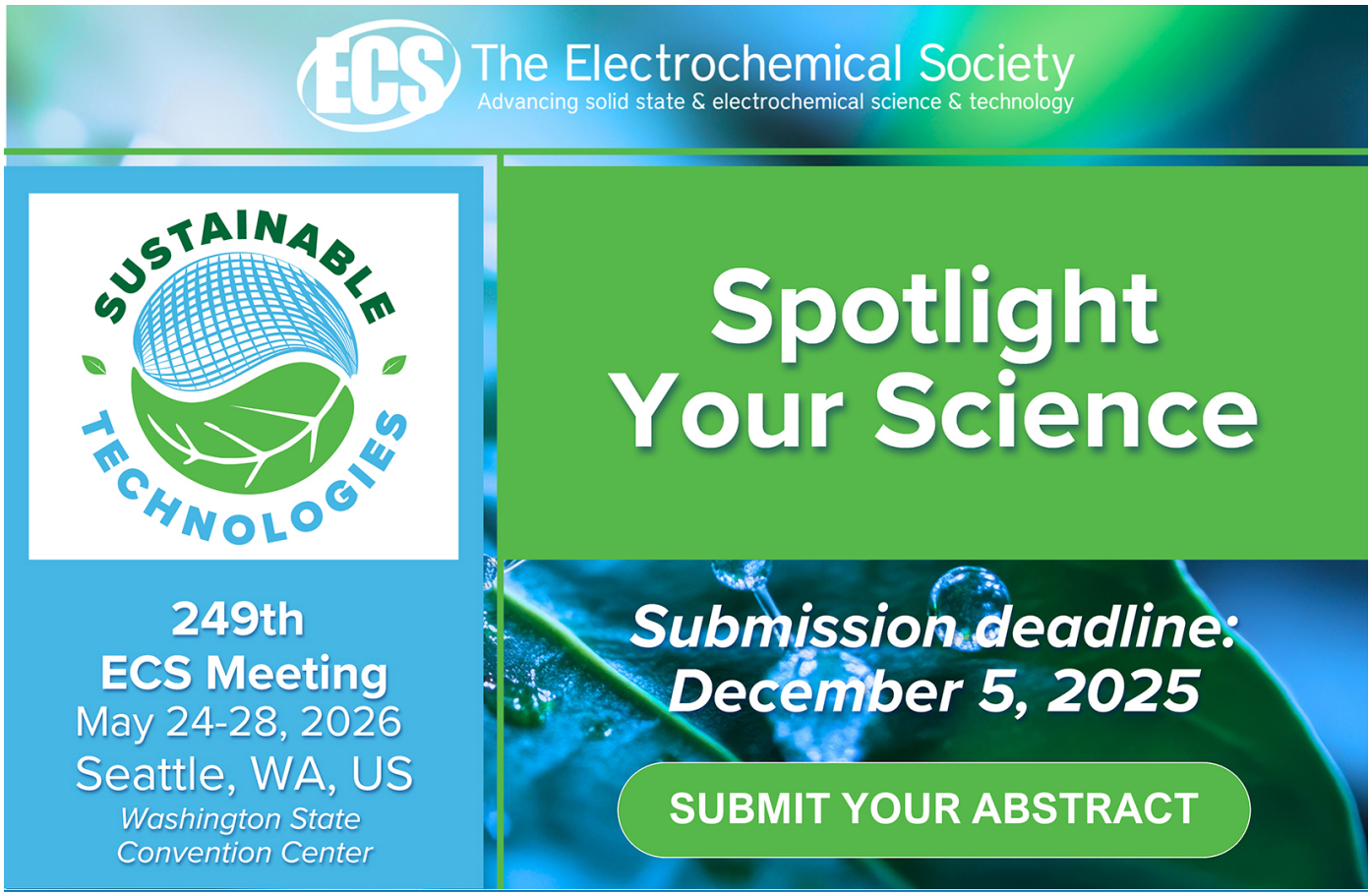
Journal of
Instrumentation

PAPER

An FPGA-based parallel deconvolution application envisaging high-energy calorimeters operating under severe pile-up conditions

To cite this article: Igo A.S. Luz *et al* 2024 *JINST* **19** P08022



Published by IOP Publishing for Sissa Medialab

Received: *April 16, 2023*
Revised: *June 22, 2024*
Accepted: *August 7, 2024*
Published: *August 28, 2024*

An FPGA-based parallel deconvolution application envisaging high-energy calorimeters operating under severe pile-up conditions

Igo A.S. Luz,[a,b,*] **Eduardo F. Simas Filho**,[b] **Paulo C.M.A. Farias**,[b] **Luciano M. de Andrade Filho**[c] **and J.M. Seixas**[d]

[a] *Federal Institute of Bahia, Rua Viriato Lobo, Santo Antônio de Jesus, Bahia, Brazil*
[b] *Digital Systems Laboratory, Electrical Engineering Program, Federal University of Bahia, Rua Prof. Aristides Novis, Salvador, Bahia, Brazil*
[c] *Department of Electrical Engineering, Federal University of Juiz de Fora, Rua José Lourenço Kelmer, Juiz de Fora, Minas Gerais, Brazil*
[d] *Coppe/Poli, University of Rio de Janeiro, Av. Horácio Macedo, Rio de Janeiro, Brazil*

E-mail: igo.luz@ifba.edu.br

Abstract: Signal pileup may arise when the system response time is larger than what would be required by the event rate in the application. In this condition, information superposition occurs, and signal estimation is deteriorated due to such information distortion. For online applications, mitigating such a pileup distortion usually requires embedded digital signal processing, which often involves FPGA-based designs when high event rates are a concern. Recently, high-energy particle experiments started facing an unprecedented increase in pileup conditions, which demanded new approaches for signal estimation. This work proposes a dedicated parallel signal processing system embedded on an FPGA platform that implements different filtering techniques for online energy reconstruction in calorimeters (energy measurement) operating under severe pileup conditions. Each designed filter is an inverse approximation of the calorimeter readout channel's impulse response, performing deconvolution of the incoming signals. Such a deconvolution approach for energy estimation was recently introduced, and the proposed filters were implemented in this work. Evaluation of the digital system design was performed using a data set that considers a typical high-energy calorimeter front-end response, as high-luminosity particle collider experiments produce high pileup distortions in calorimeter signals when processing the subproducts of the collisions. As the Large Hadron Collider (LHC) provides the most demanding conditions in terms of signal pileup, the acquisition system of the proposed solution is synchronous with its collision rate (40 MHz). The results showed that the proposed implementation may be applied as a preprocessing (pileup reduction) step in calorimeter instrumentation and could reduce the signal pileup within the fast response time required by such experiments.

Keywords: Data processing methods; Digital signal processing (DSP); VLSI circuits; Calorimeters

*Corresponding author.

https://doi.org/10.1088/1748-0221/19/08/P08022

Contents

2024 JINST 19 P08022

1 Introduction

Instrumentation systems may be affected by different types of noise and distortion, which may require a signal processing solution. When these systems operate at high speed, with a large volume of data, signal processing algorithms are usually implemented in dedicated circuits to meet stringent constraints related to the processing time. One of the types of distortion is pileup, which represents the superposition of information from subsequent events. For example, signals originating from multiple X-ray photons simultaneously incident on a photon-counting detector produce a pulse pile-up effect, thereby distorting the measured signature [1]. A similar effect is observed in nuclear spectroscopy measurement systems [2]. Such superposition occurs when the system response time expands beyond the interval between two subsequent events, and the system occupancy is high due to the event rate. The pileup arises in different fields of application, degrading the signal of interest, thus compromising the information quality. The high-energy particle experiments [3, 4] are among the applications for which signal pileup is an issue.

Considering the case study of this work, the high-energy physics experiments produce a large amount of information for proper characterization of the phenomena of interest (high-luminosity scenario) [5]. The European Organization for Nuclear Research (CERN) developed the Large Hadron Collider (LHC) [6, 7], which is currently being upgraded to increase luminosity and energy collisions. This machine upgrade will result in higher rate of detected events. This will cause an increase in the readout channel occupancy, which may lead to signal pileup in the calorimeter systems [4, 8–10], as the channel response time is often larger than the interval between subsequent events [7]. Thus, online systems face stringent operation conditions and should provide efficient algorithms for mitigating pileup distortion.

Online trigger systems are used in particle detectors to provide a fast decision whether an event shall be recorded (as it is a candidate for interesting physics) or rejected (as it was detected as belonging

to the background noise of the experiment) [11]. For this, calorimeters produce fast responses and are very important for online trigger overall performance, as the energy deposition profiles provide valuable information for particle characterization. Unfortunately, pile-up distorts measured energy deposition profiles, and results in reducing the particle detection efficiency [9].

