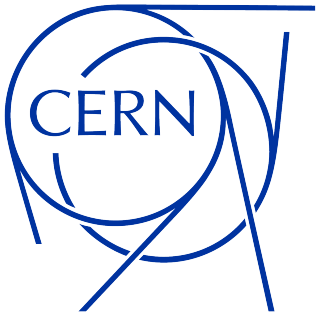


European Organization for Nuclear Research(CERN)

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

TIPP 2023 Cape Town

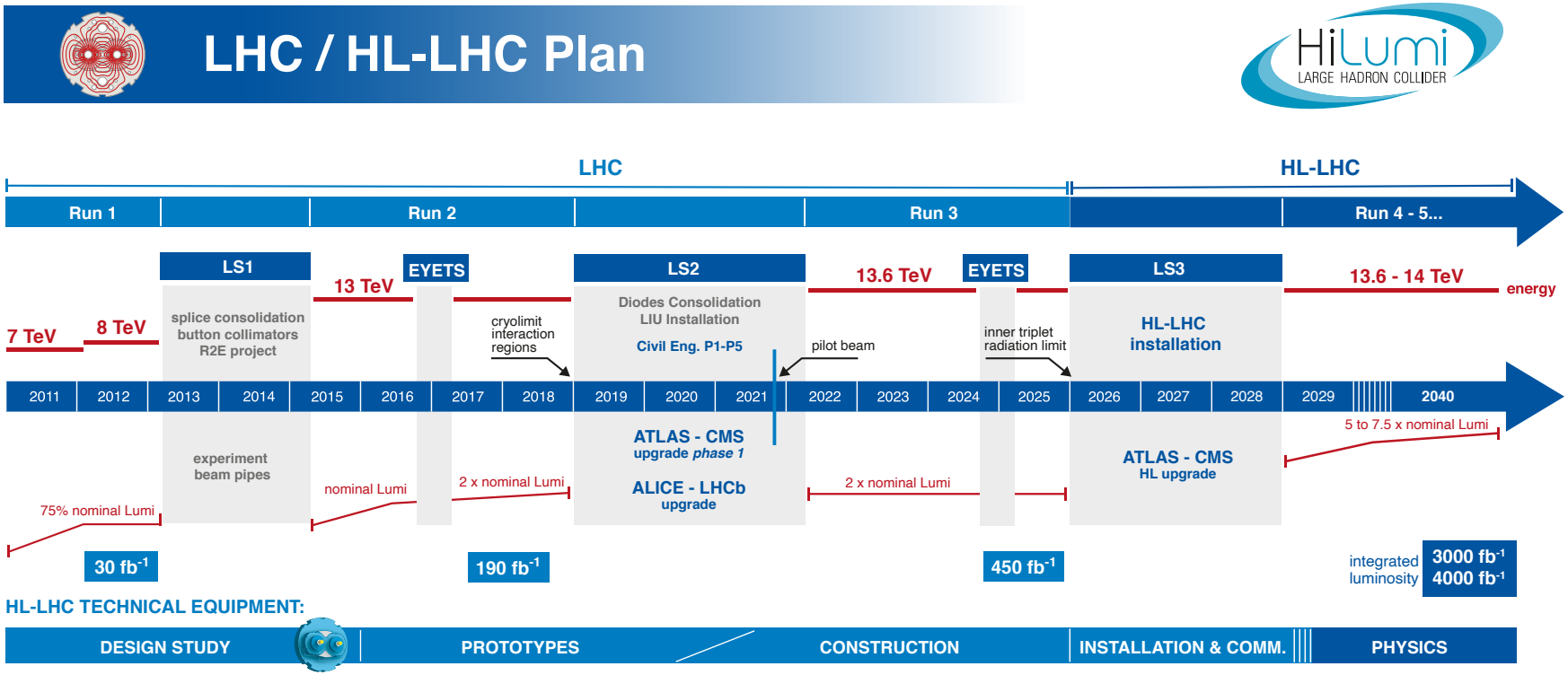
07/09/2023

Etienne FORTIN

On behalf of the ATLAS Liquid Argon Calorimeter Group

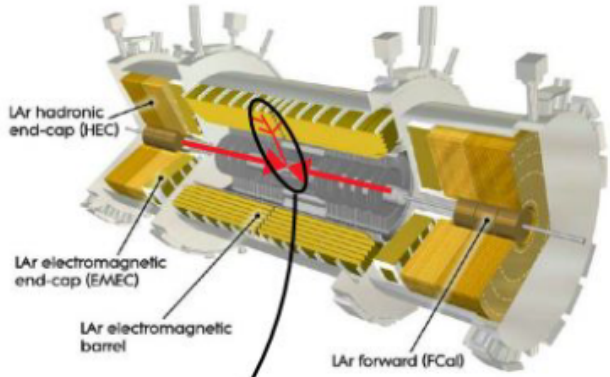
Introduction

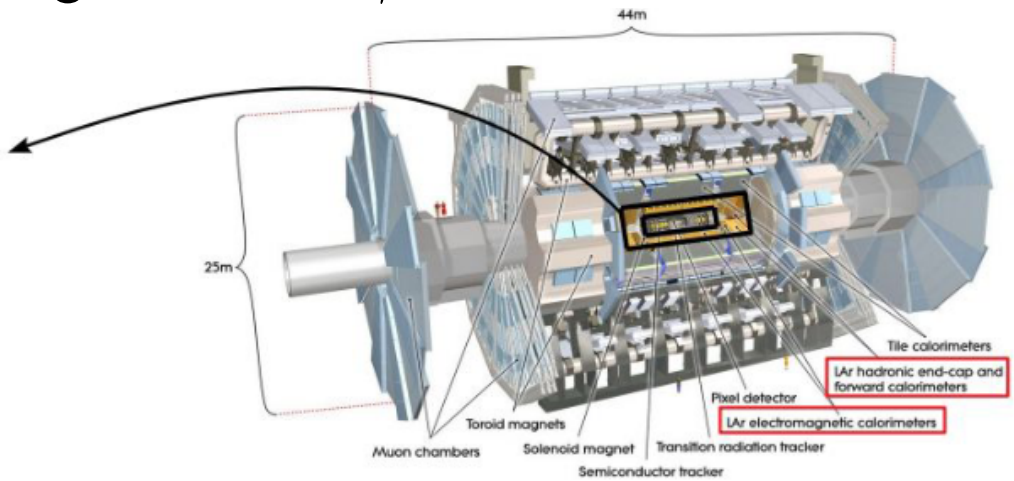
- LHC will be upgraded to HL-LHC starting in 2026.
  - Instaneous luminosity increased by a factor of 5 to 7
  - Pileup of 140 to 200 p-p collision
- To cope with the new conditions ATLAS will be upgraded
  - Trigger level 1 increased from 100 kHz to 1 MHz
  - On detector electronic need to handled higher level of radiations. . .



