Machine Learning and Hardware Acceleration for HEP Trigger System

Zhao-Zhi Liu, Shuang-shaung Zhang, Yun-qing Mao, Zi-bing Bai, Jing-xun Deng, Qi-Dong Zhou

2026.07.02

2026年超级陶粲装置研讨会

Trigger/online system

Trigger / online processing

- Raw data from detector arrive continuously
- Trigger decide whether to store events
- Low latency and high throughput are mandatory

General trigger link

Detector front-end ⇨ Hardware trigger (L1) ⇨ Software trigger (HLT) ⇨ server storage /Offline analysis

Level	Platform	Characteristics
Hardware trigger (L1)	FPGA	Low/fixed latency; simple algorithm
Software trigger (HLT)	CPU / GPU	Loosen latency; Relative complex algorithm
Offline analysis	CPU	Flexible computing resources Maximum physics performance

From tradition to hardware-accelerated neural-network

		Trigger rate	Throughput	Latency budget
Belle II	L1	Max 30 kHz	30 GB/s	Fixed 4.4 us
	HLT	30 kHz	3GB/s	s / ms level
STCF	L1	1 MHz	100 GB/s	5 us
	HLT	450 kHz	30 GB/s	Target about 100 ms

STCF and Belle II

- About 6400/8000 CPUs (added/planed) in Belle II HLT
- With same latency level, STCF has a higher trigger rate

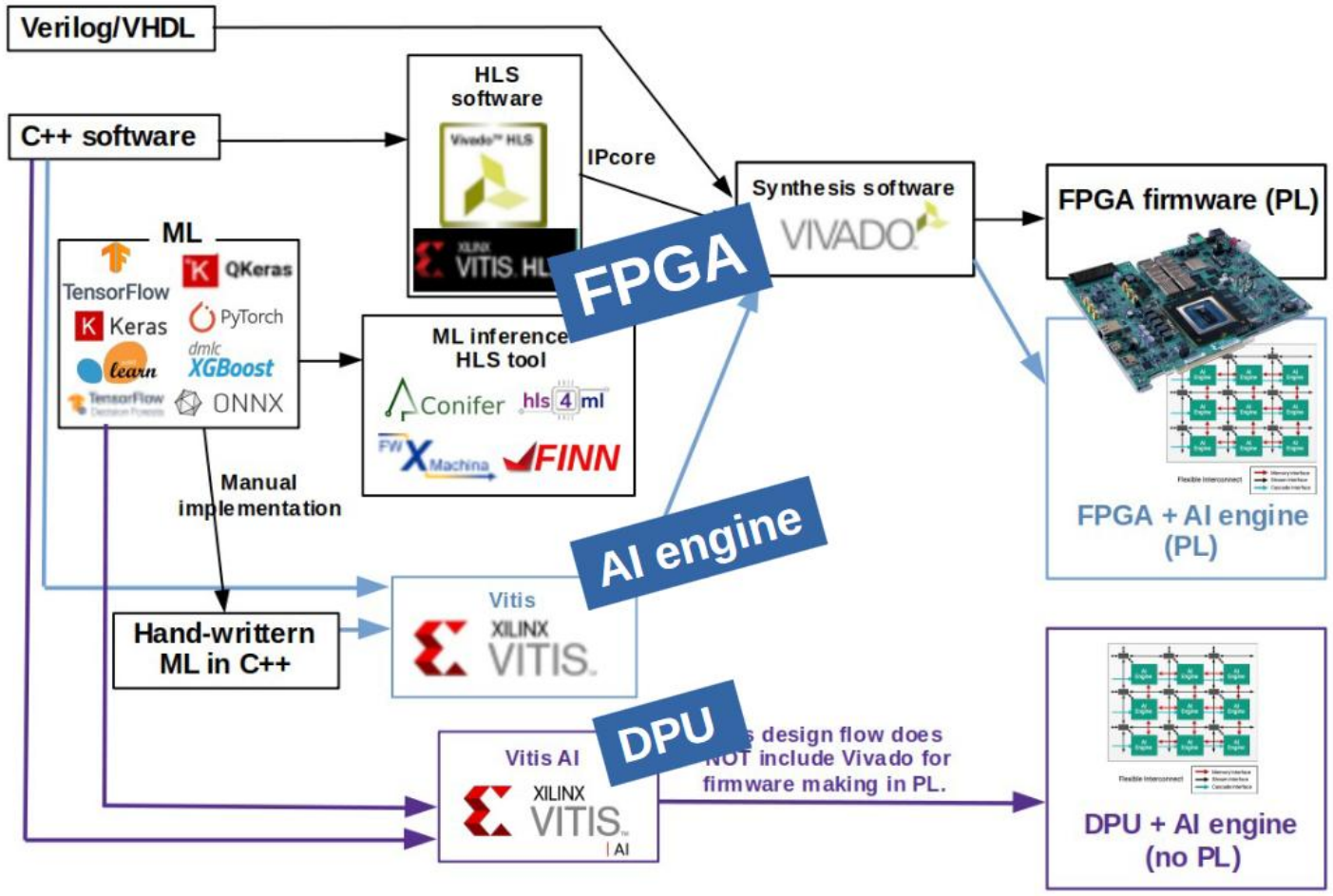
Neural-network

- Algorithm needs to upgrade for more complex jobs under same latency
- **Higher latency and hardware resource consumption**

Hardware acceleration

- Parallelism and pipelining
- Mapping onto heterogenous platforms like FPGA, AI-engine, DPU and GPU

Our studies



L1

Path I: Belle II L1 CDC z-trigger

- DNN + light transformer
- Algorithm-hardware co-design

HLT

Path II: GNN for trigger processing

- GNN
- Heterogenous design

Path III: STCF CNN on DPU

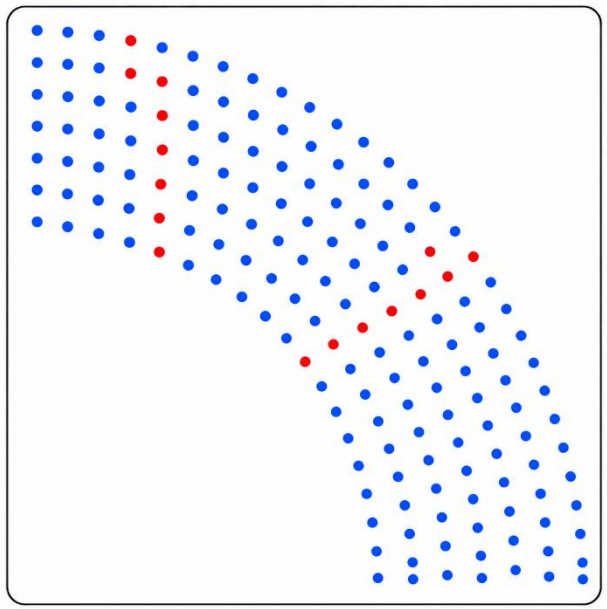
- CNN
- Deployment of CNN on DPU

Path IV: L3 trigger (+GPU)

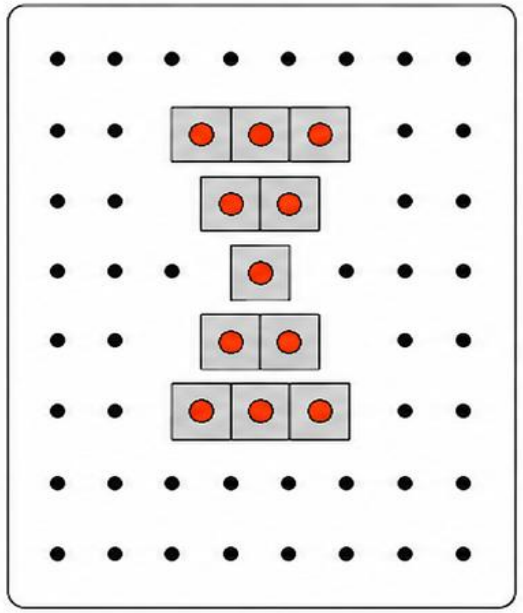
- HLT
- CPU-based reconstruction algorithm