The solution to reconstruct energy signals affected by the pileup in online systems, in general, needs to be implemented in a dedicated circuit, due to the fast response times required from the instrumentation system. Different works implement algorithms for signal reconstruction in FPGA (Field Programmable Gate Arrays). In [12–14], different FPGA-based solutions are proposed to reconstruct signals affected by pileup.

The pileup problem may be described using a convolutional model. The measured signal (affected by pileup) may result from convolution between the original information and the readout channel response. In this sense, deconvolution (or inverse filtering) methods could be used for pileup reduction [9, 15–17]. Deconvolution [3, 18] has been successfully applied to reduce measurement distortions in different signal processing applications. In [9], it was applied to remove the superposition of energy signatures from high-energy particle detectors. Other works propose different techniques to reconstruct a calorimeter signal affected by pileup:[19] proposes a deconvolution method based on sparse representation (SR), and [3] proposes two approaches, one based on an FIR Filter, and other based on an iterative technique.

This work proposes a digital circuit architecture based on FPGA that implements different techniques, formulated in [3], for online pileup reduction capable of operating in calorimetry instrumentation systems. The proposed circuit supports data processing from 12 sensors, and the output delivers the estimated energy signal of each bunch-crossing (BC) generated by the collider, as long as the analog-to-digital converter sampling clock is synchronous with the collider BC (40 MHz). The proposed hardware was described using techniques optimized for circuits that implement online systems and shall be used in state-of-the-art instrumentation systems affected by pileup. Among the main contributions of this work, we can mention: (i) FPGA implementation of two deconvolution algorithms — the first one based on a Finite Impulse Response (FIR) filter [20], which is designed to approximate the inverse function of the readout channel, and the second one based on the Van Cittert's iterative technique [21], which uses the Positive Gradient Descent (PGD) algorithm [3] to estimate the signal; as far as we know, the latter has not yet been implemented in hardware; (ii) proposition of parallel optimized and reconfigurable digital structures to accomplish the operational requirements of modern particle colliders and (iii) analysis of the proposed circuits considering aspects such as pileup reduction efficiency, processing time, device occupancy, and power consumption.

The paper is organized as follows. In section 2, the methods usually applied for free-running energy estimation in particle detectors are presented. In section 3, the proposed architecture of the digital system is detailed. The achieved results from the simulation and logical synthesis process are presented in section 4, whereas conclusions are derived in section 5.

2 Reducing the pileup through deconvolution

High-energy calorimetry signals provide a fast response to incoming particles, and the energy deposited by an incoming particle in a given calorimeter cell is usually estimated through the amplitude of the corresponding readout signal pulse. Therefore, signal pileup [22] has enormous consequences on estimating the energy. Figure 1 shows an example of piled-up signal in a scenario where the average

2024 JINST 19 P08022

occupation of the channel is 10%, which means that, in average, only one peak will appear in the measurement channel at each ten collisions. The remaining samples are expected to have only noise. The blue and black samples represent a typical $\sim$ 150 ns width unipolar pulse-shaped signal, while the orange represents the measured signal, the resulting pileup signal.

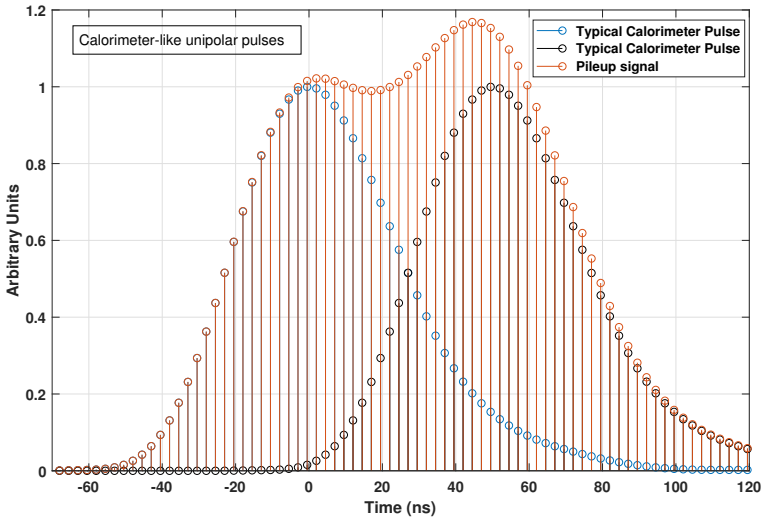


Figure 1. Representation of the pileup effect between two subsequent calorimeter-like unipolar pulses. The blue signal overlaps the black signal, resulting in the orange (distorted) signal.

The pileup can be modeled by the convolution of the original information $\mathbf{x}[n]$ with the readout channel response $\mathbf{h}[n]$, generating $\mathbf{y}[n]$, which represents the measured (distorted) information (figure 2). Equation (2.1) shows the exact mathematical formula:

$$\mathbf{y}[n] = \mathbf{x}[n] * \mathbf{h}[n] = \sum_{K=0}^{L-1} \mathbf{h}[k]\mathbf{x}[n\text{-}k] \tag{2.1}$$

With sufficient signal-to-noise ratio, the original information may be estimated by performing a convolution between the measured signal and the inverse model of the readout channel response. Other methods have also been developed. In this paper, we present techniques introduced in [3, 18], which are quickly summarized in sections 2.1 and 2.2.

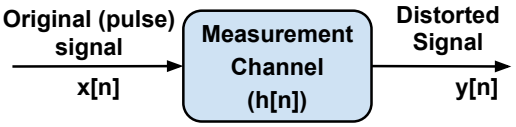


Figure 2. Signal propagation model through convolution with the measured channel considered in this work.

2.1 Digital FIR filter

Digital filters are typically used to modify signal attributes in time or frequency domains [23]. Digital filter architectures are often linear time-invariant (LTI) and, thus, interact with their input signals through a linear convolution process (equation (2.1)). The FIR filter impulse response becomes

2024 JINST 19 P08022

zero after a certain number of samples [20, 24]. Also, FIR filters may be designed to present linear phase response, which prevents phase distortions in the output signal. Equation (2.1) [20] defines the filter response in the time domain.

Different FIR filter structures result in different implementation forms: direct form, transposed structure, cascade, and others [23, 25]. The transposed structure, which was used in this work, is based on the direct form. It presents some advantages, such as no need for an extra shift register for the $\mathbf{x}[n]$ samples, nor an extra pipeline stage for the adders to obtain high throughput [23].

2.2 Positive gradient descent algorithm

The PGD algorithm is based on Van Cittert's convergence technique [26] and is considered an iterative method. The FIR output is calculated continuously as the samples arrive, based on a sum of a fixed amount of samples multiplied by filter coefficients, counting backwards from the current sample. In contrast, iterative methods operate on a set of samples and output is calculated by solving a linear system [27].

The PGD is based on linear convolution in its matrix form [27]. Equation (2.2) shows the definition of this algorithm [21, 27], highlighting the iterative mechanism.

$$\hat{\mathbf{x}}^{i+1} = \hat{\mathbf{x}}^{i} + \mu \mathbf{H}^{T}(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}^{i}) \tag{2.2}$$

where i is the iteration index, μ is a real relaxation factor, which is necessary to guarantee convergence, $\mathbf{H}$ is a matrix containing an estimation of the measurement channel response, $\mathbf{x}^{i}$ is a vector resulting from the last calculated iteration, $\mathbf{x}^{i+1}$ represents the vector resulting from the next iteration, and $\mathbf{y}$ represents the vector of samples. For example, considering the case in which the readout channel is represented by $\mathbf{h}[n] = [h_0, h_1, h_2]$, the matrix $\mathbf{H}$ is defined as:

$$\mathbf{H} = \begin{bmatrix} h_0 & 0 & 0 \\ h_1 & h_0 & 0 \\ h_2 & h_1 & h_0 \\ 0 & h_2 & h_1 \\ 0 & 0 & h_2 \end{bmatrix} \tag{2.3}$$

The PGD is a variation of the Van Cittert technique, which restricts signal processing to samples with positive amplitudes. This method is applied when the signal that will be recovered needs to be always positive. Its operating logic consists of the cancellation (zero value is assigned) of samples below a certain threshold λ at each iteration. This algorithm is illustrated in figure 3.

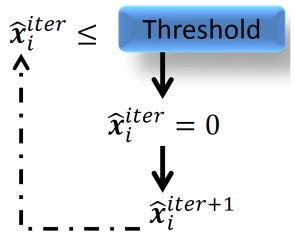


Figure 3. Ilustrative diagram of the positive threshold operation in the PGD algorithm.

