Implementing Neural Network in GRL

Riku Nomaru
University of Tokyo
2023/6/2

Presentation flow

① Introduction of hls4ml

② make a neural network model

③ evaluate the performance of model

④ implement in FPGA (GRL)

Introduction of hls4ml

What is hls4ml (high level synthesis for machine learning)

hls4ml is a library from CERN that automatically converts neural network models written in Python to FPGA firmware.

Build and train machine learning models using Keras,TensorFlow,PyTorch…

Convert the trained model to HLS project(c++)

perform higher-level synthesis and
Create IP in Vivado HLS

implement in FPGA

Introduction of hls4ml

Good point of hls4ml

- It is easy to build a model because we can use well-known open source such as Pytorch and Keras

- Once the model is created and trained, the rest process is done almost automatically.

- Synthesis time is comparatively short

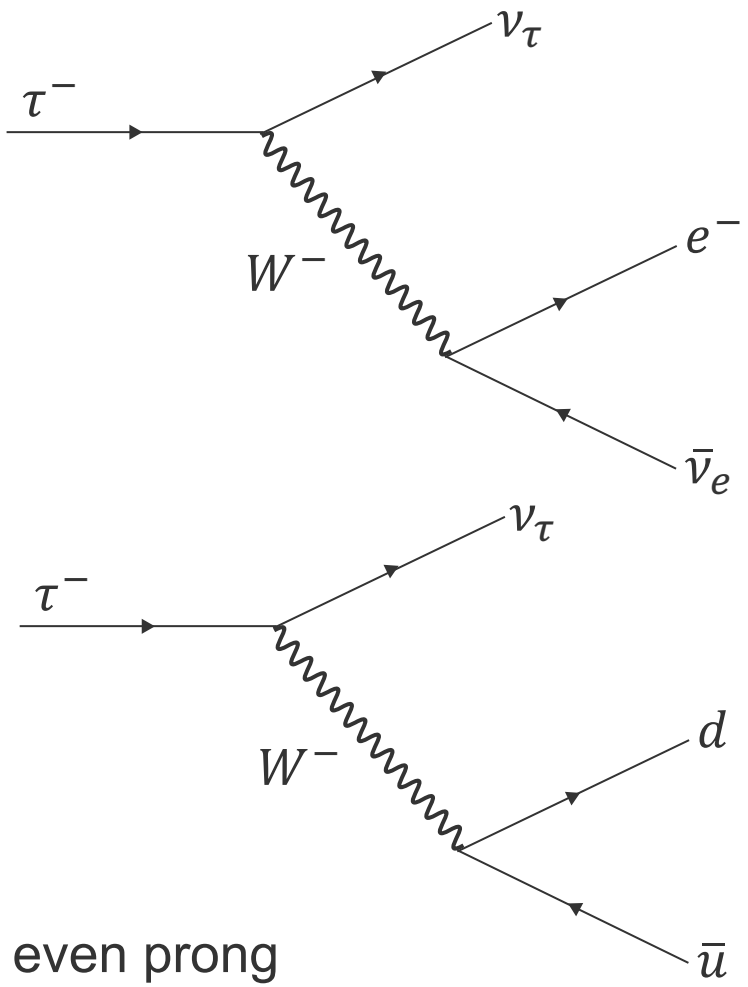
- Compression and quantization of the model are easy.
  Compression: Synapses with small weights are eliminated.
  Quantization: Quantize weights and bias values of neural network to reduce the amount of information
  →can adjust resource usage of FPGA

- Multiple calculations can be parallelized on the FPGA.
  If many calculations are performed concurrently, latency can be reduced, but more resources are used.
  Conversely, if you use fewer resources, latency will increase.

■ In this way, the model can be adjusted to meet requirements for latency, resources, and forecast accuracy.

Trigger Development

- Goal : Implement a Trigger which choose tau decay events using ECL data in GRL



- tau decay

  leptonic decay

  $\tau^- \to e^- \bar{\nu}_e \nu_\tau$, $\tau^- \to \mu^- \bar{\nu}_\mu \nu_\tau$

  hadronic decay

  - 1-prong
    $\tau^- \to \pi^- \nu_\tau$, $\tau^- \to \pi^- \pi^0 \nu_\tau$ など
  - 3-prong
    $\tau^- \to \pi^+ \pi^- \pi^- \nu_\tau$, $\tau^- \to \pi^+ \pi^- \pi^- \pi^0 \nu_\tau$ など
  - …

⇨ Tau is always generated in pairs of τ^- and τ^+, so the whole is an even prong
2-prong : 72%
4-prong : 25%
more : 3%

Trigger Development

Input data of neural network

I set up Total 19 inputs

name	quantity	Description	Detail
necl	1	The number of cluster	0~6
ecle	6	Energy	LAB frame (quantize)
ecltheta	6	Polar angle	LAB frame (quantize)
eclphi	6	Azimuth angle	LAB frame (quantize)

Data for up to six clusters can be entered.

