Supported by
MONASH WARWICK ALLIANCE
MONASH University
WARWICK
THE UNIVERSITY OF WARWICK

Real-time clustering in the LHCb electromagnetic calorimeter

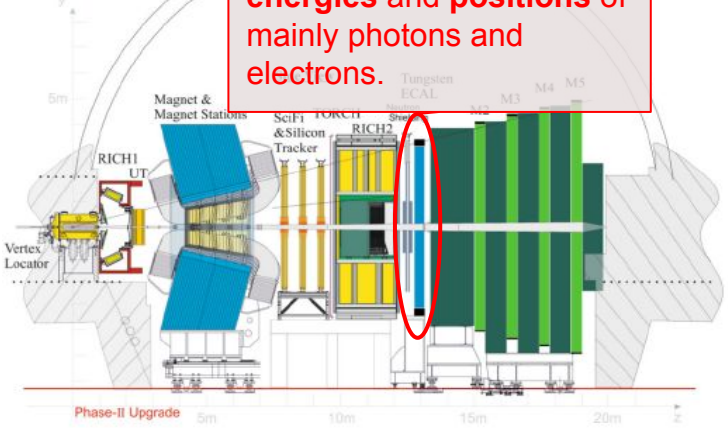
Upgrade II R&D studies

Riley Henderson, Ulrik Egede, Minni Singla

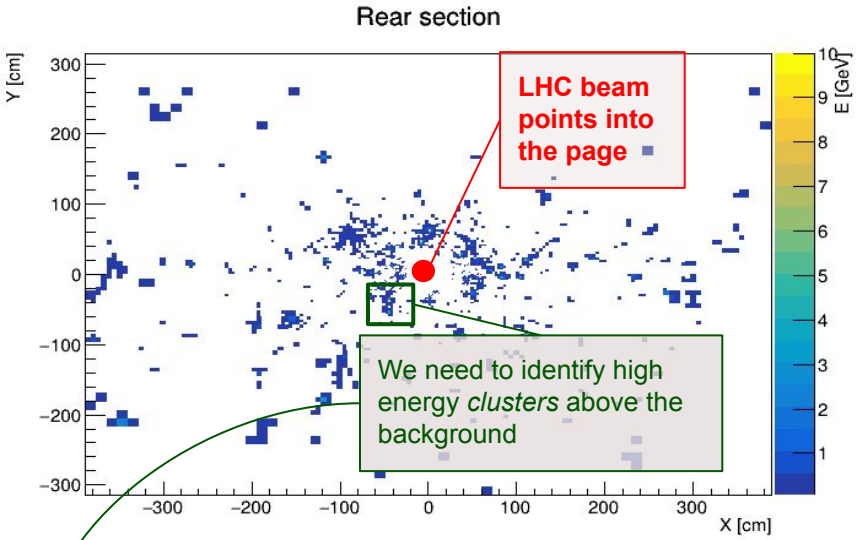
02/02/2022

The LHCb ECAL

The ECAL measures the **energies** and **positions** of mainly photons and electrons.



A schematic diagram of the Upgrade II LHCb detector



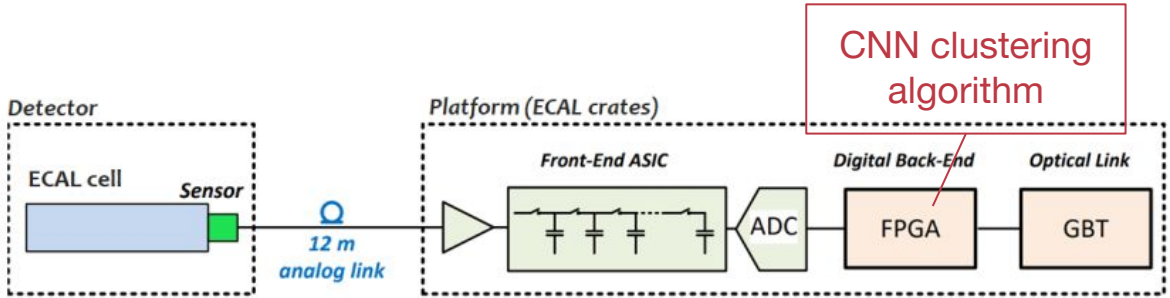
A Geant4 simulation of the ECAL readout for a single $B^0 \rightarrow K^{*0}\gamma$ event

Problem: Can we find ECAL clusters in *real-time* using convolutional neural networks (CNNs) implemented in FPGAs?

