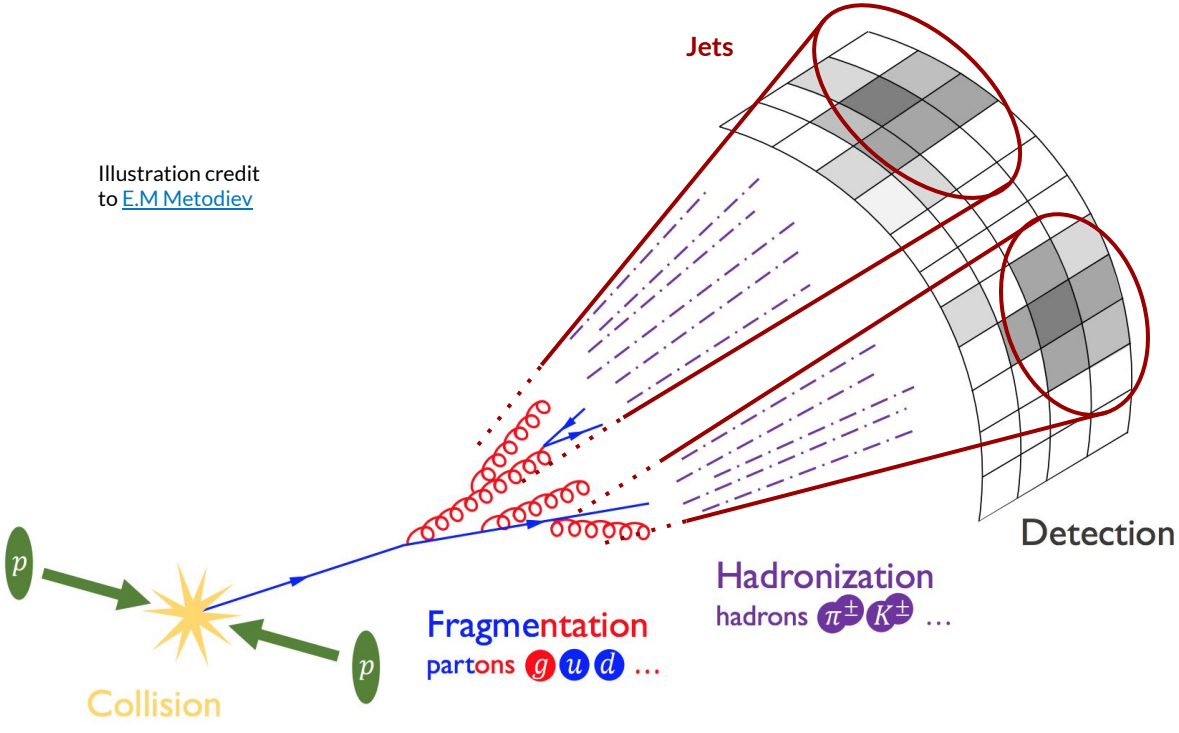
Prototype of Machine Learning for Pion Reconstruction in L0 Global Trigger FPGA for ATLAS HL-LHC

CAP 2026 – Ottawa
2026-06-23

Kelvin Leong

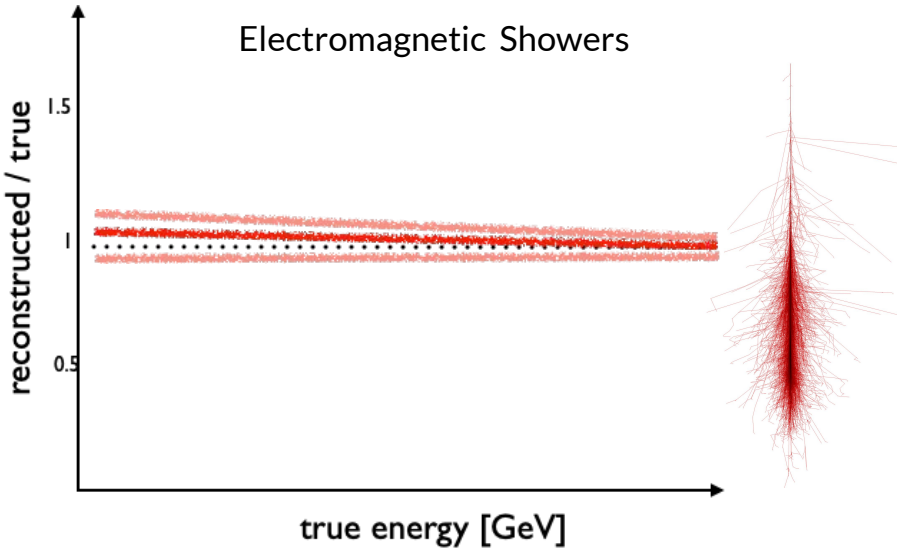
Jets

Illustration credit to E.M Metodiev



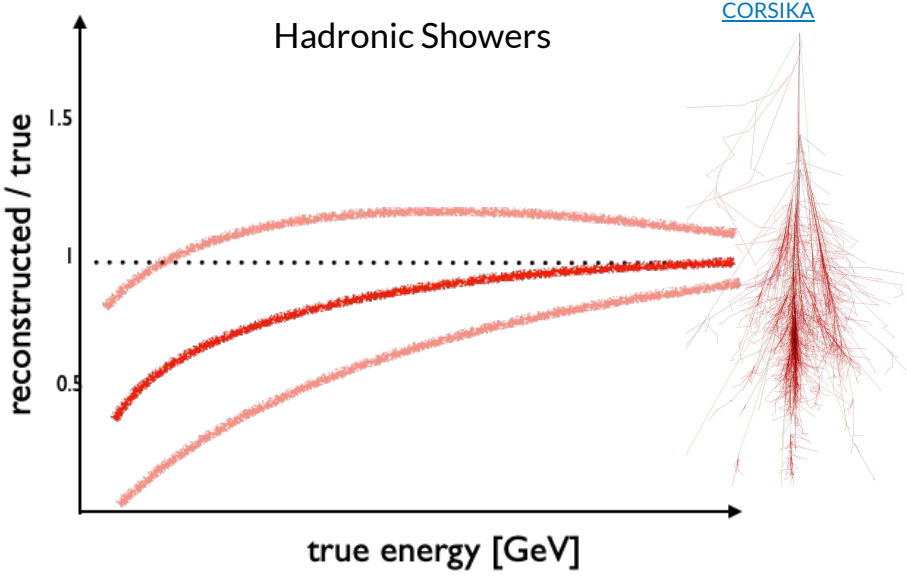
- Jets = a collimated sprays of hadrons, initiated by high-energy parton(s)
  - Important for e.g. Higgs studies ($H \rightarrow bb$)
- Initiate showers when jets interacts with matter
  - Electromagnetic (EM) shower
  - Hadronic shower
- Use trackers and calorimeters for jet detection
  - Fast triggers with calorimeters

Calorimeters & Showers



- EM showers (e, γ, $\pi^0 \rightarrow 2\gamma$) are well-calibrated, measured "correctly".
- All showers are similar, resolution is good.

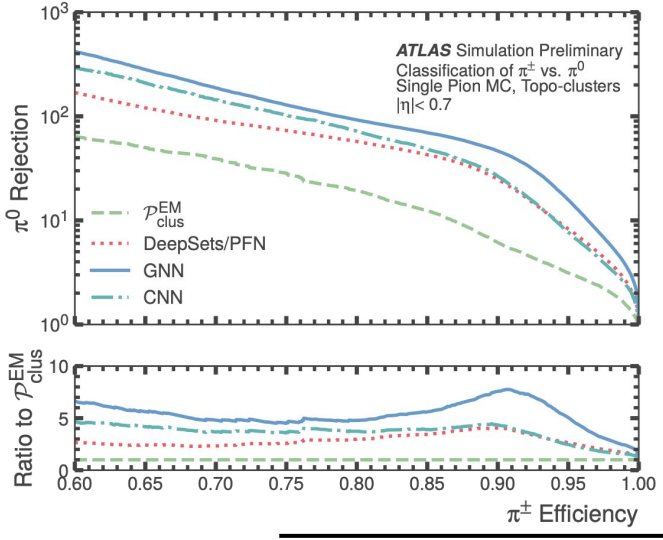
Simulations from CORSIKA

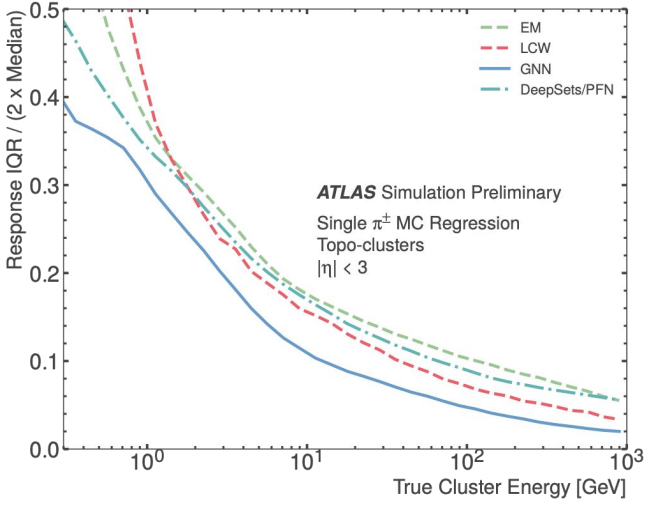


- Hadronic showers (p, n, $\pi^{\pm}$) are more difficult to calibrate
- Each shower is unique, huge resolution penalty from variations.