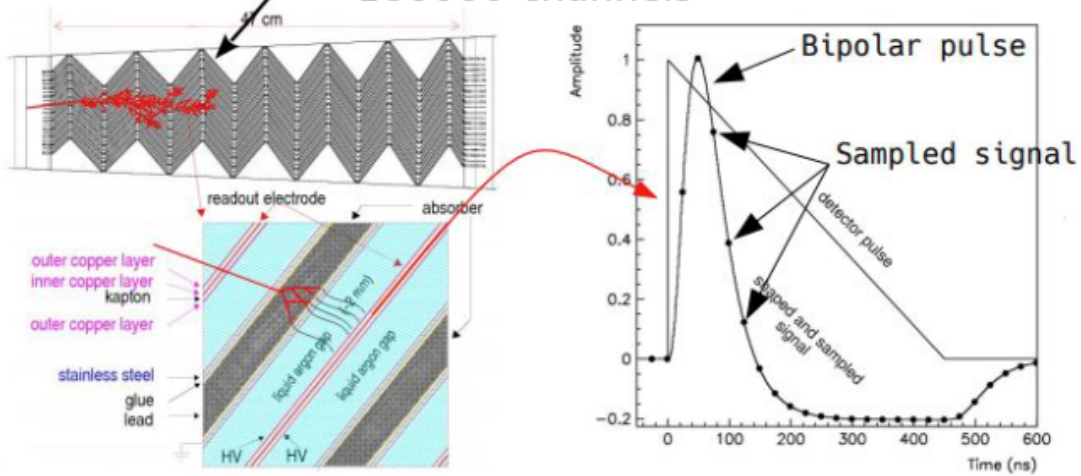
Liquid Argon Calorimeter

Measures the energies of $e^{\pm}$, γ and forward hadrons



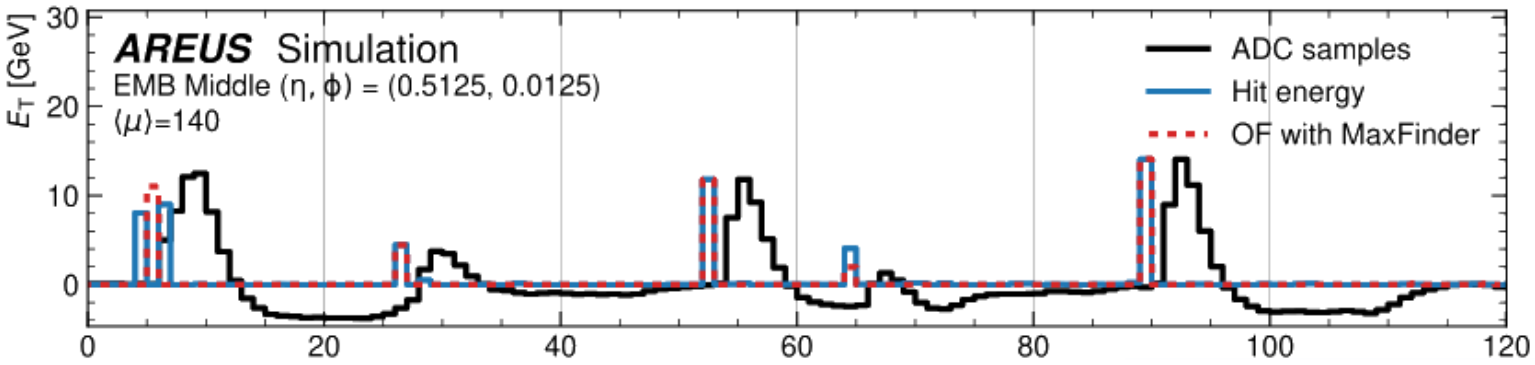


180000 channels

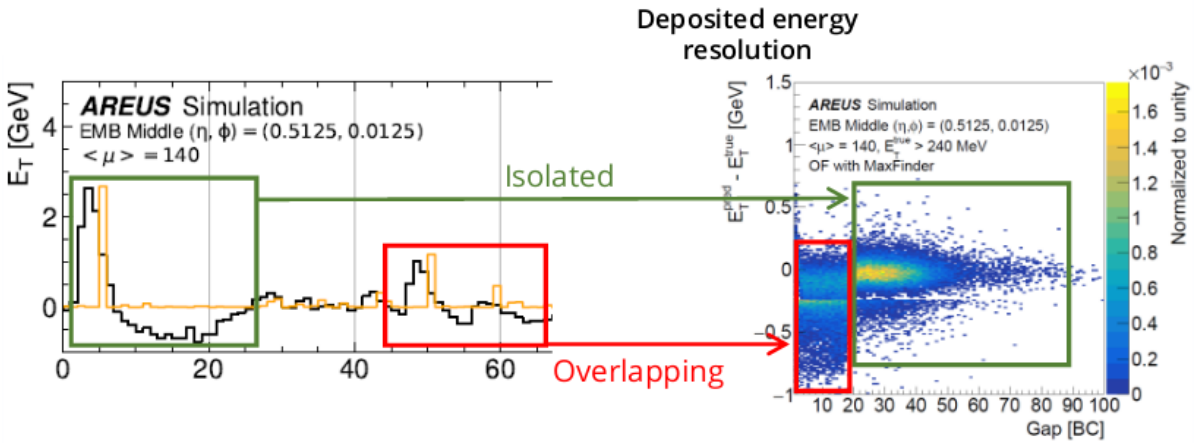


- Electronic signal with amplitude corresponding to the deposited energy in the calorimeter
- Shaped and sampled at 40 MHz
- Samples used to compute the deposited energy

Current Method for Energy measurement



- Use the "gap" as figure of merit to quantify the time distance between 2 significant energy deposits
- Optimal Filtering (OF) algorithm cannot correct past events overlapping current event with small gap



- High pile-up will cause more overlapped and distorted pulse
- To keep the performance we **need to improve the algorithms for energy computation**