Z trigger in Belle II CDC

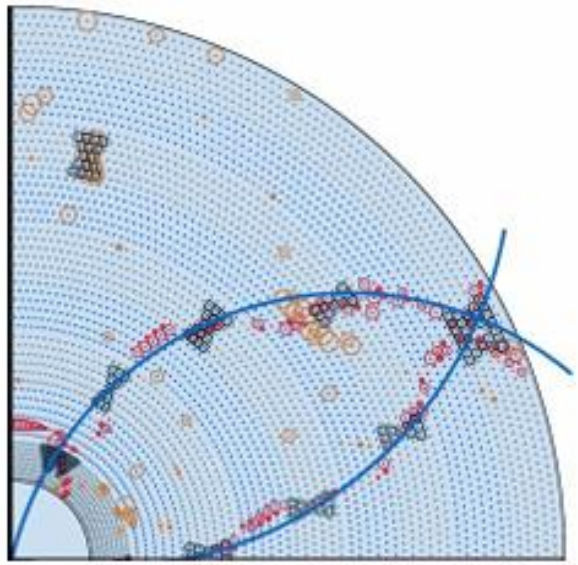
1) Central Drift Chamber hits



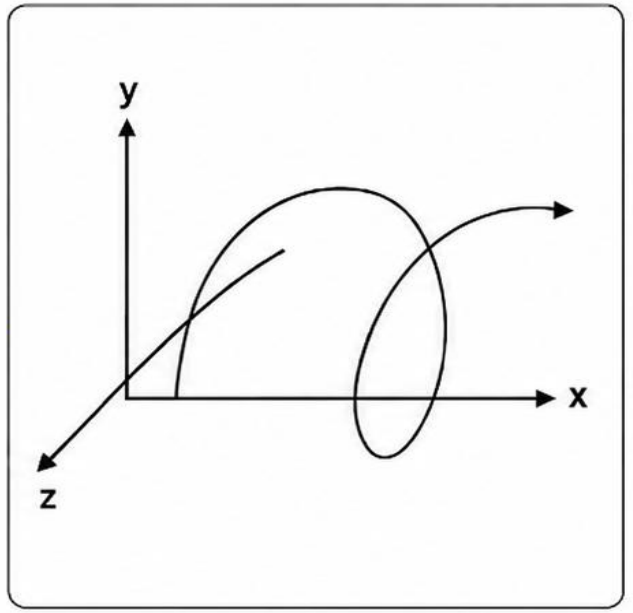
2) TSF (Track Segment Finder)



3) 2D track



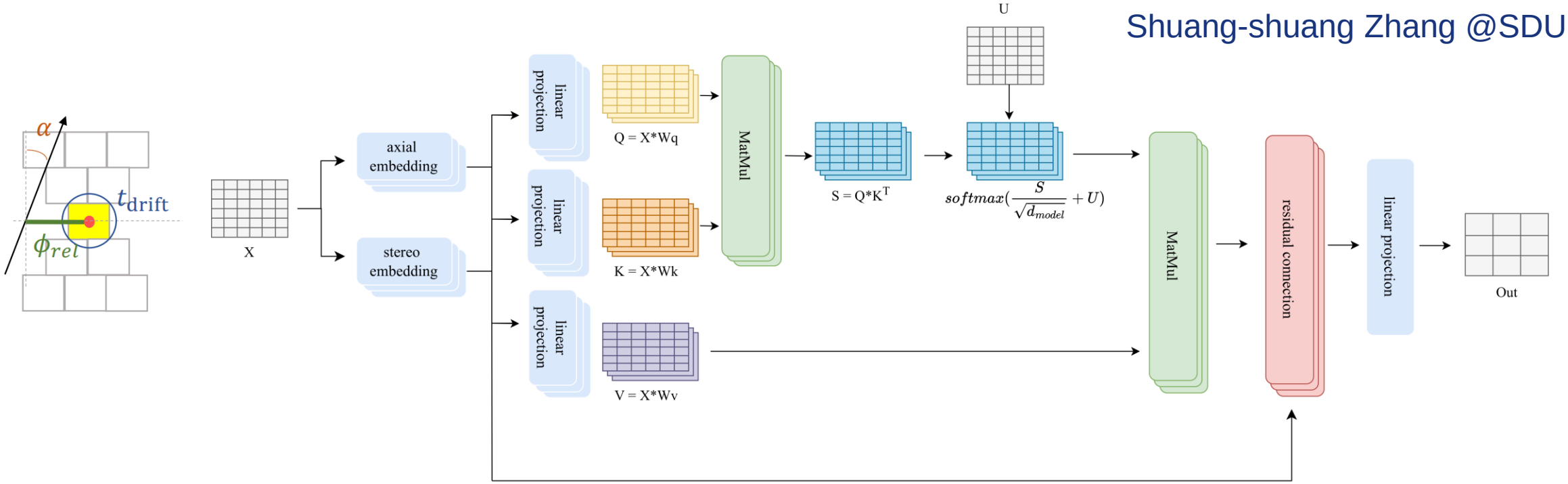
4) 3D track



Central Drift Chamber hits produced by charged particles → Find track segments by TSF → Build 2D track from TS using Hough transformation → Add stereo information to construct 3D track for z trigger

Transformer-based DNN

Shuang-shuang Zhang @SDU



Input:

Drift time t_{drift} , wire relative location Φ_{rel}, crossing angle α, Drift time for all other wires in the stereo wire

Transform-based mode:

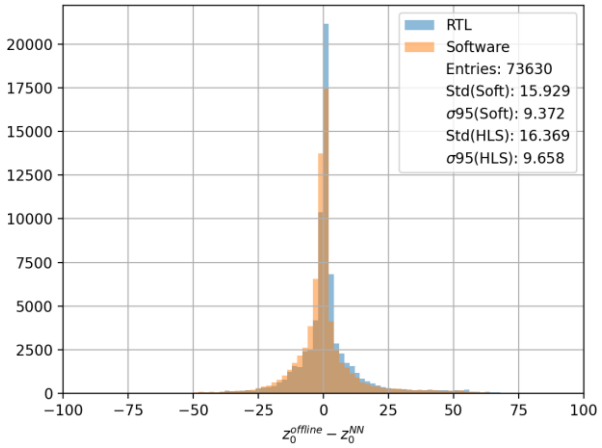
Detector-aware tokenization, **Physics-constrained attention**, FPGA-oriented lightweight design

Output:

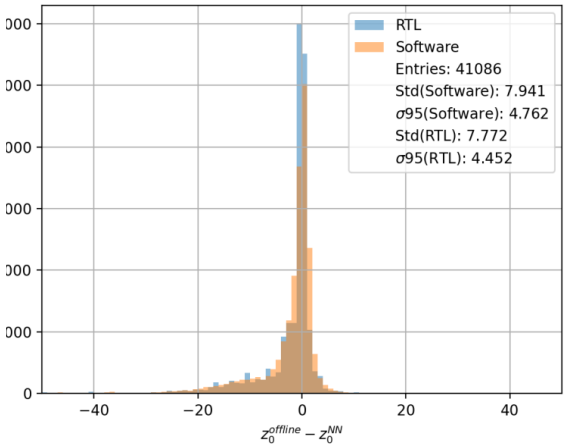
Vertex position (z0), Track polar angle (θ), Probability of classification (Q)