Machine Learning for Offline Shower Types Identification & Calorimeter Calibration

- Pions ($\pi^0, \pi^\pm$) are most abundant hadrons produced in collisions in LHC (~75%)
- ML methods (e.g. GNN, DeepSets, …) can outperform current algorithms for identifying shower types and calibrating energy in the calorimeter in offline
  - ATL-PHYS-PUB-2020-018
  - ATL-PHYS-PUB-2022-040

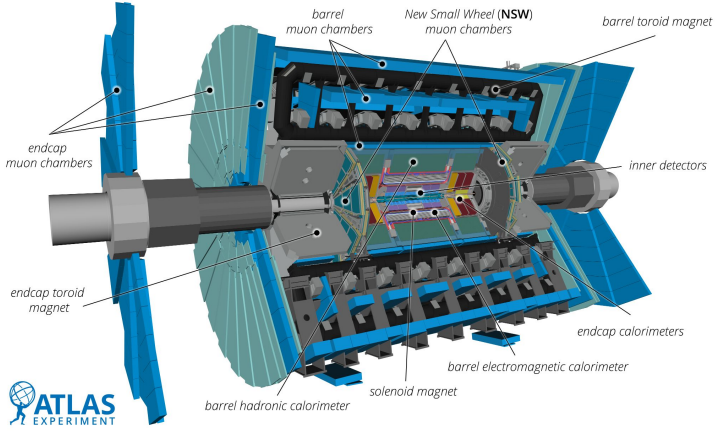


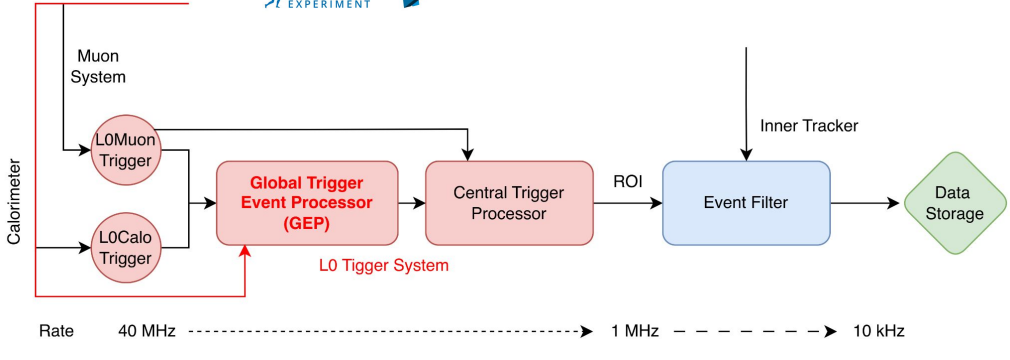


Can ML be applied to online trigger?

- Within 10μs L0 Global Trigger (40MHz → 1MHz):
  - Intakes GEP-cells from the calorimeters
  - → Forms topoclusters
  - → Forms calorimeter jets
  - → Makes decision based on jet p_T, E_T^{miss}, etc
- Local calibration of topoclusters (with single pion simulation) improves jet calibration & trigger efficiency
- **Can we apply ML in the FPGA within the L0 global trigger event processor for pion calibration?**

ATLAS-PHOTO-2022-055

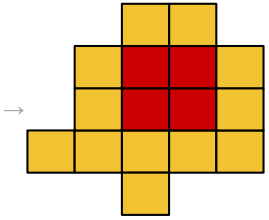




HL-LHC:

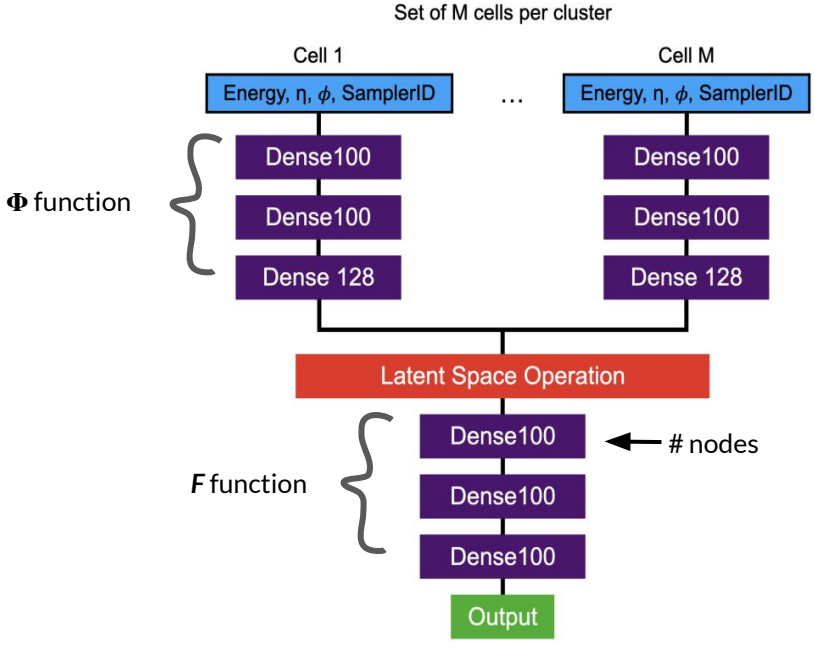
adapted from ATLAS-TDR-029-ADD-1

Topocluster = group of connected cells →

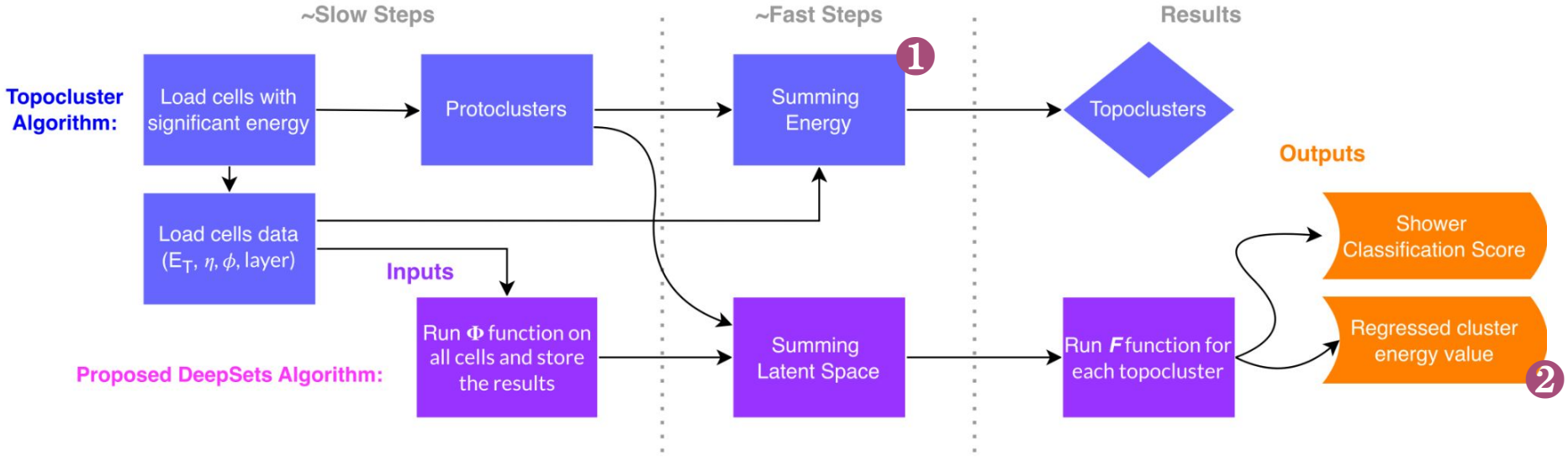


DeepSets Neural Network

- Which ML architecture should we choose?
- DeepSets: (3 stage model)
  - Run a **Φ** network (standard NN) on every cell
  - Sum all the results to “latent space”
  - Run a final ***F*** network, get outputs (classification & regression)
  - Relationships of cells are encoded in latent space
- Advantages:
  - Simpler to implement (compared to GNN, CNN)
  - Operation can start before decision of topocluster-formation arrives



Proposal:
DeepSets under the shadow of Topo-Clustering

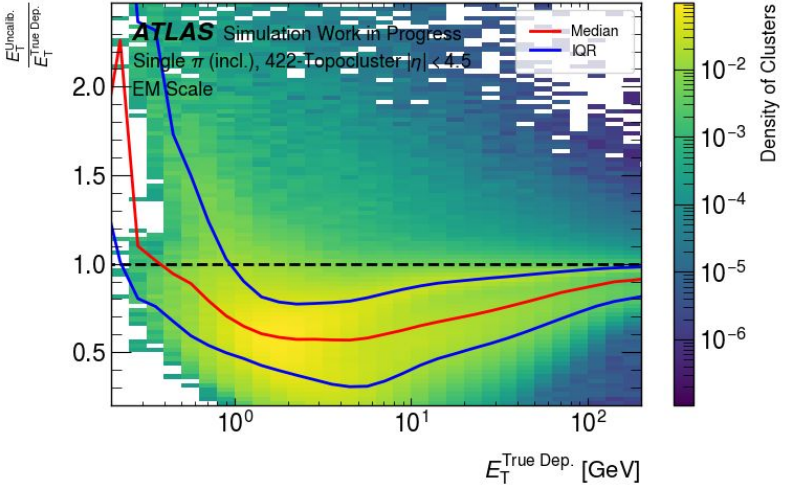


*Using cell transverse energy E_T as input, as this is the output format of GEP cells from calorimeter upstream