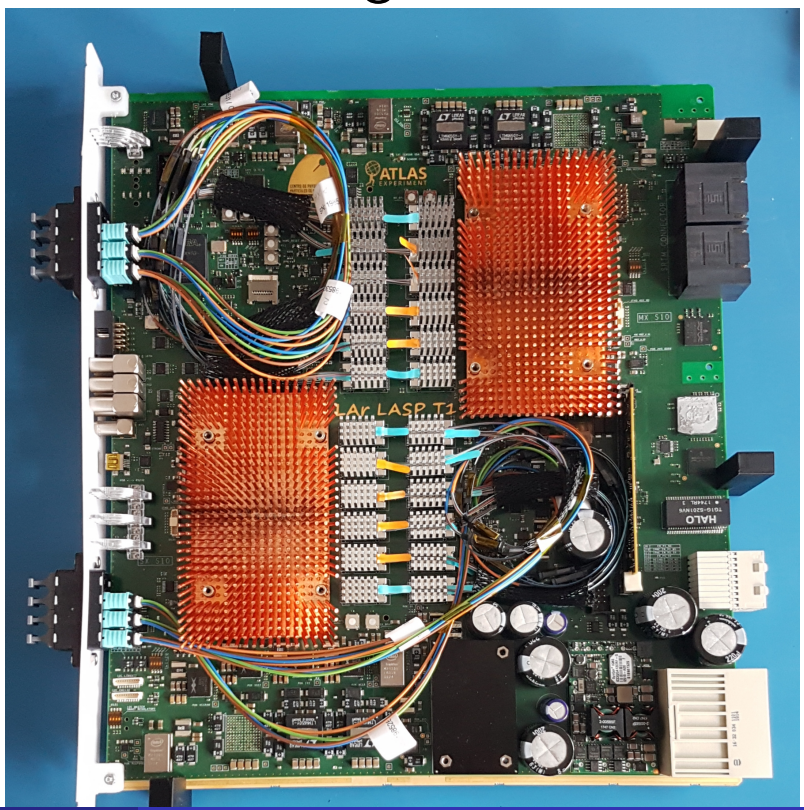
Additional environnement constrains

- Need to compute energies on 125 ns
- Need to provide energies for **each LHC bunch crossing (every 25ns)** for the trigger system.
- Want to use full granularity of dectector for better performances $->$ **need to process 182k Cells**
- **Limited space** as the processing unit is positioned underground

The proposed solution is:

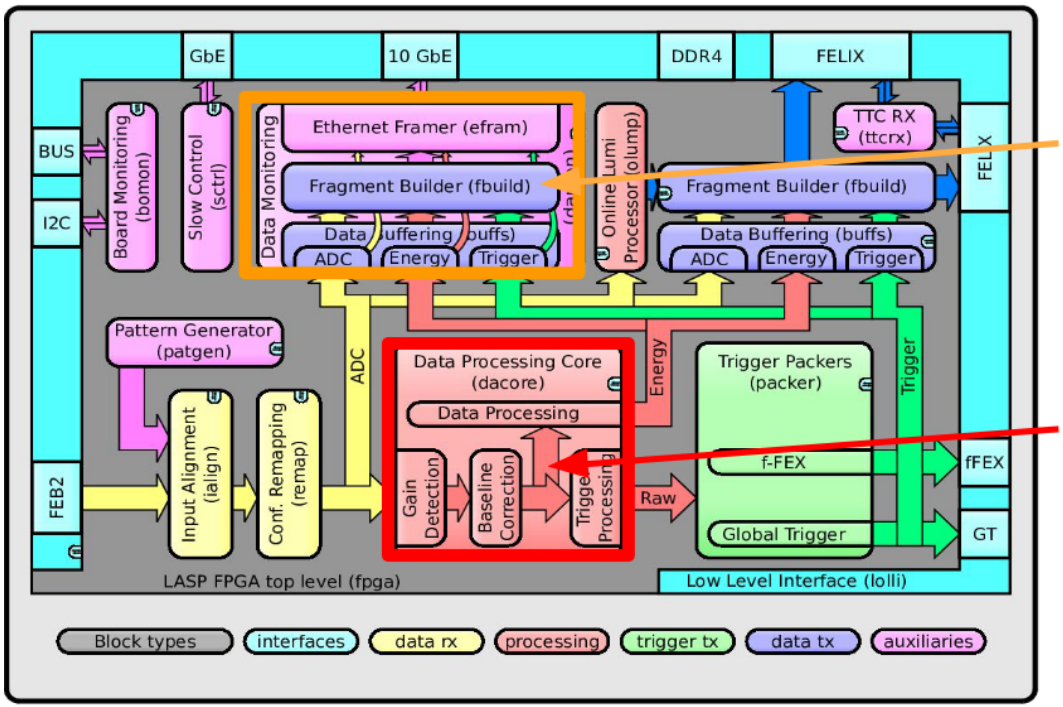
- Use High end FPGAs on ATCA boards
- 2 FPGA/boards
- High number and high speed input and output links

We call it a LASP (Liquid Argon Signal Processing)



LASP

- **LASP board containing 2 processing units based on INTEL FPGAs**
  - Demonstrator board available with Stratix 10 FPGAs
    - baseline for this talk
  - Final board will be based on Agilex FPGAs
  - Stratix and Agilex improved a lot the computation capabilities of FPGAs
- **One FPGA should process 384 channels**
  - Around 125 ns allocated latency for energy computation



Buffering waiting for L1A
Could be used to refine energy computation with less stringent latency requirements

Compute energy at 40 MHz
Assign the energy to the correct bunch crossing (collision time)