Research Framework

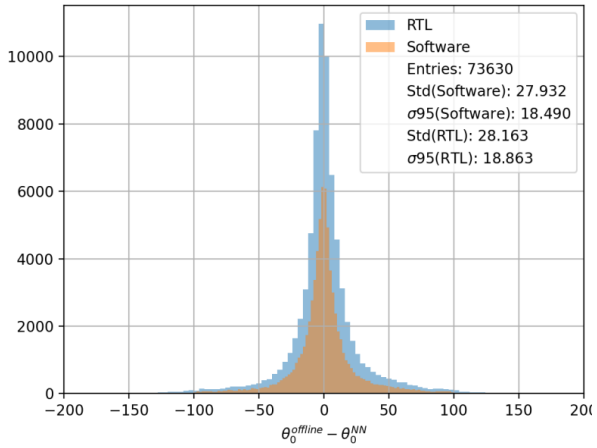
Delta track z in all range



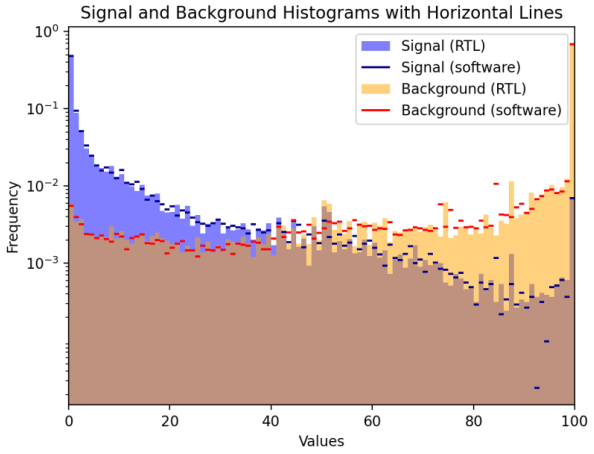
Delta track z in |z|<1cm



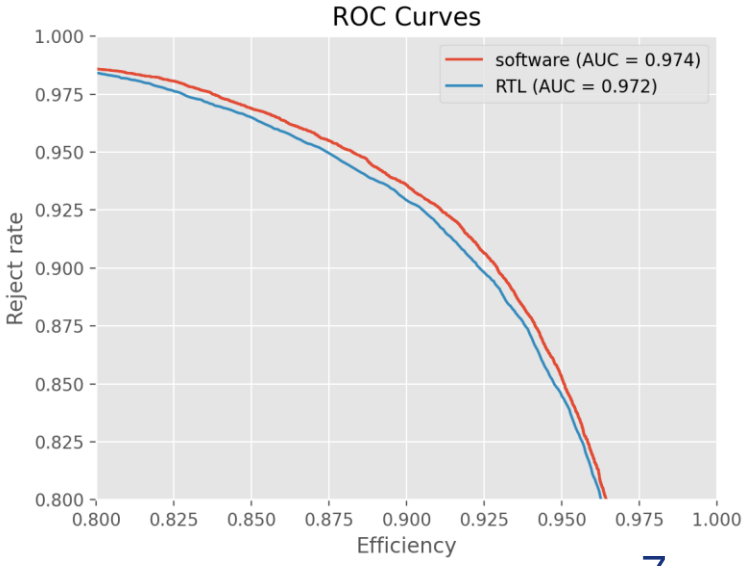
Delta track θ



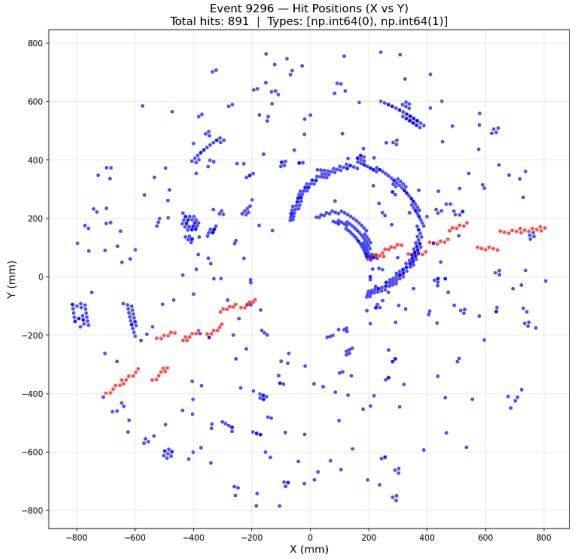
Classifier output



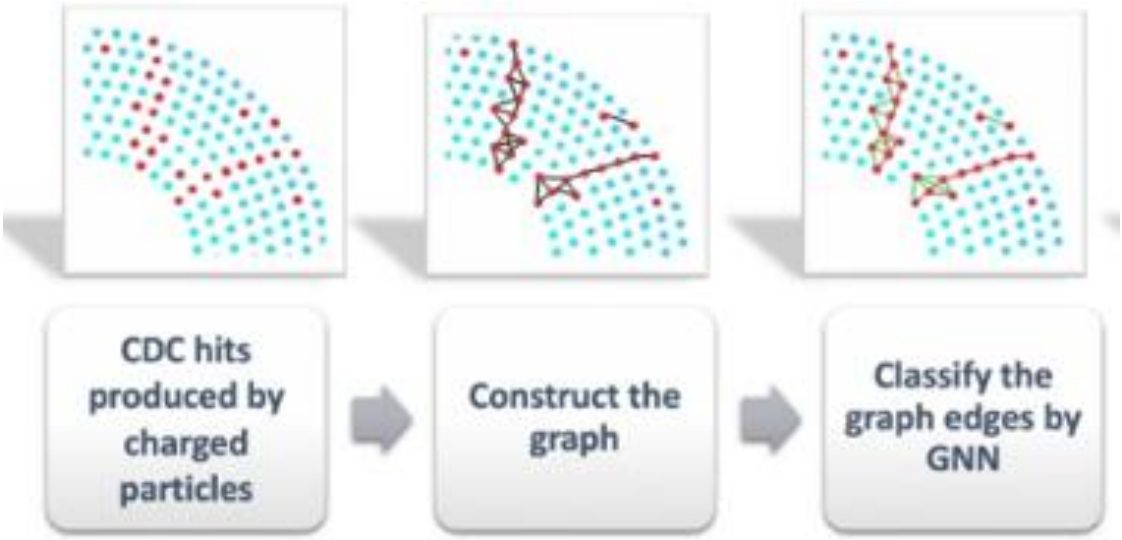
- Latency:71clock=554.7ns; require: <600ns
- FPGA resourse (VPK120:Versal premium w/0 AI-engine) resourse usage:
  - DSP: ~60%, LUT: ~65%, others<15%
- σ^{z0}=4.5cm,σ^{θ}=18°, Signal efficiency ~95%, Background rejection rate ~88%
- AUC do not get large drop comparing RTL and software simulation
- **Transform-based trigger in Hardware timing debugging phase**



GNN for CDC Noise Filtering



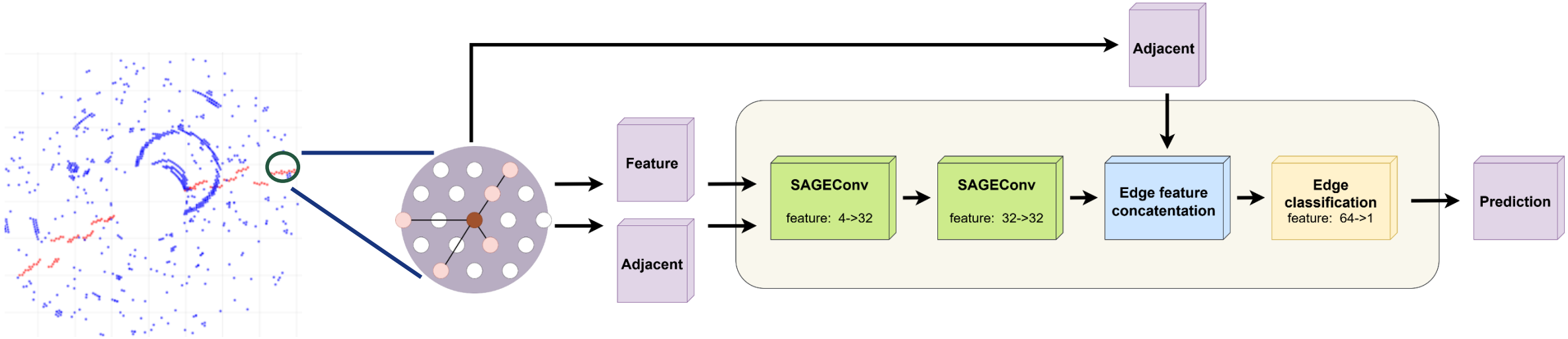
CDC raw data map



- The increasing luminosity leads to the growth of background.
- CDC denoising is a hit-association problem rather than isolated hit classification.
- Detector hits naturally form sparse relational graphs, where:
  a) nodes represent detector hits
  b) edges represent candidate hit connections
- The denoising task can thus be taken as a edge classification GNN problem.