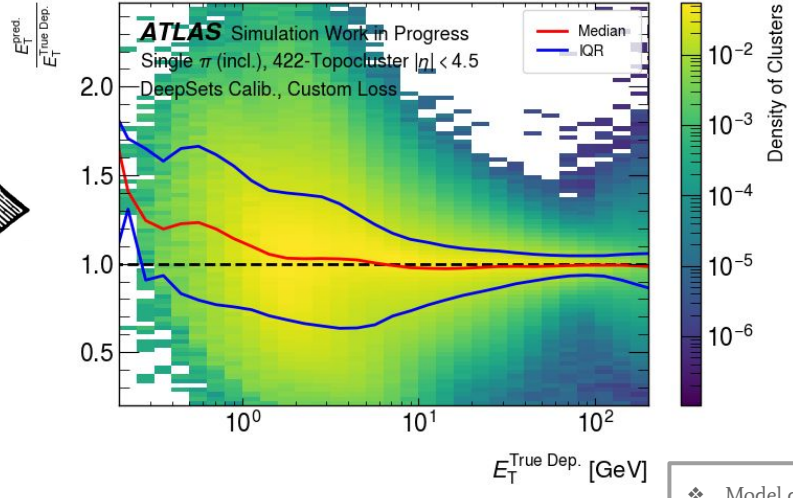
Neural network Performance

- Run 4 single pion 422-Topocluster simulation
  - True deposited cluster energy = energy within the 422-Topocluster definition + out-of-cluster energy
  - Simulate quantized cell E_T input from calorimeter electronics
  - Custom loss function, ~6.5M (2.1M) clusters for training (testing)

❶ **Before Calibration:**



❷ **After ML Calibration:**



❖ Median response matches true deposit cluster energy for E_T > 1 GeV

➢ Red (blue) line=Median (central 68% interval) response of test samples

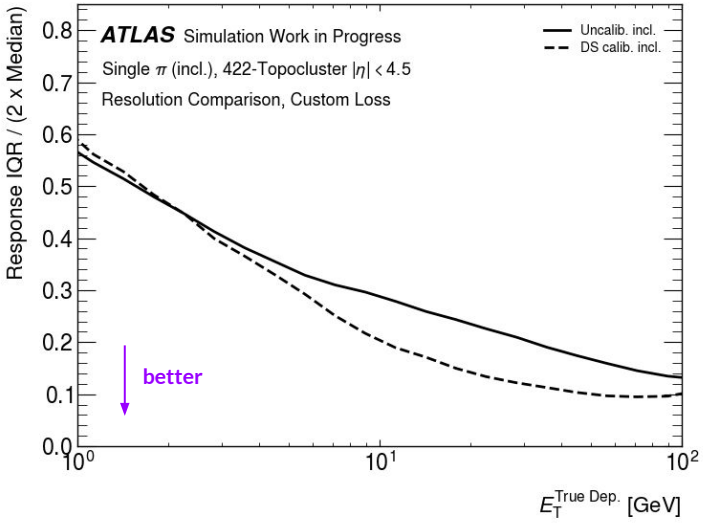
❖ Model details:

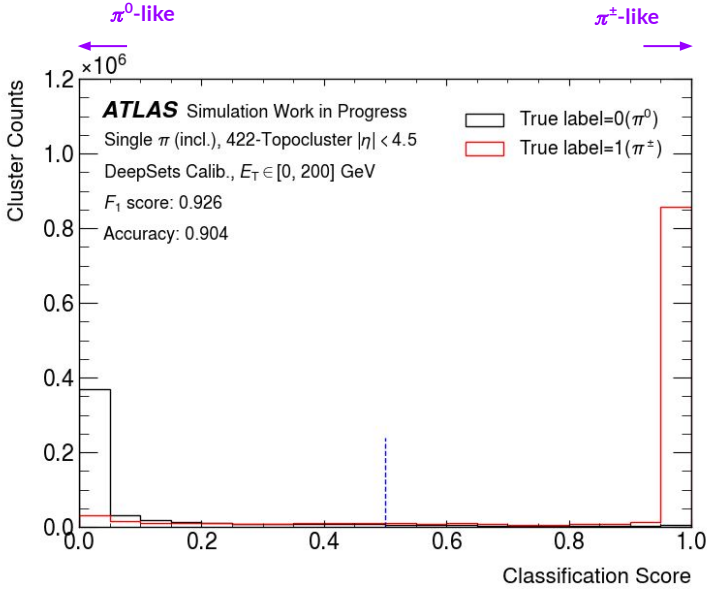
➢ Φ nodes: {25, 9}

➢ F nodes: {30, 40, 25}

➢ Total parameters: 3032

Neural network Performance

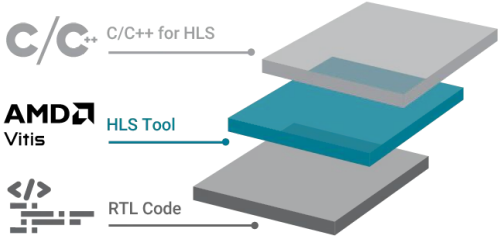




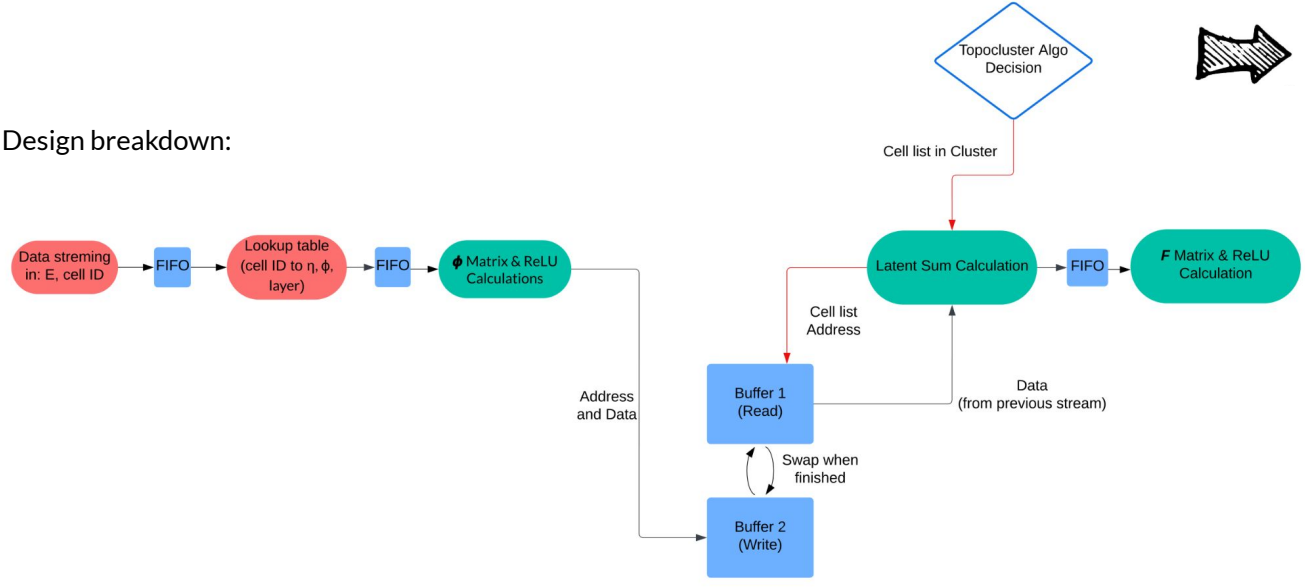
- Significant resolution improvement for $E_T \in [5, 100]$ GeV
- Good separation power of clusters originating from π^0 or $\pi^\pm$
- All achievable with this simple architecture & small parameters numbers
  - *This study excludes clusters with less than 4 cells

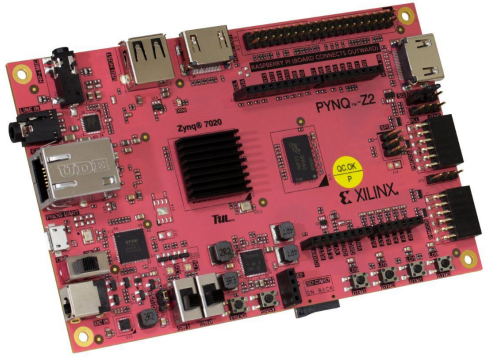
Implementation in FPGA

- Preliminary target chip: VP1802, running at 240 MHz
- Our algorithm will get ~1/20 resource (about the size of Zynq chip on PYNQ-Z2 board) & ~2μs if deploy
  - Using PYNQ board for prototype, running at 100 MHz
  - Writing NN with Vitis high-level synthesis