Since the data coming into GRL are quantized values, the NN training data are also quantized accordingly.

①Energy → LSB=5MeV & bit width=12bit

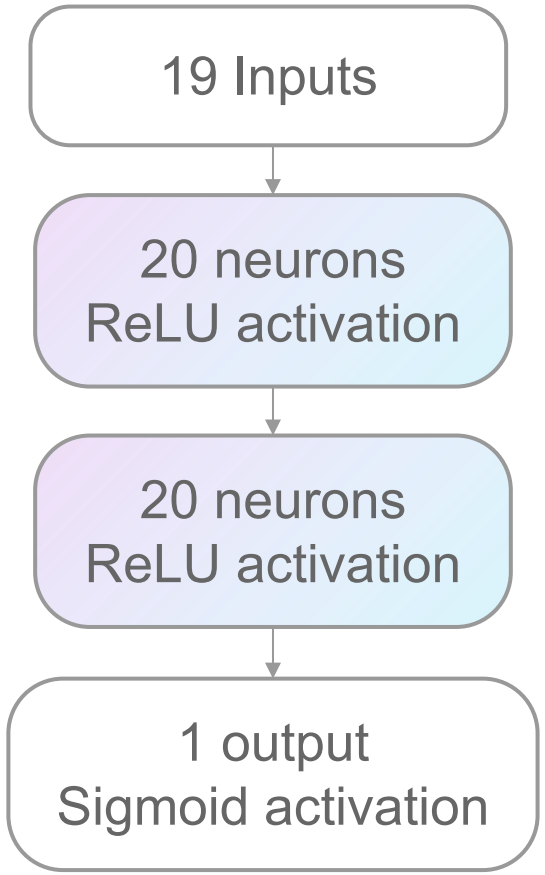
②Theta (0°~180°) → LSB=1.40625° & bit width=7bit

③Phi (0°~360°) → LSB=1.40625° & bit width=8bit

(1.40625°=360°/256)

Trigger Development

Neural Network model

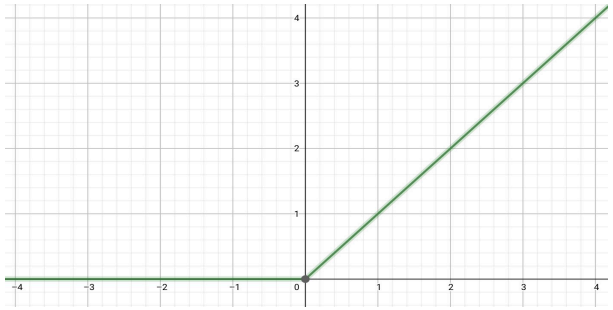


- I created a simple neural network with two hidden layers.
- The number of neurons in the output layer was set to one and the Sigmoid function was used as the activation function.

