Edge AI for accelerator controls (READS): beam loss deblending

Jeremy Arnold, Mark Austin, Jose Berlioz, Dave Bracey, Pierrick Hanlet, Kyle Hazelwood, Jerry Yao-Chieh Hu, Aisha Ibrahim, Jing Jang, Han Liu, Seda Memik, Jovan Mitrevski, Vladimir Nagaslaev, Aakaash Narayanan, Dennis Nicklaus, Gauri Pradhan, Andrea Saewert, Brian Schupbach, Kiyomi Seiya, Rui Shi, Alexis Maya-Isabelle Shuping, Mattson Thieme, Randy Thurman-Keup, Nhan Tran, Chenwei Xu

Introduction

- At the Fermilab accelerator complex, the Main Injector (MI) and Recycler Ring (RR) share a tunnel.
  - MI: 8–120 GeV proton accelerator for high-energy neutrino experiments, etc.
  - RR: 8 GeV permanent magnet ring, proton stacker for MI; beam to g-2.
- Design from the Tevatron days: RR was a storage ring for low intensity antiprotons.
- Now both the MI and RR can have high-intensity beams.
Problem

• Both the MI and RR can have significant beam loss
• About 260 Beam Loss Monitors (BLMs) spread around the tunnel to detect beam loss
• Often hard to know which ring is responsible for the beam loss: both machines can be unnecessarily tripped
• Purpose of project: better attribute beam loss to a particular machine to increase uptime
**Beam Loss**

- One can distinguish between MI or RR loss based on time / state of the machine or location
- Idea is to train a machine learning model to distinguish between the two
BLM Readout “Pirate” Cards

- BLMs are read out using 7 BLM nodes, tied to machine permit
- Standard method to read BLM values not suitable for ML deployment
- Added a custom VME reader cards (“pirate”) to each BLM node to stream out data
  - Cards use MitySOM Cyclone 5 FPGAs
  - Stream out data using DDCP protocol using UDP at 320 Hz
Central Node

- The data from the VME reader cards are collected by a central node
- Uses a REFLEX CES Achilles Arria10 SoC SOM
- The ARM cores collect the data streams, sends the samples to the ML (in the FPGA fabric), and streams the inference results out to Redis.
U-Net ML Algorithm

• After some tests settled on U-Net architecture: recognizes both global and local losses.
• Inputs: 260 BLM readings
• Outputs: 260 × 2 (nBLM × nMachines) probability that loss at a BLM is attributable to MI / RR.
• New inputs each 3 ms.
U-Net ML Algorithm

- Model is trained with Keras and uses post-training quantization
- Synthesized with hls4ml using Intel/Quartus backend
- Used Intel Quartus Prime Pro 21.4
- In order to fit on an Arria 10 had to tune the model widths:
  - 8 bit weights
  - 16 bit layer outputs, int/frac varied
- System easily meets required 3 ms system latency: U-Net latency 1.57 ms, 1.74 ms system
Offline Model Inference

• Attributes ring-wide losses well, but not local losses: a work in progress
Online performance, BLM 232C, purposeful loss in RR

![Graph showing online performance with annotations for loss attributed to RR and MI](image_url)
Implementation details

• In order to work with the skip connections, had to add a buffer size to the streams

```cpp
template <typename T, unsigned int N> using stream = ihc::stream<T, ihc::buffer<N>>;
template <typename T, unsigned int N> using stream_in = ihc::stream_in<T, ihc::buffer<N>>;
template <typename T, unsigned int N> using stream_out = ihc::stream_out<T, ihc::buffer<N>>;
```

• Required changing templates inside nnet_utils implementations

• Ran FIFO depth optimization using the Vivado backend in order to get initial buffer size estimates, increased some until cosim did not deadlock.

• Given that this is an extensive change and the Intel HLS compiler is being deprecated (with ihc::streams being replaced with pipes), decided to wait for the oneAPI version and not try to make a pull request.
Additional change

• The streaming zero-padding implementation caused errors that were not obvious:
  
  Multiple reflexive accesses from stream 'layer3_out' is not allowed

• We wrote an alternate less optimized zero-padding implementation that did not produce this error.
Conclusion

- Created a full system with custom hardware and software
- Trained an ML model to distinguish between losses from the MI and RR
- Synthesized the model using hls4ml, the Intel HLS compiler, and Quartus
- Successfully deployed an ML model in hardware as part of accelerator controls
- The first deployment of Realtime Edge AI in the Fermilab accelerator complex
Acknowledgement

This work was produced by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics. Publisher acknowledges the U.S. Government license to provide public access under the DOE Public Access Plan DOE Public Access Plan.
Backup
# Network details

<table>
<thead>
<tr>
<th>System Properties</th>
<th>U-Net Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trainable Parameters</td>
<td>134434</td>
</tr>
<tr>
<td>Default Precision</td>
<td>ap_fixed&lt;16, 7&gt;</td>
</tr>
<tr>
<td>Precision Strategy</td>
<td>Layer-based</td>
</tr>
<tr>
<td>Default Reuse Factor</td>
<td>32</td>
</tr>
<tr>
<td>Dense/Sigmoid Reuse Factor</td>
<td>260</td>
</tr>
<tr>
<td>Average System Latency</td>
<td>1.74ms</td>
</tr>
<tr>
<td>FPGA U-Net Latency</td>
<td>1.57ms</td>
</tr>
<tr>
<td>Logic Utilization</td>
<td>223674 (89%)</td>
</tr>
<tr>
<td>Total Registers</td>
<td>406123</td>
</tr>
<tr>
<td>Total Pins</td>
<td>221 (37 %)</td>
</tr>
<tr>
<td>Total Block Memory Bits</td>
<td>25275808(58%)</td>
</tr>
<tr>
<td>Total RAM Blocks</td>
<td>1818 (85%)</td>
</tr>
<tr>
<td>Total DSP Blocks</td>
<td>273 (16%)</td>
</tr>
<tr>
<td>Total PLLs</td>
<td>3 (5 %)</td>
</tr>
</tbody>
</table>