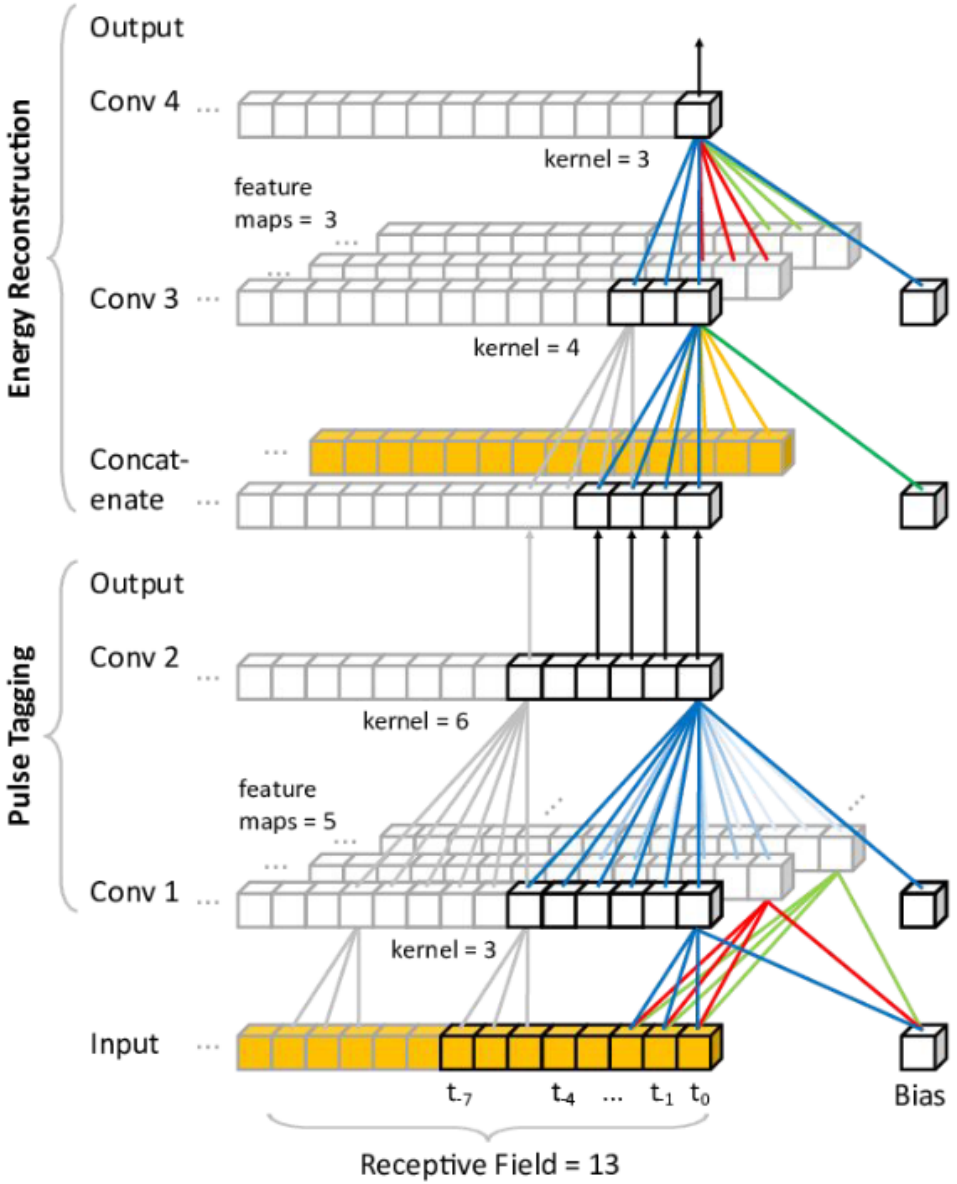
Summary

- The new generation of FPGA gives more ressoures
  - Allow more complex algorithm
- We investigates the use of neural network
  - Have to meet all the constraints (Ressources,latency...)
  - Good precision in the proccessing
- 2 Types of neural networks (NN)
  - Convolutional NN (CNN)
  - Recurrent NN (RNN)

CNN structure

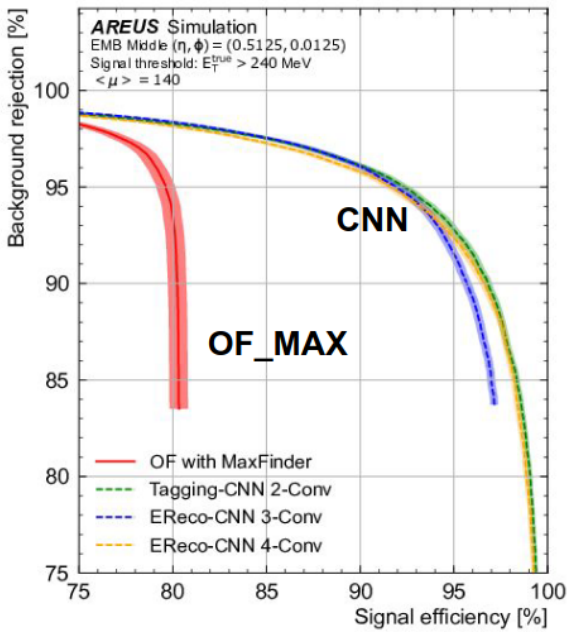
Comput Softw Big Sci 5, 19 (2021)

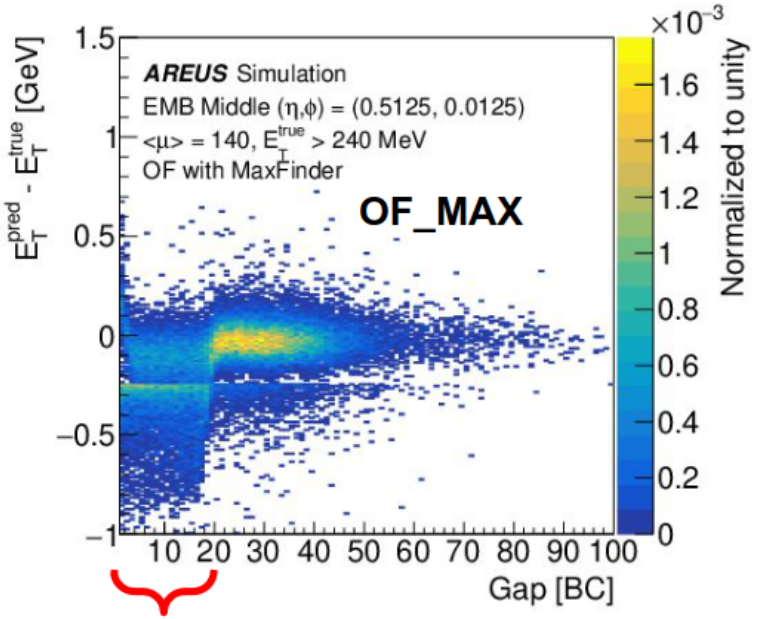
- Two blocks to identify the time of energy deposit and compute the corresponding energy
  - Two variations with 3 or 4 convolution layers (3-conv and 4-conv)
- **Energy computation block**
  - Outputs the computed energy at a given bunch crossing
- **Time (bunch crossing) identification block**
  - Outputs the probability of energy deposit (above noise) at a given bunch crossing
- 28 (13) samples as input for 3-conv (4-conv)
  - 23 (8) in the past (before the probed bunch crossing) to account for previous deposits

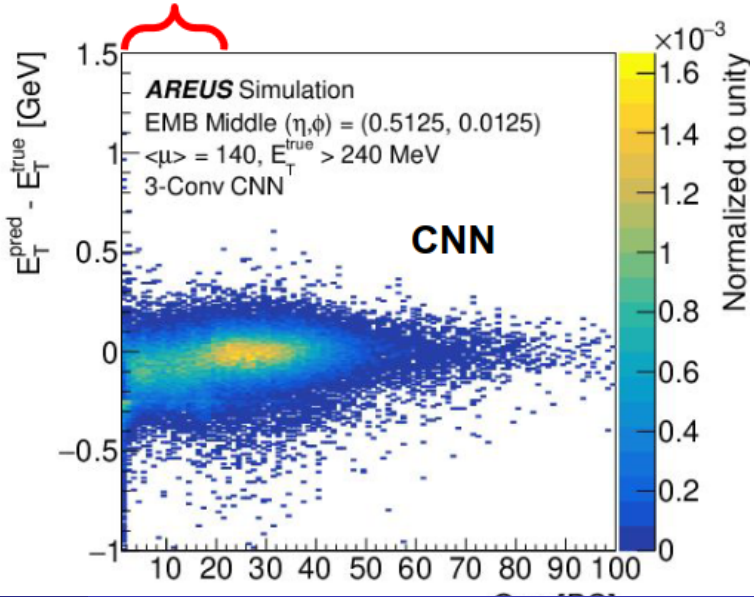


CNN performance

- Clean improvement in energy resolution in the region where pulses overlap (low time gap between two signals)
- Improved efficiency in detecting energy deposits
  - Higher rejection of noise and mis-timed signals