→The output from this model is a number in the range of 0 to 1

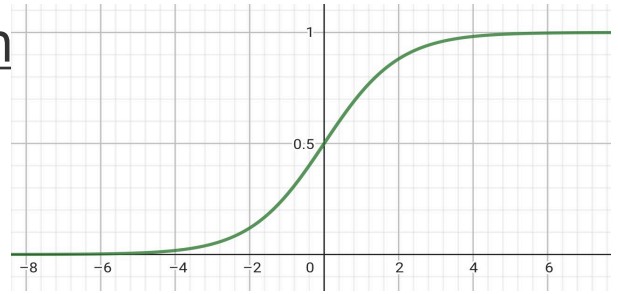
ReLU function

$$y = \begin{cases} x & (x \geq 0) \\ 0 & (x < 0) \end{cases}$$



Sigmoid function

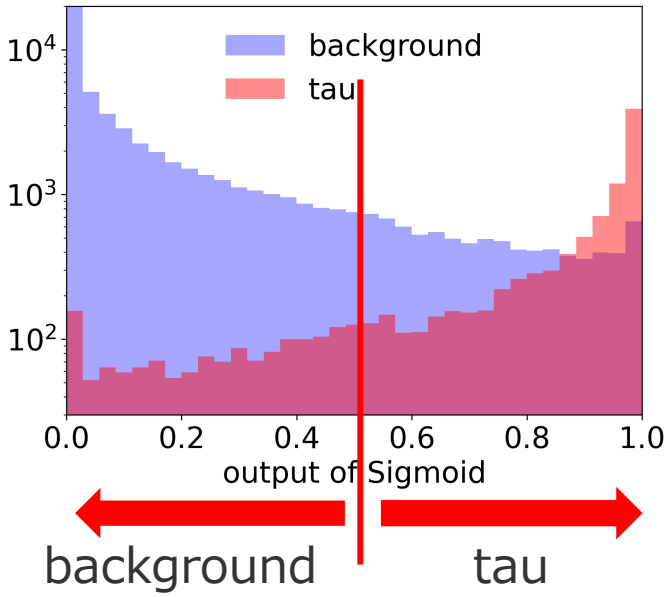
$$y = \frac{1}{1 + e^{-x}}$$



Trigger Development

- If the output is bigger than a certain threshold, it is considered a tau event;
  if it is less, it is considered a background event.

- Distribution of output after training



←The tau event is trained to be 1 and the background to be 0, so the red histogram is closer to 1 and the blue histogram is closer to 0.

- I used experimental data with offline cuts to trian NN. (700,000 events)

Performance Evaluation

Existing triggers

Introduction of existing Tau trigger with ECL information

① hie

(1) 3Dbhabha veto: $165° < \Sigma\theta CM < 190°$ && $160° < \Delta\phi CM < 200°$,
E(CL1) > 3 GeV && E(CL2) > 3 GeV &&
(E(CL1) > 4.5 GeV || E(CL2) > 4.5 GeV)

(2) In the range of $18.5° < \theta LAB < 139.3°$, Total E>1GeV

※ Improved versions hie1, hie2 and hie3 are developed to reduce background.

②ecltaub2b

(1) 2 clusters back to back angle are selected
- $130° < \Sigma\ \theta CM < 230°$
- $110° < \Delta\ \phi CM < 250°$

(2) energy of one of 2 clusters E < 1.9 GeV
(3) total energy sum in all theta region E < 7 GeV