In preliminary studies [3], two iterative algorithms, the Gradient Descent Excluding Zeros and the PGD were evaluated. The difference between them is that in the first one, the matrix resulting from

2024 JINST 19 P08022

each iteration is reduced whenever there are non-positive coefficients, while in the second one, the matrix remains with the same dimension, but its negative components are zeroed. Due to hardware resource considerations, only the PGD approach is considered for further discussion.

3 Proposed digital architecture

This work proposes developing an FPGA-based hardware architecture. The application considered by this study is a generic high-energy calorimeter system that operates at 40 MHz, taking the LHC experiment as a high-luminosity operating environment reference. The proposed circuit simultaneously receives information from multiple sensors via serial links, converts these data to a parallel form, performs filtering, and finally converts the filtered signal back to the serial form, which is then forwarded to the output. To perform tests and evaluate system performance, two FPGA devices were considered: a general-purpose device, Cyclone EP4CE115 [28] through the Altera DE2-115 [28] development board, and Xilinx XC7VX485T [29], which is a device targeting high-performance and advanced digital signal processing applications. The design targets an application in which signals are generated continuously at a high rate. Therefore, the signal processing system to be implemented should provide fast response time, and at the same time satisfy both area and power consumption constraints. Figure 4 illustrates the proposed architecture.

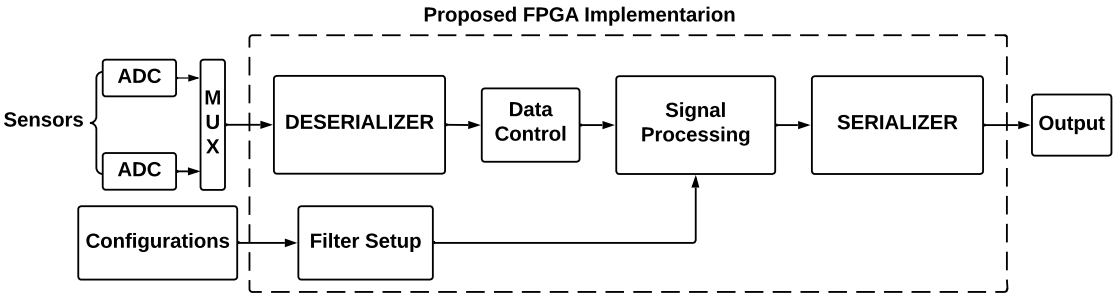


Figure 4. Block diagram of the proposed system architecture showing the internal processing steps of the proposed FPGA implementation together with the external connections to input signals, configuration information, and output signals.

The data is received by and transmitted from the FPGA via serial links. It is converted to parallel form for internal processing via the "DESERIALIZER" block and converted back to the serial form by the "SERIALIZER" block. The "Filter Setup" block contains registers that properly store the information to initialize deconvolution methods, such as the coefficient values used in the filtering process. The "Data Control" block ensures the correct parallel data handling and processing, identifying both the time stamp and the input channel. The "Signal Processing" block implements the Digital Filter in both approaches, fixed-coefficients (FIR) or iterative (PGD).

To meet the processing time requirement imposed by the 40 MHz collision rate, the proposed architecture uses pipelining techniques. Figure 5 shows the implemented pipelining structure with three stages. The system's operation is divided into several independently running stages, which prevents some parts of the system from idling while waiting for other operations to complete. As illustrated in figure 5, the proposed stages are: "Converter Serial-Parallel", "Filter Data", and "Converter

2024 JINST 19 P08022

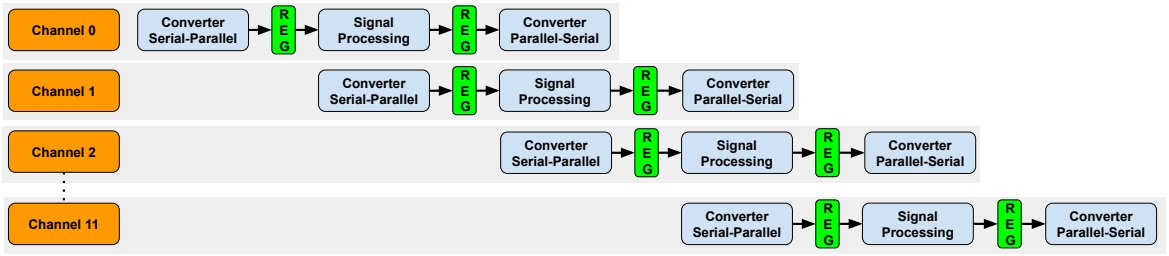


Figure 5. Diagram of the proposed parallelism strategy illustrating how information from different measurement channels are processed in parallel.

Parallel-Serial". The blocks "Channel 0" to "Channel 11" represent input signals that are to be processed. Considering this logic, when one stage finishes processing, the result is stored in a register to be available for the next stage, and it can process data from another channel. For example, when the first stage finishes converting data from Channel 0 and stores it in a register, it becomes available to process data from Channel 1, while the Filter Data stage processes the parallel data from Channel 0. Thus, all stages may run at the same time.

3.1 Digital FIR filter implementation

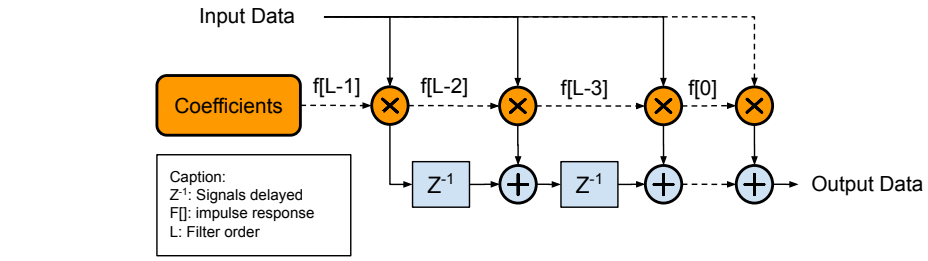
The digital FIR filter architecture was designed to optimize the utilization of structures that perform arithmetic operations (addition and multiplication). In general, the equation that defines the FIR filter in the transposed form is presented as a sum of products. The purpose was to use fewer instances of the DSP structures available in the FPGA devices, thus reducing the logic resource utilization.

The system must process signals from up to 12 sensors. Consequently, it was beneficial to design a signal architecture that would allow sharing one FIR structure among all the channels, thereby minimizing resource utilization. Figure 6(a) shows the diagram of the FIR filter implementation for one channel, and figure 6(b) shows the circuit structure for processing all 12 channels. In (b), the adders and multipliers are shared for processing all channels; for this reason, there are different registers banks for storing the channels' delayed signals.

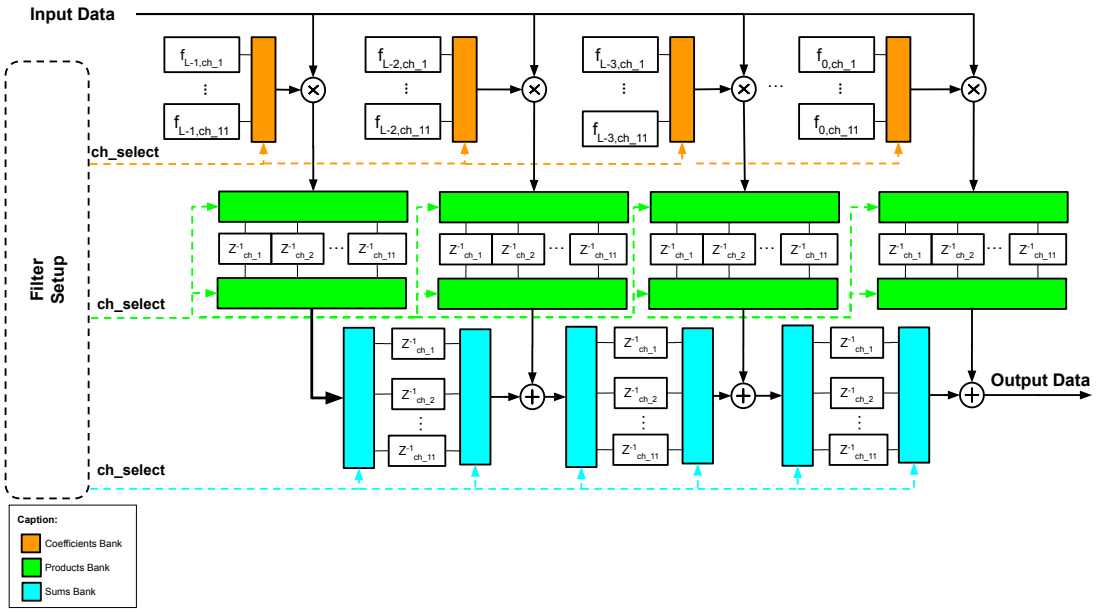
The adders and multipliers were shared to operate relative to all channels. To ensure this, three register banks were used to store the values processed from each channel. One to store the sum results (Sum Register Bank), another for the multiplication result (Product Register Bank), and a third for the coefficients (COEFF Register Bank). The logic of the "ch select" ensures that when the Input Data refers to a certain channel, the operations are performed using the values stored in the registers corresponding to it, which ensures the correct behavior of the filter algorithm, because the delayed samples of each channel remain stored for later accesses. The proposed FIR filter implementation supports designs with a maximum of 30 coefficients. The filter order and coefficient values may be configured externally from a specific input.