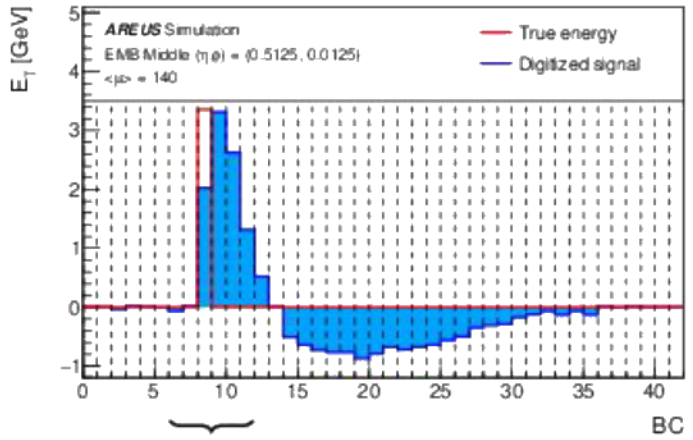


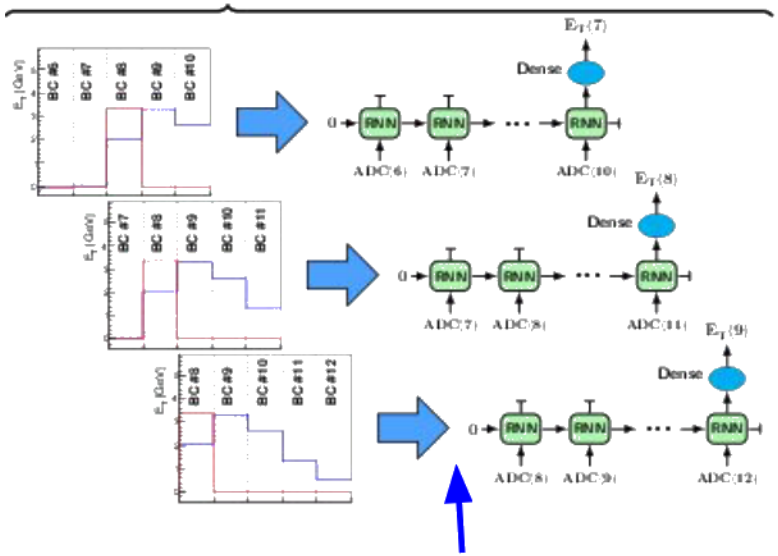


RNN structure

Comput Softw Big Sci 5, 19 (2021)

- Two architectures used
  - Single cell and Sliding windows
  - 4 samples corresponding to the signal are used
    - + 1 in the past for the sliding windows
- Two types of RNNs
  - Vanilla RNN and LSTM(Long Short-Term Memory)

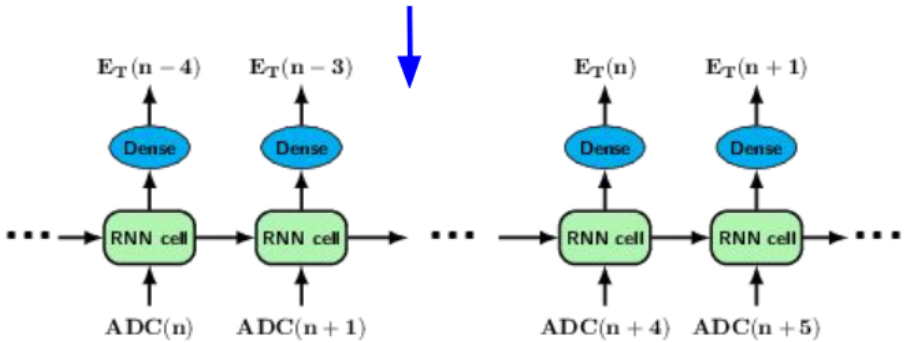




Single cell architecture

Continuous computation with a single cell

Takes into account full past info (from the beginning of run)



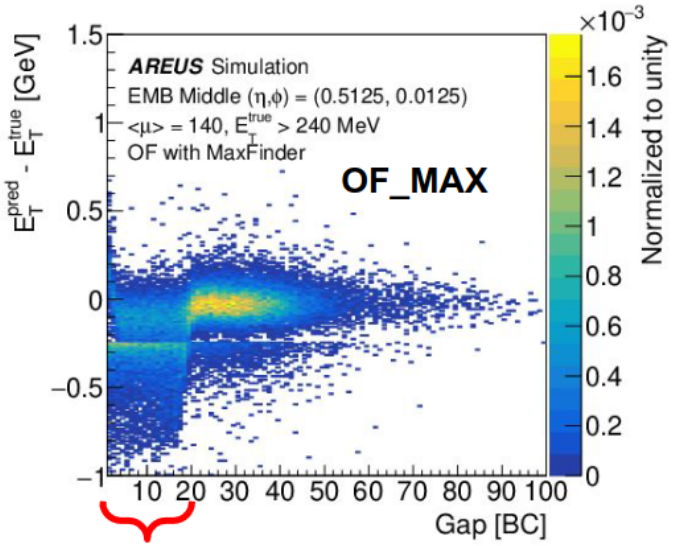
Sliding windows architecture

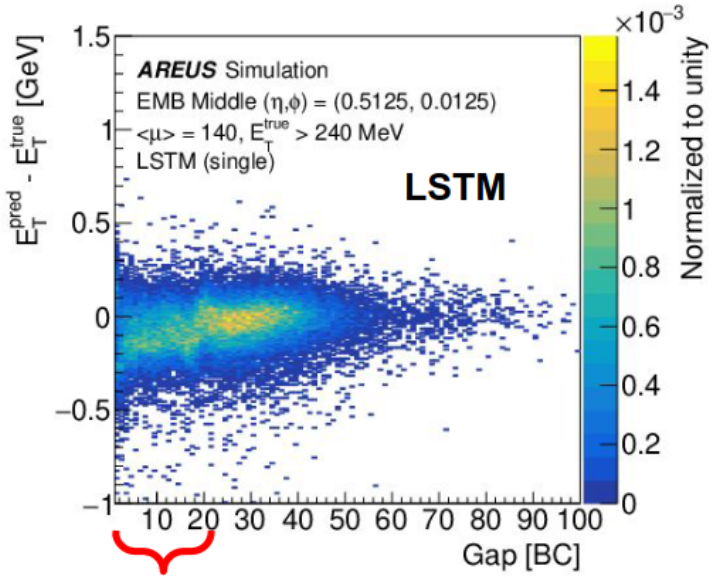
Computation on a moving slice of the data in fixed intervals