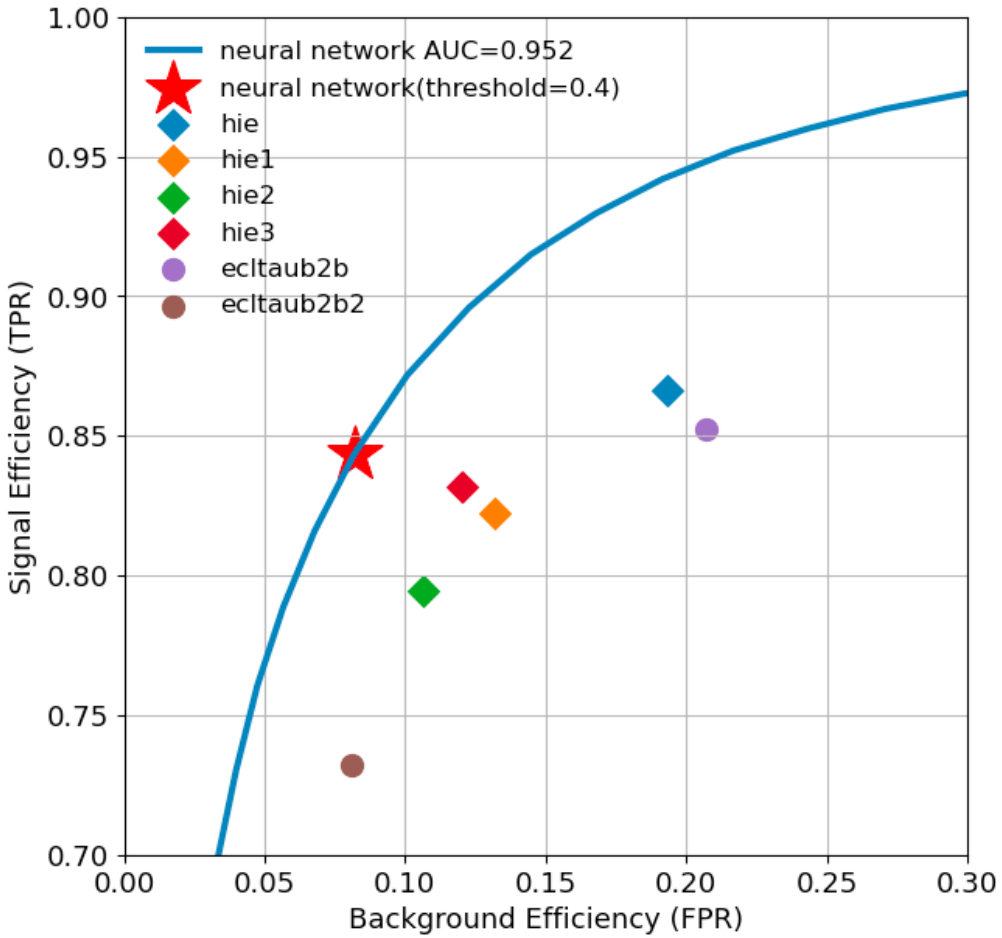
} to reduce Bhabha contribution

※ Improved versions ecltaub2b2 and ecltaub2b3 are developed to reduce background.

Performance Evaluation

ROC curve

I plotted the performance of the existing triggers I just introduced and drew a ROC curve of my NN model.



The ROC curve is to the upper left of the existing triggers' performance plot, so my neural network has a good performance.

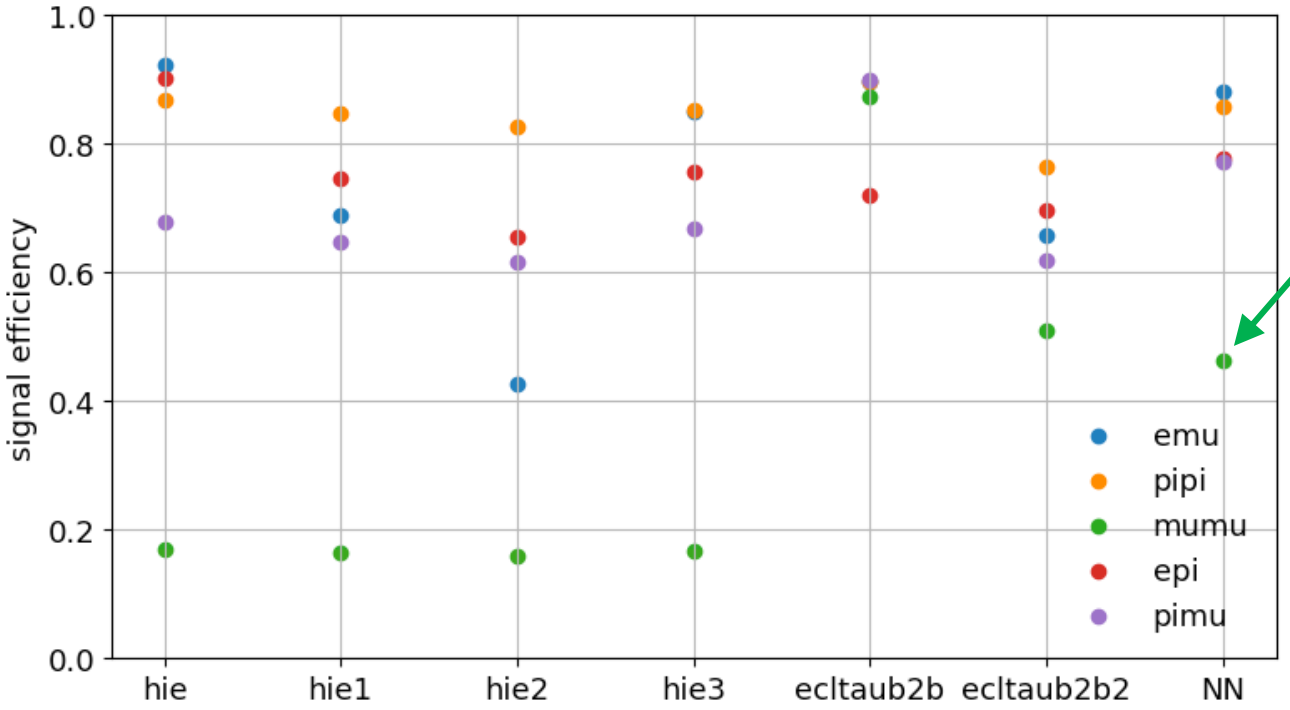
For now, the threshold of the output of the neural net is set to 0.4. (Plots marked with a red star)

Performance Evaluation

- evaluate efficiency for different final state

I plotted efficiency for each triggers, changing the color of the dots depending on what the tau particles decayed to.