Graph construction and GNN model

Zhao-zhi Liu @SDU



Epsilon-nearest neighbor (ε-nn)

- Node feature: r, phi, TDC, ADC
- Adjacent representation: source and destination node indices

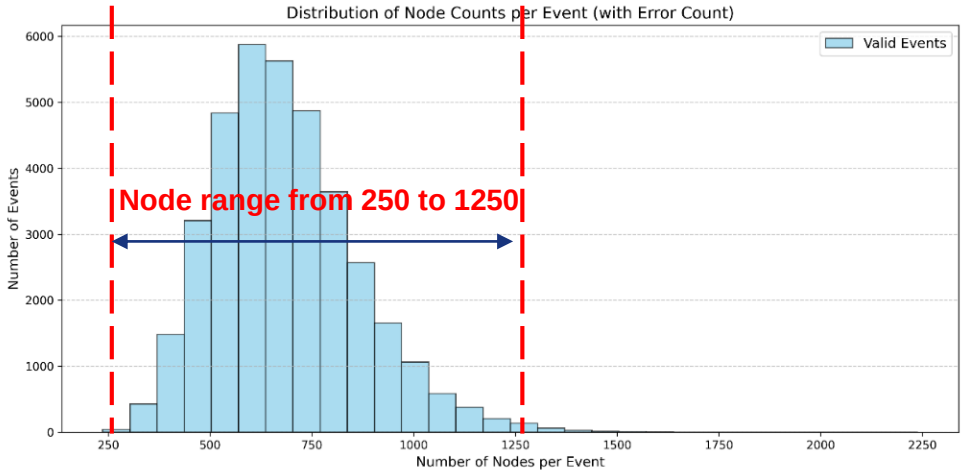
Graph neural network

- Sageconv: message passing
- Edge feature concatenation: output feature of edges
- Edge prediction score

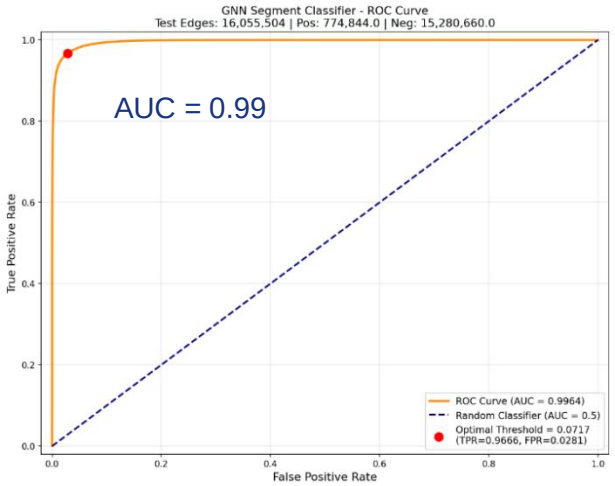
Software performance

Data set (STCF MC):

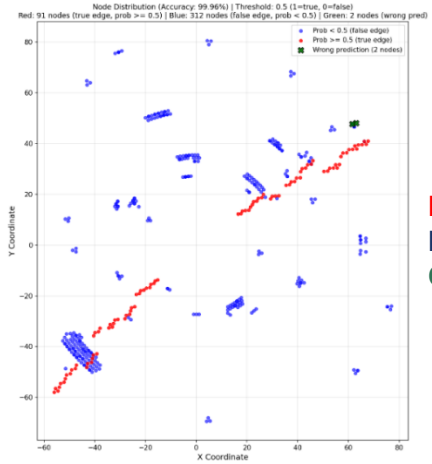
- J/psi → u+ u-
- 40,000 events (randomly split into 14,000 train, 3,000 valid, 3,000 test and 20,000 independent)
- 10 times background overlay



Distribution of nodes per event after ε-nn



ROC map



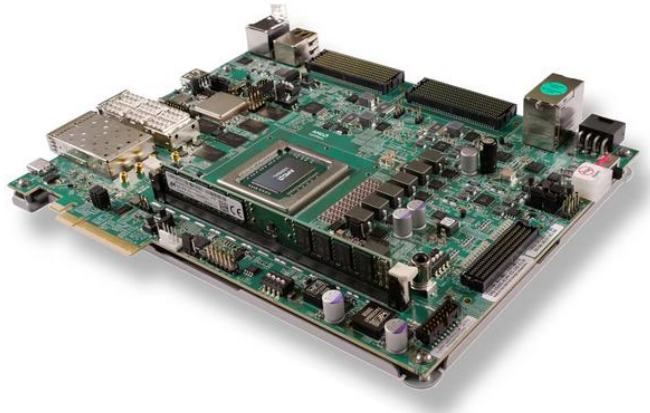
Hit map after training

Red true signal
Blue true background
Green false

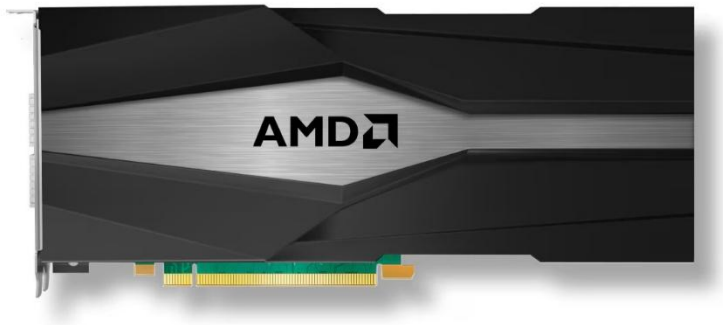
- After ε-nn, over 90% of the events has a node number of less than 1000
- The accuracy of the model is 0.99 and the AUC is about 0.99
- The signal efficiency is 0.88 and the signal purity is 0.91

Versal ACAP: Adaptive Compute Acceleration Platform

Versal premium VPK120



Versal AI core VCK190 evaluation kit

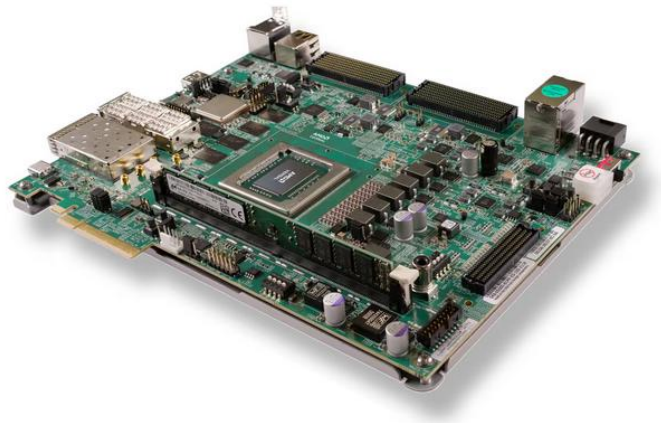


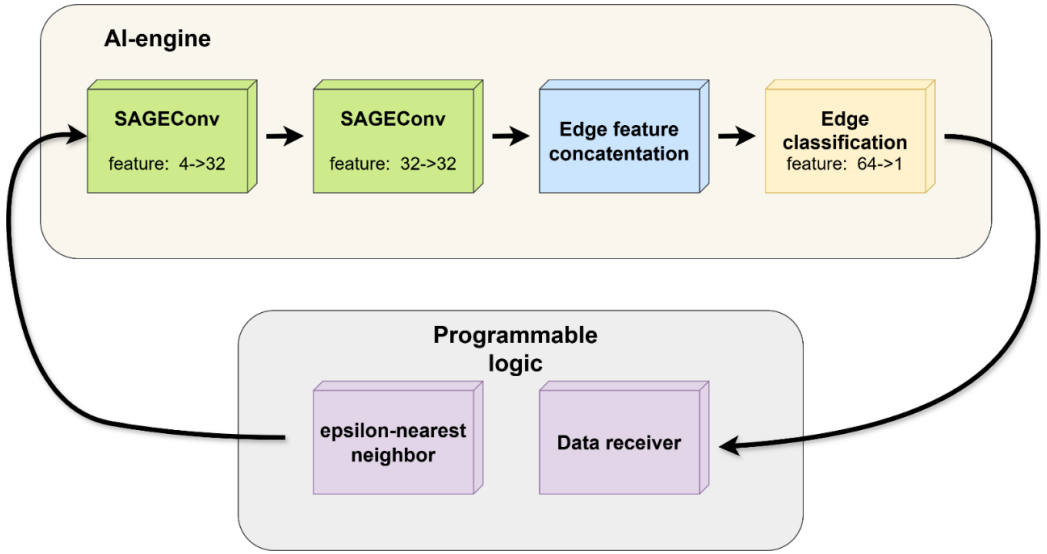
Development card VCK5000

	VPK120	VCK190	VCK5000
Main feature	100G/600G ethernet, 100+ Gb/s PAM4 receivers, etc.	AI-engines (VLIW, SIMD)	Server acceleration card
Core	VP1202	VC1902	VC1902

FPGA, AI-engines and PS are achieved by particular hardware component in different board,
But the DPU is a type of IP core on the software level

Hardware deployment





Programmable logic:

- The graph construction (ε-nn) **densifies the sparse matrix information**.
- Data storage and management.