When a new sample arrives to be processed, the "Filter Setup" logic identifies which channel generated the sample, and the "ch select" signal ensures that the values from the corresponding channel are recovered. The input data is multiplied by the coefficients. The products are stored in the registers of the respective channel (represented by the green blocks). Then, after all multiplications are completed, the cascaded sum operations are performed. For these operations, the cyan blocks' registers represent the channel's delayed samples.

2024 JINST 19 P08022



(a)



(b)

Figure 6. Data flow diagram of the proposed FIR filter: (a) illustrating the circuit structure for one channel and (b) the circuit for processing the 12 measurement channels sharing the multiplier and adder structures. The Filter Setup block provides the information on which channel shall be processed at each iteration.

3.2 PGD implementation

The general expression that defines this method is presented in equation (2.2). Direct transcription of this expression in hardware is not feasible because the synthesis process would generate too extensive data path, resulting in a long processing time. Thus, the information processing was segmented to meet the processing time and occupancy requirements. For this, eq. (2.2) was split into five steps, which are executed at different times. Equations (3.1) and (3.2) are executed in parallel, the others sequentially.

$$\mathbf{M_1} = \mu \mathbf{H}^T \tag{3.1}$$

$$\mathbf{m_2} = \mathbf{H}\mathbf{x}^i \tag{3.2}$$

$$\mathbf{m_3} = \mathbf{y} - \mathbf{m_2} \tag{3.3}$$

$$\mathbf{m_4} = \mathbf{M_1} \times \mathbf{m_3} \tag{3.4}$$

$$\mathbf{x}^{i+1} = \mathbf{x}^i + \mathbf{m_4} \tag{3.5}$$

2024 JINST 19 P08022

Data flow control was controlled by a Finite-State Machine (FSM), as shown in figure 7. Each state enables one of the computational steps for the PGD algorithm. With this strategy, it was possible to reduce the data flow path, allowing the circuit to support a higher clock frequency.

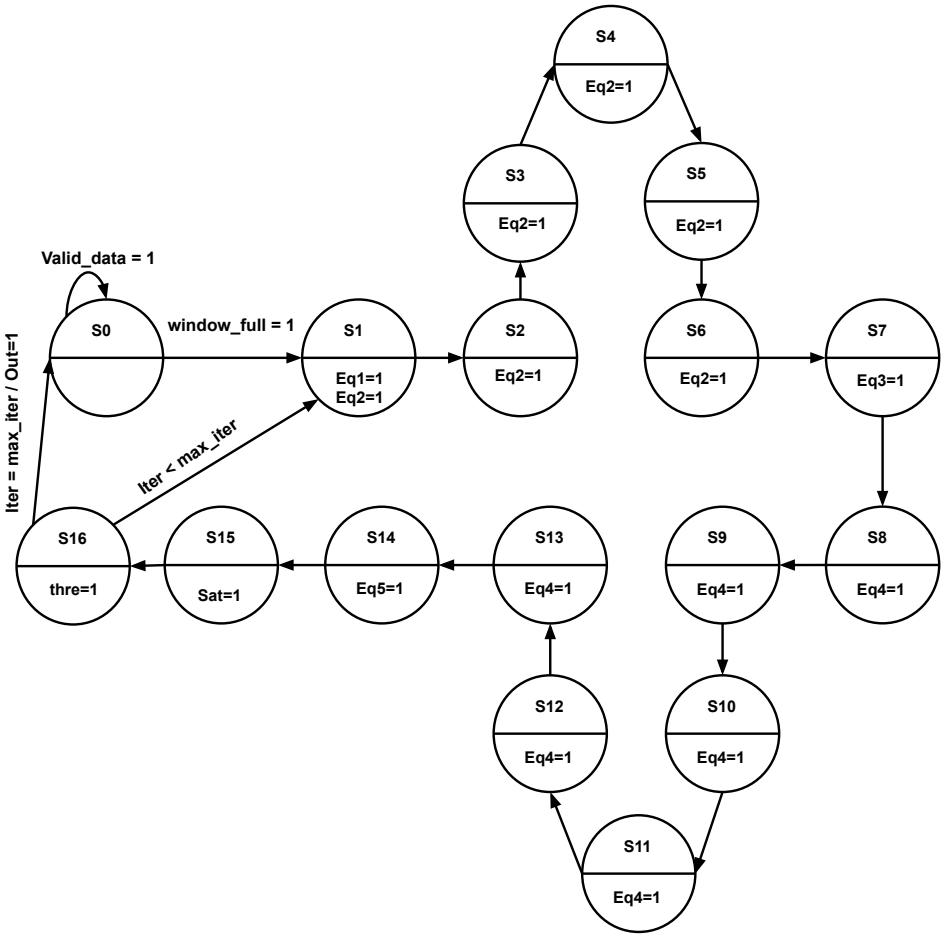


Figure 7. Diagram of the proposed finite-state machine for PGD implementation showing the detailed computational steps required to implement in hardware eqs. (3.1) to (3.5). The Eq1 to Eq5 in the drawing correspond to equations (3.1) to (3.5).

The state machine controls the execution of a fixed number of iterations, defined in "max_iter" and evaluated in state S16. In figure 7, S0 generates the control signals to ensure proper data storage for a given window of samples. S1 represents the beginning of the algorithm, and equations (3.1) and (3.2) are executed at this state, with the latter equation representing a multiplication between a matrix and a vector. The arithmetic operations for calculating the multiplication involving matrix are composed of cascading multiple sums, so the propagation delay caused by combinatorial logic would require reduction of clock frequency. Therefore, to avoid decreasing execution speed, it was decided to perform them in six different states, using six clock pulses. The first state performs the multiplication operations, while additions are spread among states S2 to S6, given the number of rows of the **H** matrix.

State S7 controls the execution of equation (3.3). States S8 through S13 control computation of equation (3.4), using the same strategy as for equation (3.2), to obtain better processing time. State S14 controls the execution of equation (3.5), resulting in the output vector for a given iteration.

2024 JINST 19 P08022

State S15 controls the application of the saturation technique, where the elements of the resulting vector are analyzed and saturated to be represented using the number of available output bits. This technique is necessary because the number of bits of the operands in the multiplication operations is greater than the number of bits in the output. Finally, state S16 controls the application of the threshold λ, changing the array elements that have values below λ to zero. In state S16, the number of iterations that were executed with a certain window is verified, and when the maximum number of iterations is reached, the processing of that window ends and the signal that controls the output of the result of the processed window is enabled "Out=1".

4 Results

The proposed architecture was designed and validated using the selected FPGA devices. Functional verification was performed using a MATLAB® script that implemented both deconvolution techniques. For simulation, a model that considers the operation of a high-energy calorimetry system was developed, generating the signal for each event. This simulator generates the signals affected by pileup and the true energy information. It also considers the phase deviation at the moment of signal acquisition, the maximum amplitude, and channel occupancy. The phase deviation is related to the time-shift of the simulated instrumentation pulse-shape at every signal acquisition (sampling). This effect is implemented to simulate that particles may have different time-of-flights as they propagate from the collision point towards the calorimeter sensor cell. The simulator considers this deviation to be uniformly distributed in the range of ±5 ns. The maximum amplitude parameter is the maximum value that the simulator can assign to each generated signal. Occupancy represents the percentage of bunch crossing with pileup noise.

The method of generating simulation data for a given occupancy level was as follows. First, a random set of BCs is selected from a set of two million consecutive bunches. Next, for the selected BCs, the "true" amplitude values are randomly generated from an exponential distribution, simulating minimum bias events[1] with mean parameter set to 600 MeV. Then, the measured signal is based on the superposition sequence of true signals and system's impulse response. Finally, bandwidth-limited Gaussian noise is added to emulate the effects of electronic noise.