■ 1prong × 1prong

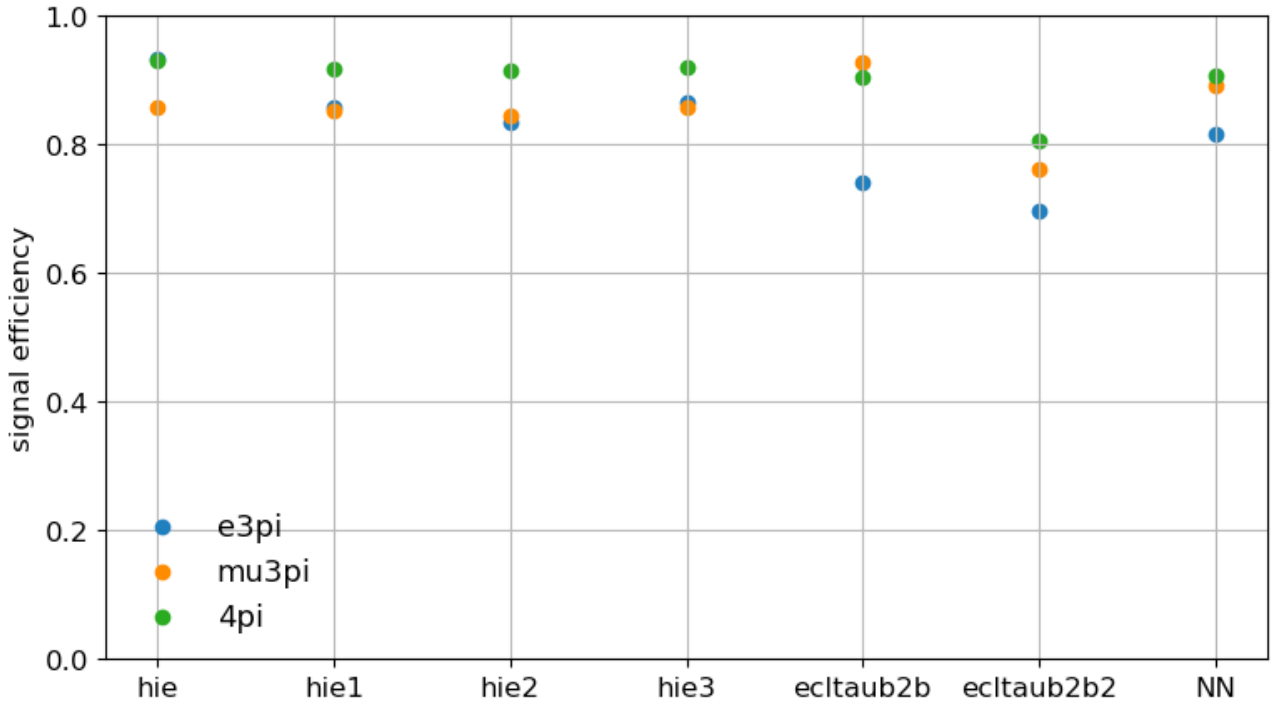


NN is insensitive to mumu than other decay mode

Adding KLM information to the NN's inputs would make the NN more sensitive to muons and it would increase the NN’s performance.