AI-engine:

- Dense operation
- SAGEConv, MLP, aggregation, edge classification

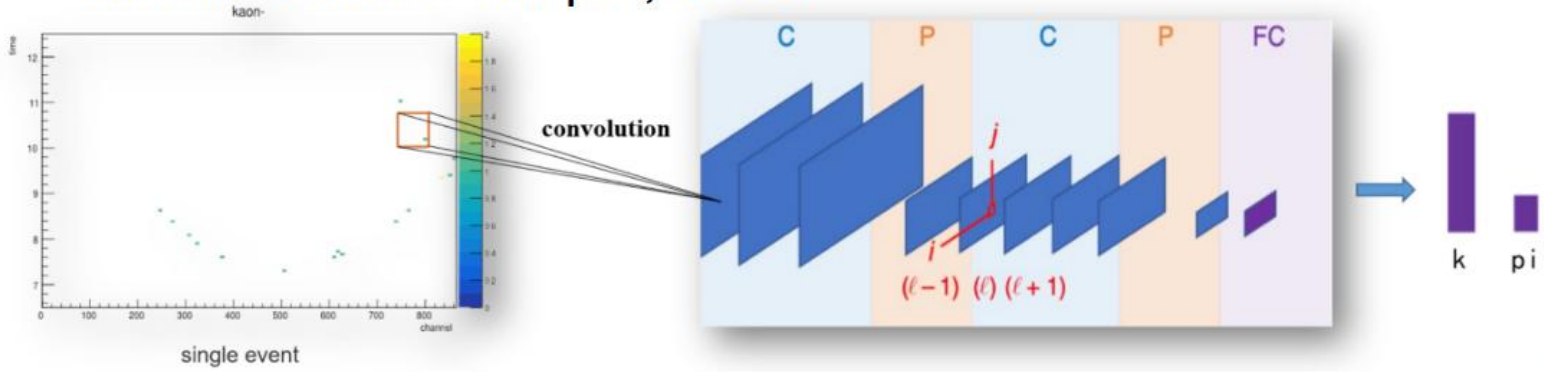
Event for test	CLK frequency	ε-nn latency	SAGEConv1 latency	SAGEConv2 latency	Edge classification	Total latency
210 nodes 1712 edges	100 MHz	~ 1 ms	526 us	1.2 ms	6us * 1712	12ms

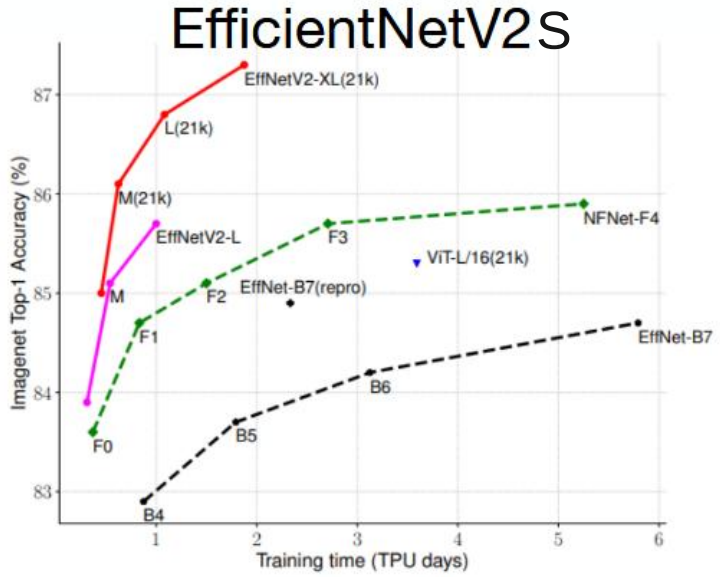
Baseline version deployment is about to finish, optimization will start soon

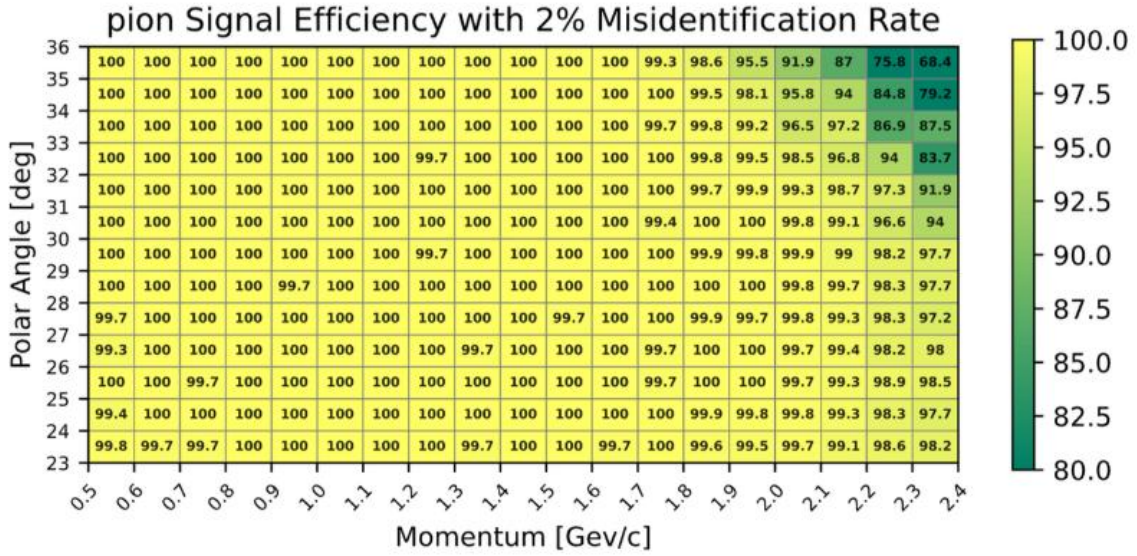
CNN for STCF PID

- DTOF as a PID subdetector of STCF
- CNN algorithm developed for Kaon/pion identification
- Kaon/Pion MC simple, 800w

Z. Yao et al.@SDU



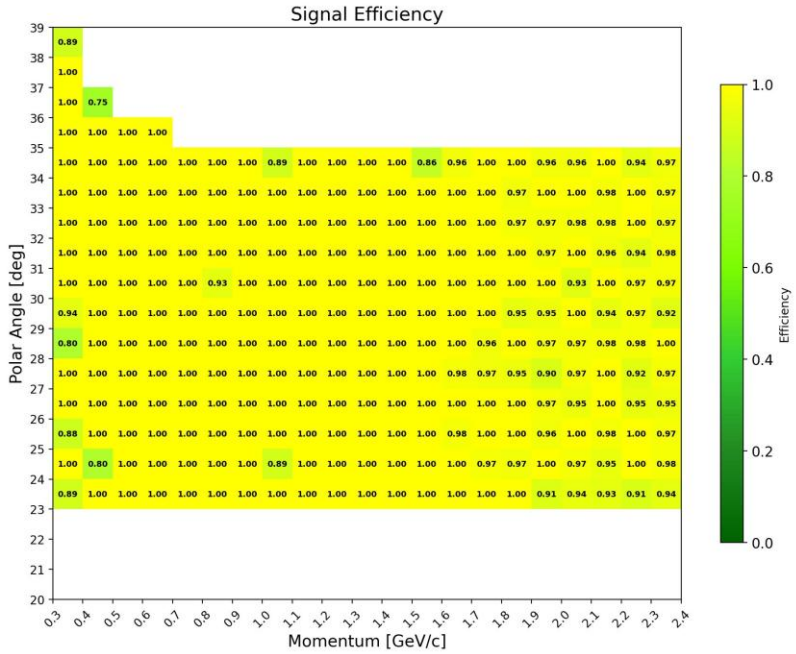




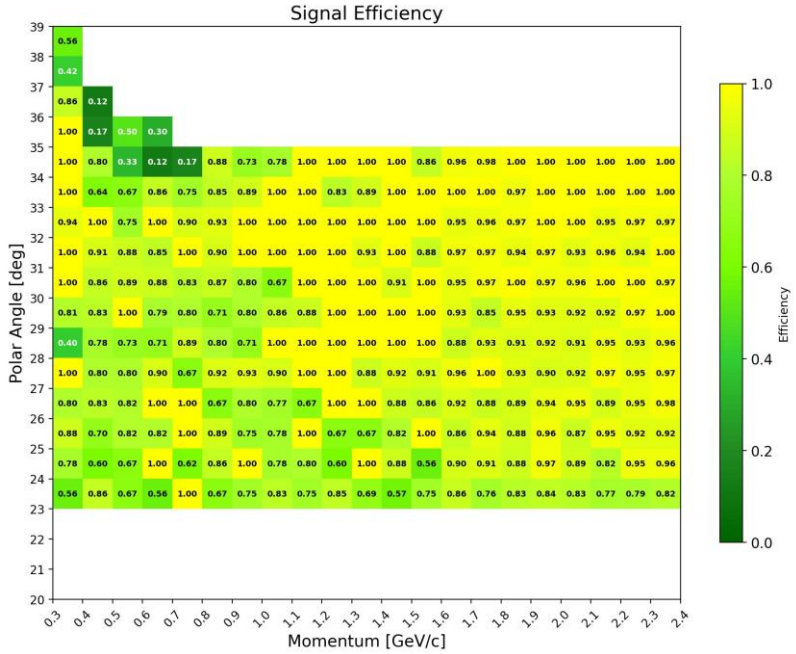
CNN performance

For deployment on the VCK5000 DPU, the CNN model is partial user-defined to adapt the DPU operators