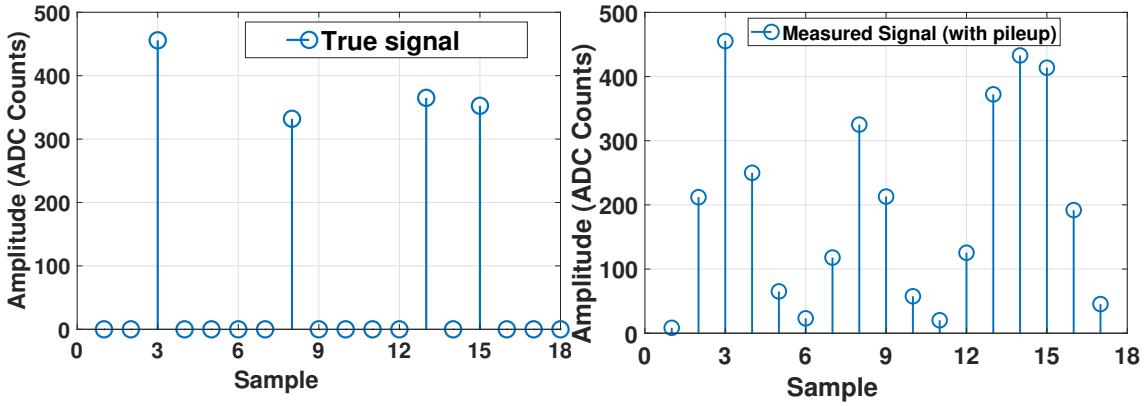
To generate the results of this work, the following case was considered: sensors providing unipolar pulses, pulse length of 7 samples, 30% occupancy level. Figure 8(a) shows the simulated true signal corresponding to energies deposited in the calorimeter, assuming a 25 nanoseconds period. Figure 8(b) shows the effect of the simulated pileup.

For simulations, a verification environment was used. It was responsible for simulating the hardware design and monitoring its output. In this way, it was ensured that the signals were sent to the hardware considering the previously defined protocol and that the outputs were monitored considering the expected behavior of the output protocol.

In the case of the FIR method, simulations were executed by varying the filter order and using the same set of simulated signals. The energy signals were represented using 12 bits and filter coefficients with 18 bits. Such digital accuracies reflect preliminary analyses aimed at reducing information loss. The simulated filter orders were 5, 10, 15, and 20.

[1]Minimum bias (MB) is a generic term that refers to selected events with a "very loose" trigger (with as little bias as possible) that accepts a significant fraction of the overall inelastic cross-section of a particle collider [30].

2024 JINST 19 P08022



(a) (b)

Figure 8. Graphical representation of (a) simulated true signals corresponding to energy deposited in the calorimeter and (b) signals affected by pileup resulting from system's response. The sampling clock is considered to be synchronous with the particle accelerator bunch-crossings.

To evaluate the estimation performance from the FPGA implementation of the FIR filter approach, the Root Mean Square Error (RMSE) was employed:

$$\text{rmse} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(a_t - a_s)_i^2} \tag{4.1}$$

where a_t is the amplitude of the simulated true signal, a_s amplitude of the estimated signal and N represents the number of estimated energy signals generated by the simulation.

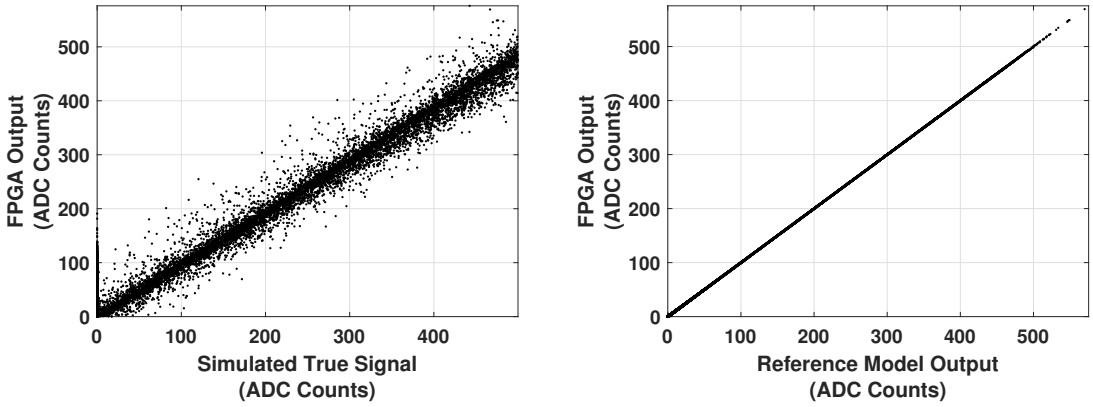
Table 1 shows a comparison (root mean square error estimation) between the output of the FPGA FIR filter and that of the offline reference model implemented using the corresponding FIR function in MATLAB. One can observe that the hardware design performs well and closely follows the reference model. The differences between offline and FPGA implementations appear mainly due to finite word-length effects. In the FPGA implementation, 12-bit word-length precision was used, and in this case, the RMSE deviations from the offline reference model are smaller than 0.5 ADC count.

Table 1. Root Mean Square Error results (in ADC Counts) of the FPGA FIR filter implementation (fixed-coefficients method) by varying the filter order. The result off all collum was calculated with the eq. 4.1. The Circuit vs True Value column represents the calculated error between the FPGA output and the simulated true values. The Reference Model vs True Value represent the error between the offline implementation (MATLAB) and the simulated true values. The last column represents the error between the FPGA output and the reference model implemented in MATLAB.

Filter Order	Circuit vs True Value	Reference Model vs True Value	Circuit vs Reference Model
5th order	24.95	24.95	0.37
10th order	19.86	19.84	0.45
15th order	19.33	19.32	0.46
20th order	19.21	19.20	0.47

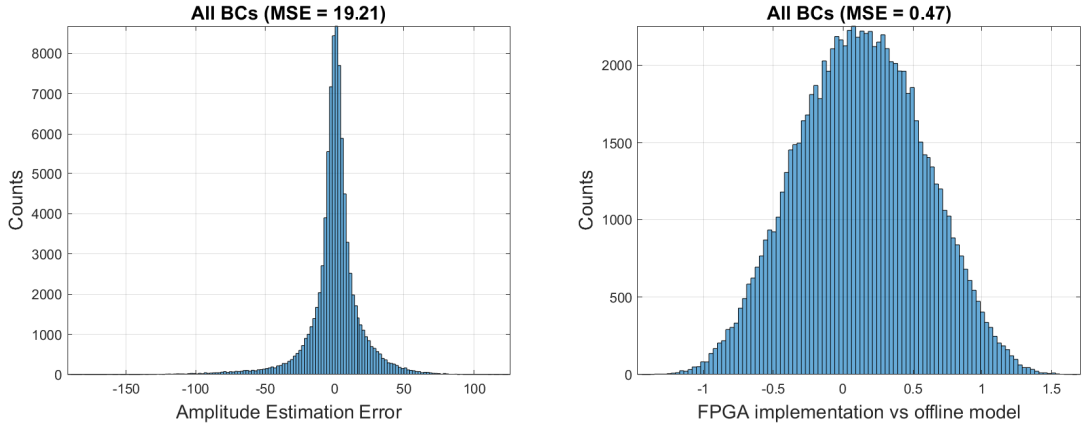
2024 JINST 19 P08022

Figure 9 and figure 10 show scatter plots and histograms for the proposed system's response when using the 20th-order FIR filter, compared to either the simulated ideal signals or to the response of the offline reference model implemented in Matlab. Through these analyses, one can confirm that the FPGA-based system closely follows the offline reference model.



(a) (b)

Figure 9. Scatter Plots for the energy estimated using the FPGA FIR filter implementation (20th order) compared to (a) the true signals and (b) the offline FIR filter implementation.



(a) (b)

Figure 10. Histograms of the amplitude estimation errors using the FPGA FIR filter implementation (20th order) compared to (a) the simulated true signals and (b) the offline FIR filter implementation.

Different from the FIR design, the iterative method requires window-based signal processing. According to previous studies [3], when the calorimeter operates under high pileup conditions, a window length of 25 and above stabilizes the energy estimation error when PGD is applied. Due to the deconvolution operation and sensor characteristic pulse being described by 7 samples, when an input window with 25 samples is considered, after successive iterations, the FPGA output will generate 25 - 6 = 19 time samples. Furthermore, such a windowed iterative technique also requires that the number of iterations is set. Thus, to perform this analysis, the window length was fixed,

2024 JINST 19 P08022

and simulations were performed, changing the number of iterations up to 100 iterations, increasing it in step of 5. The error stabilized between 45 and 50 iterations.

Table 2 shows the results in terms of the RMSE, and figure 11 shows how the RMSE varies when the number of iterations is increased.

Table 2. Root Mean Square Error results (in ADC Counts) of the iterative method by varying the number of iterations. The result off all collum was calculated with the eq. 4.1. Circuit vs True Value column represents the calculated error between the FPGA output and the simulated true values. The Reference Model vs True Value represent the error between the offline implementation (MATLAB) and the simulated true values. The last column represents the calculated error between the FPGA output and the reference model implemented in MATLAB.

Iterations	Circuit vs True Value	Reference Model vs True Value	Circuit vs Reference Model
10	9.52	9.55	0.10
20	9.06	9.08	0.10
30	8.88	8.89	0.08
40	8.82	8.83	0.07
50	8.79	8.80	0.06
60	8.78	8.79	0.06
70	8.78	8.79	0.06
80	8.78	8.79	0.06
90	8.79	8.80	0.06
100	8.79	8.80	0.06

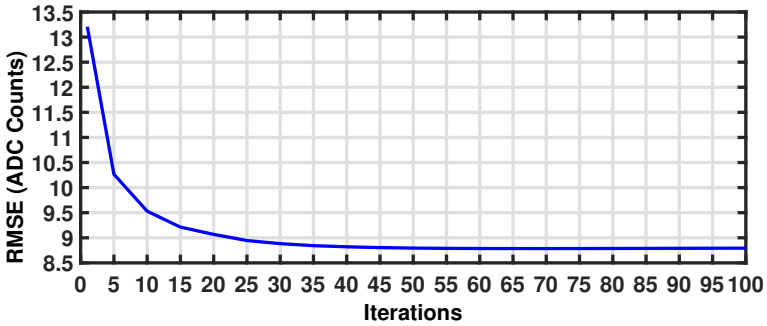


Figure 11. Root Mean Square Error for the FPGA PGD method computed varying the number of iterations.

In the iterative method, the estimation error might increase when samples of a given event are in two subsequent windows, known as the edge effect. To mitigate this issue, samples from each estimated signal window's edges (upper and lower) were discarded. To ensure that all samples will be reconstructed, the step in which the window slides through the input signals must consider the number of discarded samples. The discarding procedure is illustrated in the figure 12. In the scenario where the window size is 25 samples, according to the analysis presented in figure 13, the step was of three samples.

The impact of discarding samples was analyzed using an input window with 25 samples processed for 60 iterations. Figure 13 shows how the estimation error varies with the amount of discarded samples at each window edge. The error shows to stabilize from three discarded samples on each edge.

2024 JINST 19 P08022

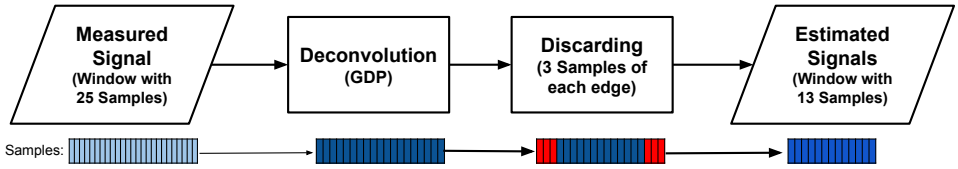


Figure 12. Block diagram illustrating the process of discarding samples from the PGD output window. The PGD processes a window of 25 samples and generates an output with 19 samples, of which six are discarded from each (upper and lower) edge, generating a result with 13 samples.

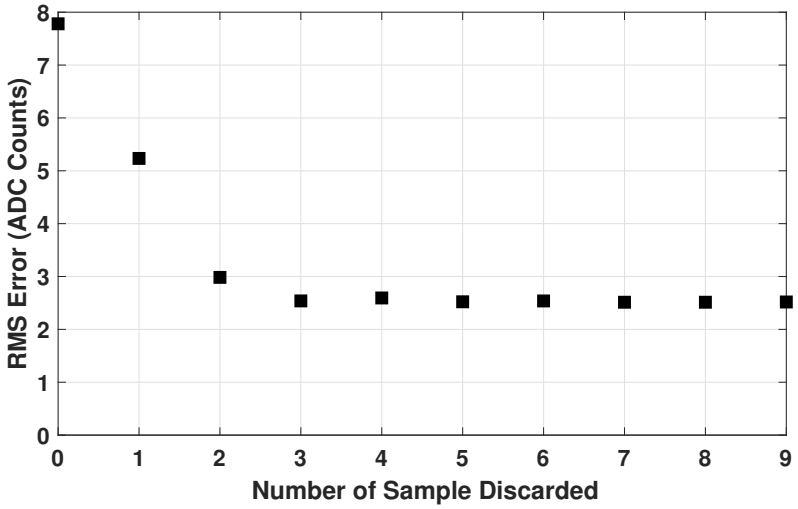


Figure 13. RMSE of the PDG method FPGA implementation versus the number of discarded samples on each window edge.

Figure 14 shows the scatter plots for the iterative implementation. A total of 100 iterations were considered. As was shown for the FIR implementation, the iterative approach performed similarly to its reference model.

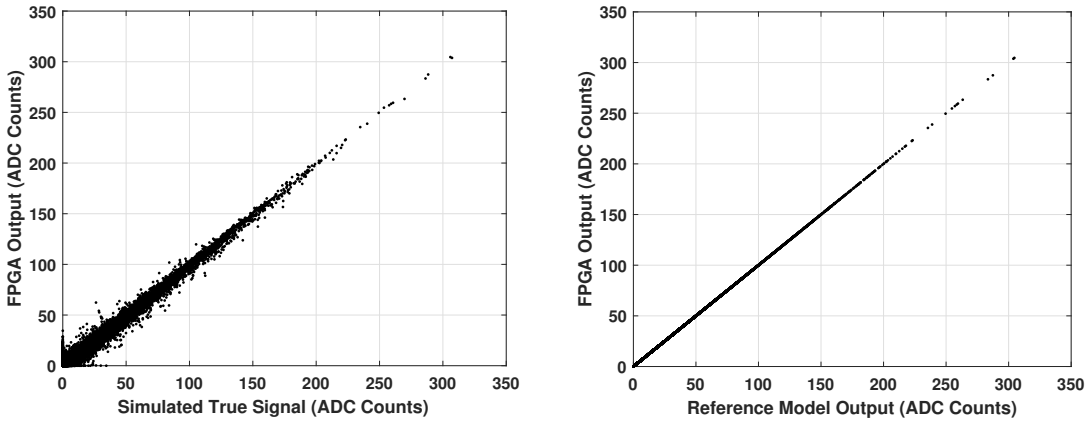
4.1 Synthesis results

In addition to the simulation phase, the FPGA synthesis was developed considering both signal estimation approaches. For this, the use of logical resources was evaluated. Considering the FIR filter approach, figures 15 and 16 illustrates the use of slice registers and DSP structures according to the filter order.

For the Xilinx device, the critical signal path for processing speed is determined by the DSP structures used for multiplication operations. In the first version, this path was synthesized with a delay of 3.351 ns. However, following the instruction of the manufacturer [31], those structures could be optimized using three pipeline stages because this allowed running the DSP block at full speed. Figure 17 shows the datapath for multiplication of the DSP48 device, where the pipeline registers are shown as A1, B1, M, and P. After this optimization, the minimum clock period for the proposed architecture was 1.879 ns (achieving 44% reduction).

The timing analysis provided by the Intel® Quartus® Prime Timing Analyzer tool for the DE2-115 board showed that the worst case for the data path was the multiplication of the filter coefficients with

2024 JINST 19 P08022



(a) (b)

Figure 14. Scatter Plots for the amplitude estimation using the FPGA PDG implementation compared to (a) the simulated true signals and (b) the offline PDG implementation.

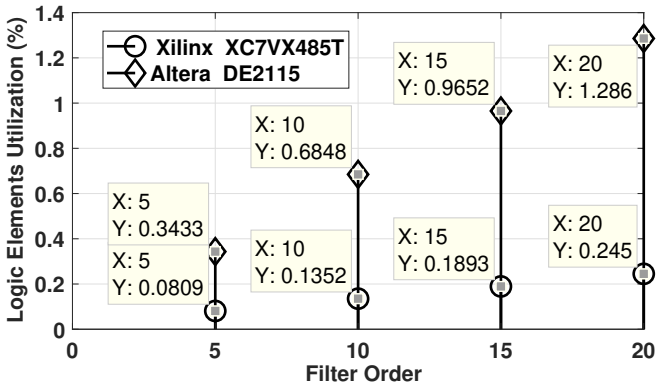


Figure 15. Slice Registers utilization of the proposed FPGA implementation as the FIR filter order increases considering two different FPGA evaluation boards (XC7VX485T and DE2115).