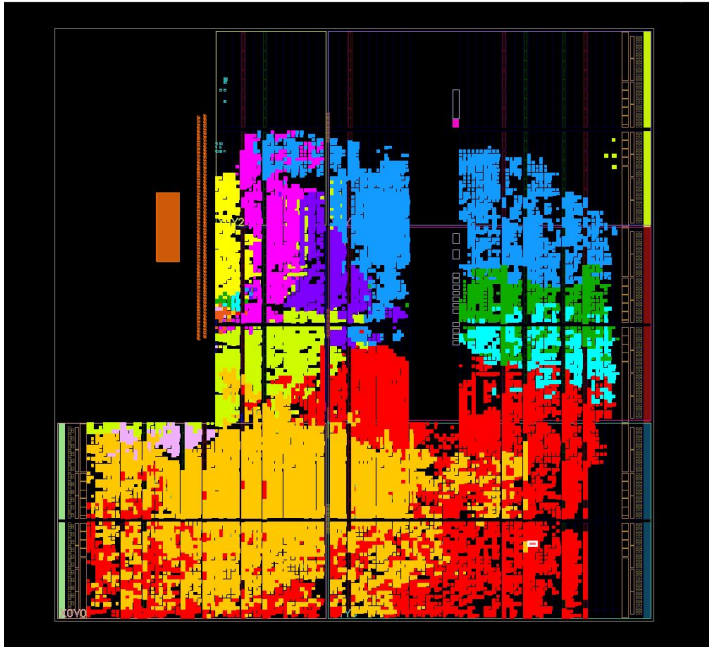
Design breakdown:





Implementation in FPGA

- Prototyping using AMD tools and PYNQ-Z2 board
  - Vivado:



Design implementation on the PYNQ board

Magenta, purple, pink: Generators & Result Storage

Blue: Φ network

Green: Latent Buffer

Teal: Latent Sum

Red, orange: *F* network

Yellow, lime green: Logic interconnect

Terracotta: System control

Design usage on the PYNQ board:

	PYNQ-Z2 limit	Φ net usage (%)	F net usage (%)	Design total usage (%)
LUTs	53200	2247 (4.2%)	9264 (17.4%)	14971 (27.7%)
FFs	106400	4197 (3.9%)	18465 (17.4%)	27410 (25.4%)
BRAMs	140	7 (5%)	48 (34.3%)	64 (45.7%)
DSPs	220	13 (5.9%)	93 (42.3%)	106 (48.2%)

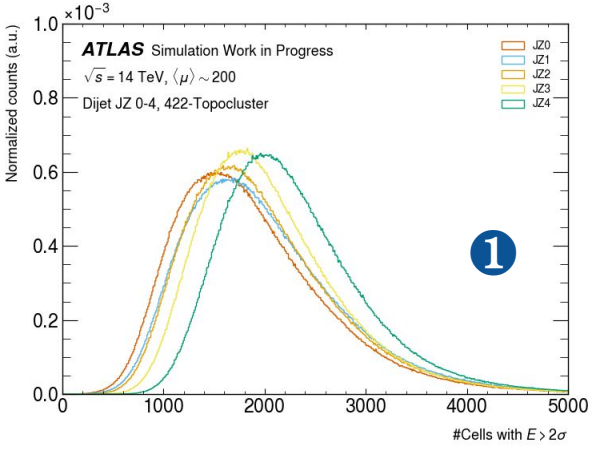
Latency (at 100 MHz):

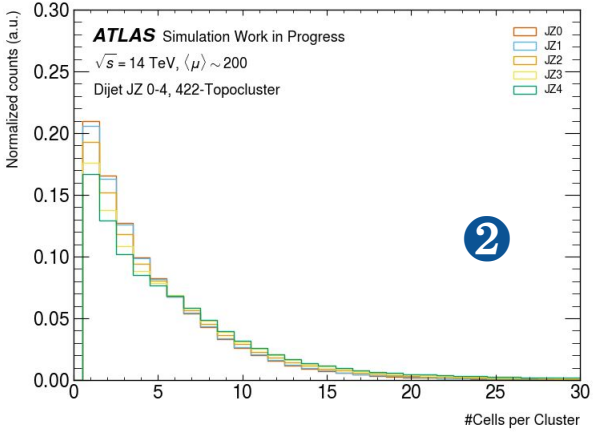
- **Φ** net: avg 43 clock ticks (=430 ns)
- ***F*** net: 89 clock ticks (=890 ns) for first cluster, and finish processing all consecutive clusters every 48 clock ticks (=480 ns)

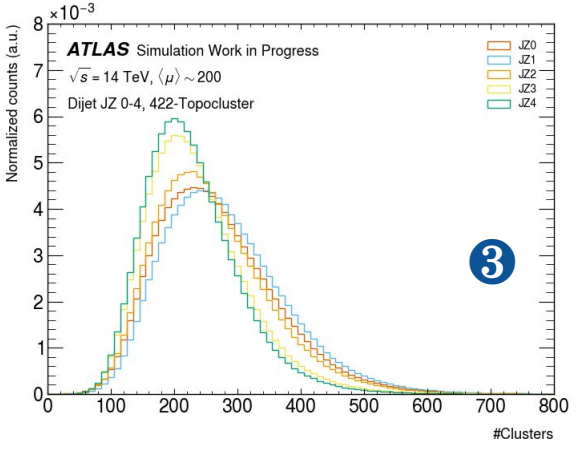
Note: We have only placed 1 Φ net & 1 *F* net on this design!

A more realistic environment

- Using dijet samples to simulate the more realistic environment expected from calorimeter







- The 95th percentile of moderate-energy ($p_T \in [400, 800]$ GeV) jets events has:
  1. 3656 cells that **Φ** net needs to process ← **can place 10 Φ net in design, process ~50 cells at 240 MHz within ~1μs now**
  2. 27 cells per cluster that latent sum needs to process
  3. 371 clusters that ***F*** net needs to process ← **can place 2 *F* net in design, process ~6 clusters at 240 MHz within ~1μs now**
- Still a few improvements needed for implementation to handle rates in L0 Global
  - A few clear directions for future works

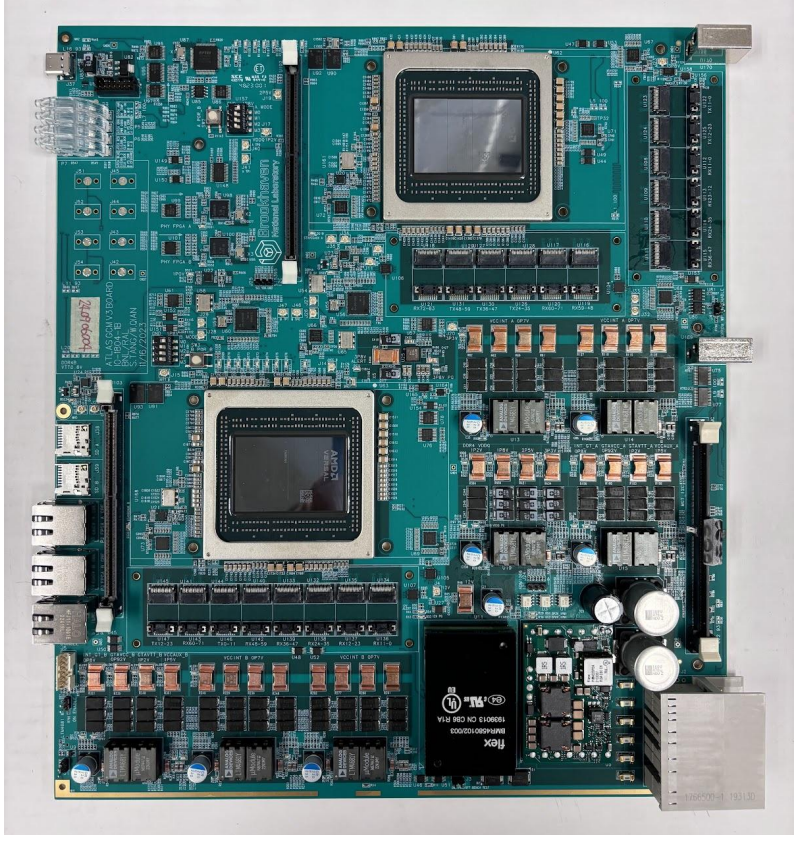
Conclusion

- **Problem trying to solve:**
  - Improve calorimeter calibration for online trigger during the HL-LHC
- **Why ML on pions?**
  - Pions ~75% final state; ML performs better than current algorithms in pion type classification and energy calibration
- **Results:**
  - **Successful prototype DeepSets implementation on FPGA!**
- **Challenges ahead:**
  - Large number of cells and clusters per event, limited resources & latency for parallelization
- **Future Steps:**
  - Test design on a development board VPK180 (containing a VP1802 FPGA) at 240 MHz
  - Continue optimizing the design
  - Study effects of our topocluster calibration scheme on jet calibration in L0 Global

Backup

ATLAS L0 Global Preliminary Target Board

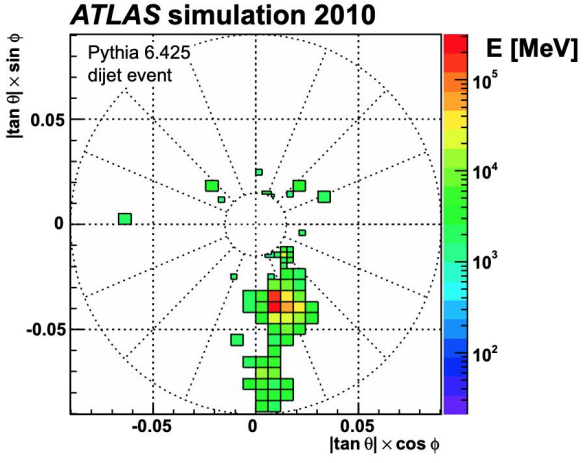
- With two VP1802 FPGA:

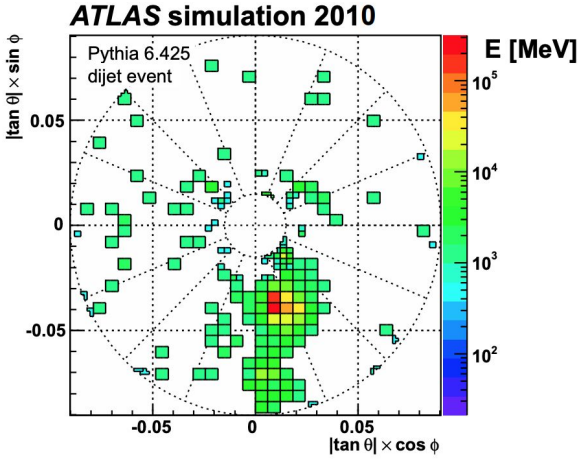


Topo-Clustering for HL-LHC in Lo Global

- 422-Topocluster definition:

arXiv:1603.02934





❶ Initial seed collection

- All cells above seed threshold:

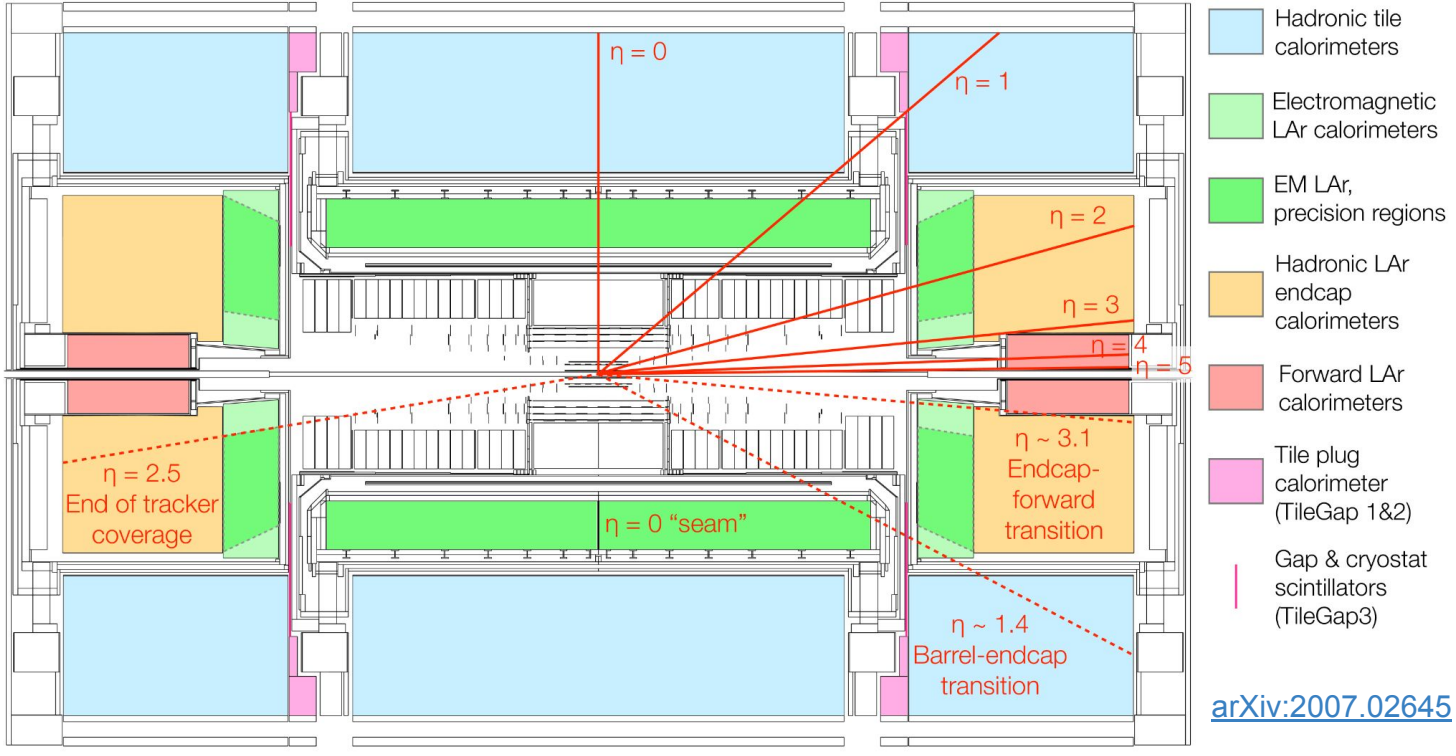
$$\left|E_{\text{cell}}^{\text{EM}}\right| / \sigma_{\text{noise,cell}}^{\text{EM}} > 4$$

❷ Cells above growth control threshold collection

- Cells above growth threshold:

$$\left|E_{\text{cell}}^{\text{EM}}\right| / \sigma_{\text{noise,cell}}^{\text{EM}} > 2$$

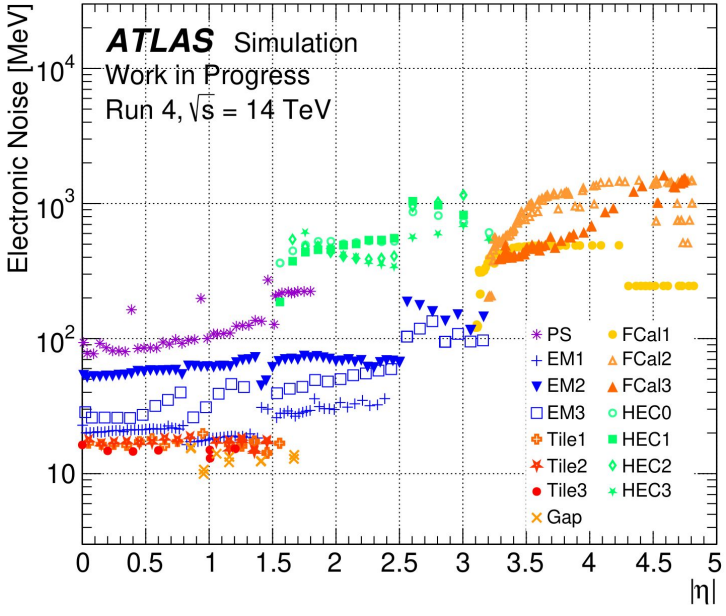
Geometry Reference

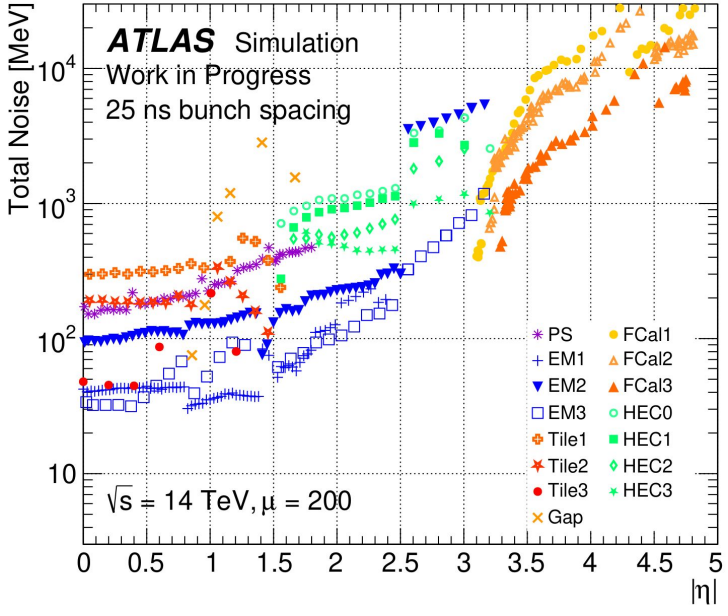


arXiv:2007.02645

Noises in Calorimeter

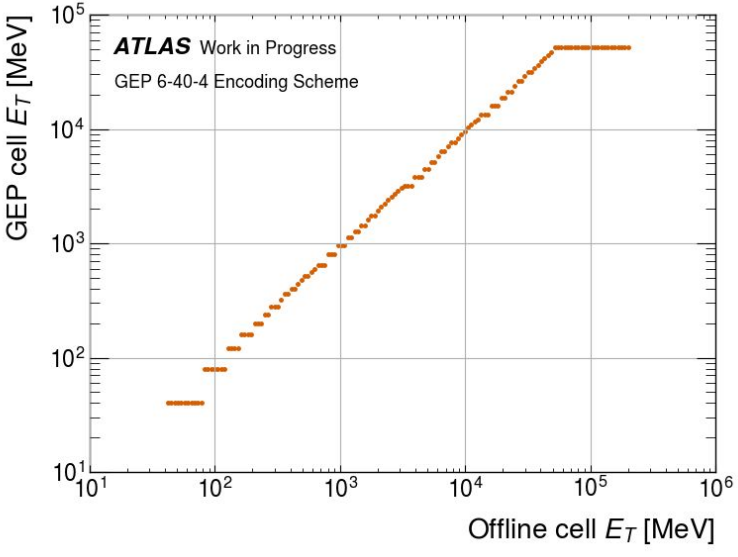
$$\sigma_{\text{noise}} = \sqrt{\left(\sigma_{\text{noise}}^{\text{electronic}}\right)^2 + \left(\sigma_{\text{noise}}^{\text{pile-up}}\right)^2}$$