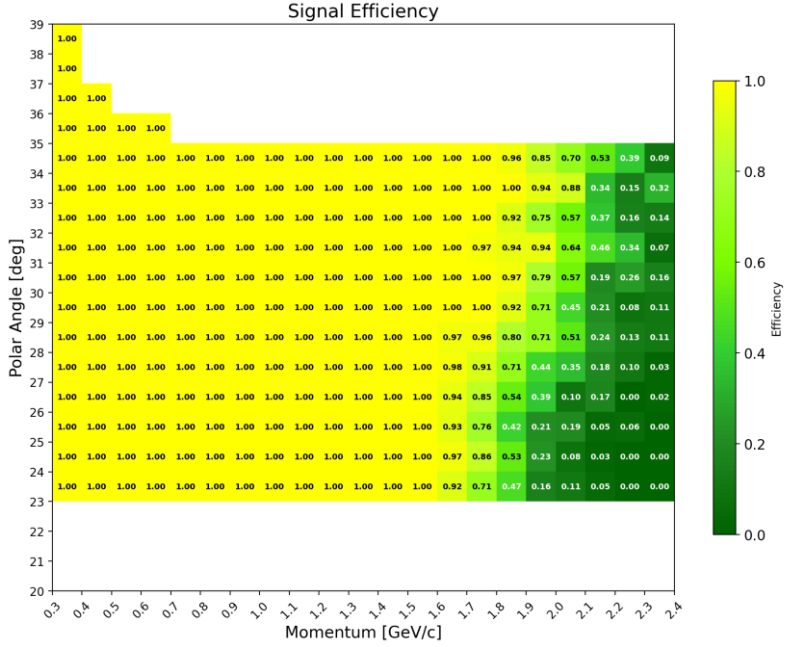
Yun-qing Mao @SDU



Model performance (software)



Deployed on VCK5000 with default quantization

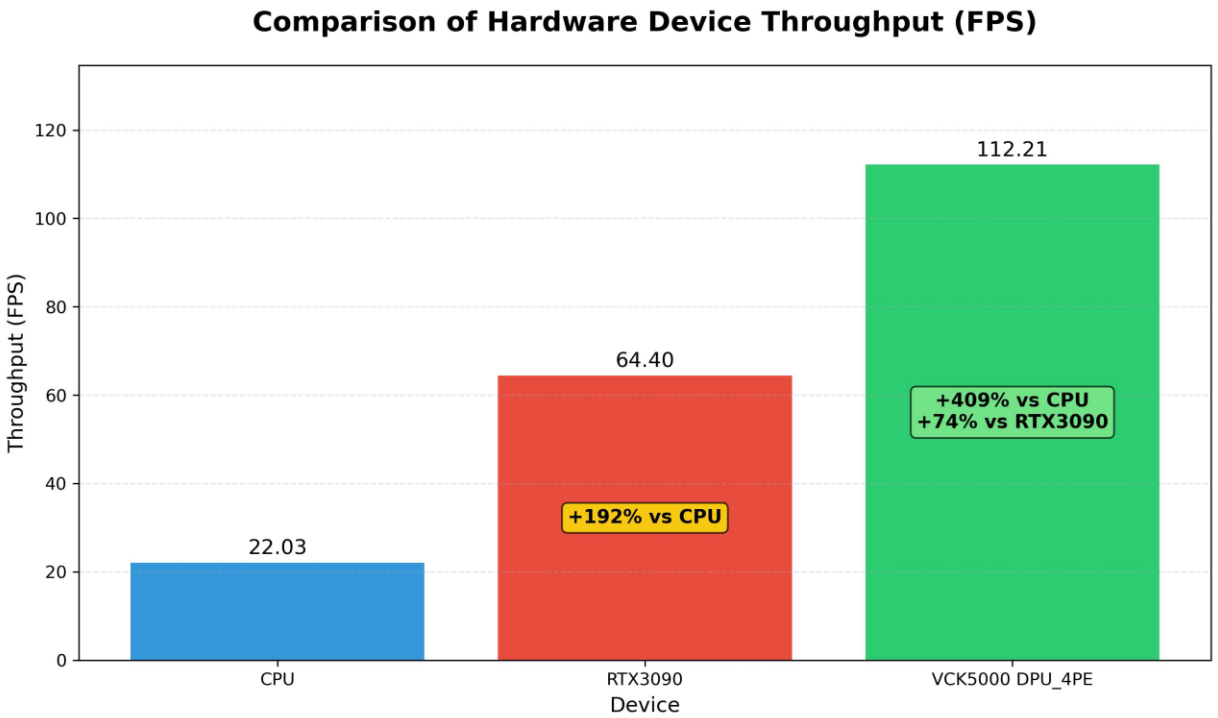


Quantization strategies applied cross-layer; 2^n;per-tensor

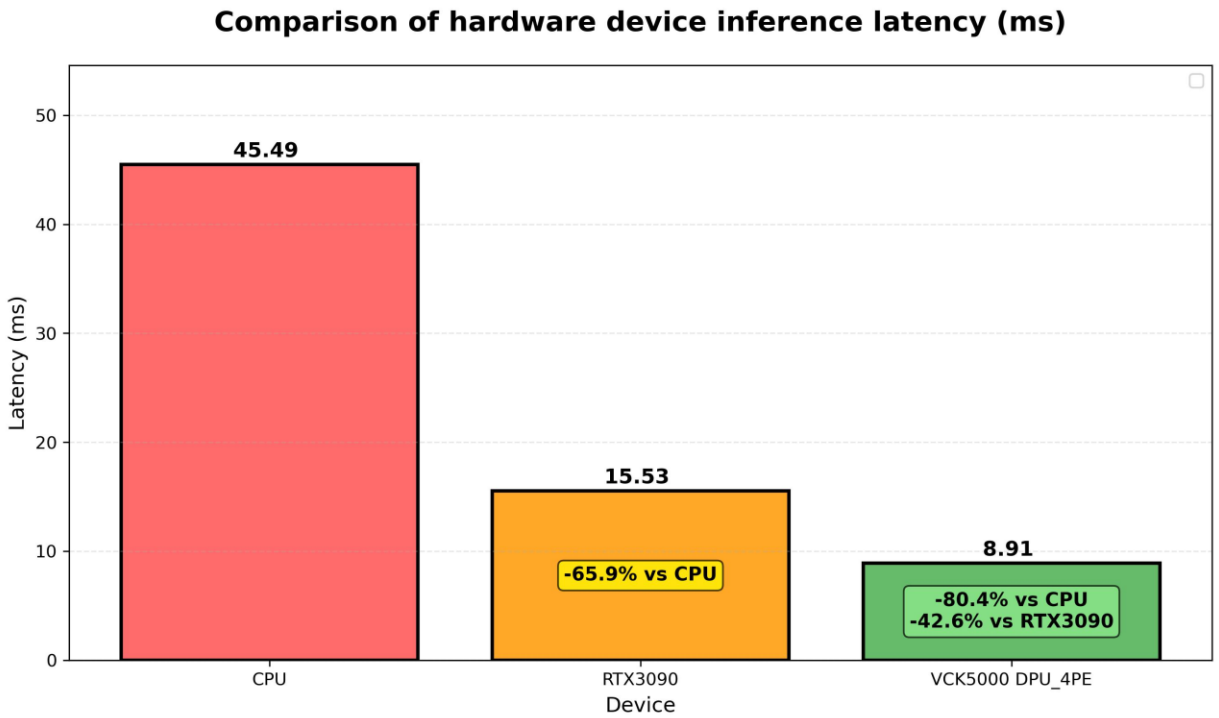
- Redesigned CNN model has improved the PID efficiency at high momentum region

