APEIRON: composing smart TDAQ systems for high energy physics experiments

Alessandro Lonardo

(INFN Roma, APE Lab)

for the APEIRON team

21st International Workshop on Advanced Computing and Analysis Techniques in Physics Research
ACAT 2022
Bari – 27th October 2022

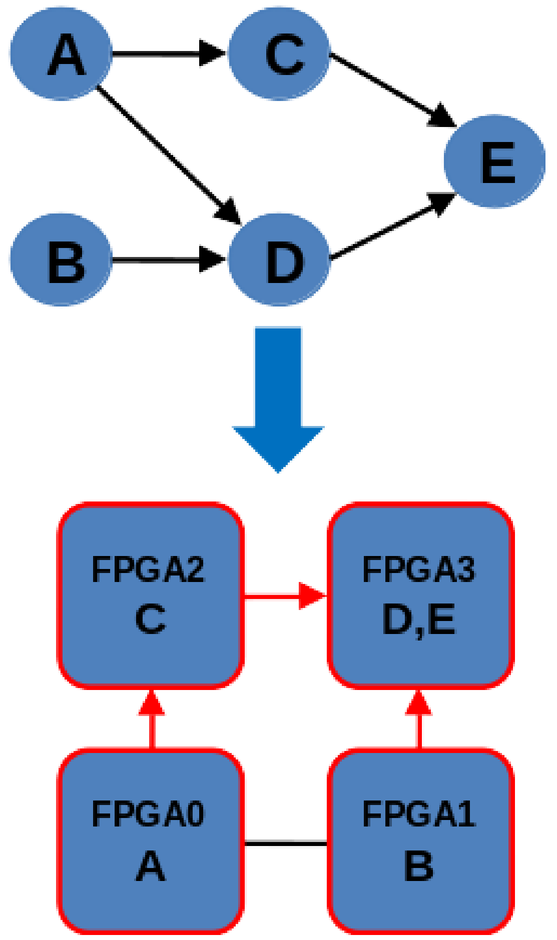
This work is supported by the TEXTAROSSA project (G.A. n. 956831), as part of the EuroHPC initiative, and by INFN National Scientific Committee 5.

APEIRON: an overview

- **Goal:** develop a framework offering hardware and software support for the execution of real-time dataflow applications on a system composed by interconnected FPGAs .

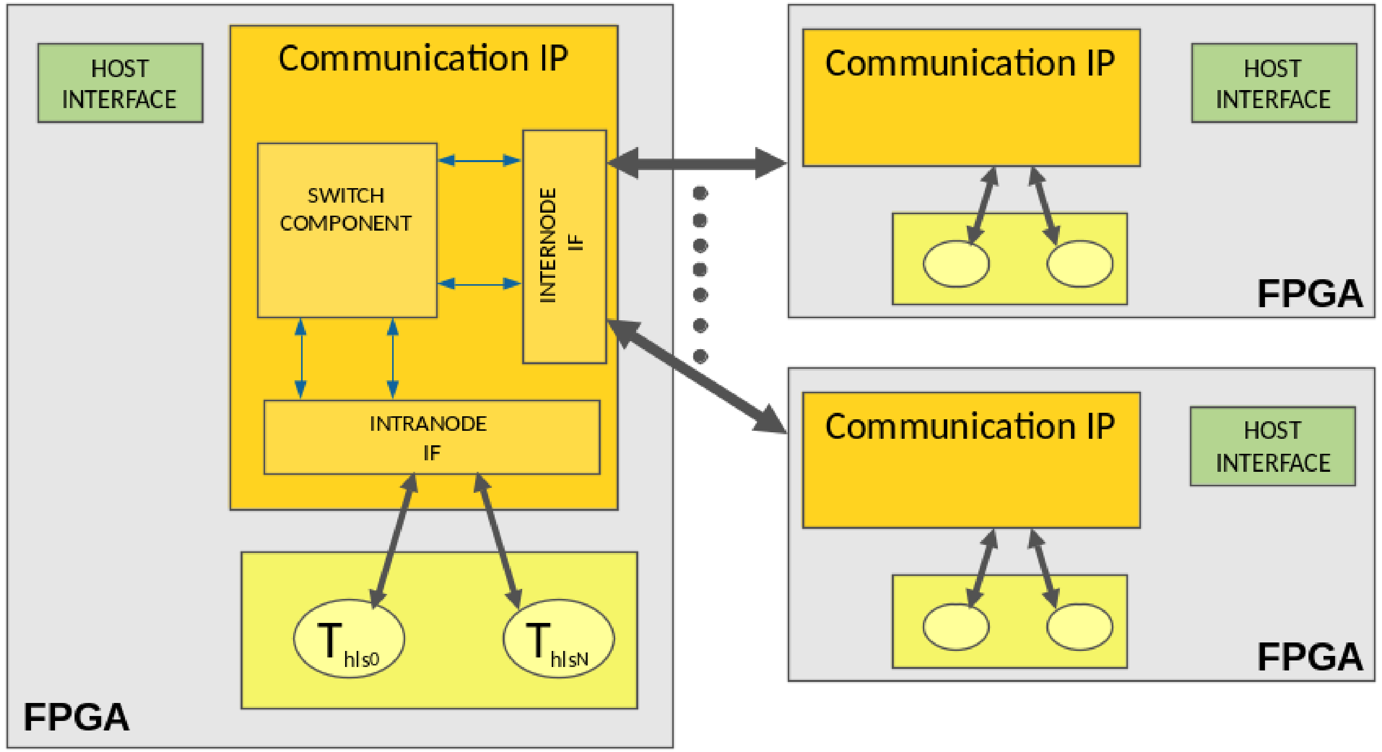
  - Map the dataflow graph of the application on the distributed FPGA system and offers runtime support for the execution.
  - Allow users with no (or little) experience in hardware design tools, to develop their applications on such distributed FPGA-based platforms
    - Tasks are implemented in C++ using High Level Synthesis tools (Xilinx Vitis).
    - Lightweight C++ communication API
      - Non-blocking *send()*
      - Blocking *receive()*

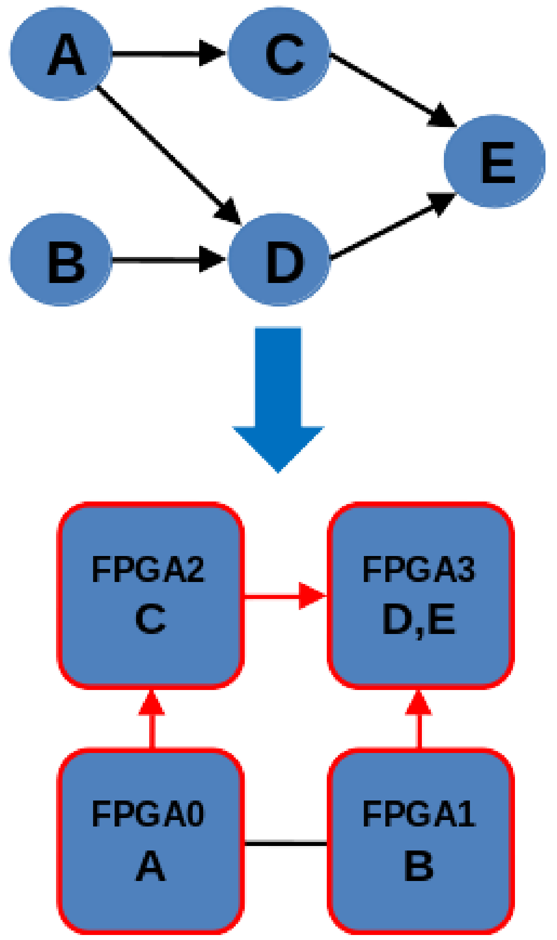
  - **APEIRON is based on Xilinx Vitis High Level Synthesis framework and on INFN Communication IP**



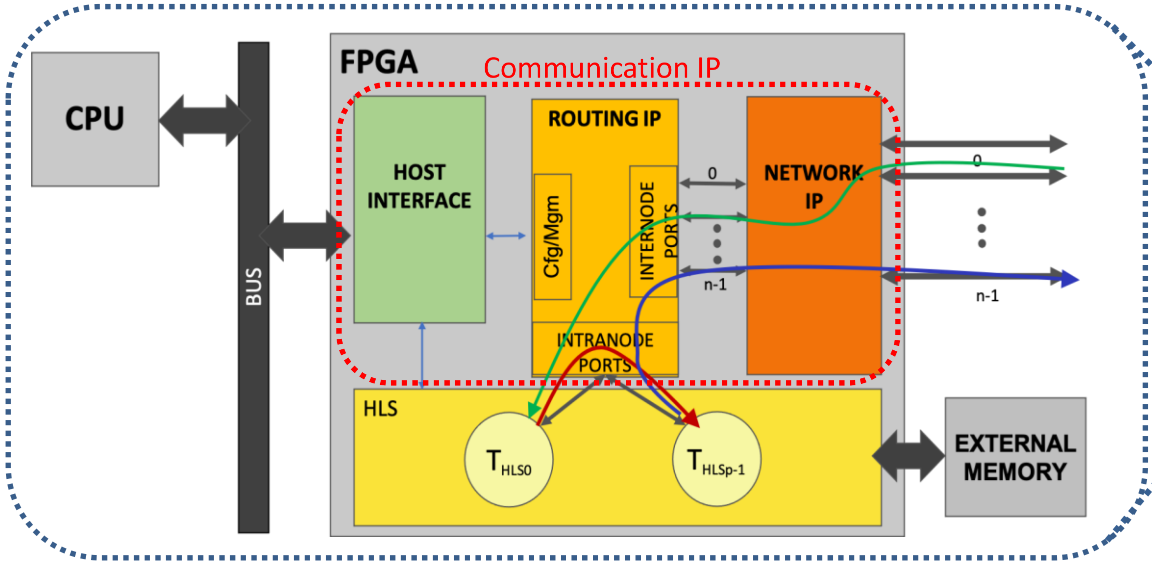
APEIRON: INFN Communication IP

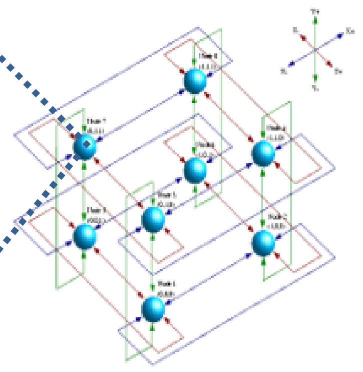
- INFN is developing the IPs implementing a **direct network** that allows **low-latency** data transfer between processing tasks deployed on the same FPGA (intra-node communication) and on different FPGAs (inter-node communication).





APEIRON: the Node

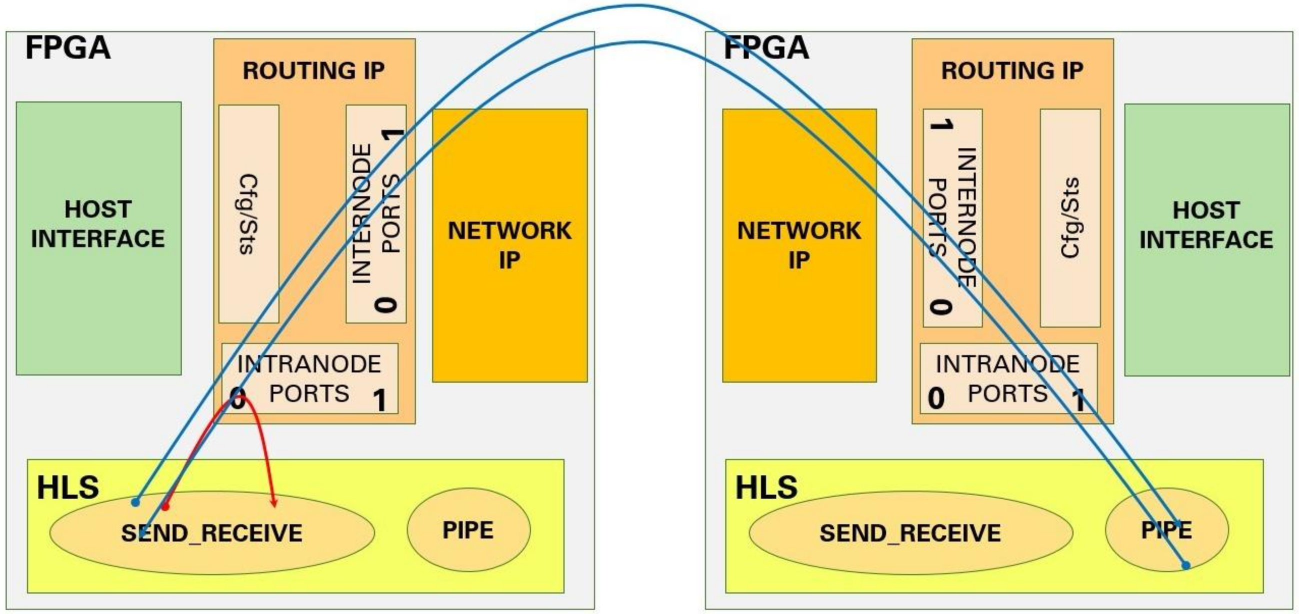




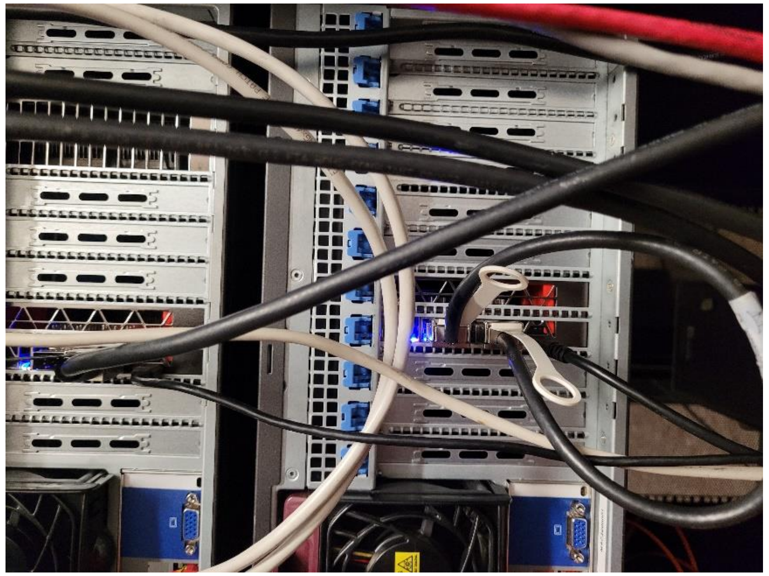
APEIRON node in a Torus three-D configuration

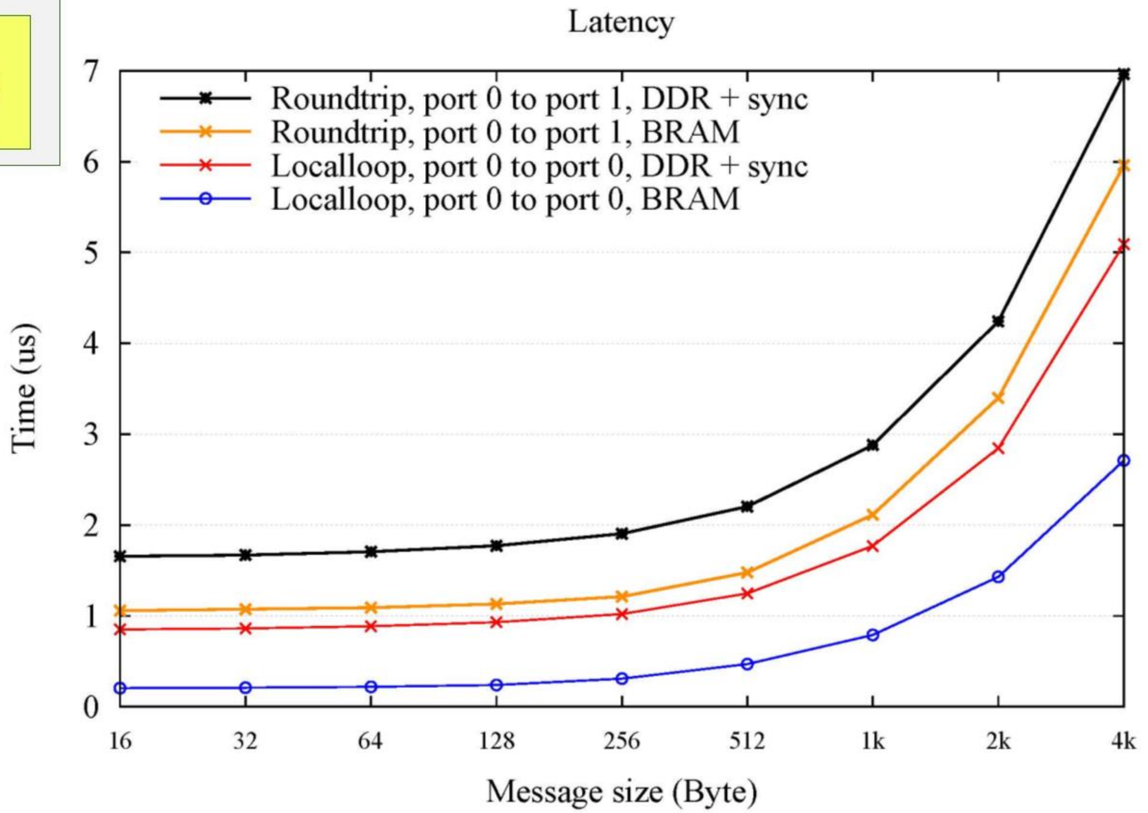
- Host Interface IP: Interface the FPGA logic with the host through the system bus.
  - Xilinx XDMA PCIe Gen3
- Routing IP: Routing of intra-node and inter-node messages between processing tasks on FPGA.
- Network IP: Network channels and Application-dependent I/O
  - APElink 20 Gbps → 80 Gbps
  - UDP/IP over 1/10 GbE → 25/40/100 GbE
- HLS Kernels: user defined processing tasks

APEIRON: Communication Latency



2 Alveo U200 Cards, 2 QSFP+ ports each, ring topology.

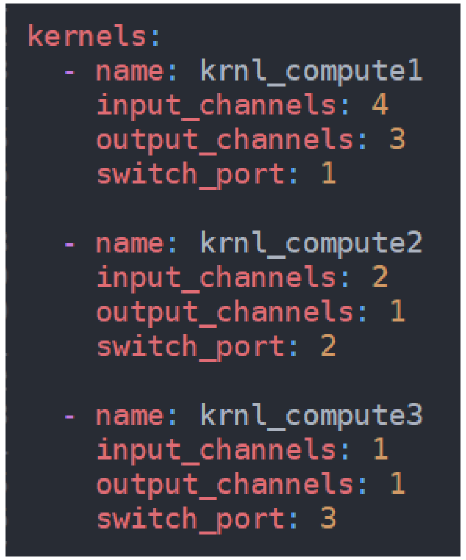



Latency
Roundtrip, port 0 to port 1, DDR + sync
Roundtrip, port 0 to port 1, BRAM
Localloop, port 0 to port 0, DDR + sync
Localloop, port 0 to port 0, BRAM
Time (us)
0
1
2
3
4
5
6
7
16
32
64
128
256
512
1k
2k
4k
Message size (Byte)

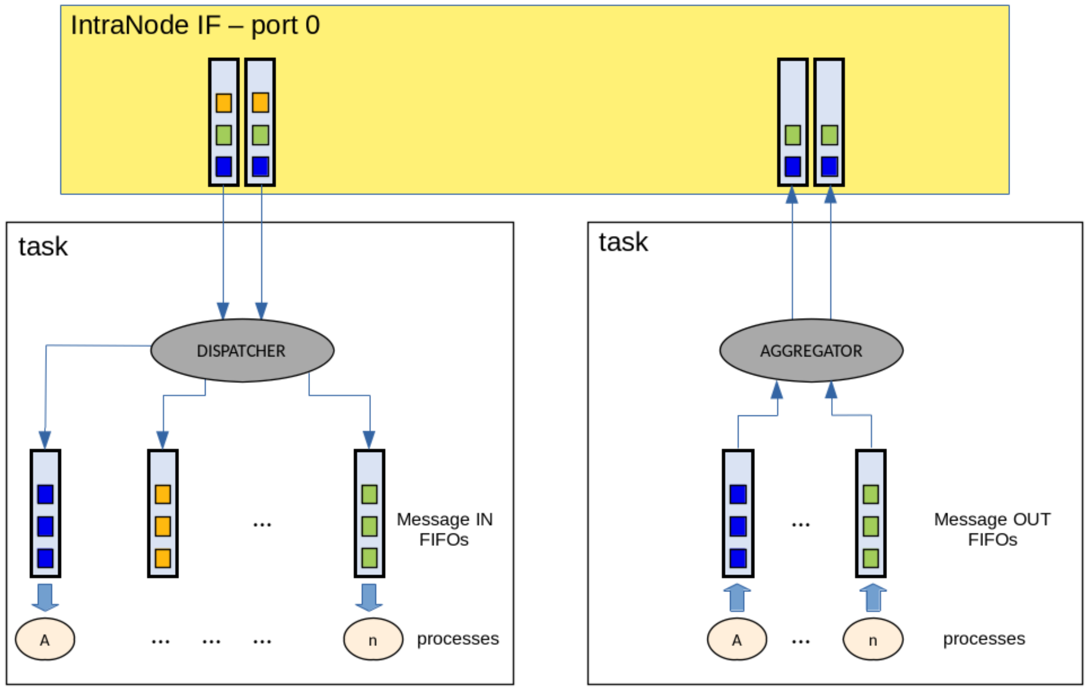
APEIRON: Workflow for automatic FPGA bitstream generation

- The HLS task must have a generic interface, implementation is free.
- A YAML configuration file is used to describe the kernels interconnection topology, specifying how many input/output channels they have

```
void example_task(
[list of optional kernel specific parameters],
message_stream_t message_data_in[N_INPUT_CHANNELS],
message_stream_t message_data_out[N_OUTPUT_CHANNELS])
{…}
```

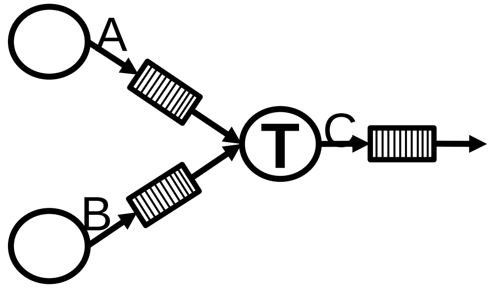


```
kernels:
  - name: krnl_compute1
    input_channels: 4
    output_channels: 3
    switch_port: 1

  - name: krnl_compute2
    input_channels: 2
    output_channels: 1
    switch_port: 2

  - name: krnl_compute3
    input_channels: 1
    output_channels: 1
    switch_port: 3
```



- Adaptation toward/from IntraNode ports of the Routing IP is done by the automatically generated Aggregator and Dispatcher kernel templates.

APEIRON: HLS C++ Communication Primitives



– **send**(msg, size, dest_node,
task_id, ch_id)

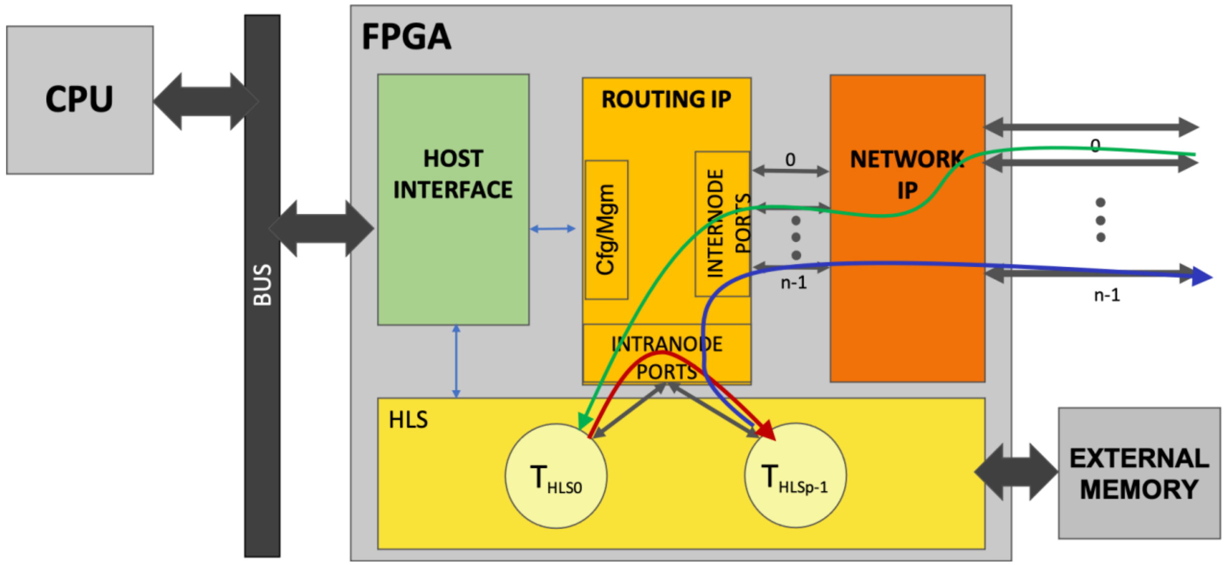
– **receive**(ch_id)

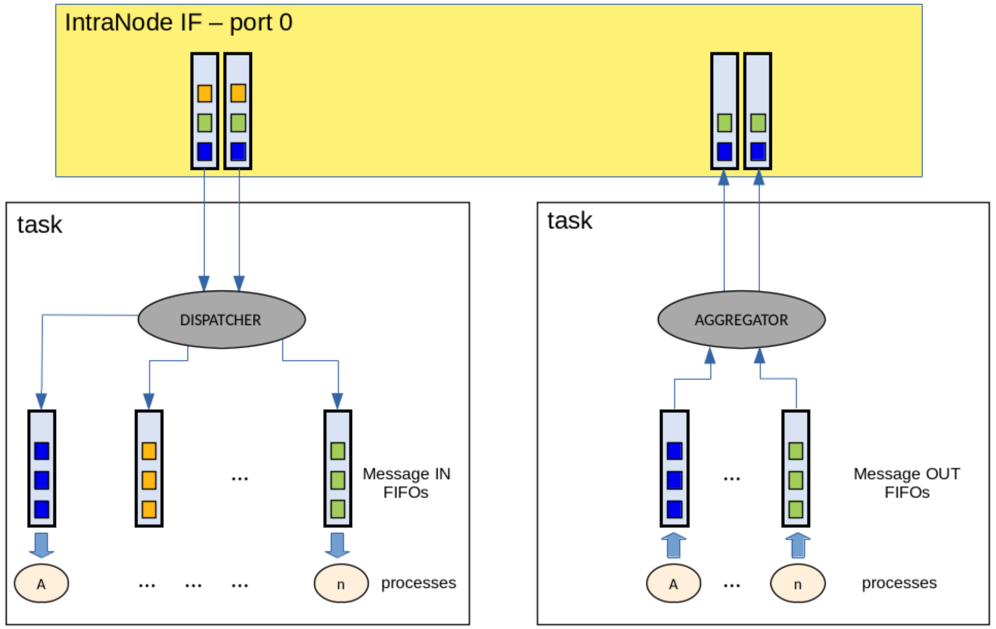
Where :

dest_node are the n-Dim coordinates of the destination node (FPGA) in a n-Dim torus network.

task_id is the local-to-node receiving task (kernel) identifier (0-3).

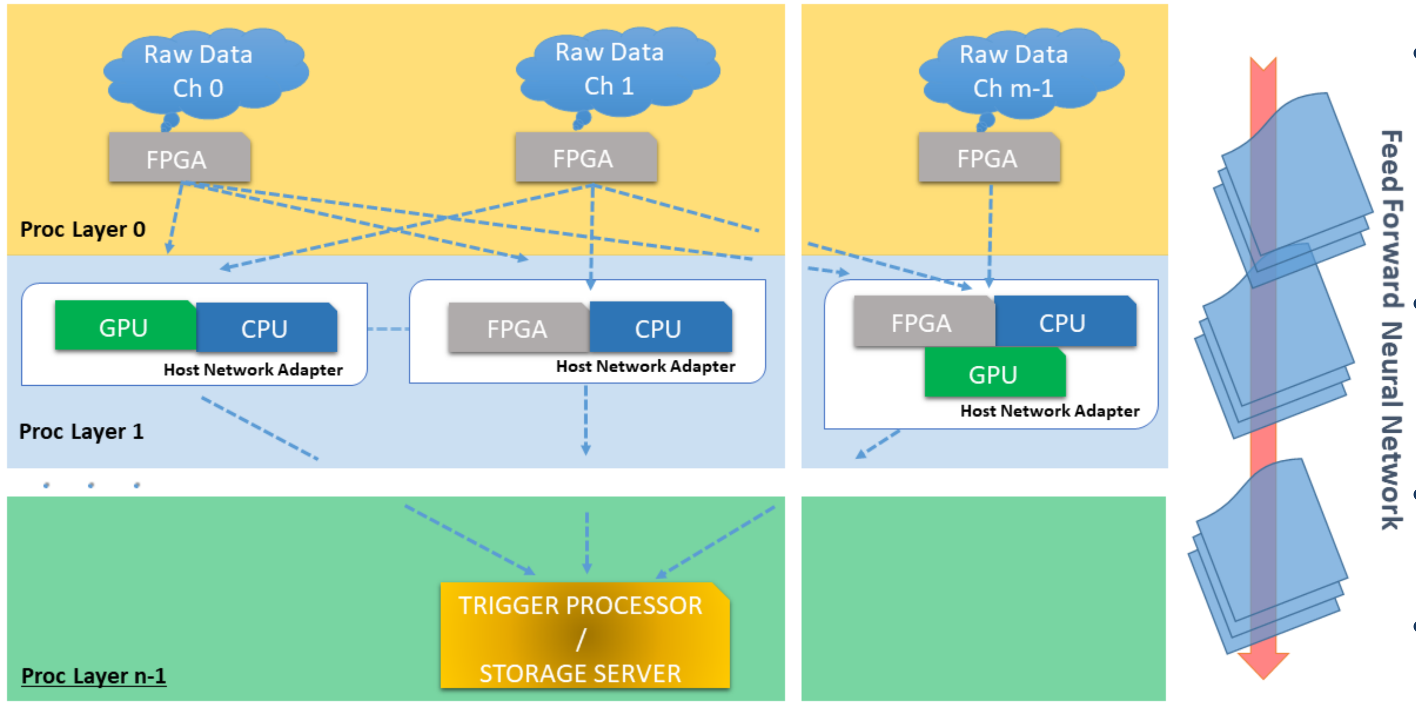
ch_id is the local-to-task receiving fifo (channel) identifier (0-127).





APEIRON: Use in Trigger and Data Acquistion Systems

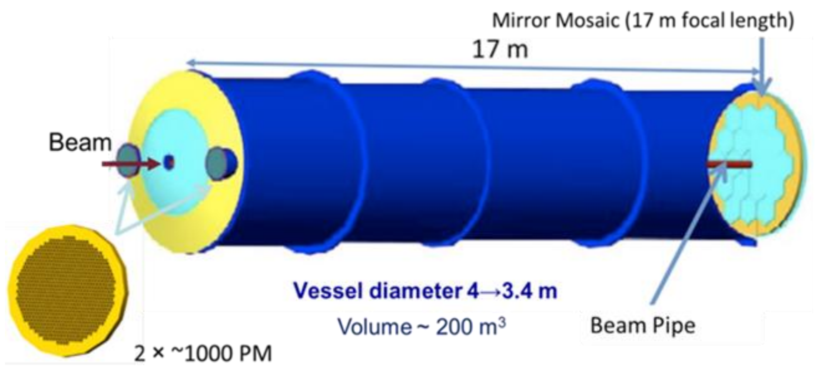
Abstract **P**rocessing **E**nvironment for **I**ntelligent **R**ead-**O**ut systems based on **N**eural networks

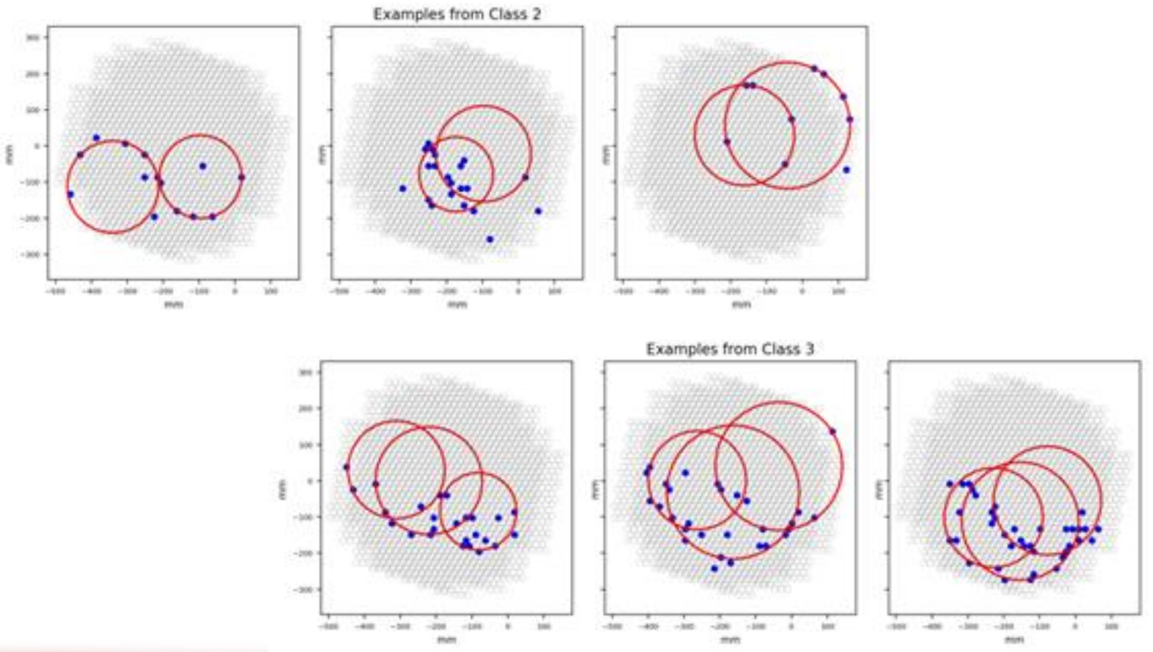


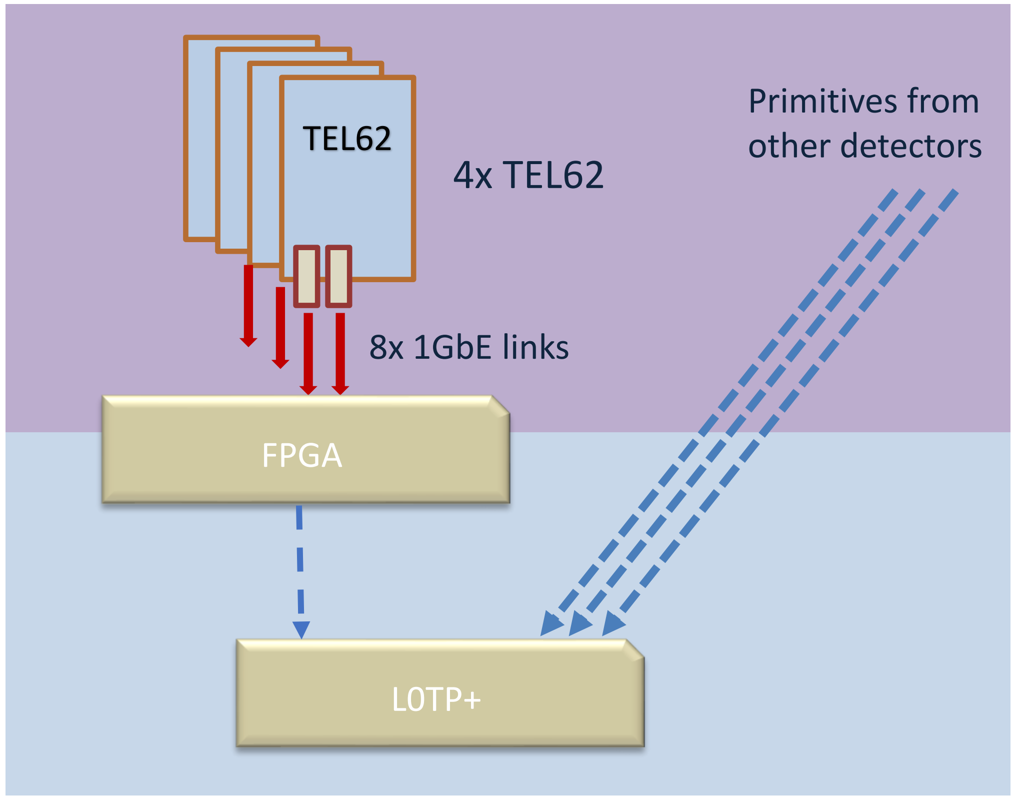
- Input data from several different channels (data sources, detectors/sub-detectors).
- **Data streams** from different channels recombined through the processing layers using **a low-latency, modular and scalable network infrastructure**
- Distributed online processing on heterogeneous computing devices (FPGAs for the moment) in *n* subsequent layers.
- Typically features extraction will occur in the first NN layers on RO FPGAs.
- More resource-demanding NN layers can be implemented in subsequent processing layers.
- Classification produced by the NN in last processing layer (e.g. pid) will be input for the **trigger processor/storage online data reduction stage for triggerless systems.**

PID in NA62 with the RICH using NN on FPGA at L0 Trigger

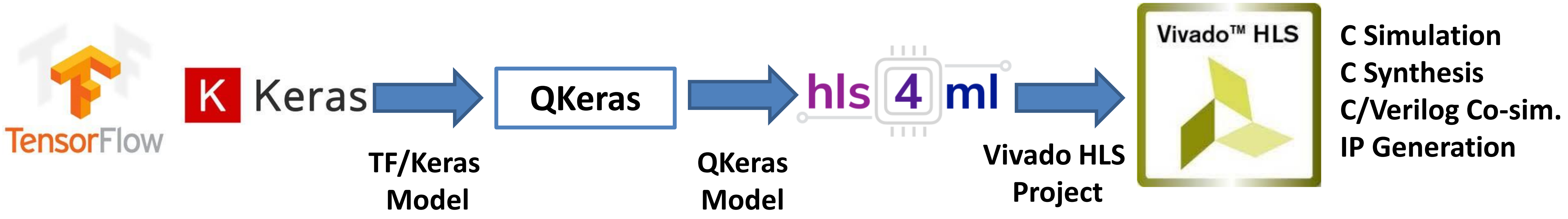
- Goal: for any event detected by the RICH provide an estimate for the **number charged particles** and the **number of electrons**
- Streaming readout processing on FPGA using Neural Networks for classification (10 MHz).
- Produce a new primitives stream for L0TP+
- **The main challenge is the proc. throughput**







Design and Implementation Workflow

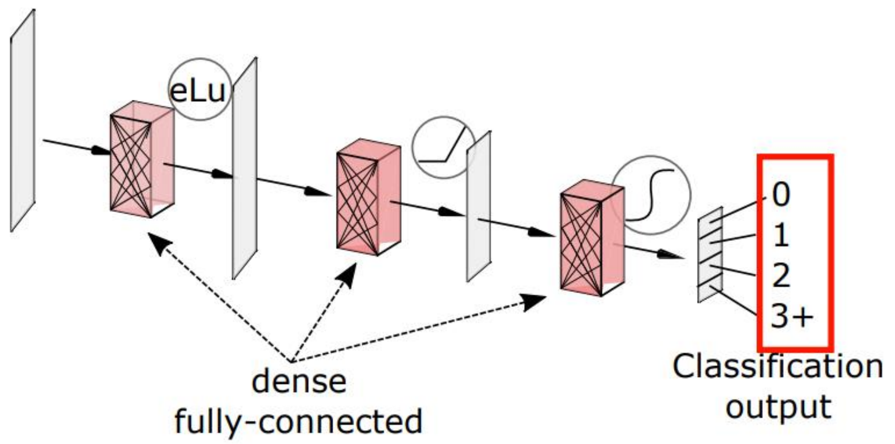


Design targets (efficiency, purity, throughput, latency) and constraints (mainly FPGA resource usage) must be taken into account and verified at any stage:

- Generation strategy of training and validation data sets.
- **TF/KERAS** NN architecture (number and kind of layers) and **representation of the input**
    - Training strategy (class balancing, batch sizes, optimizer choice, learning rate,...).
- **QKeras** Serach iteratively the minimal representation size in bits of weights, biases and activations, possibly by layer.
- **hls4ml** Tuning of REUSE FACTOR config param (low values -> low latency, high throughput, high resource usage), clock frequency.
- **Vivado HLS** co-simulation for verification of performance (experimented very good agreement with QKeras Model)

NN Architectures: Dense Model

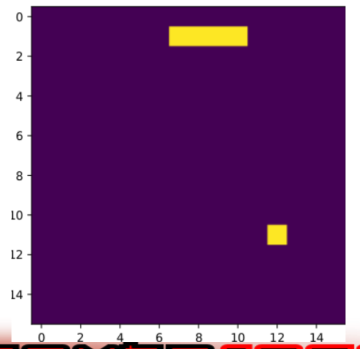
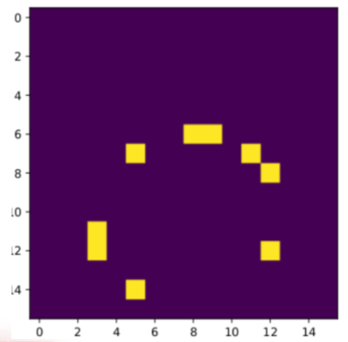
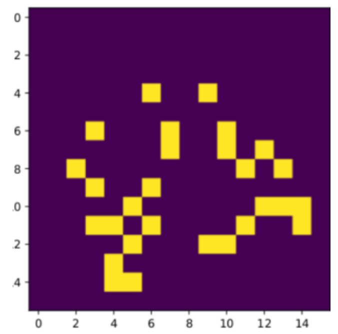
- Input representation: normalized hitlist (max 64 hits per event)
- **Output: 4 classes (0, 1, 2, 3+ rings)**
- **Quantization (fixed point)**
  - Weights and biases: 8 bits <8, 1>
  - Activations: 16 bits <16, 6>
- **FPGA resource usage (VCU118)** LUT 14%, DSP 2%, BRAM 0%
- **Latency: 22 cycles @ 150MHz**
- **Initiation Interval (II): 8 cycles**
- **Throughput: 18.75 MHz**

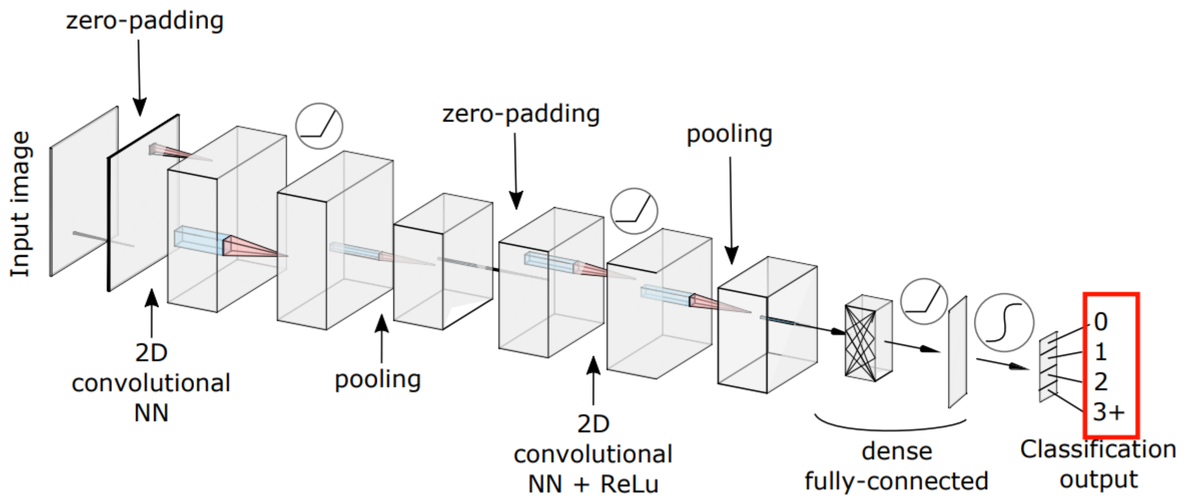


```
Layer (type)                 Output Shape              Param #
=================================================================
input1 (InputLayer)          [(None, 64)]              0

fc1 (Dense)                  (None, 64)                4160

act1 (Activation)            (None, 64)                0

fc2 (Dense)                  (None, 16)                1040

act2 (Activation)            (None, 16)                0

fc3 (Dense)                  (None, 4)                 68

softmax (Activation)         (None, 4)                 0

=================================================================
Total params: 5,268
```

NN Architectures: Convolutional Model

- Input representation: 16x16 images
- Output: 4 classes (0, 1, 2, 3+ rings)
- Quantization (fixed point):
  - Weights and biases: 8 bits <8, 1>
  - Activations:16 bits <16, 6>
- FPGA resource usage (Alveo U200)
  LUT 5.2%, FF 1.5%, DSP 4.8%,
  BRAM 0.05%
- Latency: 388 cycles @ 220MHz
- Initiation Interval (II): 369 cycles
- Throughput: 0.6 MHz

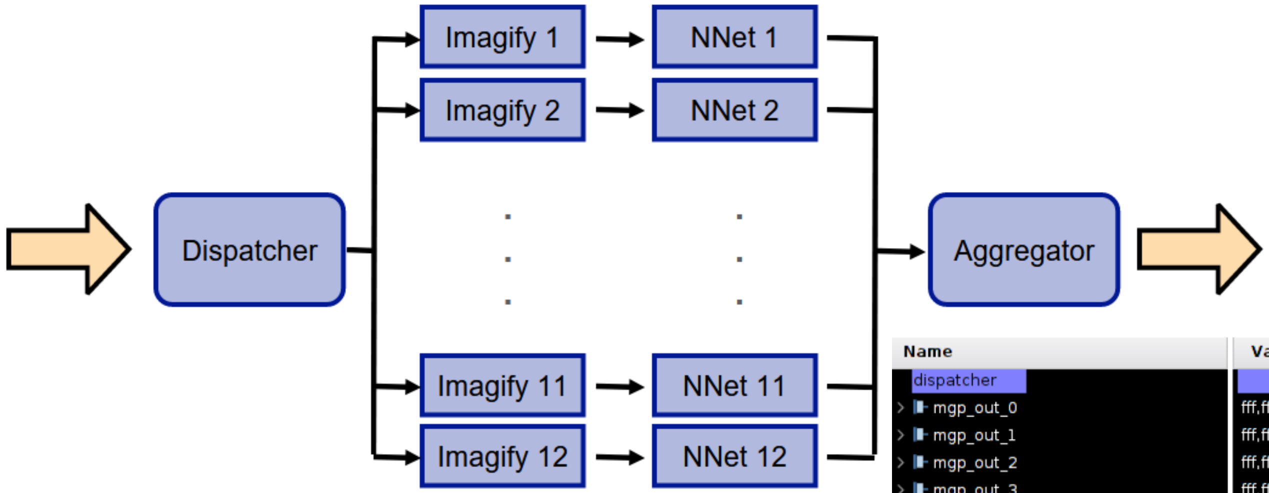




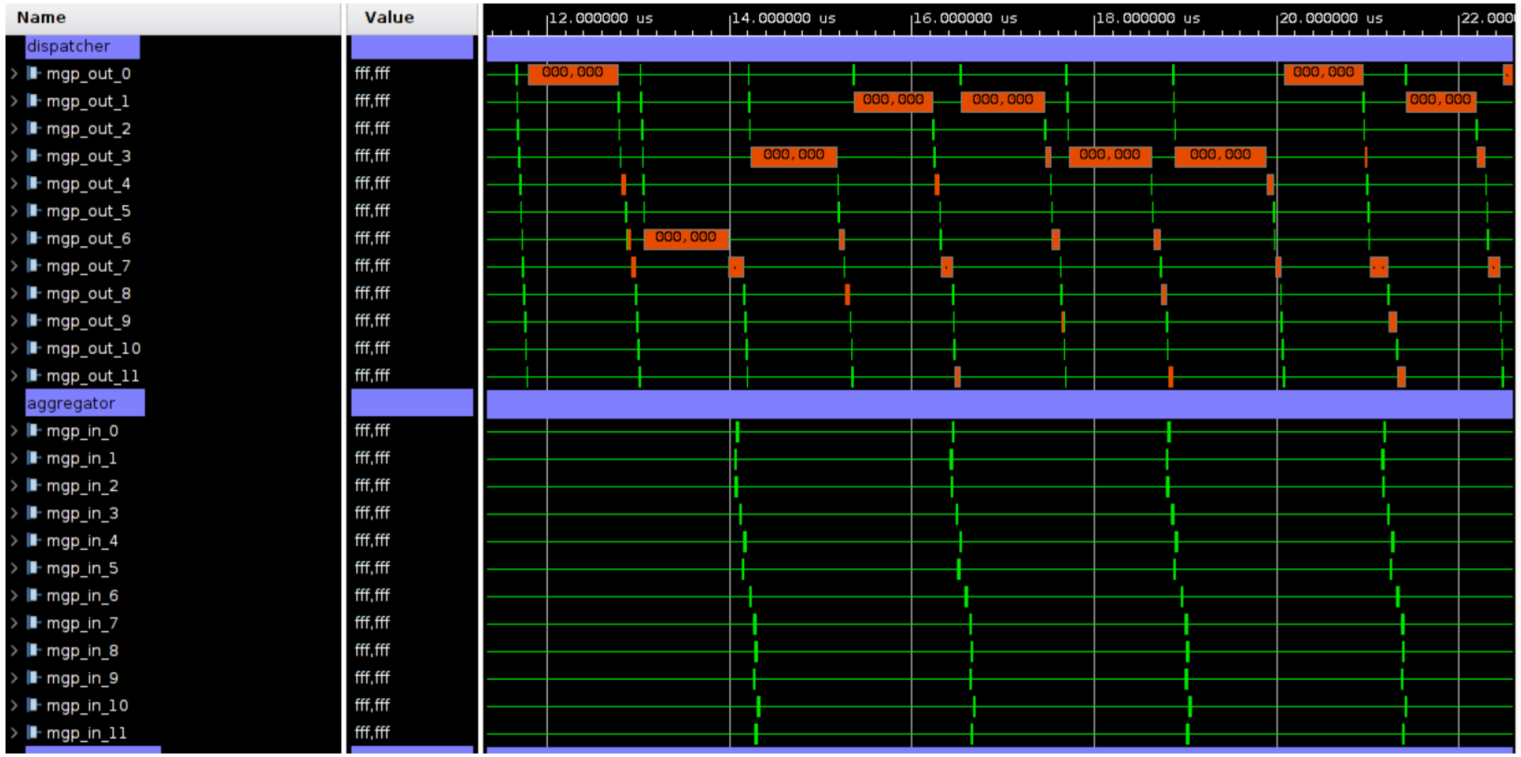


```
Layer (type)                  Output Shape              Param #
=================================================================
input1 (InputLayer)           [(None, 16, 16, 1)]       0

conv1 (Conv2D)                (None, 16, 16, 8)         80

act1 (Activation)             (None, 16, 16, 8)         0

maxp1 (MaxPooling2D)          (None, 8, 8, 8)           0

conv2 (Conv2D)                (None, 8, 8, 8)           584

act2 (Activation)             (None, 8, 8, 8)           0

maxp2 (MaxPooling2D)          (None, 4, 4, 8)           0

flatten (Flatten)             (None, 128)               0

fc3 (Dense)                   (None, 16)                2064

act3 (Activation)             (None, 16)                0

fc4 (Dense)                   (None, 4)                 68

softmax (Activation)          (None, 4)                 0

=================================================================
Total params: 2,796
```

Convolutional model – Kernel replication

Throughput is not enough to sustain L0 rate, but we can replicate the network multiple times, also on multiple devices if necessary.





Resources usage for 12 replicas:

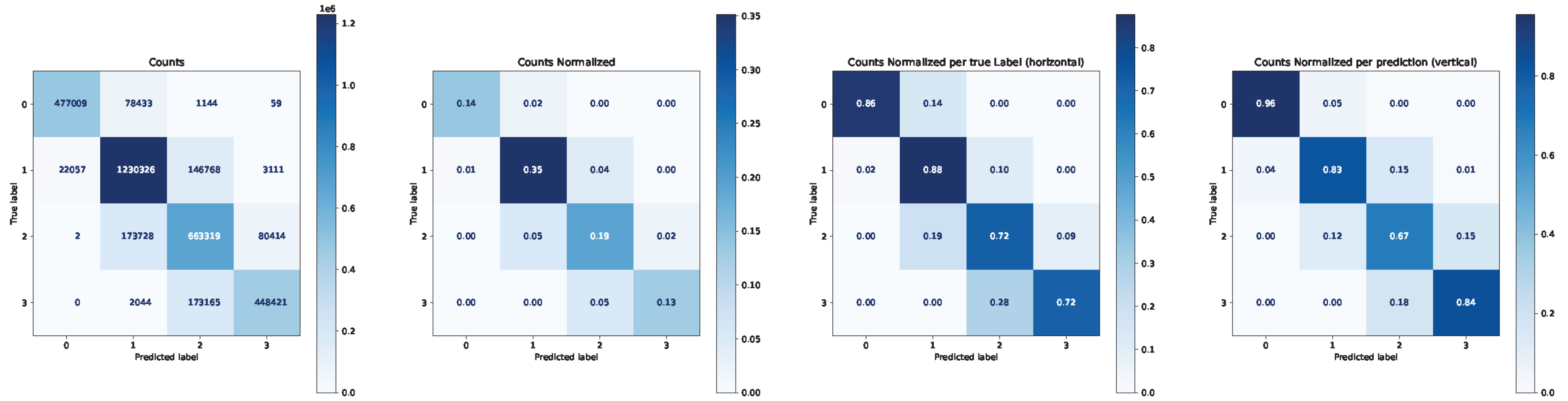
- LUT 74%
- FF 17%
- DSP 61%
- BRAM 1.4%

Processing time @220MHz: 137 ns per event

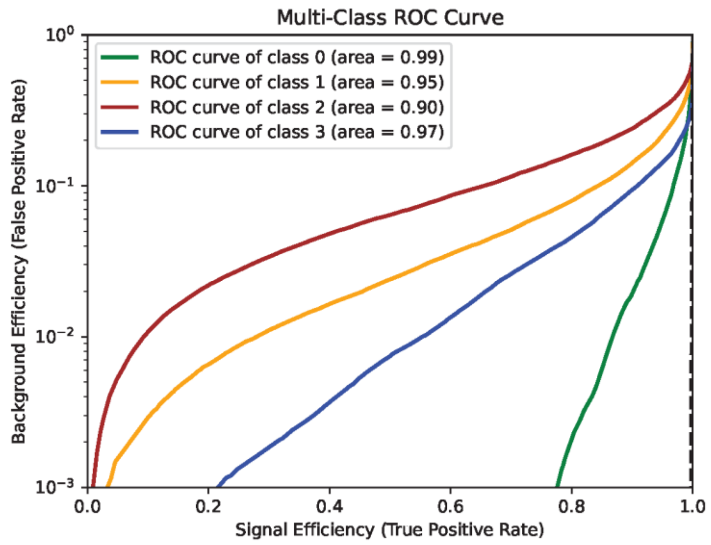
Processing throughput: 7.2 MHz

Dense Model: results for classification of number of rings

- Trained on 3 Mevents from run 8011, Validated on 3.5 Mevents from run 8893, ground truth label 1



Class 0 (0 rings) Efficiency 85.7 Purity 95.6
Class 1 (1 rings) Efficiency 87.7 Purity 82.9
Class 2 (2 rings) Efficiency 72.3 Purity 67.4
Class 3 (3+ rings) Efficiency 71.9 Purity 84.3

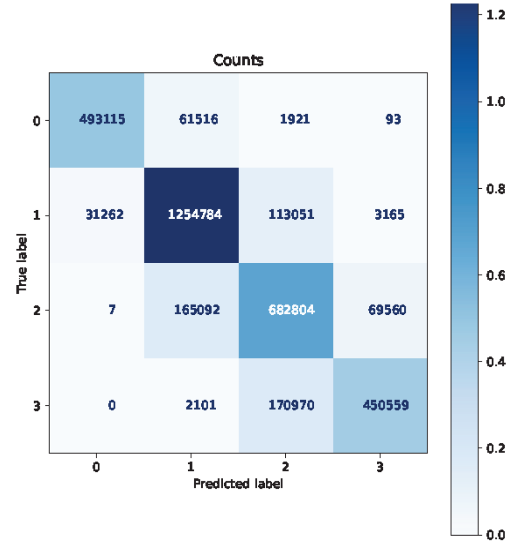


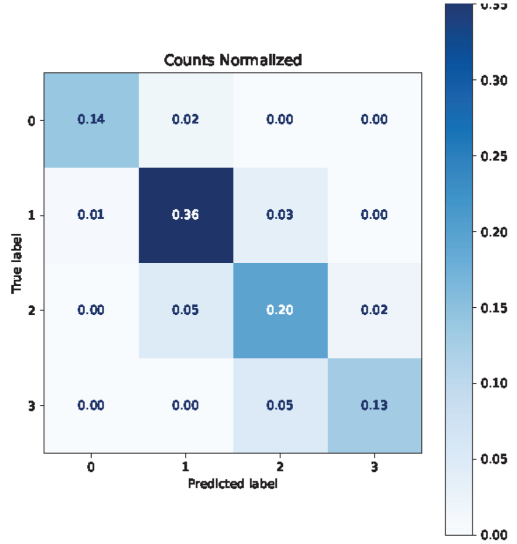
Efficiency = TP / (TP + FN)

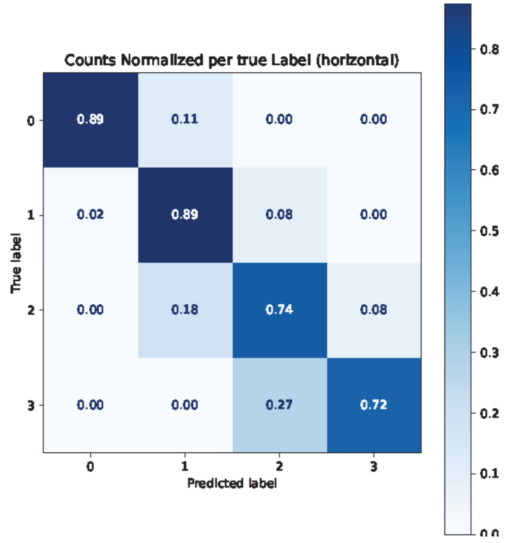
Purity = TP / (TP + FP)

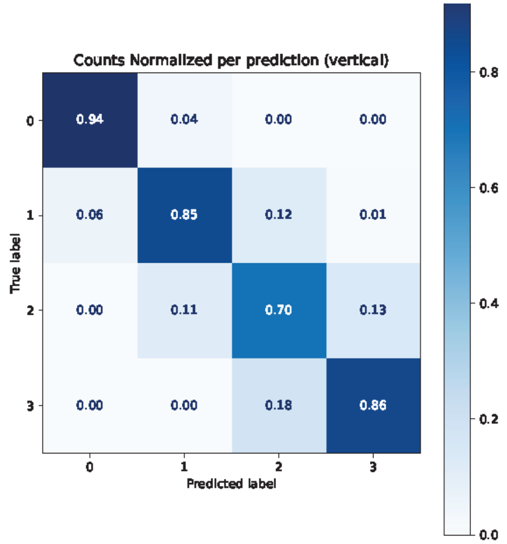
Convolutional Model: results for classification of number of rings

- Trained on 3 Mevents from run 8011, Validated on 3.5 Mevents from run 8893, ground truth label 1

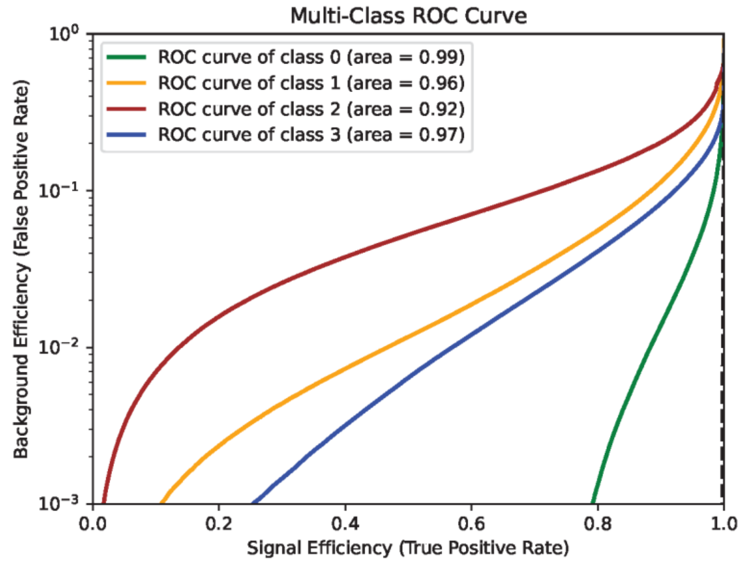








Class 0 (0 rings) Efficiency 88.6 Purity 94.0
Class 1 (1 rings) Efficiency 89.5 Purity 84.6
Class 2 (2 rings) Efficiency 74.4 Purity 70.5
Class 3 (3+ rings) Efficiency 72.2 Purity 86.1

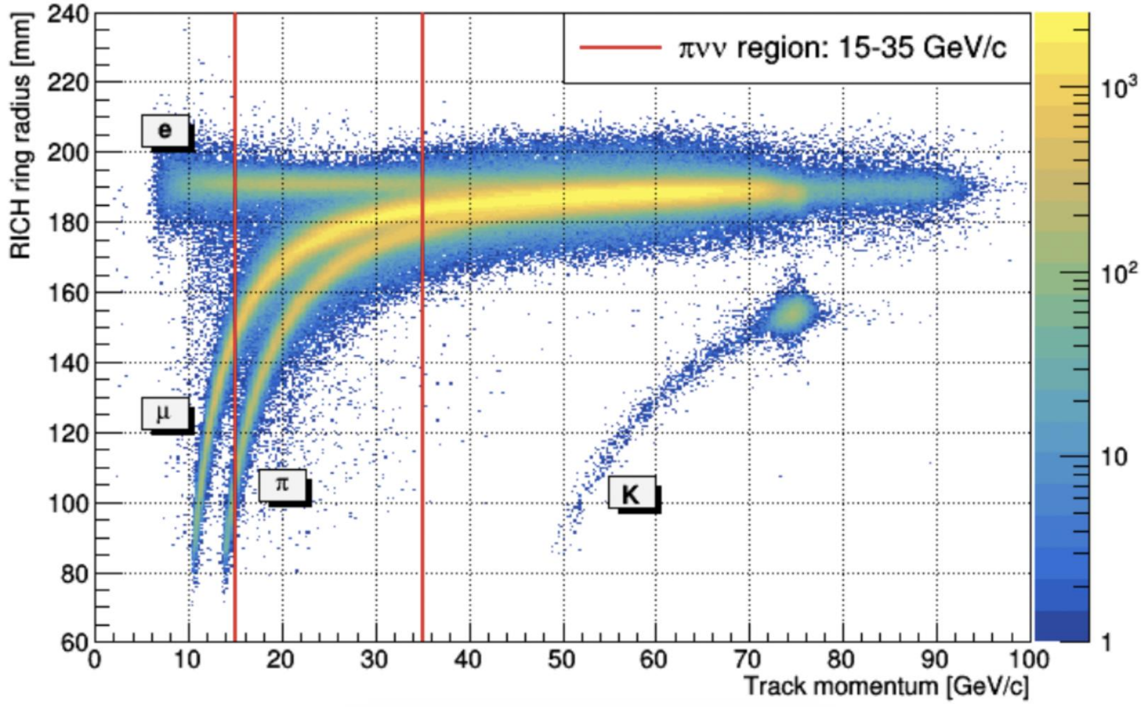


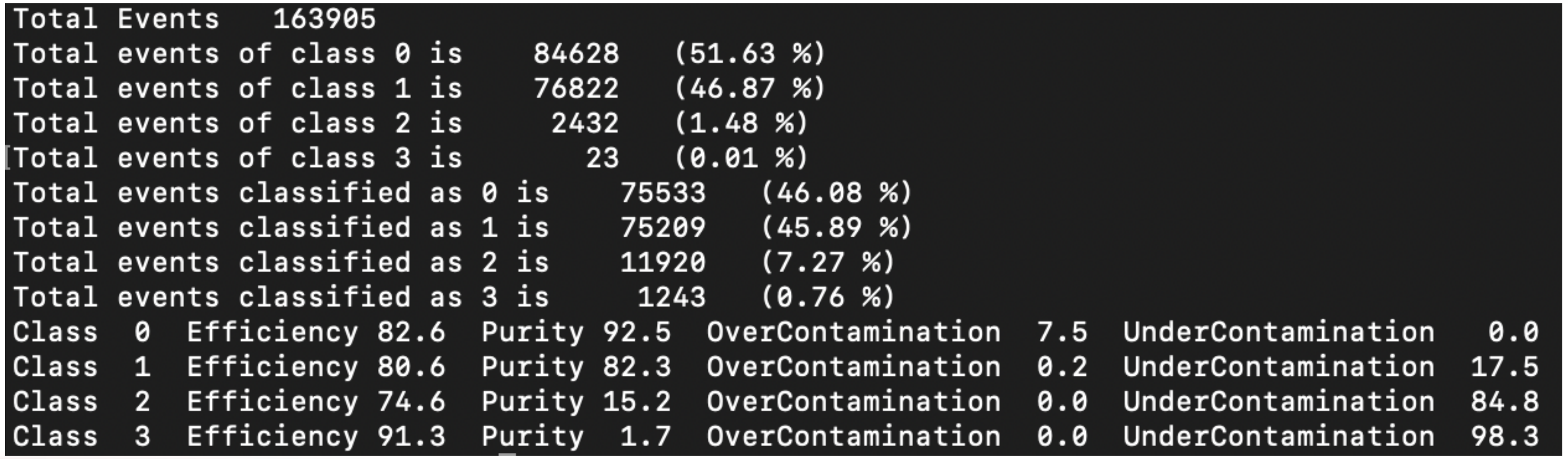
Efficiency = TP / (TP + FN)

Purity = TP / (TP + FP)

Results for classification of number of electrons

- Preliminary results for online classification of the number of "electrons" show that even the very simple NN architectures that we tested are capable, below 35 GeV/c momentum, of reaching a non-negligible performance (see terminal picture below).
- It can be improved for the online unfiltered event stream using a dedicated NN receiving in input data from other detectors (e.g. L0CALO).





```
Total Events    163905
Total events of class 0 is     84628    (51.63 %)
Total events of class 1 is     76822    (46.87 %)
Total events of class 2 is      2432    (1.48 %)
Total events of class 3 is        23    (0.01 %)
Total events classified as 0 is     75533    (46.08 %)
Total events classified as 1 is     75209    (45.89 %)
Total events classified as 2 is     11920    (7.27 %)
Total events classified as 3 is      1243    (0.76 %)
Class  0  Efficiency 82.6  Purity 92.5  OverContamination  7.5  UnderContamination   0.0
Class  1  Efficiency 80.6  Purity 82.3  OverContamination  0.2  UnderContamination  17.5
Class  2  Efficiency 74.6  Purity 15.2  OverContamination  0.0  UnderContamination  84.8
Class  3  Efficiency 91.3  Purity  1.7  OverContamination  0.0  UnderContamination  98.3
```

Conclusions

- We control the workflow for the implementation of real-time/high throughput classifiers on FPGA.
- The use case of the PID for the RICH shows that it is possible to reach good performance (at least for number of rings for now) even with a very limited usage of FPGA resources.
- This hints for applying the methodology also to:
  - less capable (i.e. front-end) FPGAs
  - complex design making use of a large fraction of FPGA resources (e.g. L0TP+)

  Either to improve the L0 trigger performance or to online tag and select events in a future streaming readout DAQ.
- Going to exploit the APEIRON framework to ingest and process primitive streams from other detectors (e.g. L0CALO to improve electron identification).
- We are going to deploy and test the described system in parasitic mode in NA62 next year.
- We are interested in the application of the framework in other contexts.

The APEIRON Team

@INFN Roma – APE Lab

A. Lonardo

P. Vicini

F. Lo Cicero

F. Simula

M. Martinelli

P. S. Paolucci

R. Ammendola

A. Biagioni

P. Cretaro

O. Frezza

C. Rossi

M. Turisini

@INFN Roma

A. Ciardiello

THANK YOU!