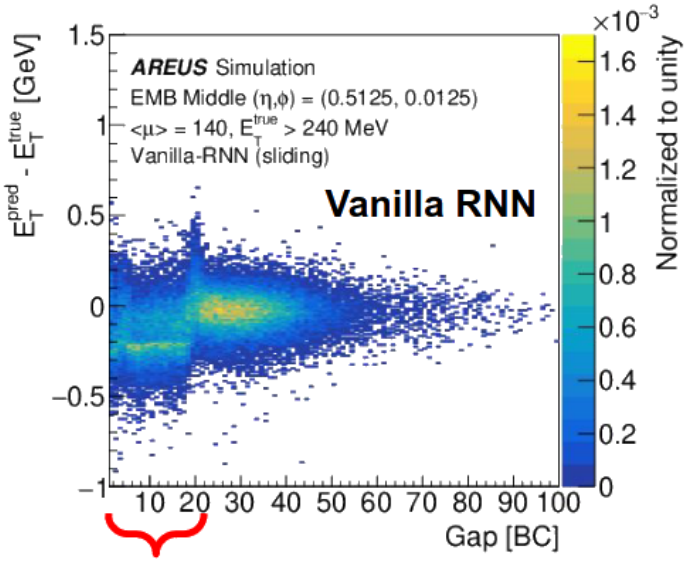
Takes into account a limited set in the past (1 sample in the past in this talk)

RNN performances

- LSTM with single cell recovers best the regions with small gap
  - Usage of full information from past events
- Vanilla RNN shows significant improvement with only 1 sample from the past



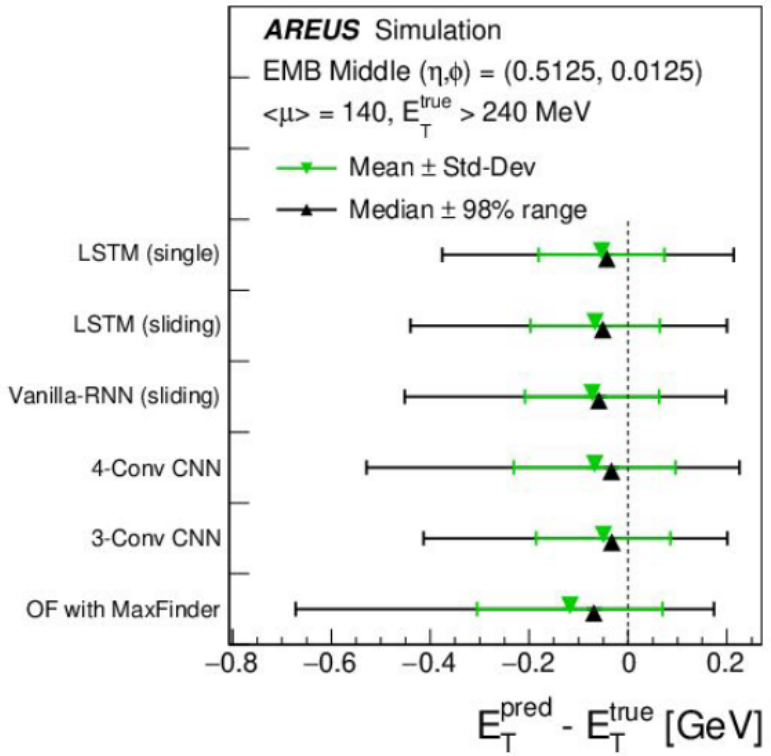




NN Results

Comput Softw Big Sci 5, 19 (2021)

- Overall better energy scale and resolution for all NNs with respect to OF_MAX
  - Lower tails as seen from the 98% median range
- All NNs optimized to reduce as much as possible the required computing resources
- LSTM perform best but have a larger number of parameters
  - Harder to fit in the FPGAs



Algorithm	LSTM (single)	LSTM (sliding)	Vanilla (sliding)	CNN (3-conv)	CNN (4-conv)	Optimal filtering
Number of parameters	491	491	89	94	88	5
MAC units	480	2360	368	87	78	5

Firmware implementation

Focus on Vanilla RNN implementation in HLS and VHDL

384 channels per FPGA

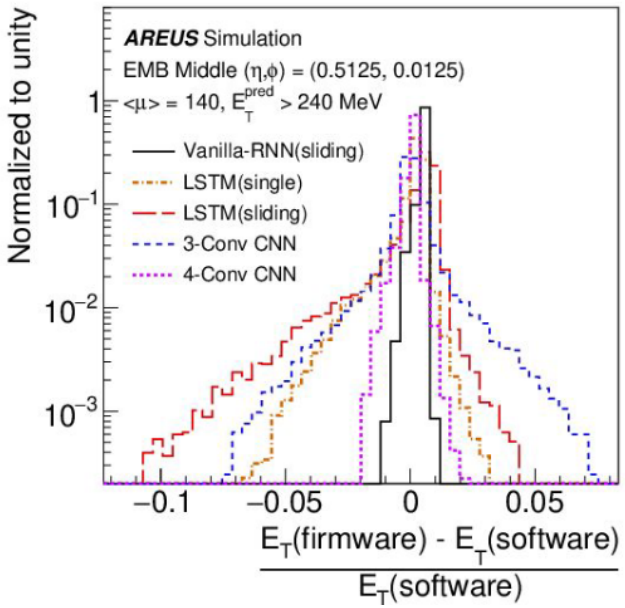
125 ns latency

Firmware implementation

Comput Softw Big Sci 5, 19 (2021)

- Implemented on Stratix 10 FPGA
  - reference 1SG280HU1F50E2VG
- CNN implemented in VHDL
- RNN implemented in HLS
- CNN and Vanilla RNN look feasible for the LAr usecase with time multiplexing

	3-Conv CNN	4-Conv CNN	Vanilla RNN (sliding)	LSTM (single)	LSTM (sliding)
Frequency F_{max} [MHz]	493	480	641	560	517
Latency clk_{core} cycles	62	58	206	220	363
Initiation interval clk_{core} cycles	1	1	1	220	1
Resource Usage					
#DSPs	46 0.8%	42 0.7%	34 0.6%	176 3.1%	738 12.8%
#ALMs	5684 0.6%	5702 0.6%	13115 1.4%	18079 1.9%	69892 7.5%



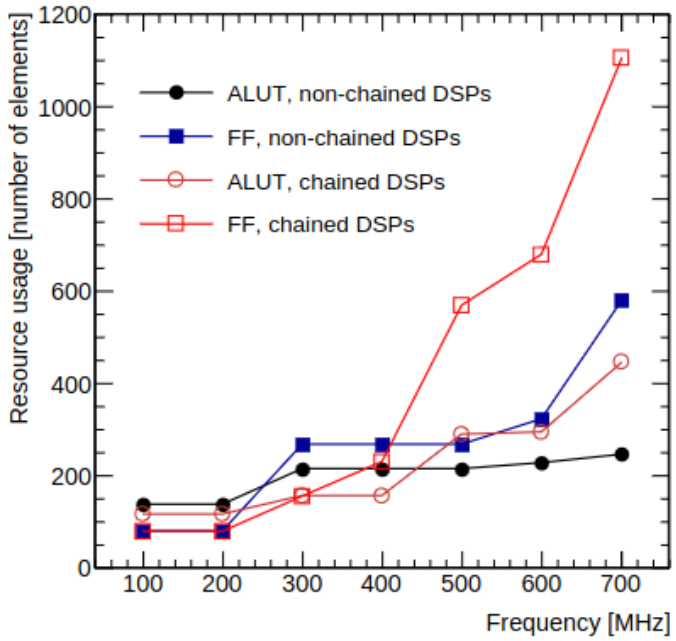
- Energy computed with quartus simulation and verified on target
- O(1%) resolution due to firmware approximations
  - LUT for activation functions
  - Fixed point arithmetics

HLS optimisations (Vanilla RNN) G. Aad et al 2023 JINST 18 P05017

- Optimisation needed to fit vanilla RNNs that can compute 384 channels within 125 ns
  - Multiplexed RNNs use more resources as we add more networks to the FPGA
- Several optimisations are performed
  - Activation functions in LUT
  - Number of bits in fixed point representation
  - Rounding and truncation in arithmetic operations
  - Implementation of vector/matrix multiplication (Dot product)

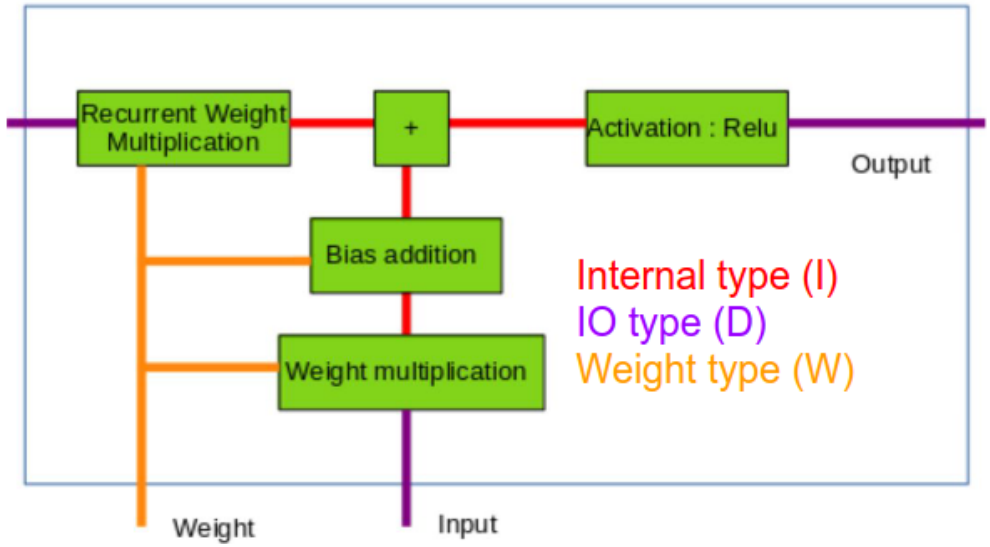
- Dot product implementations
  - Naive C++: let HLS do it all
  - Accumulate (sum) in DSPs by chaining them (ACC37)
  - Accumulate (sum) in ALUT (ACC19)

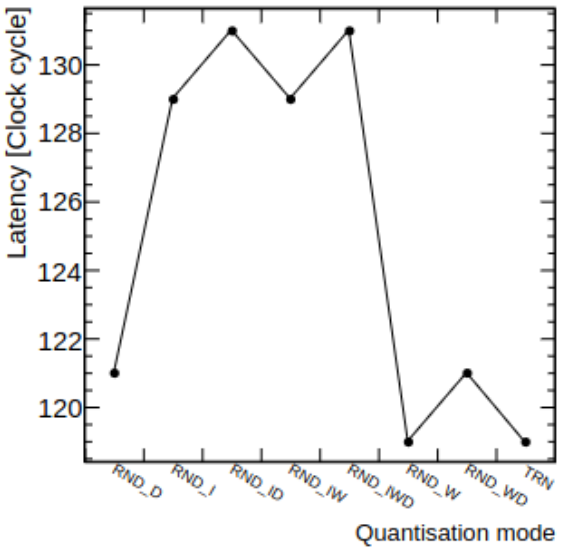
	Implementation	ALUTs	FF	DSP
@100 MHz	C++ style	709	222	8
	ACC37	116	79	4
	ACC19	137	78	4

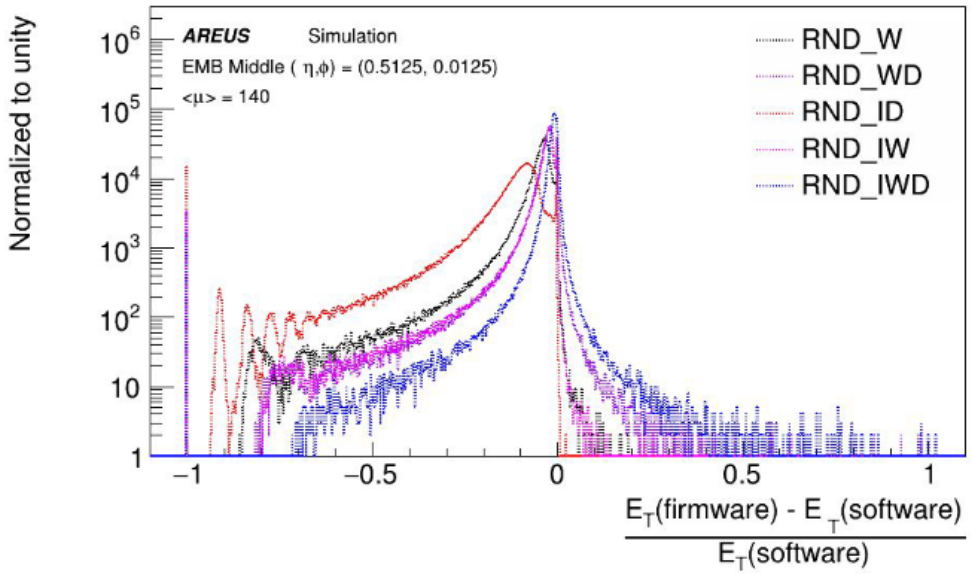


Rounding (Vanilla RNN)