Performance Evaluation

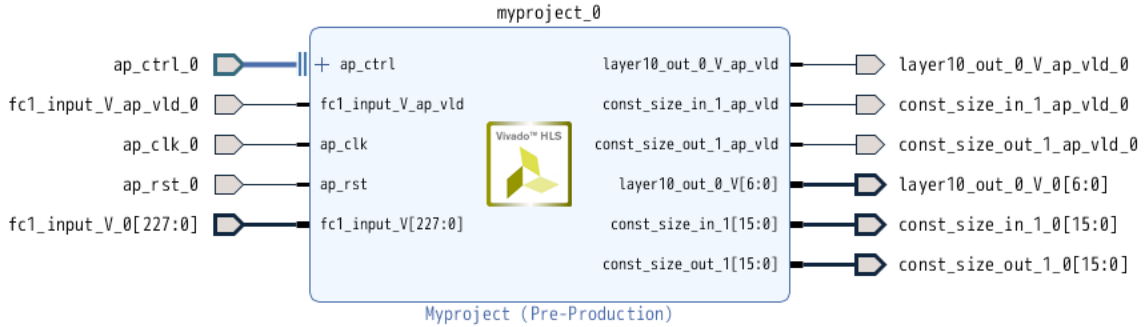
■ 3prong × 1prong



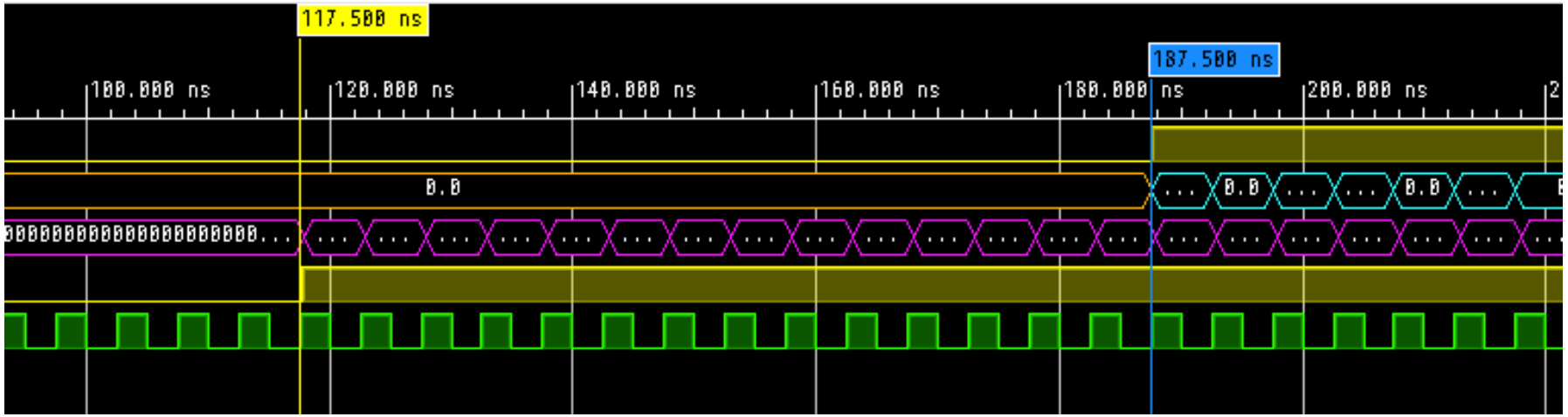
3prong × 1prong has a higher overall efficiency than 1prong × 1prong
Basically, 3prong × 1prong is considered easier to identify.

Implementation

- create neural network IP core and implement it in GRL



- simulation result of this IP core



latency=14clk

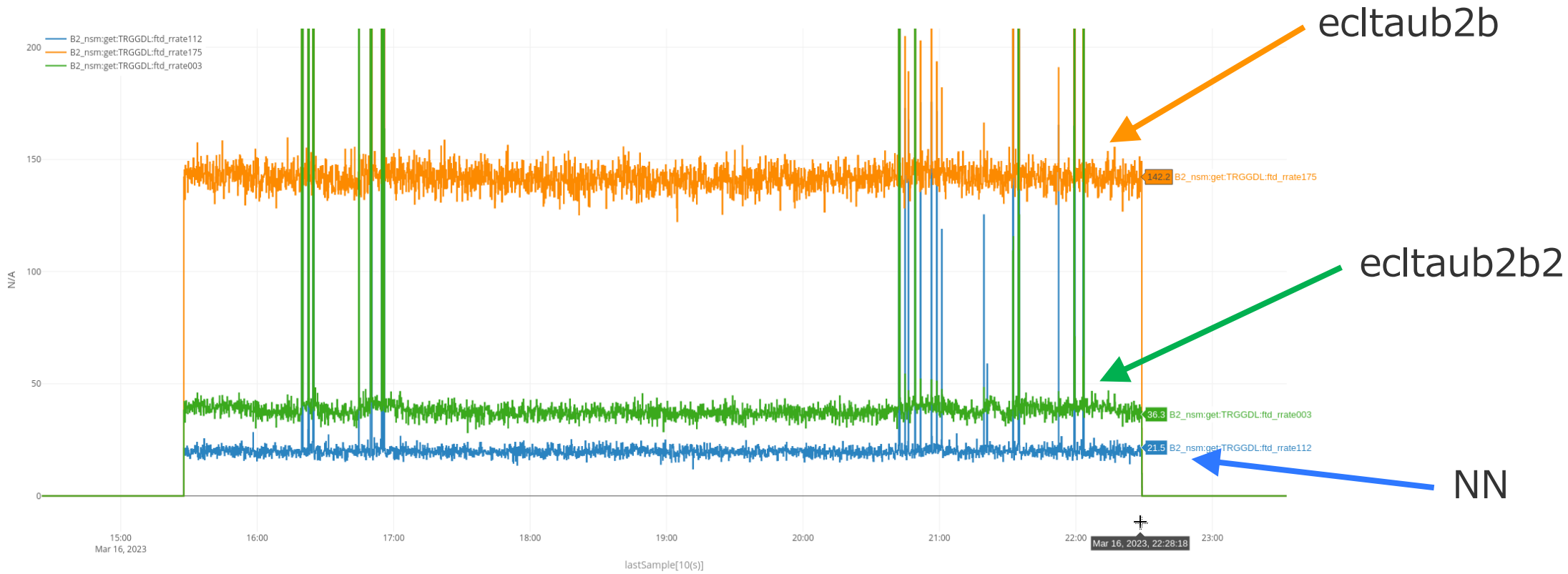
It can accept input and make output at each clock cycle. (pipeline processing)