Riley Henderson

Real-time clustering

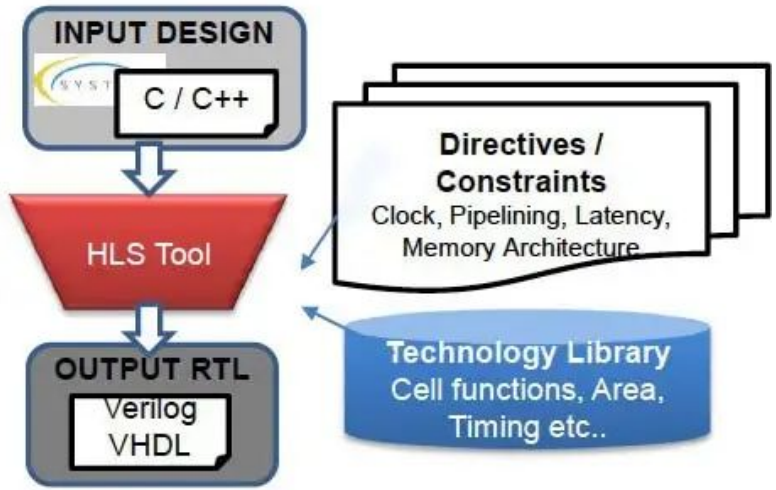
- Real-time clustering is required to reduce the ECAL data rate by only transmitting cluster level information instead of the complete raw readout.
- The LHC bunch crossing rate is 40 MHz.
- Such processing speeds can only be achieved using specialised hardware such as FPGAs.



Riley Henderson

High-level synthesis

- Programming an FPGA directly can be very time consuming!
- Thankfully, high-level synthesis (HLS) tools have been developed to bypass this difficulty.
- One particularly useful tool — **hls4ml** — is specifically designed for neural network translation.



Hls4ml

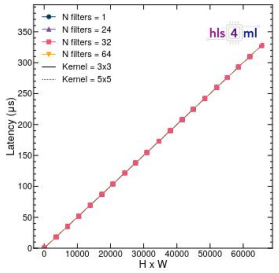
- **Hls4ml** opts for an *everything-on-chip* implementation.
  - Less limited by I/O speeds.
- Highly transparent and configurable:
  - Specific latency optimisation strategy.
  - Tunable parallelisation — i.e. resource utilisation.
- I have experimented with various configurations, clustering window sizes, etc.
- **Minimum Latency**: 0.5 μs for an 8x8 clustering window.

```
* Product family: zynq
* Target device:  xcu250-figd2104-2L-e
```

```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+---------+----------+------------+
    |  Clock |  Target | Estimated| Uncertainty|
    +--------+---------+----------+------------+
    |ap_clk  | 5.00 ns | 4.190 ns |   0.62 ns  |
    +--------+---------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+-----------+-----------+-----+-----+----------+
    | Latency (cycles)  |  Latency (absolute)   |  Interval | Pipeline |
    |   min   |   max   |    min    |    max    | min | max |   Type   |
    +---------+---------+-----------+-----------+-----+-----+----------+
    |    16879|    16879| 84.395 us | 84.395 us |  5409|  16225| dataflow |
    +---------+---------+-----------+-----------+-----+-----+----------+
```

```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+------+
|       Name      | BRAM_18K| DSP48E|   FF   |  LUT   | URAM |
+-----------------+---------+-------+--------+--------+------+
|DSP              |        -|      -|       0|       0|     -|
|Expression       |        -|      -|       0|      32|     -|
|FIFO             |      196|      -|    6370|   12632|     -|
|Instance         |      338|      0|   27391|  166257|     -|
|Memory           |        -|      -|       -|       -|     -|
|Multiplexer      |        -|      -|       -|      36|     -|
|Register         |        -|      -|       6|       -|     -|
+-----------------+---------+-------+--------+--------+------+
|Total            |      534|      0|   33767|  178957|     0|
+-----------------+---------+-------+--------+--------+------+
|Available SLR    |     1344|   3072|  864000|  432000|   320|
+-----------------+---------+-------+--------+--------+------+
|Utilization SLR (%)|      39|      0|       3|      41|     0|
+-----------------+---------+-------+--------+--------+------+
|Available        |     5376|  12288| 3456000| 1728000|  1280|
+-----------------+---------+-------+--------+--------+------+
|Utilization (%)  |      9  |      0|      -0|      10|     0|
+-----------------+---------+-------+--------+--------+------+
```



Latency is strongly correlated with input size!

Latency and throughput

- A naive estimate of throughput based on the hls4ml latency estimate is:
  - Throughput (MHz) = 1/Latency (μs)
- With this estimate, we can indeed achieve MHz operating frequencies if we process 8x8 clustering windows.
- This fits well with the *already proposed* 8x8 processing window for a given FPGA in the LHCb ECAL.
- However, it may be possible to make further use of pipelining/buffering to enhance throughput.
  - This means we could potentially afford a somewhat higher latency.
  - This may allow processing larger windows and enhancing the level of data reduction further.