- Compromise between energy resolution and resource usage and latency
  - Truncation of IO and Internal types leads to important reduction of latency with small impact on energy resolution
  - Weight type rounded in software (no impact on latency)



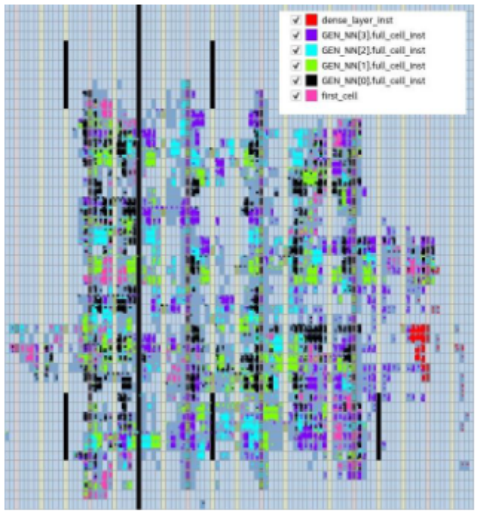




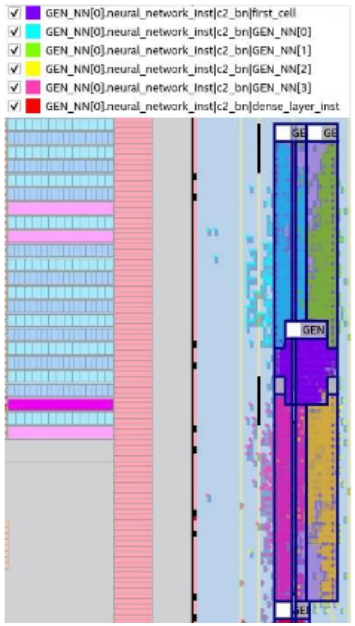
VHDL implementation of Vanilla RNN

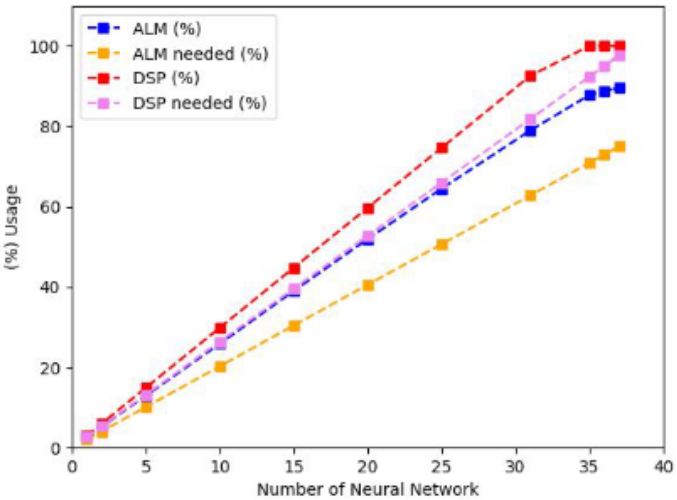
- HLS does not allow to reach the target frequency and resource usage
  - Increase of the RNN ALM resources and reduction of FMax as we add networks to the FPGA
- Move to VHDL for the final fine tuning
- Force placement of the RNN components
  - Allow to better tackle timing violations and improve FMax
- Use incremental compilation
  - Keep networks with no timing violations and recompile only the rest

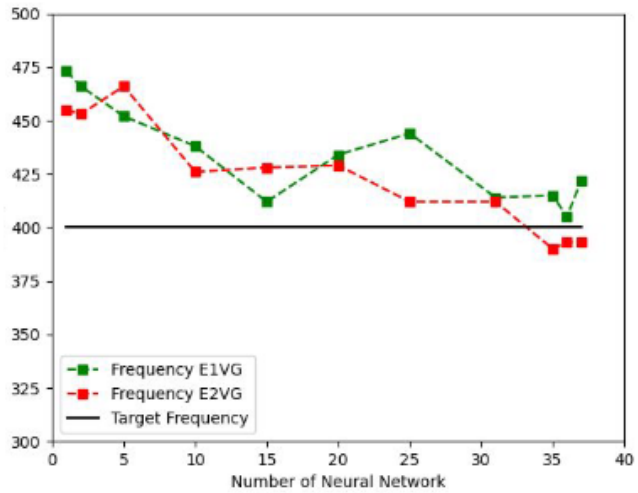
HLS placement



VHDL forced placement







RNN firmware results

- HLS allows fast development and optimisation of the firmware
  - Multiple developments and optimisations of RNN firmware in a short time
  - RNN for INTEL FPGAs implemented in HLS4ML for wider usage
- VHDL is needed to fine tune the design and fit the LAr requirements
- Vanilla RNN firmware produced and fit the requirements with Stratix 10
  - Better performance expected with the Agilex FPGA
  - However still need to test it within the full LASP firmware

	N networks x multiplexing	ALM	DSP	FMax	latency
target	**384 channels**	**30%***	**70%***	**-**	**125 ns**
HLS (no multiplexing)	384x1	226%	529%	-	322 ns
HLS optimized	37x10	23%	100%	414 MHz	302 ns
VHDL optimized	28x14	18%	66%	561 MHz	121 ns

*based on experience with the phase I upgrade

Conclusion

- CNN and RNN networks outperform the optimal filtering algorithm for the energy reconstruction in the ATLAS LAr Calorimeter
- All networks are designed to reduce to a maximum the resource usage while keeping the performance
- CNN and Vanilla RNN are serious candidates that can fit the stringent requirements on the LASP firmware
- Vanilla RNN and CNN optimisations allow to reach the requirements in terms of resource usage and latency
  - Testing on hardware (INTEL DevKits and LASP prototypes) started and shows good results
  - Need to check timing violations with all other components of the LASP firmware
  - Next step is to quantify the effect on object (electrons, photons) reconstruction and physics performance

Opening the way for AI usage to process raw detector data online and at trigger level

Any questions ?