Quantization development on DPU is still ongoing

Throughput and latency



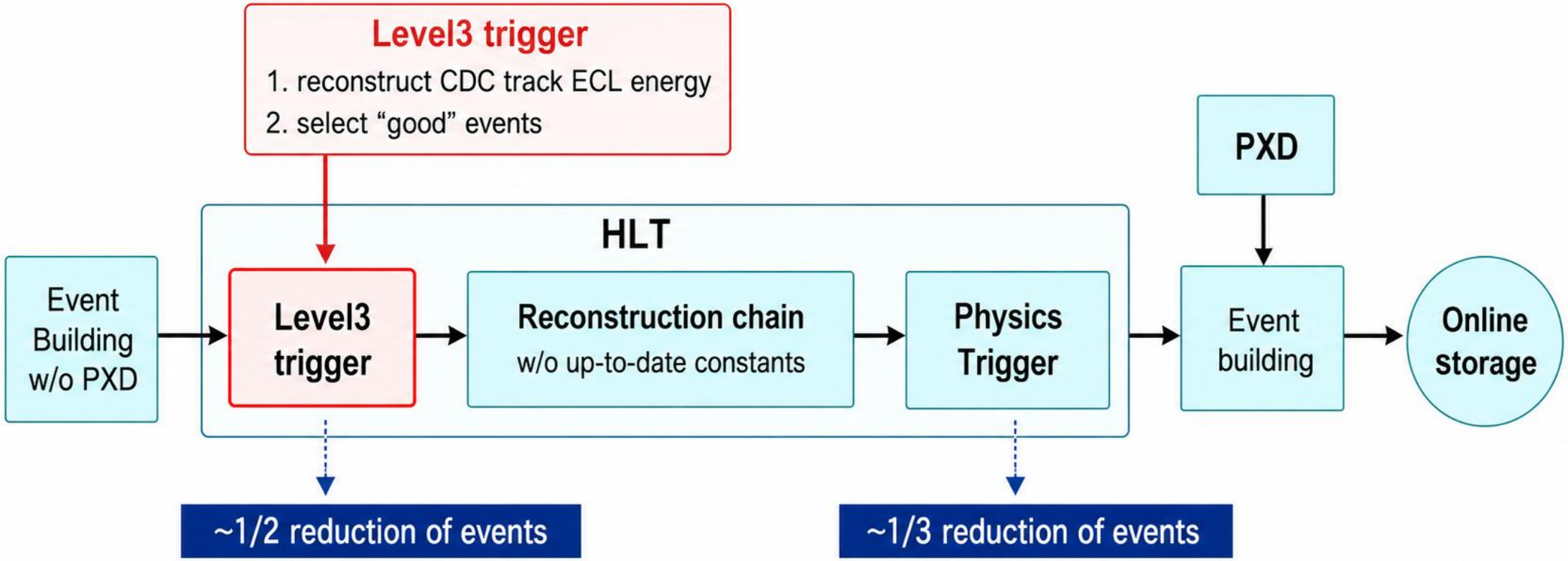
Throughput



Latency

VCK5000 DPU shows a **5 times** throughput than CPU and **74% more** throughput than RTX 3090
A **80**% shorter latency than CPU and **42**% shorter than that of RTX 3090

L3 trigger



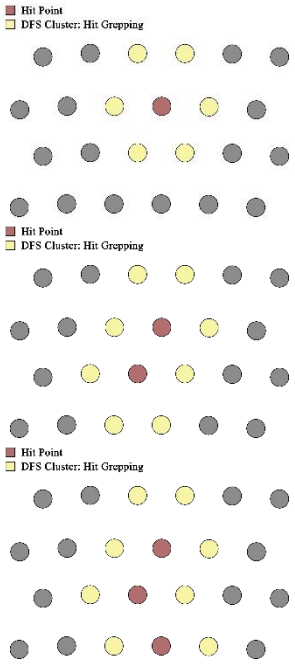
- Currently developed version is inherited by Kakuno-san
  - The module is known as "Fzisan" , continued from Belle
- This L3 project is basically for CPU-only HLT, but Belle II is trying to run it on GPUs

L3 trigger algorithm

Zi-bing Bai @SDU

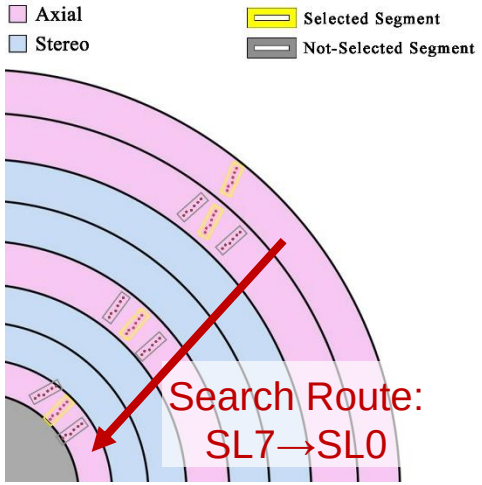
1) Reconstruct Segment

DFS Clustering



2) Link Axial Segment

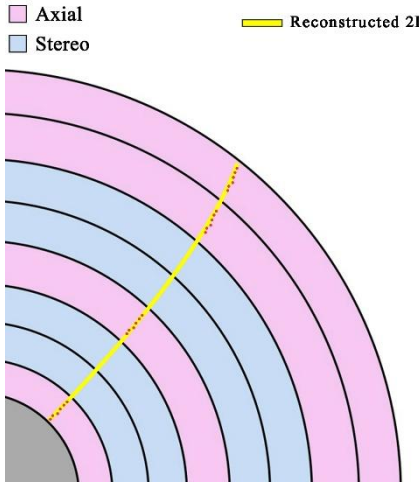
Min χ^2 of Segments' κ



3) Fit 2D Track

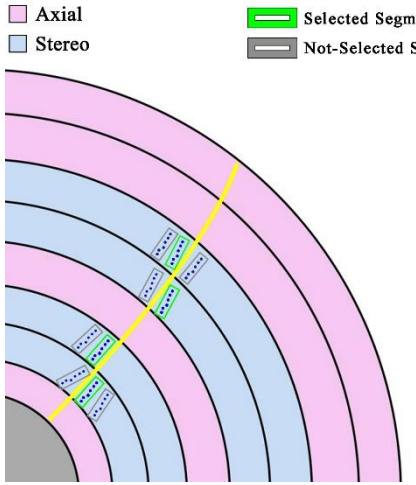
Algebraic Circle Fit

($r-\phi$ plane)



4) Link Stereo Segment

Winner of $s-z$ Consistency

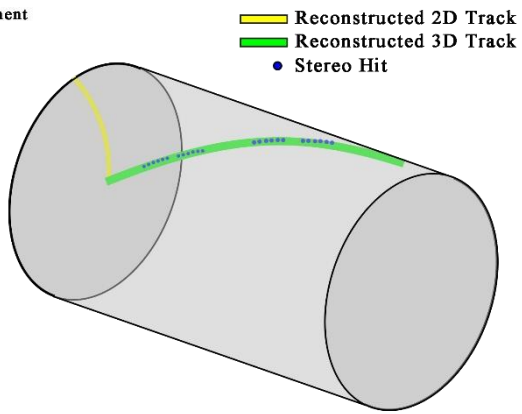


*Check Details about $s-z$ Consistency in Backup

5) Fit 3D Track

WLS Line Fit

($s-z$ plane)