Implementation

- test with cosmic ray

I used cosmic rays to see if the IP is working properly.

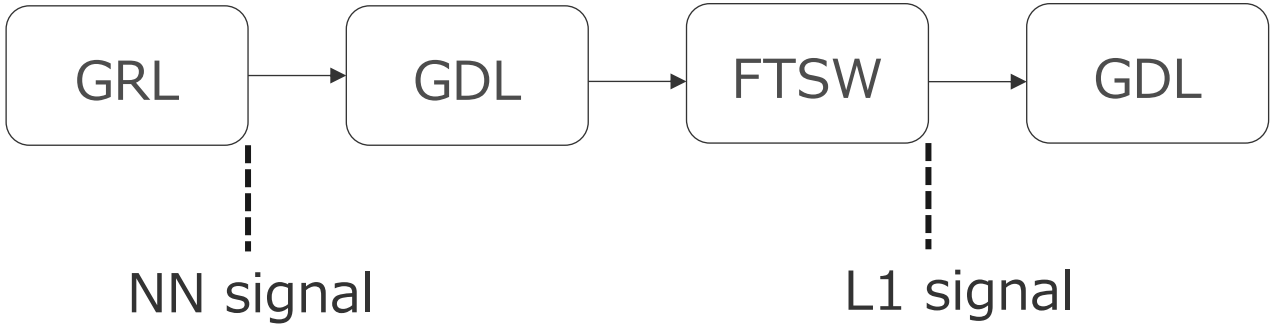
Trigger rate ↓

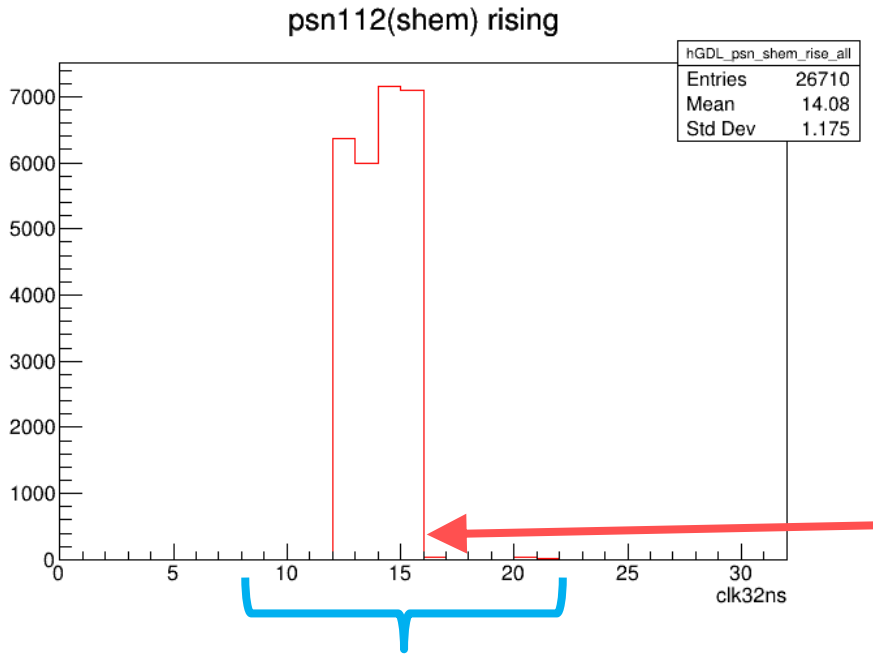


I was able to confirm that the trigger signal was emitted from my NN.