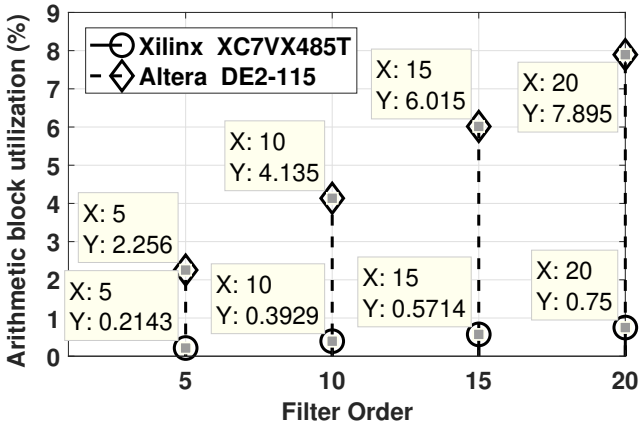


Figure 16. DSP structures utilization of the FPGA implementation as the FIR filter order increases considering two different FPGA evaluation boards (XC7VX485T and DE2115).

2024 JINST 19 P08022

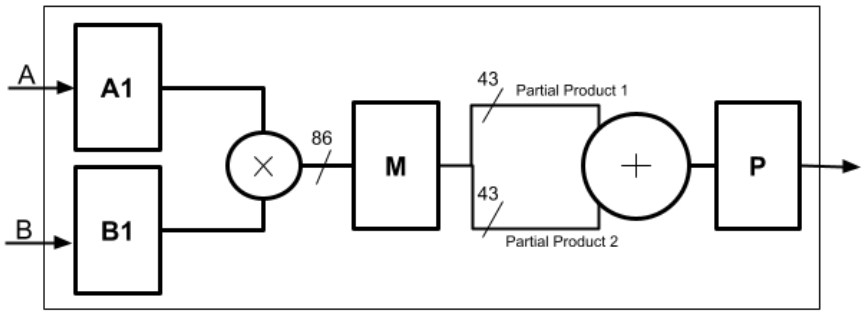


Figure 17. Diagram of the DSP48 multiplication datapath used in the proposed FPGA implementation [31, 32].

the energy signal. The generated timing report showed that the maximum frequency supported by this design would be 250 MHz, mainly limited by the minimum period restriction of the I/O toggle rate.

For the iterative method, the critical data path was the multiplications required to obtain the energy estimation at each iteration. The largest complexity comes from the implementation of eq. 3.2 and eq. 3.4 due to matrix multiplications.

To achieve a viable operating frequency, it was necessary to use hardware design strategies to improve the performance. In this case, the critical data path needed to be reduced through the remodeling of the calculation steps of eq.2 (3.2) and eq.4 (3.4). These modifications consisted in executing the multiplications and sums necessary to calculate each term of the resulting matrices in distinct states by modifying the state machine. With this technique, despite increasing the number of clock cycles for the execution of an iteration, it was possible to remove the cascaded DSP instances responsible for the longer data path delay. Figure 18 illustrates the matrix multiplications described directly in hardware with a large combinational circuit. Figure 19 shows the way we split the combinational circuit. As a result, the Xilinx timing analysis tool reported that the critical path for execution time was reduced from 12.43 ns to 1.896 ns (85% reduction). For the DE2-115 board, the Intel® Quartus® Prime Timing Analyzer tool reported 117.15 MHz as the maximum supported frequency.

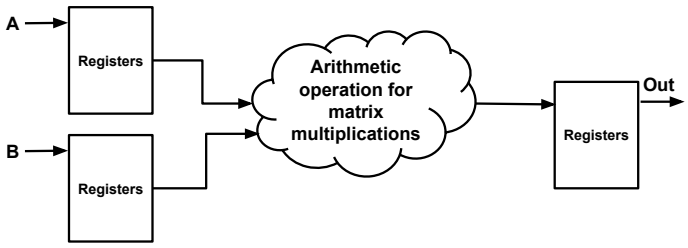


Figure 18. Diagram of the required arithmetic operations for matrix multiplications.

The results obtained from the synthesis tool concerning the use of logic elements, registers, and instances of DSPs structures used to perform the arithmetic operations of the iterative method are shown in table 3. The excessive occupation of multipliers is due to the large number of multiplication operations associated with executing operations in parallel to achieve the requirements for a very short processing time.

The power consumption extracted from the synthesis report is shown in table 4. Static power represents the amount consumed when the digital cells are not switching, and dynamic one represents

2024 JINST 19 P08022

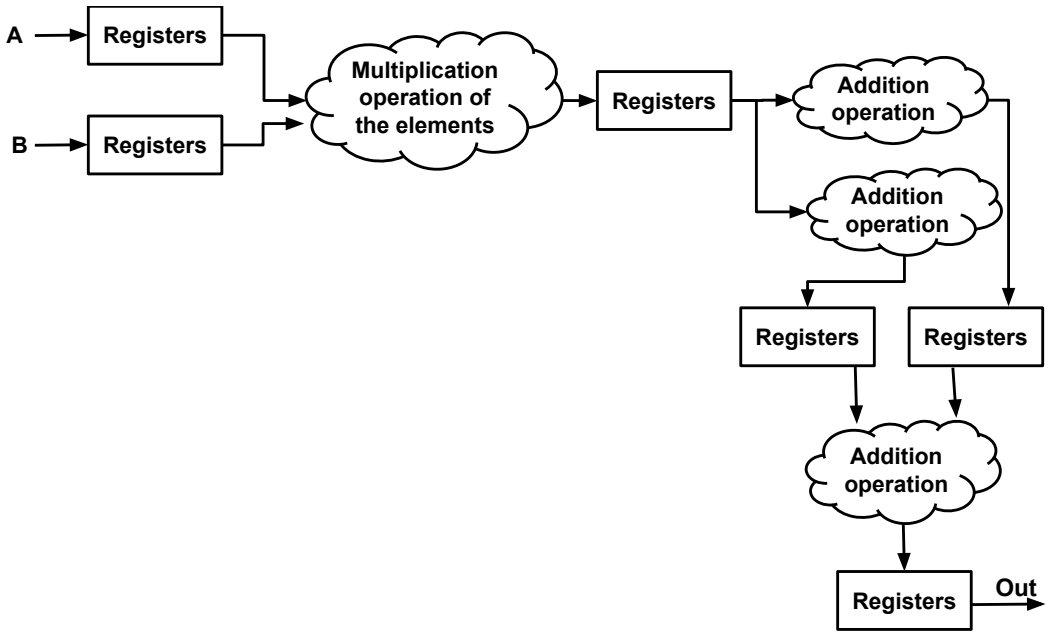


Figure 19. Diagram of the proposed combinational circuit for matrices multiplication to reduce the response time.

Table 3. Utilization of FPGA logic resources in the iterative method considering different evaluation boards (DE2-115 and XC7VX485T).

Component	Used	Available	Utilization
Altera DE2-115			
Logic elements	33416	114480	29%
Registers	24885		
Embedded Multiplier 9-bit elements	532	532	100%
Xilinx XC7VX485T			
LUTs	16036	303600	5%
Registers	27738	607200	4%
DSPs	480	2800	17%

the power when the chip is active and the cells are switching. It was observed that dynamic power contributes to increasing power consumption; this may be related to the used arithmetic structures. So, greater power is required for the iterative implementation.

4.2 Discussion

Reductions on both used area and DSP utilization were made possible by sharing the addition and multiplication structures. Optimization of processing speed was achieved by applying the parallelism technique. Considering the fixed-coefficients method, it was possible to use only one FIR structure for all channels. For the iterative method, using a finite state machine, it was possible to share the DSP structures and control the execution of each operation in different clock cycles, reducing the critical time data path.

2024 JINST 19 P08022

Table 4. Power report considering the implementation of both FIR filter and iterative method considering different evaluation boards (DE2-115 and XC7VX485T).

	Static (mW)	Dynamic (mW)	Total (mW)
FIR Filter - Altera DE2-115			
5th Order	98.58	23.57	164.39
10th Order	98.63	31.09	171.96
15th Order	98.68	39.79	180.65
20th Order	98.73	49.37	190.35
FIR Filter - Xilinx XC7VX485T			
5th Order	241.72	89.83	331.55
10th Order	241.92	112.65	354.57
15th Order	242.16	139.99	382.15
20th Order	242.53	181.13	423.66
Iterative Method - Altera DE2-115			
20 Iterations	102.84	677.74	848.69
Iterative Method - Xilinx XC7VX485T			
20 Iterations	243.67	307.18	550.85

Analyzing the achieved performance from both pileup mitigation approaches, it became clear that they fit the selected FPGA devices. Concerning information reconstruction, the PGD algorithm presented better performance: the error stabilized after 50 iterations and was considerably smaller than that obtained from the FIR method. In contrast, the processing time for the FIR filter was shorter. The FIR filter circuit needs six clock pulses to generate the result sample, while the PGD circuit needs 16 clock pulses to process one iteration and, considering 60 iterations, 961 clock pulses to generate the result for a window of samples. Considering FPGA occupation, due to its greater complexity in comparison to the FIR filter, the iterative method required higher usage of logical resources, mainly DSPs, which resulted in higher power consumption.