Global Event Processor Multilinear Encoding

- 6-40-4 encoding scheme:
  - Total number of bits: 6
  - Least significant bit (minimum encoded energy): 40 MeV
  - Gain factor: 4



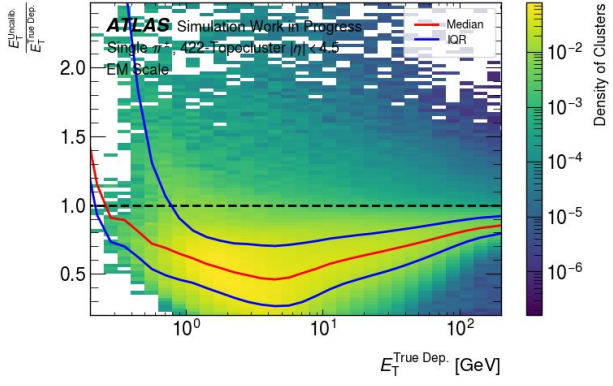
Readout range bases	Step sizes
0	40 MeV
640 MeV	160 MeV
3.2 GeV	640 MeV
13.44 GeV	2.56 GeV

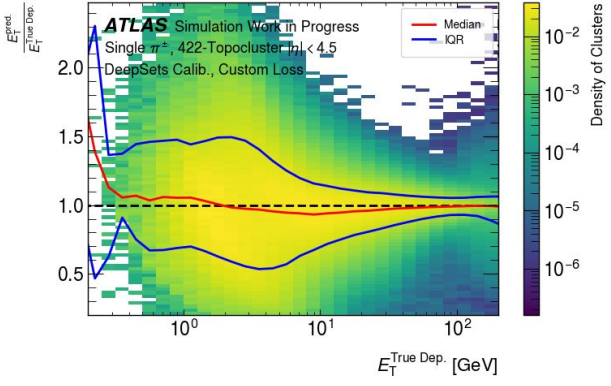
- Cell E_T saturates at 51.84 GeV
  - Affects the resolution achievable by the NN at high E_T

Regional Plots from Fitting

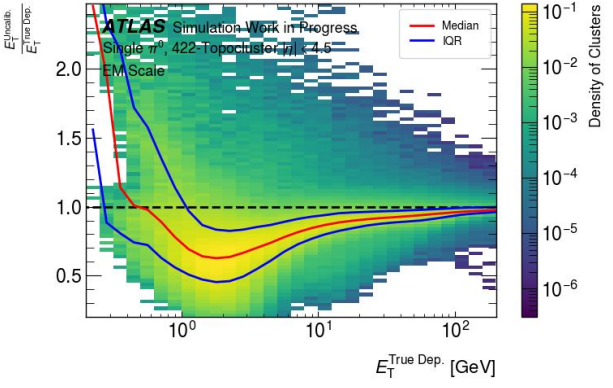
$$\text{Custom Loss} = \begin{cases} \frac{|y-\hat{y}|}{\hat{y}} + \frac{(y-\hat{y})^2}{\hat{y}^4} & , (y-\hat{y}) > 0 \\ \frac{|y-\hat{y}|}{\hat{y}} + \beta * \frac{1}{2}\left(\text{erf}\left(\frac{\hat{y}-a}{s_1}\right) - \text{erf}\left(\frac{\hat{y}-b}{s_2}\right)\right) * (y-\hat{y})^2 & , (y-\hat{y}) < 0 \end{cases}$$

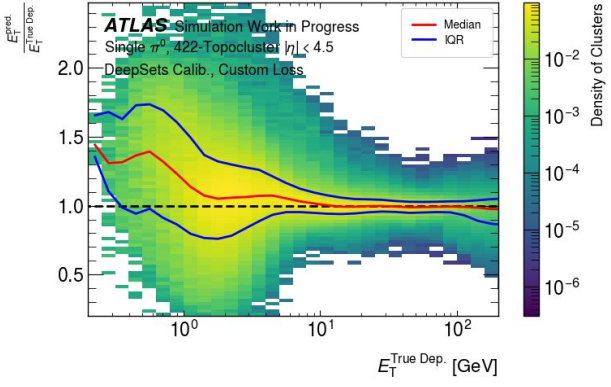
$\pi^{\pm}$:





π^{0}:

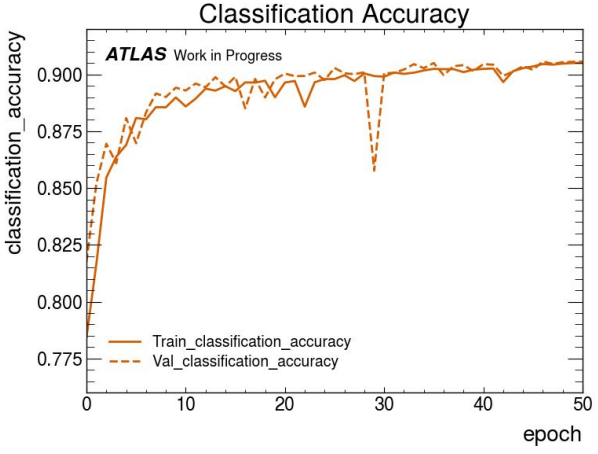


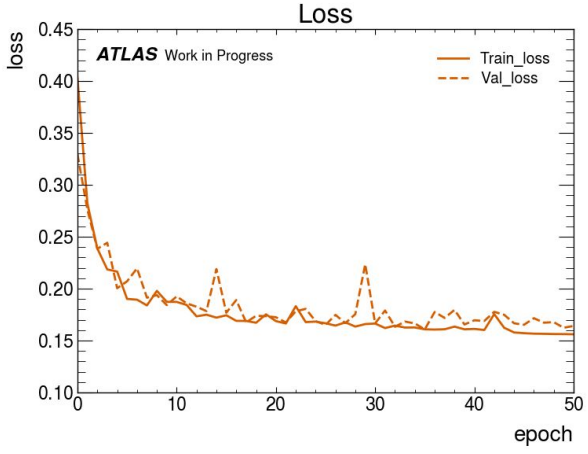


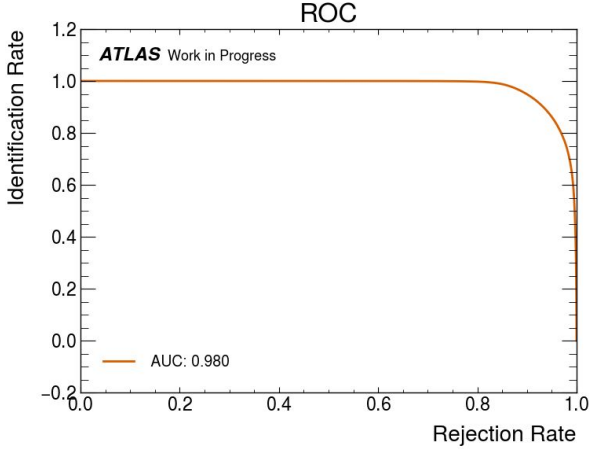
- Training details:
  - ~6.5M clusters with π^0 & $\pi^{\pm}$ labels (mc21)
  - Epochs: 100
  - Learning rate: 0.001
  - EarlyStopping: 15 epochs
  - Loss weights:
    - Regression: 65%
    - Classification: 35%
- Model details:
  - $\boldsymbol{\Phi}$ nodes: {25, 9}
  - $\boldsymbol{F}$ nodes: {30, 40, 25}
  - Total parameters: 3032

Accuracy, losses, ROC plots

$$\text{Custom Loss} = \begin{cases} \frac{|y-\hat{y}|}{\hat{y}} + \frac{(y-\hat{y})^2}{\hat{y}^4} & ,(y-\hat{y}) > 0 \\ \frac{|y-\hat{y}|}{\hat{y}} + \beta * \frac{1}{2}\left(\text{erf}\left(\frac{\hat{y}-a}{s_1}\right) - \text{erf}\left(\frac{\hat{y}-b}{s_2}\right)\right) * (y-\hat{y})^2 & ,(y-\hat{y}) < 0 \end{cases}$$