Other ongoing studies

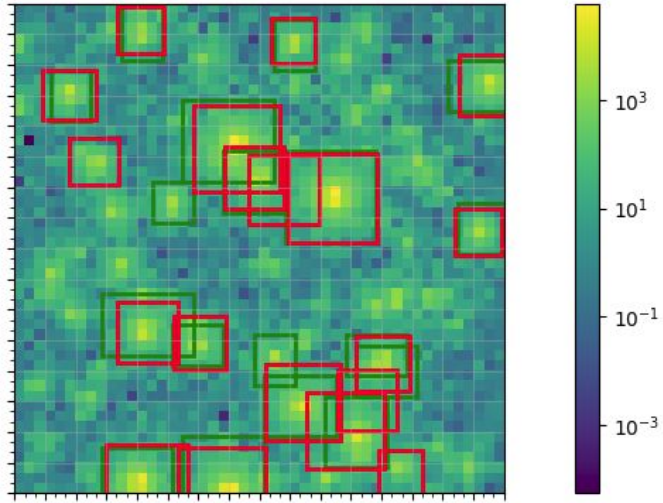
- Physics performance evaluation
  - Interface the clustering model with the simulation of real physics processes in the ECAL.
  - Quantify energy resolution, time resolution, etc.
  - Translate this to:
    - Invariant mass resolution and background rejection in radiative decays.

Riley Henderson

Backup slides

ECAL clustering

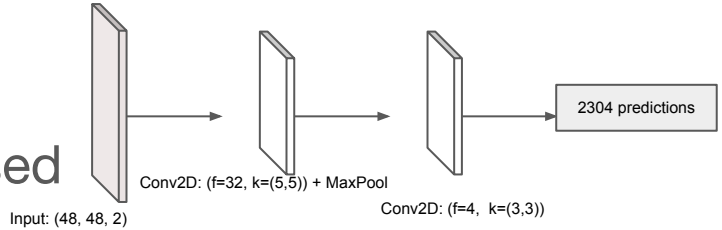
- There are several approaches one can take. Two important ones are:
  1. Cellular automata (used at LHCb currently to find ECAL clusters).
     - Straightforward and very fast
     - ❌ Cannot handle Upgrade II luminosity.
  2. Neural networks.
     - Another possible solution

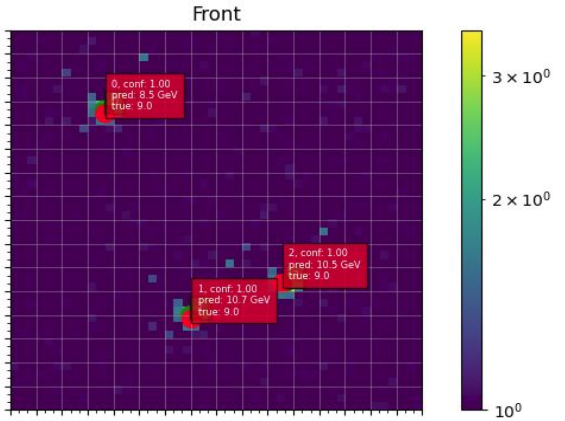


(Above) Example output from a calorimeter clustering algorithm. The green boxes are true particle hits, while the red boxes are the identified clusters.

SSD-ECAL algorithm

- My previous studies involved creating a convolutional neural network (CNN) based clustering algorithm.
- The resulting algorithm is based on the SSD network for object detection — the SSD-ECAL model.
- The model can find clusters reasonably well in high-pileup environments.
  - ✓ Candidate solution for Upgrade II luminosity.





(Top) Overview of the minimal SSD-ECAL network architecture.
(Bottom) Basic output from the model: cluster centroid + incident particle energy

HLS tools

- I have independently made use of two HLS tools for CNNs:
  - **Vitis AI**
    - Directly integrated with Xilinx VCK5000 development card.
    - We can physically deploy and test the resulting model.
    - ❌ Generic and not directly focused on raw processing speed.
  - **Hls4ml**
    - Developed specifically for LHC trigger applications ⇒ directly focussed on raw processing speed.
    - ❌ Not directly interfaced with any device ⇒ a few more steps before we can physically deploy and test the model.

Riley Henderson

Initial tests: Vitis AI

- I was able to create a fully operational CNN based clustering model with the **Vitis AI** HLS tool.
- I physically measured the throughput in frames per second (FPS).
- Device: Xilinx VCK5000 AI inference card.
- **Max frame rate: ~10–20 kHz.**
  - Some dependence on image size and level of multithreading.