Considering the FPGAs, the results show that the selected high-performance device (Xilinx XC7VX485T) may operate with a higher frequency and with lower occupancy of the logical resources. Table 5 presents the main results obtained for each implementation on this device.

Table 5. Comparison of the results obtained with the FPGA implementation of both methods using Xilinx XC7VX485T evaluation board.

	FIR (20th order)	Iterative
RMSE (ADC Counts)	19.21	8.79
Minimum Clock period (ns)	1.879	1.896
Utilization of Logic elements (%)	1	5
Utilization of Registers (%)	0.245	4
Utilization of DSP Structure (%)	0.75	17
Total Power Consumption (mW)	423.66	550.85

2024 JINST 19 P08022

5 Conclusion

This paper presented a reconfigurable hardware architecture developed in FPGA technology to mitigate the pileup effect in fast online systems. Considering recent results from the literature, the proposed technique for signal recovery was inverse filtering (deconvolution) using both fixed-coefficients and iterative approaches. The achieved solution focused on reducing area, power consumption, utilization of the DSP structures, and processing time. From the perspective of signal recovery, the iterative approach, implemented through the Positive Gradient Descent method, seems to fit better to such signal processing tasks. Two FPGA families were used as target devices for circuit synthesis, one addressing general-purpose applications and another focused on high-performance digital signal processing designs. The simulation results indicated that both devices could be used, but as expected, the high-performance FPGA family showed better timing and resource occupancy performance. In future works, different designs of digital filters using other interactive methods and neural networks will be considered.

Acknowledgments

The authors thank FAPESB, CNPq, FAPERJ, and RENAFAE for their financial support (Brazil). This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior — Brasil (CAPES) Finance Code 001 and by the CAPES-COFECUB program.

References

[1] K. Murata and K. Ogawa, *Influence of pulse pile-up effects on material decomposition with photon-counting CT*, in the proceedings of the *IEEE Nuclear Science Symposium and Medical Imaging Conference*, Boston, MA, U.S.A., 31 October–7 November 2020 [DOI:10.1109/nss/mic42677.2020.9507751].

[2] H. Yang, J. Zhang, J. Zhou and J. Fan, *Efficient pile-up correction based on pulse-tail prediction for high count rates*, *Nucl. Instrum. Meth. A* **1029** (2022) 166376.

[3] J.P.B.S. Duarte et al., *Online energy reconstruction for calorimeters under high pile-up conditions using deconvolutional techniques*, 2019 *JINST* **14** P12017.

[4] J.L. Marin, *Energy Reconstruction Performance in the ATLAS Tile Calorimeter Operating at High Event Rate Conditions Using LHC Collision Data*, in the proceedings of the *27th International Conference on Systems, Signals and Image Processing*, Niteroi, Brazil, 1–3 July 2020 [DOI:10.1109/IWSSIP48289.2020.9145451].

[5] G. Apollinari, O. Brüning, T. Nakamoto and L. Rossi, *Chapter 1: High Luminosity Large Hadron Collider HL-LHC*, in *CERN Yellow Rep.* (2015), pp. 1–19 [DOI:10.5170/CERN-2015-005.1] [arXiv:1705.08830].

[6] ATLAS collaboration, *The ATLAS Experiment at the CERN Large Hadron Collider*, 2008 *JINST* **3** S08003.

[7] ATLAS collaboration, *Operation and performance of the ATLAS Tile Calorimeter in Run 1*, *Eur. Phys. J. C* **78** (2018) 987 [arXiv:1806.02129].

[8] R. Wigmans, *Calorimetry: Energy measurement in particle physics*, Oxford University Press, second edition (2017) [DOI:10.1093/oso/9780198786351.001.0001].

2024 JINST 19 P08022

[9] N. Nedjah, H.S. Lopes and, L. de Macedo Mourelle, eds., *Designing with Computational Intelligence*, Springer International Publishing (2017) [DOI:10.1007/978-3-319-44735-3].

[10] M. Livan and R. Wigmans, *Calorimetry for Collider Physics, an Introduction*, UNITEXT for Physics, Springer International Publishing (2019) [DOI:10.1007/978-3-030-23653-3].

[11] ATLAS collaboration, *Technical Design Report for the Phase-II Upgrade of the ATLAS TDAQ System*, CERN-LHCC-2017-020, ATLAS-TDR-029, CERN, Geneva (2017) [DOI:10.17181/CERN.2LBB.4IAL].

[12] J.L. Ortiz Arciniega, F. Carrió and A. Valero, *FPGA implementation of a deep learning algorithm for real-time signal reconstruction in particle detectors under high pile-up conditions*, 2019 *JINST* **14** P09002 [arXiv:1903.02439].

[13] P.T. Komiske, E.M. Metodiev, B. Nachman and M.D. Schwartz, *Learning to Remove Pileup at the LHC with Jet Images*, *J. Phys. Conf. Ser.* **1085** (2018) 042010.

[14] N. Chiedde, *Machine learning for real-time processing of ATLAS liquid argon calorimeter signals with FPGAs*, 2022 *JINST* **17** C04010 [arXiv:2111.08590].

[15] M. Gal et al., *Deconvolution enhanced direction of arrival estimation using one- and three-component seismic arrays applied to ocean induced microseisms*, *Geophys. J. Int.* **206** (2016) 345.

[16] S. Del Favero, A. Facchinetti, G. Sparacino and C. Cobelli, *Improving Accuracy and Precision of Glucose Sensor Profiles: Retrospective Fitting by Constrained Deconvolution*, *IEEE Trans. Biomed. Eng.* **61** (2014) 1044.

[17] Z. He et al., *A Deconvolutional Reconstruction Method Based on Lucy–Richardson Algorithm for Joint Scanning Laser Thermography*, *IEEE Trans. Instrum. Meas.* **70** (2021) 1.

[18] L.M. de A. Filho, B.S. Peralva, J.M. de Seixas and A.S. Cerqueira, *Calorimeter Response Deconvolution for Energy Estimation in High-Luminosity Conditions*, *IEEE Trans. Nucl. Sci.* **62** (2015) 3265.

[19] D.P. Barbosa et al., *Sparse Representation for Signal Reconstruction in Calorimeters Operating in High Luminosity*, *IEEE Trans. Nucl. Sci.* **64** (2017) 1942.

[20] R.G. Lyons, *Understanding Digit. Signal Process.*, third edition, Pearson Education (2010).

[21] P. Bandžuch, M. Morháč and J. Krištiak, *Study of the Van Cittert and Gold iterative methods of deconvolution and their application in the deconvolution of experimental spectra of positron annihilation*, *Nucl. Instrum. Meth. A* **384** (1997) 506.

[22] C. Clement and P. Klimek, *Identification of Pile-up Using the Quality Factor of Pulse Shapes in the ATLAS Tile Calorimeter*, in the proceedings of the *Nuclear Science Symposium and Medical Imaging Conference*, Valencia, Spain, 23–29 October 2011, pp. 1188–1193 [ATL-TILECAL-PROC-2011-014] [DOI:10.1109/NSSMIC.2011.6154599].

[23] U. Meyer-Baese, *Digital Signal Processing with Field Programmable Gate Arrays*, Springer Berlin Heidelberg (2014) [DOI:10.1007/978-3-642-45309-0].

[24] M. Gnanasekaran and M. Manikandan, *High throughput pipelined implementation of reconfigurable FIR filter for SDR*, in the proceedings of the *International Conference on Current Trends in Engineering and Technology (ICCTET)*, Coimbatore, India, 3 January 2013, pp. 362–364 [DOI:10.1109/icctet.2013.6675986].

[25] J.G. Proakis and D.K. Manolakis, *Digit. Signal Process.: Principles, Algorithms and Applications*, fourth edition, Pearson (2006).

[26] A.A. Dunca, *Estimates of the discrete van Cittert deconvolution error in approximate deconvolution models of turbulence in bounded domains*, *Appl. Numer. Math.* **134** (2018) 1.

2024 JINST 19 P08022

[27] S. Haykin, *Adaptive Filter Theory*, Always learning, Pearson (2014).

[28] *DE2-115 User Manual*, Terasic Technologies (2017).

[29] *7 Series FPGAs Data Sheet: Overview*, Xilinx (2018).

[30] R. Field, *Min-Bias and the Underlying Event at the LHC*, *Acta Phys. Polon. B* **42** (2011) 2631 [arXiv:1110.5530].

[31] *7 Series DSP48E1 Slice*, Xilinx (2018).

[32] H.Y. Cheah, F. Brosser, S.A. Fahmy and D.L. Maskell, *The iDEA DSP Block-Based Soft Processor for FPGAs*, *ACM Trans. Reconfig. Technol. Syst.* **7** (2014) 1.

2024 JINST 19 P08022