Implementation

- timing constraints





Horizontal axes :
(Time GDL received L1 signal from FTSW)
- (Time GDL received NN signal)

about 16clk

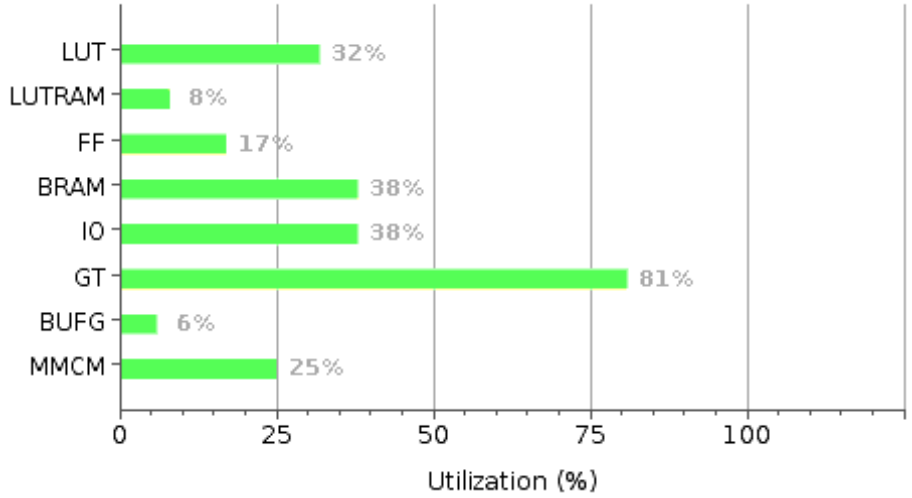
If time difference is within the range of 8~22clk ,it is OK.

Implementation

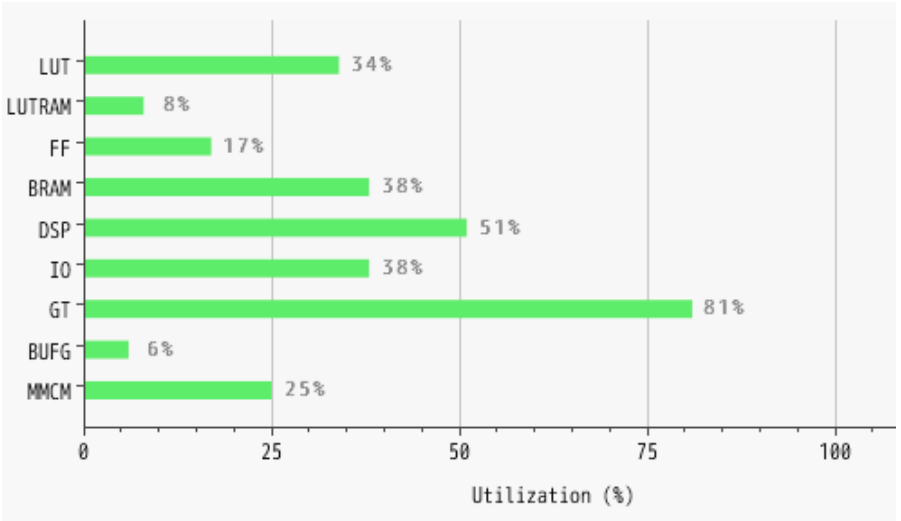
resource

I checked resource utilization of GRL.

Before implementing NN



After implementing NN



DSP usage increased significantly.
This is due to the large amount of multiplication in the hidden layer of the NN.
If it becomes necessary to implement even larger NN, DSP usage will be a bottleneck.

Summary

- Using hls4ml, I was able to create a trigger using a neural network and implement it in GRL.
- In this study, I created a tau trigger using ECL information and found that it seems to be able to improve performance over existing triggers.
- The same procedure can be used to create triggers using neural networks not only for tau but also for other decay modes.
- It would be interesting to combine the information from the ECL as well as from other detectors to create a neural network.