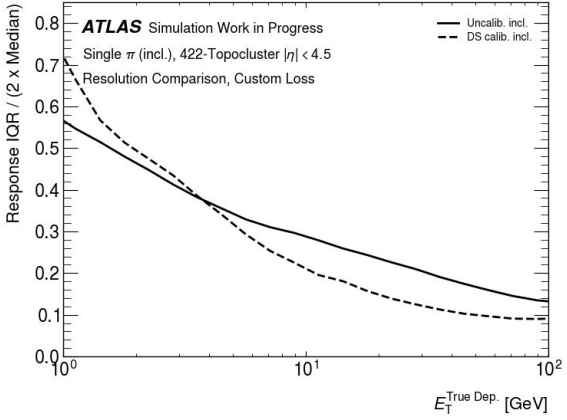


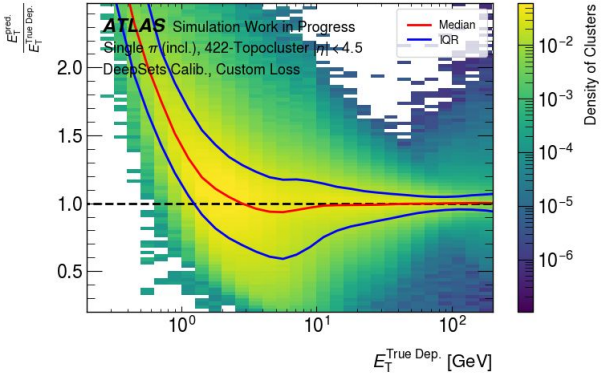


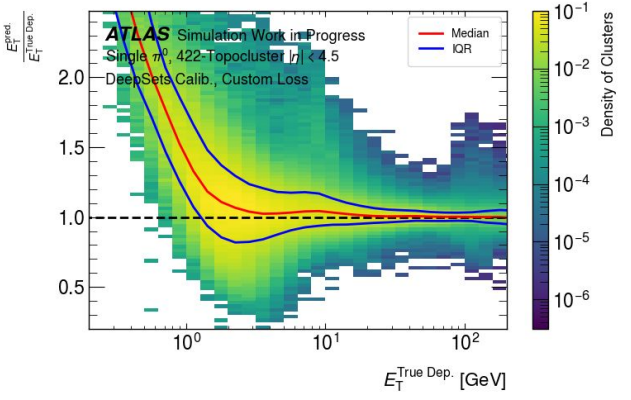


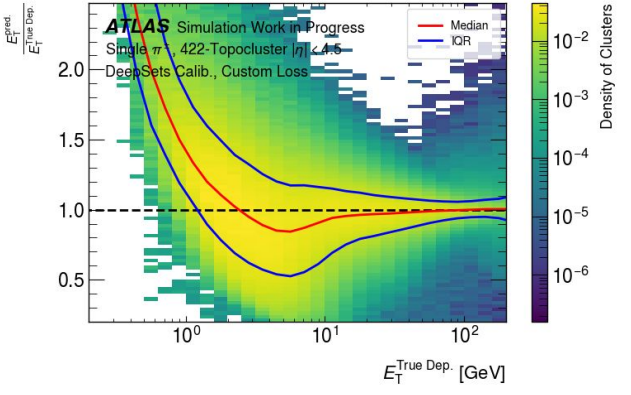
Other loss functions

L1 loss: $|y - \hat{y}|$



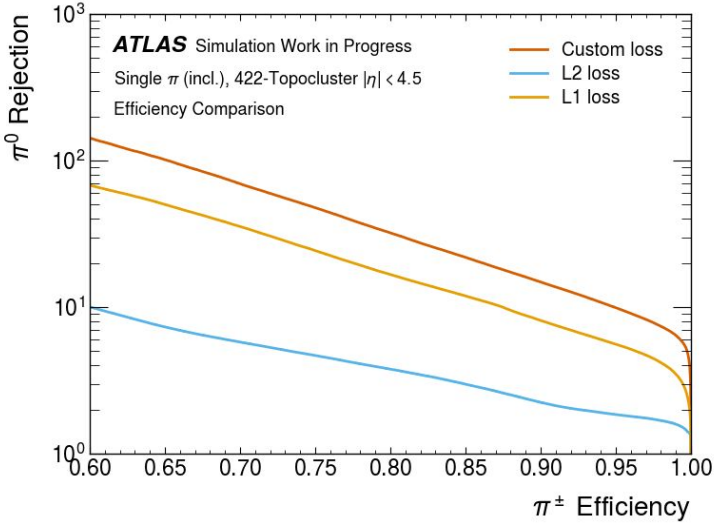


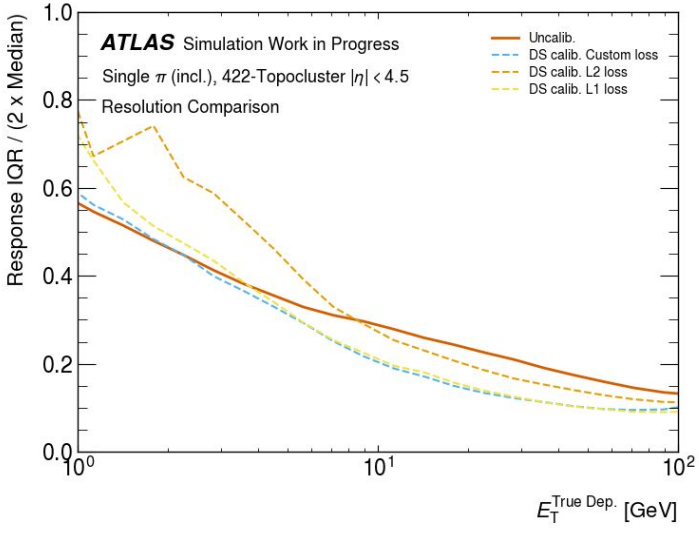




- Similar resolution compared to the custom loss model, but prediction incorrect for low-energy

Other loss functions

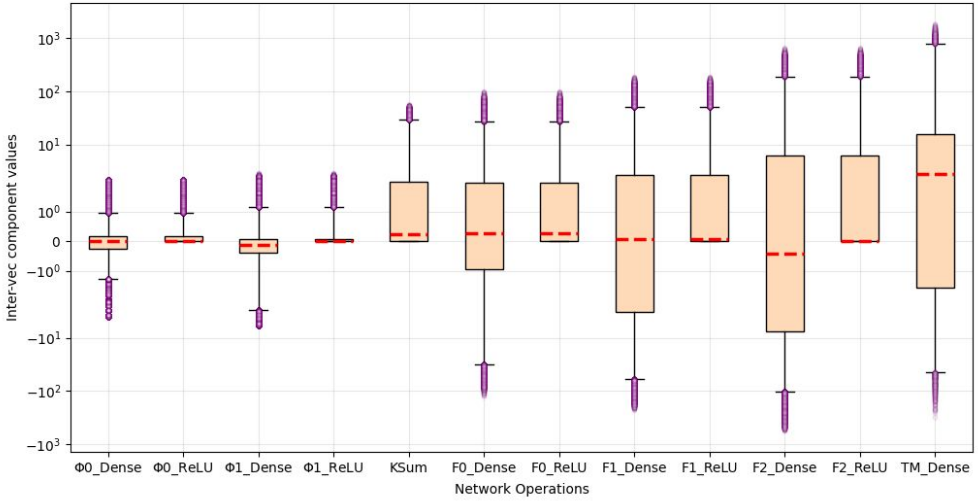




Post-Quantization Study

$S < 16, 6 > : \underline{101101.1010000000}_2 = -18.375_{10}$

- Network weights and biases (w&b) require 5 bits to encode integer part

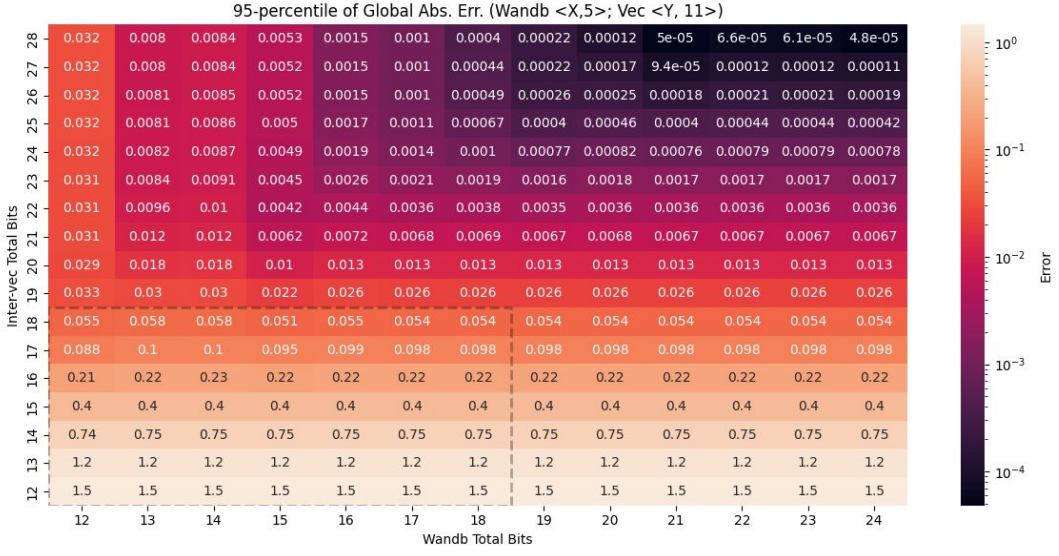


- Testing trained network with 30k clusters
- Inter-layer vector components at absolute maximum require 11 bits to encode integer part

Post-Quantization Study

S < 16, 6 > : $101101.1010000000_2 = -18.375_{10}$

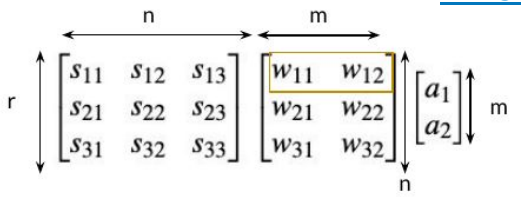
- 95th-percentile of all absolute errors (|(floating point) - (fixed point)| of inter-layer vector components) from test data:



