Prepared for submission to JINST

TWEPP2024 - Topical Workshop on Electronics for Particle Physics
30 September 2024 to 4 October 2024
Glasgow, UK

Machine Learning for Real-Time Processing of ATLAS Liquid Argon Calorimeter Signals with FPGAs

Johann Christoph Voigt on behalf of the ATLAS Liquid Argon Calorimeter group

IKTP, TU Dresden, Germany

E-mail: johann_christoph.voigt@tu-dresden.de

Abstract: To deal with the higher pileup after the high-luminosity upgrade of the LHC, the ATLAS liquid argon calorimeter readout electronics will be replaced. 556 Intel Agilex-7 FPGAs will be installed as part of the off-detector electronics, enabling a more sophisticated digital energy reconstruction. Instead of using a linear optimal filter like the current readout, we evaluated the use of recurrent and convolutional neural networks. These artificial neural networks are trained on signal-enriched simulated detector pulses for a set of representative cells. The network size is limited to approximately 500 multiply-accumulate operations per detector cell. They show improvements in energy reconstruction especially for the case of overlapping signal pulses, as they will appear more often under high-luminosity conditions. An FPGA prototype firmware has been developed for both RNNs and CNNs, that supports processing the required number of 384 detector cells on one FPGA. This uses a low-level VHDL approach to minimize the resource usage. The ANN architecture is fixed at compile time, but the weights are configurable per cell. The compilation results show that such a neural network based readout is feasible with the available hardware. This is possible through the use of time multiplexing, i.e. by running the FPGA at a multiple of the 40 MHz input data frequency.

Keywords: Calorimeters, Electronic detector readout concepts, Data processing methods

Contents

1 ATLAS Liquid Argon Calorimeter after Phase-II Upgrade

The liquid argon (LAr) calorimeters are a set of sampling calorimeters used by the ATLAS detector [1] in the electromagnetic barrel and end-cap region with a pseudorapidity $|\eta|$ up to 3.2, as well as in an $|\eta|$ range of 1.5 to 4.9 for the hadronic end-cap and forward calorimeters.

During the upcoming Long-Shutdown 3 of the LHC [2], the calorimeter readout electronics will be replaced as part of the larger Phase-II upgrade project [3] of the ATLAS detector. This is necessitated mainly by the increased pileup at a projected mean number of simultaneous collisions per bunch crossing (BC) $\langle\mu\rangle$ of up to 200, as well as the increase in the targeted first-level trigger accept rate of 1 MHz.

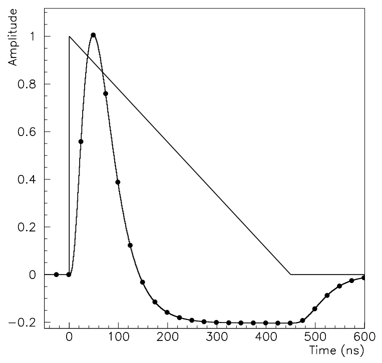

Figure 1. The raw LAr pulse has a triangular shape with a duration in the order of 500 ns. This is amplified and shaped into a bipolar pulse and sampled every 25 ns (i.e. at 40 MHz) before being digitized. The energy deposited in the detector cell is encoded in the amplitude of the pulse. [4]

The LAr calorimeters consist of a total of 182 468 detector cells, which will each be read out individually at the LHC BC frequency of 40 MHz. Figure 1 shows a typical LAr pulse. This constitutes the input to the off-detector side of the readout. Here, away from the radiation environment, the new LAr signal processor system will be equipped with 556 Intel Agilex-7 FPGAs to compute the energy deposited in the calorimeter cells in real-time.

2 Digital Readout

2.1 Current Plans and Requirements

The current readout electronics use an Optimal Filter (OF) [5] for the energy reconstruction. The filter calculates a weighted sum over 4 samples using filter coefficients, that optimize the signal-to-noise ratio. The new readout is designed to reconstruct the cell-level energies in real-time with the full granularity.

As a potential improvement over the baseline OF, we propose the use of small artificial neural networks (ANNs) for the digital energy reconstruction [6]. The inference will be executed on the FPGAs in real-time, which sets constraints on the possible network architectures and firmware implementation. The latency budget available for the energy calculation is approximately 150 ns, since the results are used as input to the ATLAS trigger system. For an ANN, this means that the number of layers should be as small as possible. On the firmware side, this necessitates a well optimized pipeline. One FPGA is required to process the 40 MHz input streams from up to 384 detector cells in parallel. The possible size of the neural networks is thus severely limited to the order of 500 multiply-accumulate (MAC) operations. This assumes time multiplexing, in order to partially serialize the processing of the detector cells and reduce the required number of parallel firmware instances of the ANN, as further detailed in section 3. The targeted Intel Agilex-7 FPGA has dedicated multiplication units, that can execute two simultaneous multiplications when using 18-bit fixed-point numbers. Quantization aware training can be used to specifically train networks for this bit width.

The compilation of the firmware is in itself already a time-intensive process. To reduce the compilation effort and make the deployment easier, it is planned to only have a single compiled firmware version that is used for all 556 FPGAs. This mandates that the ANN weights can be loaded over an Ethernet connection during runtime. Furthermore, all detector cells will be processed by the same network architecture. The common approach, when deploying ANNs on FPGAs, of extensively using pruning is therefore not viable, as one would have to guarantee that all network weights used for any detector cell can be pruned in the same way. Pruning would also conflict with the multiplexing approach.

2.2 Artificial Neural Network Architectures

Figure 2 shows the structure of a vanilla RNN cell. The full RNN consists of 5 of these cells with 8 internal dimensions each and a dense neuron that extracts the output at the last cell. This results in a total of 368 multiplications. The network operates in a sliding window mode, where at every BC the network is reinitialized and each RNN cell is provided one input sample out of the sequence. The field of view, i.e. the number of samples included in the calculation of one output value, is therefore 5. If necessary, this can later be extended using a dense layer preceding the first RNN cell to better compensate the influence of previous pulses.

The CNN consists of two layers, where the first layer operates on the input data and can make use of parallel filters. With dilation, this network reaches a larger field of view of 22 BC. The network shown contains approximately 100 MAC operations. Both networks use a rectified linear unit (ReLU) as activation function, as it is cheap to implement on FPGAs and generally very suitable for regression problems.

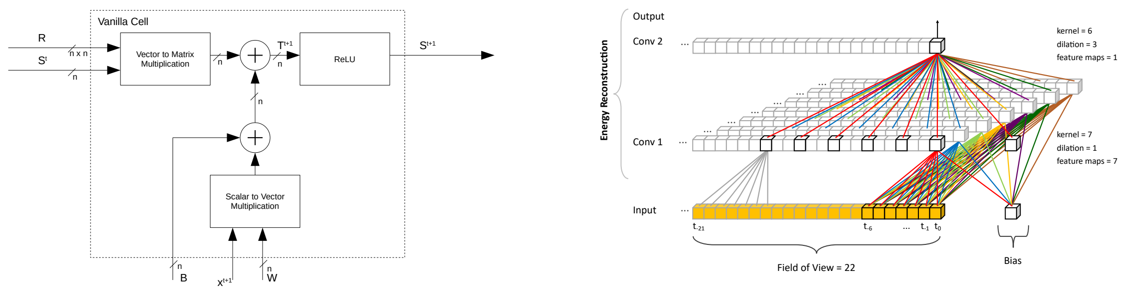

Figure 2. Left: Structure of a vanilla RNN cell. The state vector of the previous cell S^t is multiplied with the recurrent kernel weight matrix R. The digitized input sample x^{t+1} is multiplied with the kernel weight vector W. The internal state vector T^{t+1} is the sum of these and a bias. This is passed through the ReLU activation to obtain the new state vector S^{t+1}. [7]
Right: Architecture of a 2-layer CNN. The input sequence at the bottom is densely screened by the first layer with dilation 1. The last layer increases the field of view to 22 BC by sparsely screening the output of the first layer with a dilation of 3. [8]

2.3 Training and Performance

The networks are trained on signal-enriched simulated detector sequences with pileup. This allows the use of the true deposited energy as a training target. The training is done with Keras [9] using the QKeras extension for quantization.

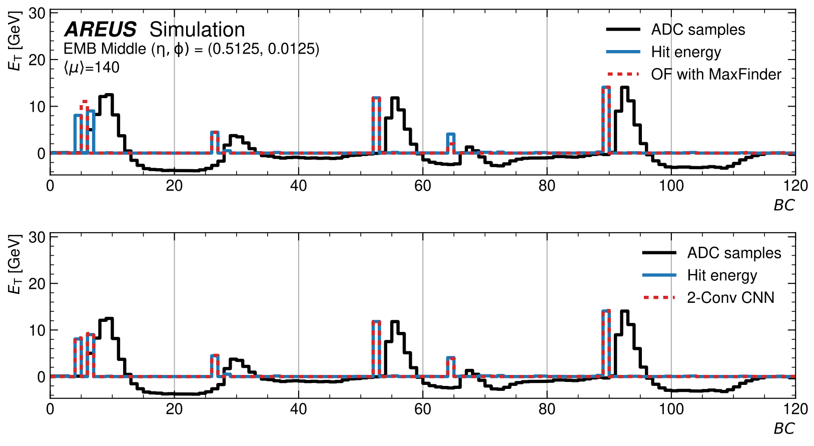

Figure 3. Comparison of OF (top, red) and CNN (bottom, red) energy reconstruction for an example sequence simulated with $\langle\mu\rangle = 140$. The ADC samples (black) shown are the input for the networks and the true hit energy is the desired output. [8]

Figure 3 shows an example sequence in terms of true deposited energy and the resulting ADC sequence, which is the input to the networks. One can see that the optimal filtering approach combined with a maximum finder works very well for isolated hits (e.g. at BC 89). However, the ANNs show increased accuracy for the case of overlapping pulses (e.g. at BC 5 and 64), in particular the case of energy deposits within the undershoot of a preceding pulse. [6]

3 FPGA Firmware

3.1 VHDL Implementation of RNN and CNN Inference

The first version of the RNN inference firmware [6] uses Intel's high-level synthesis (HLS) language. This allowed further studies into the effects of rounding and truncation in different parts of the network [7]. This version supports multiplexing. By running the FPGA at a multiple of the frequency of the input data, every RNN firmware block can process the input from multiple detector cells in a serial manner, therefore reducing the required number of parallel instances of the RNN firmware block on the FPGA. Since this firmware did not meet the requirements, the RNN inference has also been implemented in the low-level VHDL language. Further reductions in resource consumption and improvements of the maximum clock frequency are possible through the reuse of common results between RNN cells, manual placement constraints, as well as incremental compilation, where RNN instances that do not meet the timing constraints are recompiled, while the rest of the larger layout is preserved. The CNN inference has been implemented in VHDL directly and also utilizes multiplexing to reduce the number of required instances. With a targeted multiplexing factor of 12, this requires a clock frequency of 480 MHz. The firmware is custom-tailored for Intel Stratix and Agilex FPGAs, by manually assigning all multiplications to the digital signal processors (DSPs) available for this purpose. This is best achieved with 18-bit fixed-point numbers for both the network weights and the data. The DSPs can be chained together for efficient MAC cascades, like they appear in the CNN structure. The particular architecture of the CNN is described by a configuration file, which fixes the structure at compilation time. This configuration is automatically generated from the Keras output files obtained from the training.

Table 1. Estimates of the resource usage for Intel Stratix-10 (reference 1SG280HU2F50E2VG) and Agilex-7 (reference AGIC040R39A2E2VR0) FPGAs. [10]

FPGA	Network	#MACs	Multiplex.	#Cells	f_{max}	ALMs	DSPs
Stratix-10	RNN (HLS)	368	10×	370	393 MHz	90 %	100 %
	RNN (VHDL)	368	14×	392	561 MHz	18 %	66 %
	CNN	91	12×	396	415 MHz	8 %	28 %
Agilex-7	CNN	91	12×	396	539 MHz	4 %	13 %

Table 1 summarizes the resource consumption and maximum clock frequency of the different firmware versions as reported by the Intel Quartus compiler. Initial development was targeting the Straix-10 FPGA, but the project switched to the Agilex-7 later. The VHDL versions show that the resource usage, especially in terms of Adaptive Logic Modules (ALMs), of a full design supporting at least 384 detector cells is low enough to also fit the other required components of the larger firmware project on the same FPGA. DSPs are not as critical, as they will be almost exclusively used by the ANNs. With the final board using the Agilex FPGA, the demonstrated timings are sufficient for the planned time multiplexing factor for both RNNs and CNNs.

3.2 Testing the Firmware on Stratix-10 Development Kit

To show that the developed firmware also works on hardware, test projects have been set up for both architectures. They combine the ANNs with memory blocks to store example input data for

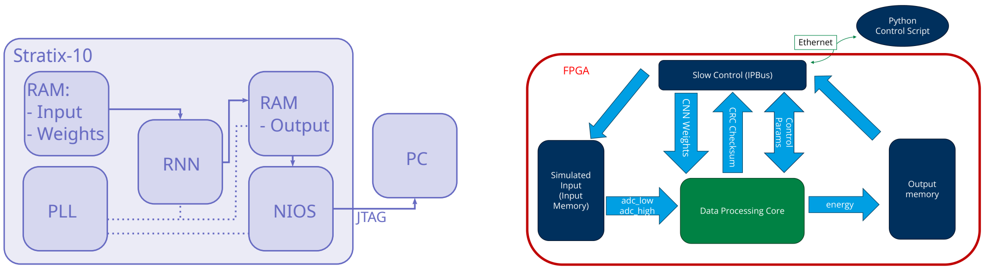

Figure 4. Schema of the wrapper firmware project to test RNN (left) and CNN (right) on a devkit.

the network, as well as the resulting output. In case of the RNNs, the input and output memory blocks can be accessed from a NIOS soft-processor, that is connected via JTAG to a PC. The CNN prototype uses a control interface based on IPbus [11] to access the memory blocks over an Ethernet connection. Both demonstrator projects have been successfully run on a Stratix-10 development kit and the results match the expectations. While the RNNs run at 560 MHz, a reduced clock frequency of 125 MHz has been used for the CNNs so far. Running the CNNs at their nominal frequency of 480 MHz is planned for tests on an Intel Agilex development board.

4 Summary and Outlook

It has been shown in previous studies [6] that both RNNs and CNNs can outperform the OF, especially for the case of overlapping signals. For both architectures, an FPGA implementation using VHDL has been developed to demonstrate the feasibility of this new signal filtering approach using the FPGAs that will be deployed as part of the Phase-II upgrade. The low occupancy of the firmware on the Agilex FPGA, combined with its higher possible clock frequencies, motivate the training of larger networks in the future. The behaviour of both implementations has been verified on development boards. Going forward, the ANN firmware will be further integrated into the larger firmware project of the LAr off-detector readout. Separate studies on how the neural network based readout influences the reconstruction of physics objects, like electrons or jets, have been started.

Acknowledgments

This work was in part supported by the German Federal Ministry of Education and Research within the projects 05H19ODCA9 and 05H24OD9 and by the DFG fund "NFDI 39/1" in the framework of the PUNCH4NFDI consortium. The project leading to this publication has received funding from Excellence Initiative of Aix-Marseille Université - A*MIDEX, a French “Investissements d’Avenir” programme, AMX-18-INT-006 and from the French “Agence National de la Recherche”, ANR-20-CE31-0013. This work received support from the French government under the France 2030 investment plan, as part of the Excellence Initiative of Aix-Marseille University - A*MIDEX (AMX-19-IET-008 - IPhU).

References

[1] ATLAS Collaboration, *The ATLAS experiment at the CERN large hadron collider*, *Journal of Instrumentation* **3** (2008) S08003.

[2] L. Evans and P. Bryant, *LHC machine*, *Journal of Instrumentation* **3** (2008) S08001.

[3] ATLAS Collaboration, *ATLAS Liquid Argon Calorimeter Phase-II Upgrade: Technical Design Report*, Tech. Rep. CERN-LHCC-2017-018. ATLAS-TDR-027, CERN, Geneva (Sep, 2017), https://cds.cern.ch/record/2285582.

[4] ATLAS Collaboration, *ATLAS liquid-argon calorimeter: Technical Design Report*, Tech. Rep. CERN-LHCC-96-041, CERN, Geneva (1996), https://cds.cern.ch/record/331061.

[5] W. Cleland and E. Stern, *Signal processing considerations for liquid ionization calorimeters in a high rate environment*, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **338** (1994) 467.

[6] G. Aad, A.-S. Berthold, T. Calvet, N. Chiedde, E.M. Fortin, N. Fritzsche et al., *Artificial neural networks on FPGAs for real-time energy reconstruction of the ATLAS LAr calorimeters*, *Computing and Software for Big Science* **5** (2021) .

[7] G. Aad, T. Calvet, N. Chiedde, R. Faure, E. Fortin, L. Laatu et al., *Firmware implementation of a recurrent neural network for the computation of the energy deposited in the liquid argon calorimeter of the ATLAS experiment*, *Journal of Instrumentation* **18** (2023) P05017.

[8] ATLAS LAr Calorimeter Group, "Public liquid argon calorimeter plots on upgrade." https://twiki.cern.ch/twiki/bin/view/AtlasPublic/LArCaloPublicResultsUpgrade.

[9] F. Chollet et al., "Keras." `https://keras.io`, 2015.

[10] Voigt, Johann Christoph, *Machine learning for real-time processing of ATLAS liquid argon calorimeter signals with FPGAs*, *EPJ Web of Conf.* **295** (2024) 09025.

[11] C.G. Larrea, K. Harder, D. Newbold, D. Sankey, A. Rose, A. Thea et al., *IPbus: a flexible Ethernet-based control system for xTCA hardware*, *Journal of Instrumentation* **10** (2015) C02019.