(Above) The Xilinx VCK5000 Versal Development Card for AI Inference

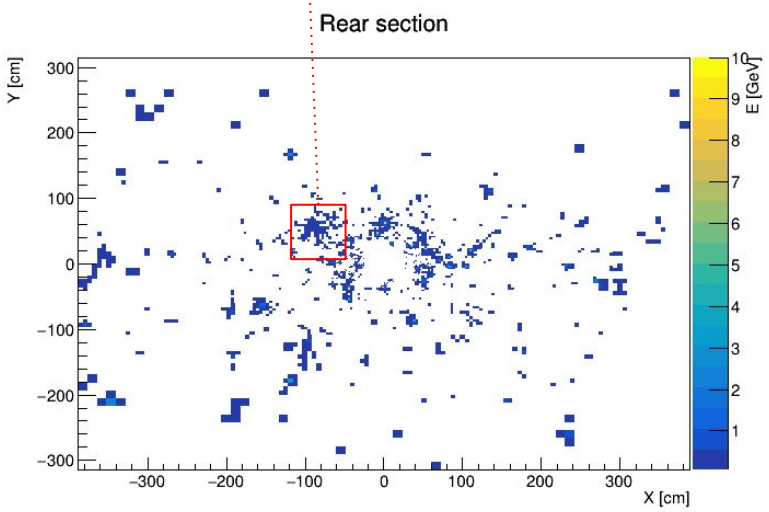
What's next?

- Performance evaluation
  - Simulation of real physics processes in the ECAL
- Additional challenges

ECAL simulation

- In order to assess the performance of the clustering algorithm, we need realistic datasets.
- We can simulate real process typical of LHCb physics analyses — e.g. $B^0 \rightarrow K^{*0}\gamma$
- We can also simulate minimum bias background processes and combine them.

We can slice out portions of these full ECAL simulations and run the clustering algorithm on them.

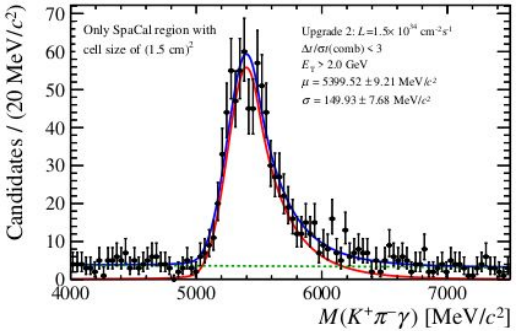


(Above) A Geant4 simulation of the ECAL readout for a single $B^0 \rightarrow K^{*0}\gamma$ event.

Performance metrics

- There are several performance metrics we can use to evaluate the performance of the algorithm.
- The goal is make the clustering evaluation easily translatable into higher-level physics analysis goals.
  - For example, the *position and energy resolution* for photons in the ECAL is an important factor for the mass resolution in radiative B meson decays.

Example: $B^0 \rightarrow K^{*0}\gamma$ photon position and energy resolution are critical for reconstructing the B^0 mass.



Riley Henderson

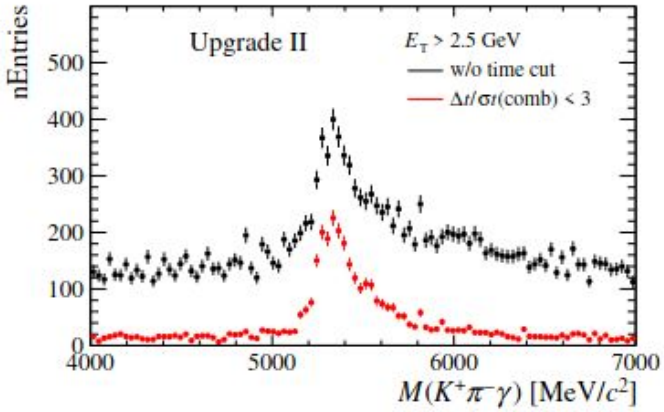
Performance metrics cont.

- Another useful metric is the particle *arrival time resolution* which will be crucial for:
  - Background suppression
  - Correct track and primary vertex association.

Example: $B^0 \rightarrow K^{*0}\gamma$ photon arrival time resolution is important for background reduction.

An arrival time window cut imposed around the expected arrival time of the photon dramatically reduces the background.

The level of reduction depends on the ECAL arrival time resolution.



Additional challenges

- How to deal with particles that hit the ECAL on the boundary of multiple FPGA processing windows?
  - The algorithm should avoid missing these partial clusters.
  - The algorithm should also avoid double counting these partial clusters.
  - Cross-board communication may not be available to ensure consistency.
- How to deal with varying cell sizes across the ECAL?
  - CNNs inherently assume a uniform grid.
  - May be avoidable with parallel processing of small (varying sized) slices as likely already required for achieving the latency target.

Riley Henderson