- Outside the gray dashed box, # of DSPs required in design would double
- Decided to use <18,5> for w&b, <18,11> for inter-vec

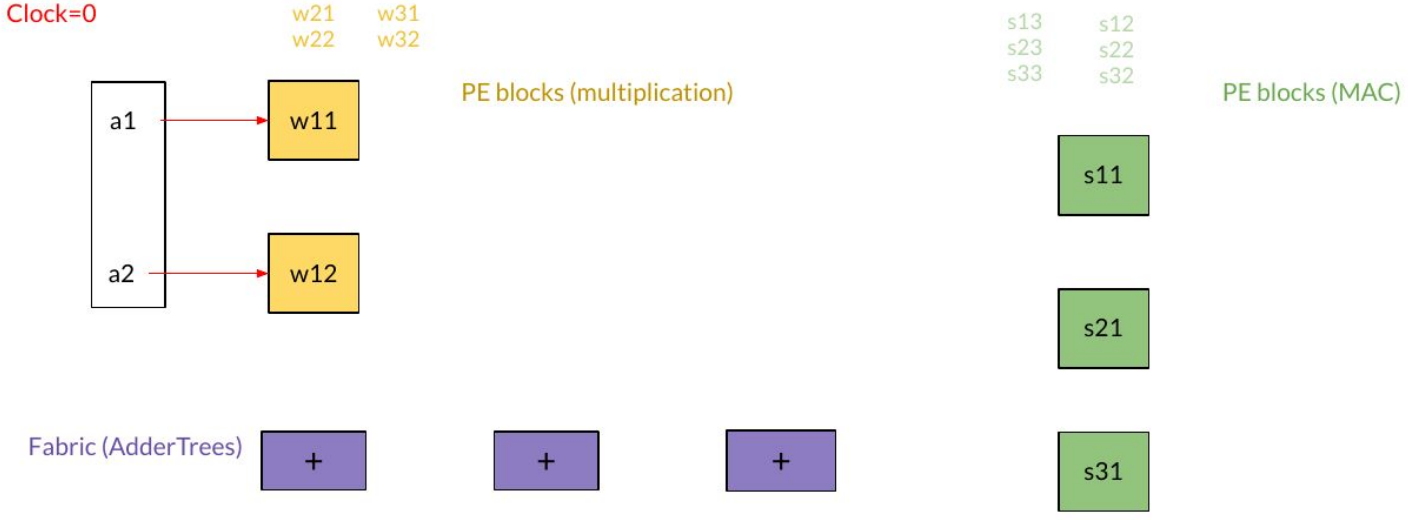
Pipeline matrix-vector calculation

MEISSA paper



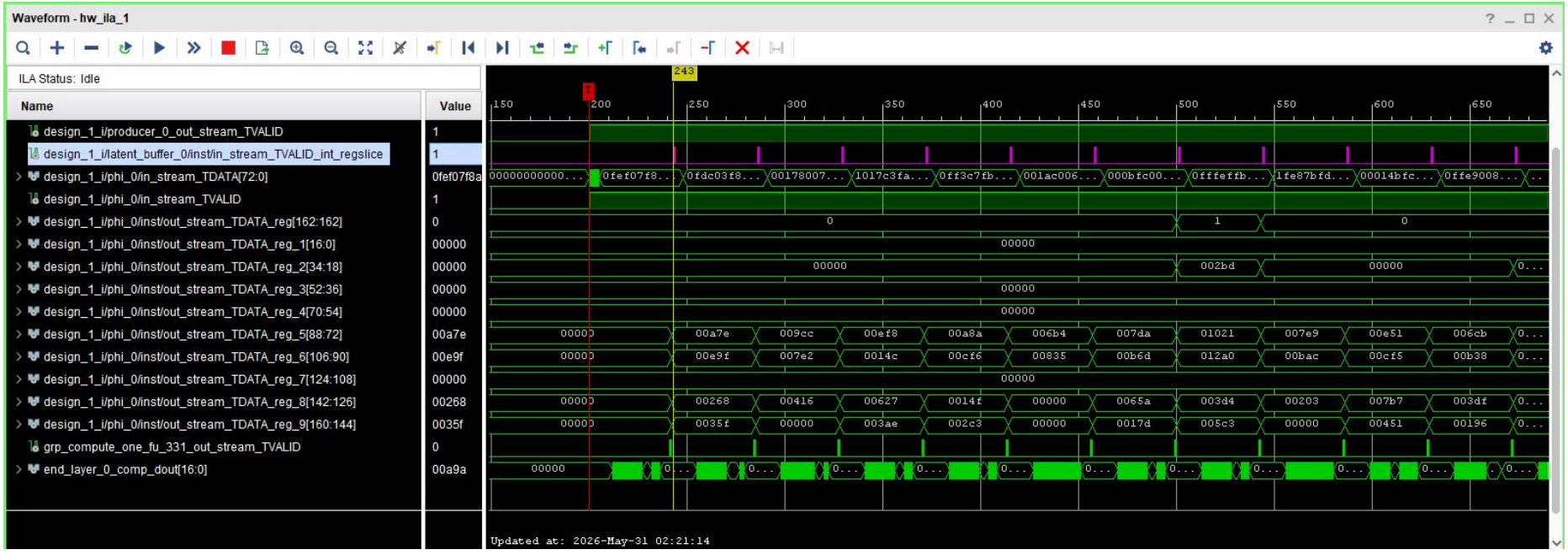
- Meissa + TMac Architecture:
  - 1st layer (Meissa)
    - Set up PEs on cols instead of rows (save DSPs if m < n)
    - Separate accumulation out from PEs → less routing needed
  - 2nd layer (TMac)
    - Use partial results from first layer to do transpose multiplication and addition → Don't need to wait

Gif:



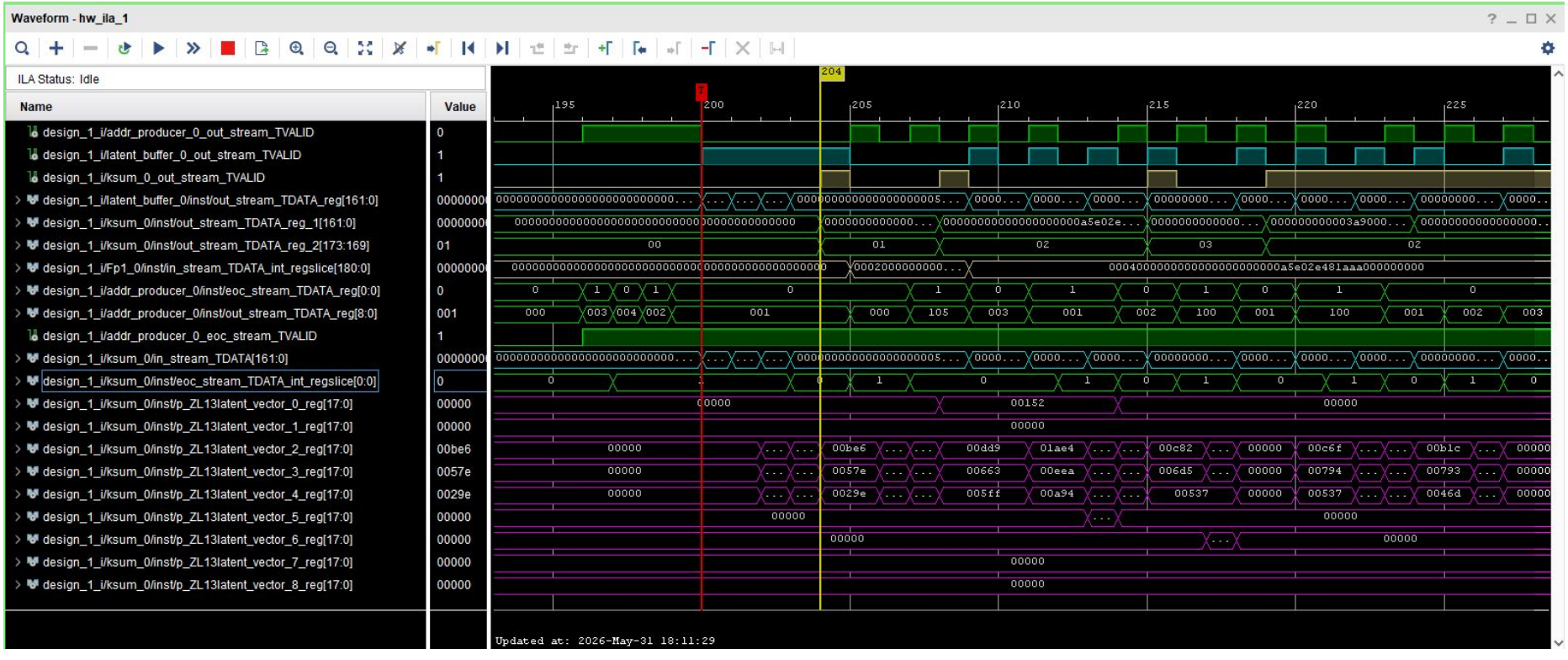
Waveform

- **Φ** network:



Waveform

- Latent Sum:



Waveform

- ***F*** network: