GNN-ETM: A Hardware Perspective

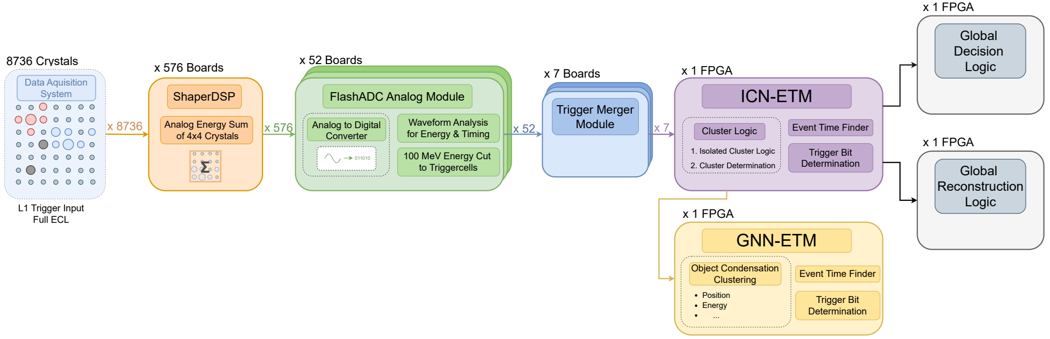
TRG Parallel Session - 50th B2GM

M. Neu, I. Haide, T. Justinger, T. Rädler, V. Dajaku, J. Becker, T. Ferber | Monday 24th February, 2025

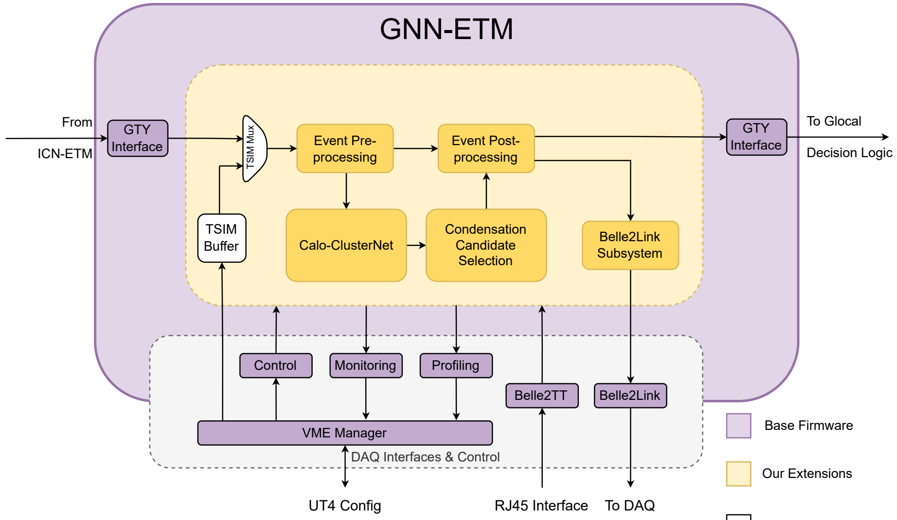
Agenda

- Today, I will give you an overview of the GNN-ETM architecture
- In addition, I will describe two paths for deployment:
  (a) the full integrated implementation on the UT4 (AMD Ultrascale XCVU-190). This implementation has been demonstrated on Christmas. Big Thanks to Unno-San!
  (b) a prototype implementation on the AMD VCK190.
- I will go in details about our current simulation setup for the GNN-ETM firmware.
- Last, I will give an overview of our plans for GNN-ETM from the hardware side.

ECL Trigger Overview



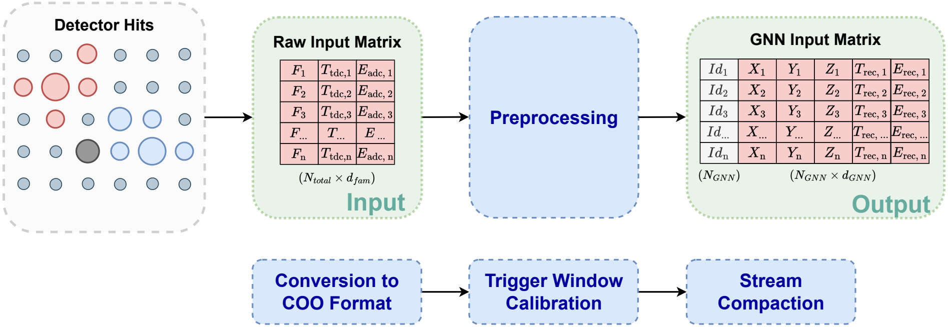
The GNN-ETM



🟪	Base Firmware
🟨	Our Extensions
☐	Under Construction

Event Preprocessing

- Extract all TCs above a $100\,$MeV, conversion into Coordinate-Offset-Offset (COO) format.
- Apply trigger window and reconstruct *event*T_0
- Conversion of E_{adc} into E_{rec} in GeV and T_{tdc} into E_{rec} in us.
- Written in Chisel, I plan to extend this module as a FuseSoc IP Core



F_1	$T_{\text{tdc},1}$	$E_{\text{adc},1}$
F_2	$T_{\text{tdc},2}$	$E_{\text{adc},2}$
F_3	$T_{\text{tdc},3}$	$E_{\text{adc},3}$
$F_{...}$	$T_{...}$	$E_{...}$
F_n	$T_{\text{tdc},n}$	$E_{\text{adc},n}$

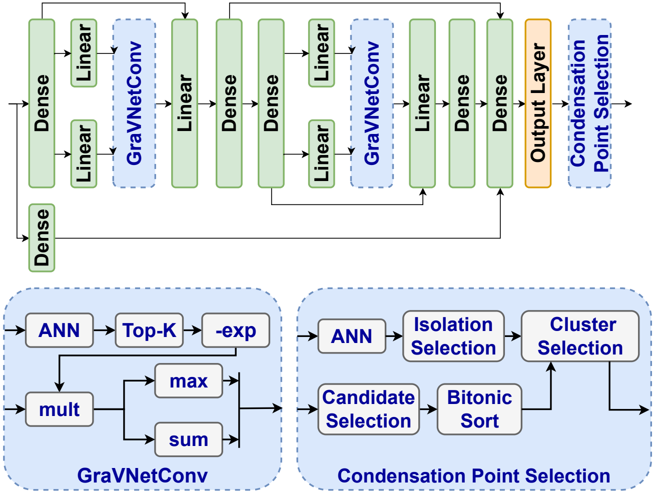
$(N_{total} \times d_{fam})$

Id_1	X_1	Y_1	Z_1	$T_{\text{rec},1}$	$E_{\text{rec},1}$
Id_2	X_2	Y_2	Z_2	$T_{\text{rec},2}$	$E_{\text{rec},2}$
Id_3	X_3	Y_3	Z_3	$T_{\text{rec},3}$	$E_{\text{rec},3}$
$Id_{...}$	$X_{...}$	$Y_{...}$	$Z_{...}$	$T_{\text{rec},...}$	$E_{\text{rec},...}$
Id_n	X_n	Y_n	Z_n	$T_{\text{rec},n}$	$E_{\text{rec},n}$

(N_{GNN}) $(N_{GNN} \times d_{GNN})$

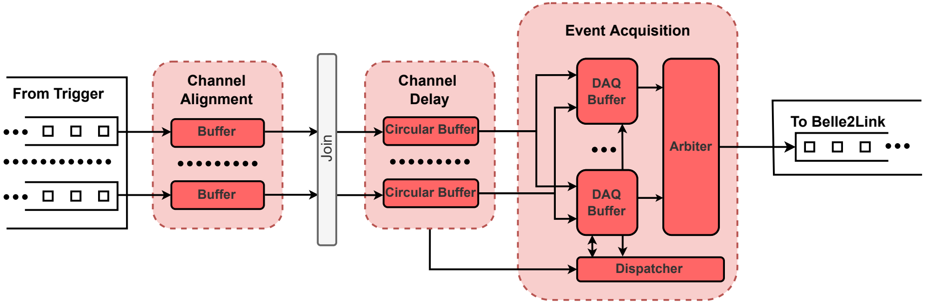
CaloClusterNet

Karlsruhe Institute of Technology



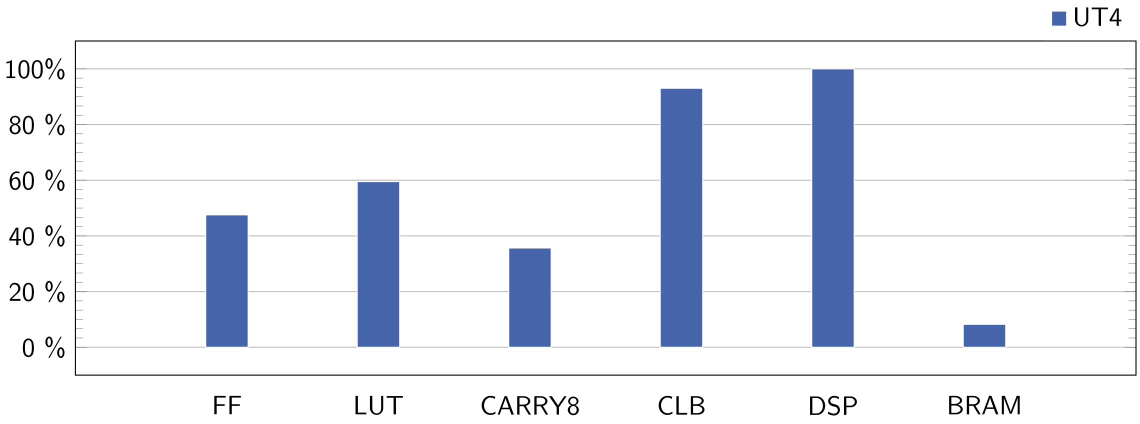
Belle2Link Buffer

- Adapted from ICN-ETM. Rewrite in Chisel.
- Configurable number of input channels, automatic alignment of different input streams-
- Configurable buffer depth and delay.
- I plan to make this module available as FuseSoC IP Core.



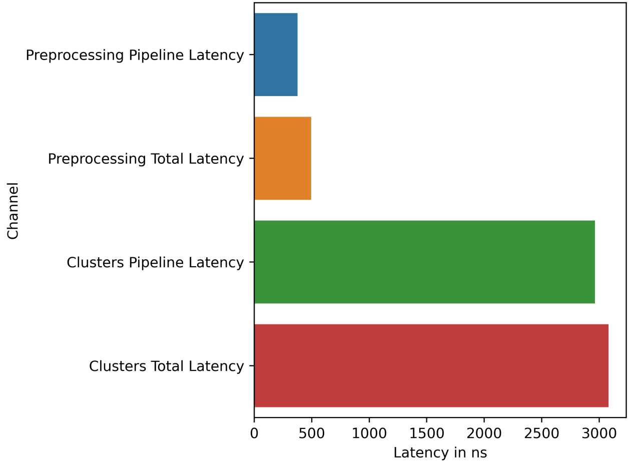
GNN-ETM Utilization

- I have implemented the GNN-ETM firmware on the UT4 (XCVU 190).
- The successful implementation required around $12\,\mathrm{h}$ for implementation.

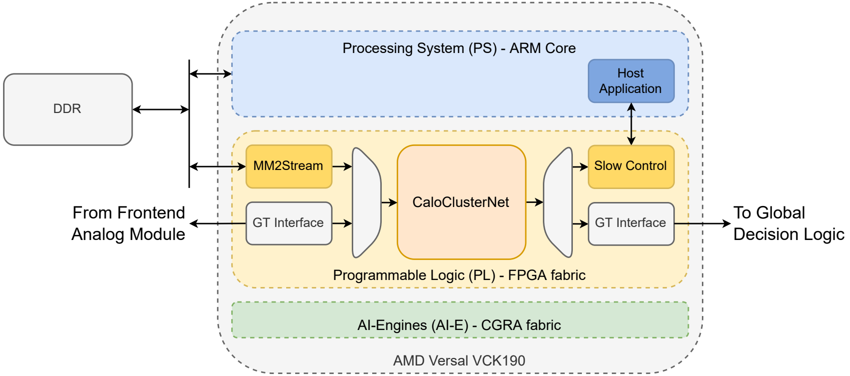


GNN-ETM Latency

- We use the following system frequencies:
  - $f_{sys} = 127\,\text{MHz}$
  - $f_{pre} = 254\,\text{MHz}$
  - $f_{gnn} = 127\,\text{MHz}$
- V31 requires a total latency of $3\,\text{us}$, which is above the required $1.6\,\text{us}$.
- Two options for improving the overal latency
  - Switch f_{gnn} to $254\,\text{MHz}$
  - Optimizations on algorithmic level
- **We have to improve the latency for using GNN-ETM actively in the first-level trigger system.**

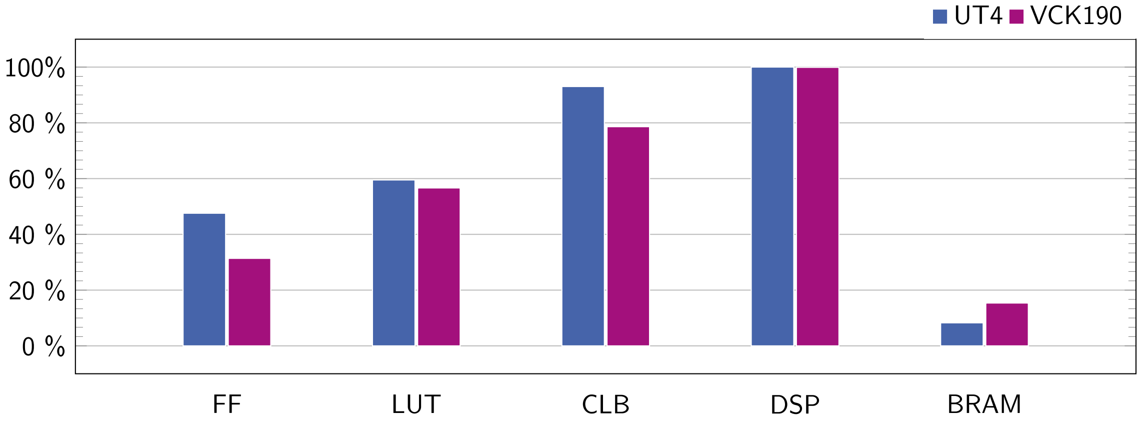


The VCK190 Prototype


Processing System (PS) - ARM Core
Host Application
DDR
MM2Stream
Slow Control
CaloClusterNet
From Frontend Analog Module
GT Interface
GT Interface
To Global Decision Logic
Programmable Logic (PL) - FPGA fabric
AI-Engines (AI-E) - CGRA fabric
AMD Versal VCK190

VCK190 Implementation

- I have implemented the CaloClusterNet on the AMD VCK190.
- No utilization of AI-Engines.
- The implementations required around $3\,\mathrm{h}$.



Thoughts to Go

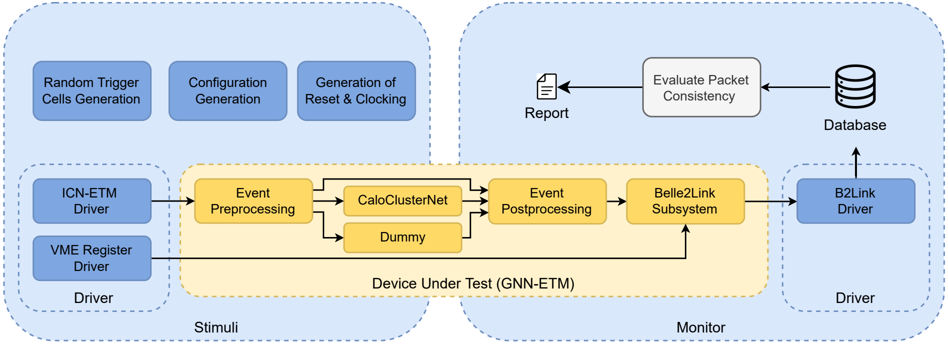
- I have successfully implemented our Caloclusternet GNN on FPGA, achieving a latency of 3 us.
- We have tested our model during physics run during Christmas. Big thanks to Unno-San :).
- We find that, the utilization on the VCK190 is slightly lower than on the UT4, even though VCK190 is smaller than UT4.
- **However, timing is much easier to achieve on the VCK190**.
- In order to include GNN-ETM in the active trigger decision, we must
  (a) reduce the overal latency below 1.6 us
  (b) find remaining inconsistencies between the software model and the hardware implementation.

Simulation

- Debugging on directly on hardware is often time-consuming.
- Therefore, I have implemented a wide range of simulation options for our system.
- I differentiate between two levels of simulations
  - **Control-level Behavioral Simulation:** Make sure that there is no deadlock in the system, interfaces to upstream systems, e.g. DAQ readout, work as expected. Check integrity of data packets.
  - **Data-level Behavioral Simulation:** Check the actual values inside data packets or on streaming interfaces. Full validation of the digital design.

Control-Level Behavioral Simulation

- We identify test cases for various operation conditions: Writing registers, link instabilities, transmission of B2Link packets.
- I have written test cases in CoCoTb using Python. QuestaSim 2023.4 for all simulations.
- In the future, solved bugs will receive a specific test case which covers the issue.



Data-Level Behavioral Simulations

- Based on the control-level simulations, we also conduct data-level simulations
- From top to bottom, accuracy and simulation time increases
- Post-Route-Sim is extracted from Design Check Point for a given firmware version.
- End-To-End-Sim is currently under construction.

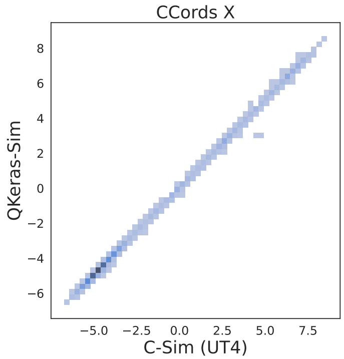
Name	Description
QKeras-Sim	Limited accuracy, optimized for efficient training.
C-Sim	Bit-accurate simulation before Vitis HLS.
Co-Sim	Cycle-accurate behavioral simulation using Vitis HLS
RTL-Sim	Cycle-accurate behavioral simulation after Vitis HLS.
Post-Route-Sim	Cycle-accurate behavioral simulation after synthesis, placement and routing
End-To-End-Sim	Cycle-accurate behavioral simulation including Pre- & Postprocessing, excluding Belle2Link
Hardware	Actual execution on the target platform, either UT4 or VCK190.

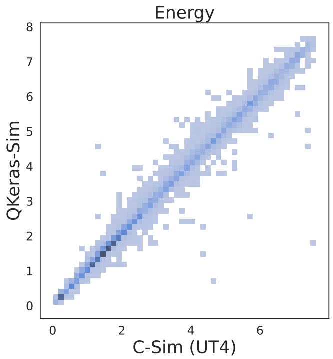
Simulation Results for UT4 (GNN-ETM)

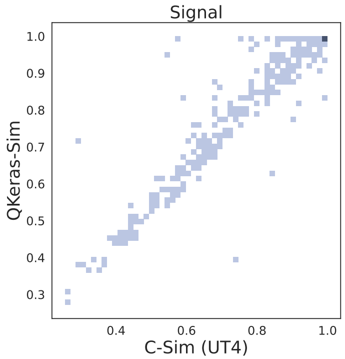
- ✓ indicates a 100% match
- ● indicates a reasonable good match
- ✗ indicates something is very wrong
- So far, I was not able to reproduce the errors we saw on the Christmas runs with GNN-ETM
- **Evaluations on hardware are required**

	Q-Keras-Sim	C-Sim	Co-Sim	RTL-Sim	Post-Route-Sim	End-To-End-Sim	Hardware
QKeras-Sim	✓						
C-Sim	●	✓					
Co-Sim	●	✓	✓				
RTL-Sim	●	✓	✓	✓			
Post-Route-Sim	●	✓	✓	✓	✓		
End-To-End-Sim							
Hardware	✗	✗	✗	✗	✗		✓

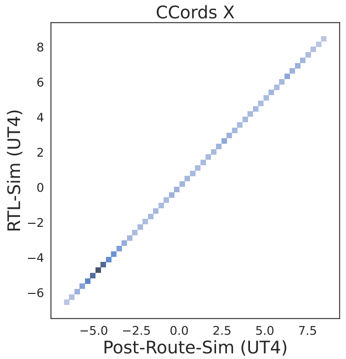
C-Sim (UT4) vs QKeras-Sim Examples

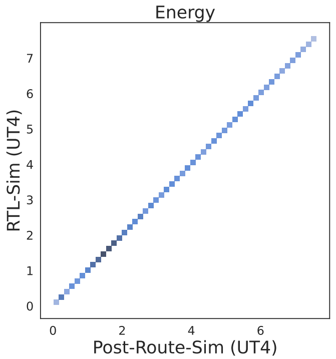


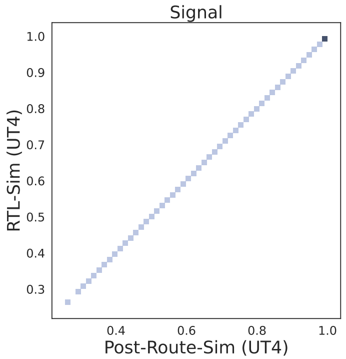




RTL-Sim (UT4) vs Post-Route-Sim (UT4) Examples







Design Study on VCK 190

- I have implemented CaloClusterNet without preprocessing, on the VCK190
- Achieved 100% match between C-Sim and hardware execution
- Bug which I had reported to AMD has not yet been answered
- Workaround: Changing *ap_ufixed* to *ap_fixed* resolved the issue.
- Downside is a slight increase in hardware utilization

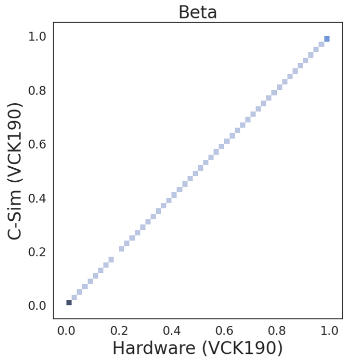
Name	UT4	VCK190	Comment
QKeras-Sim	✓	✓	Identical
C-Sim	✓	✓	
Co-Sim	✓	✓	
RTL-Sim	✓	✓	
Post-Route-Sim	✓	✗	Could be implemented, but time consuming
End-To-End-Sim	✓*	✗	Infeasible on VCK190
Hardware	✓	✓	No Preprocessing on VCK190

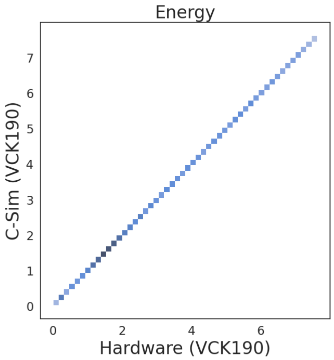
Simulation Results for VCK190

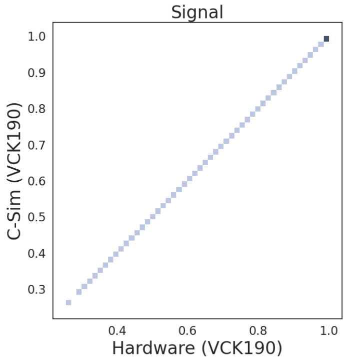
- ✓ indicates a 100% match
- ● indicates a reasonable good match
- ✗ indicates something is very wrong
- AMD Toolchain works as expected
- **It seems our model QKeras differs from our hardware implementation. Investigation ongoing**

	Q-Keras-Sim	C-Sim	Co-Sim	RTL-Sim	Hardware
QKeras-Sim	✓				
C-Sim	●	✓			
Co-Sim	●	✓	✓		
RTL-Sim	●	✓	✓	✓	
Hardware	●	✓	✓	✓	✓

Hardware (VCK190) vs C-Sim (VCK190) Examples







Next Steps

- I will be here at KEK until 6th March.
- I am currently learning how to do local and cosmic runs at Belle II
- I have prepared a list of tests for my stay at KEK, using GNN-ETM V0.31
  - **Issue a** Check if the bitstream generation was faulty for some reason. Same synthesis as baseline, but alternative implementation strategy.
  - **Issue b** Check with dummy, whether features are transmitted correctly via B2LinkBuffer.
  - **Issue 3** We would like to check the alignment between ICN- and GNN-ETM in depth, by adjusting the delay parameter.
  - **Issue 5** We would like to adjust the beta and isolation criteria for the cluster selection algorithms separately.

Next Steps

- During our Christmas runs, we observed a number of open issues. Most importantly, the algorithmic performance differs between the hardware implementation and our software mode.
- I am to find the root cause of this issue as soon as possible
- I have prepared a list of issues in Gitlab (here)
- I would like to migrate the GNN-ETM repository into FuseSoc.

Conclusion

- I have implemented our CaloClusternet on the GNN-ETM and started to validate its functionality in the experiment **during christmas**. Big Thanks to Unno-San for his great support!
- I have set up a comprehensive simulation environment for the GNN-ETM in order to identify all remaining issues which we found during the Christmas runs.
- I will continue to improve the current firmware version, reducing the overall latency and preparing the system for deployment.
- In parallel, we will conduct studies on the scalability of the Caloclusternet in the first-level trigger, also considering an ECL upgrade.