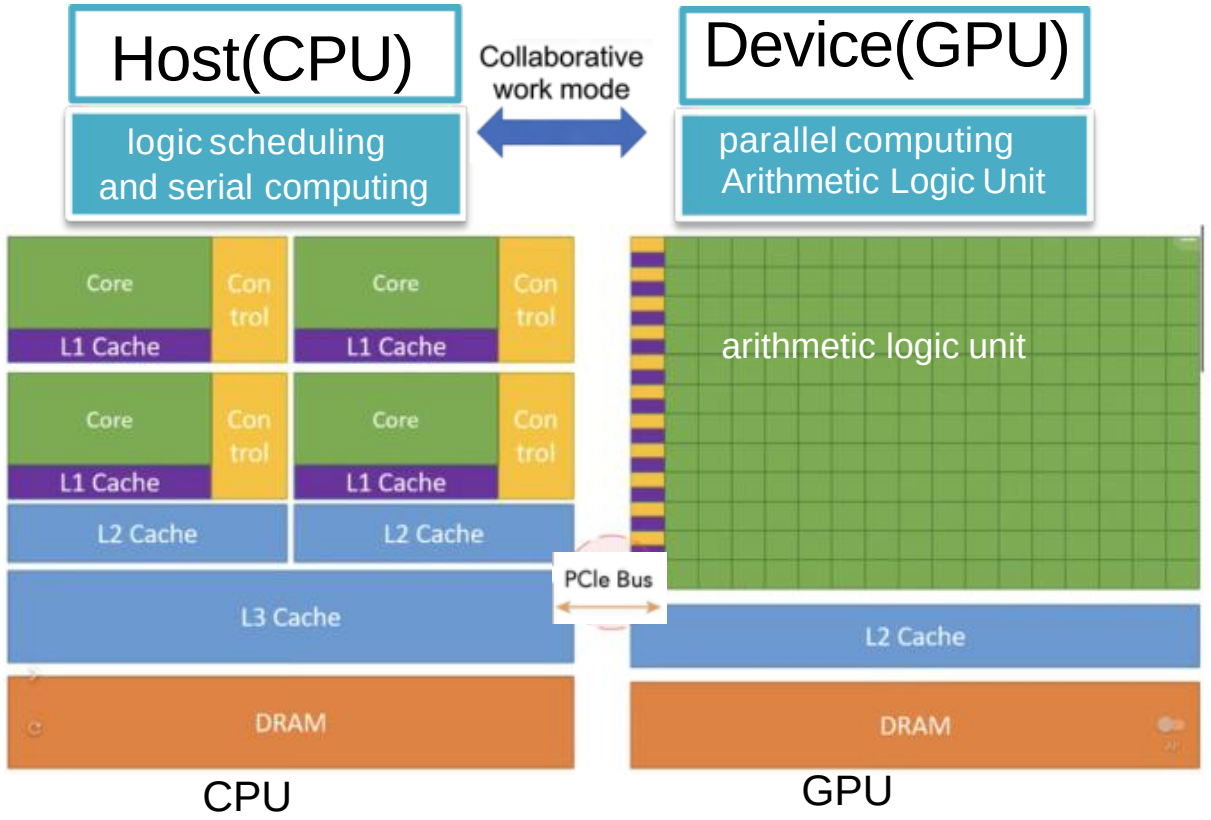


Hits to **Segments** → **Link** Segments in **Axial SLs** → **Hits** in Linked Segments to Reconstructed **2D Track** & **Collision Point** → **Link** Segments in **Stereo SLs** with Reconstructed **2D Track** → **Hits** in Linked Pair to Reconstructed **3D Track**

GPU Parallelization of L3 Energy Trigger

Jing-xuan Deng @SDU

- Host + Device Collaborative Architecture



- ★ Low compute density
- ★ Complex control logic
- ★ Large caches (L1$/L2$, etc.)
- ★ Optimized for serial operations
  - Fewer execution units (ALUs)
  - Higher clock speeds

- ★ High compute density
- ★ High Computations per Memory Access
- ★ Built for parallel operations
  - Many parallel execution units (ALUs)
  - Graphics is the best known case of parallelism

6-Step Standard CUDA Program Flow

1. Host memory allocation&Host data initialization
2. Device memory allocation
3. Host → Device data transfer
4. Kernel launch & parallel computing
5. Device → Host result transfer
6. Resource release & result verification

GPU Parallelization of L3 Energy Trigger

➢ Current challenge
- Higher background → more noise hits/events
- CPU load grows before design luminosity

Work Mainline

Raw event data
CDC hits, ECL clusters, segments

→

Flatten data
POD structure
remove pointers / STL dependence

→

GPU batch test
CUDA kernel
parallel reduction / segment check

→

Return decision
trigger flag / selected events

Testing and Results

Kernel-level acceleration

L3 energy-sum test

10M clusters
CPU serial: ~46 ms
GPU kernel: ~0.16 ms

Meaning: parallel computation is effective; end-to-end time is now transfer/setup dominated.

Data transport correctness

FTSegment POD test

Wire coordinates + superlayer parameters copied to GPU
CPU/GPU values matched in validation

Meaning: GPU-readable Belle II data layout is feasible.

Integration direction

Batch + CUDA Stream pipeline

next plan:
CPU/GPU consistency tests
basf2 module integration
Belle II-compatible GPU-L3 modules

Summary

Belle II transformer-based trigger for L1
- Development of light transformer with physics constraint
- Deployment on VPK120 PL
- Satisfy the latency requirement of the L1 trigger

Trigger-oriented GNN
- Finish the GNN operator on VCK190 AI-engine
- From X86 simulation to board-level test
- A demo with 250 maximum nodes input and 12ms baseline latency

STCF PID CNN
- Development of the PID CNN with Vitis AI compatibility
- Deployment of the CNN on VCK5000 with quantization

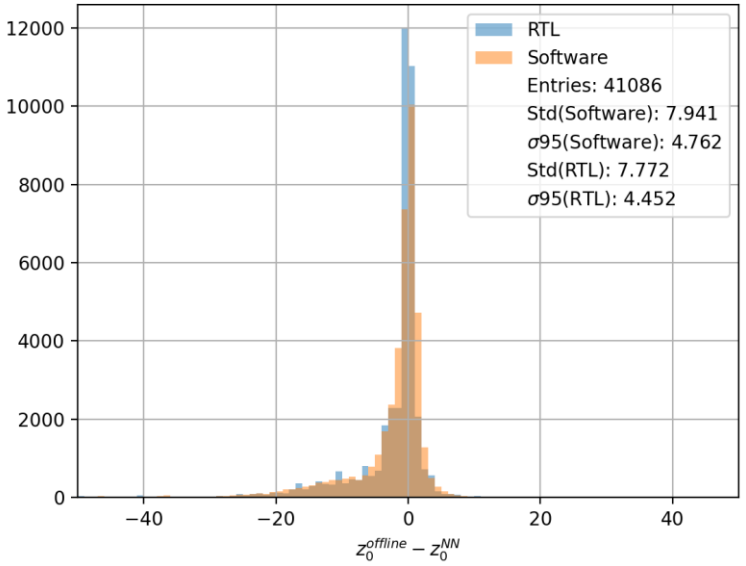
L3 trigger
- Development of the adaption of the L3 trigger from Belle II to STCF
- Ongoing exploration of the capability of the L3 trigger algorithm
- Several GPU acceleration tests are performed

END

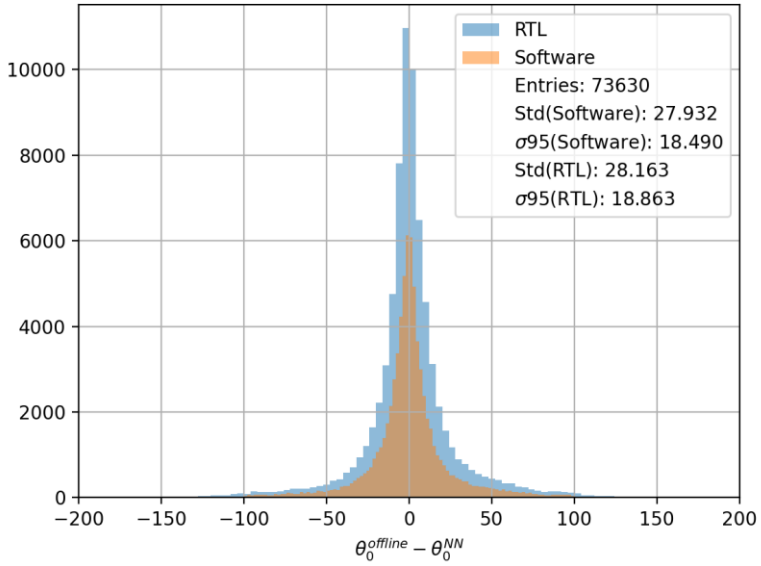
BACK UP

Performance

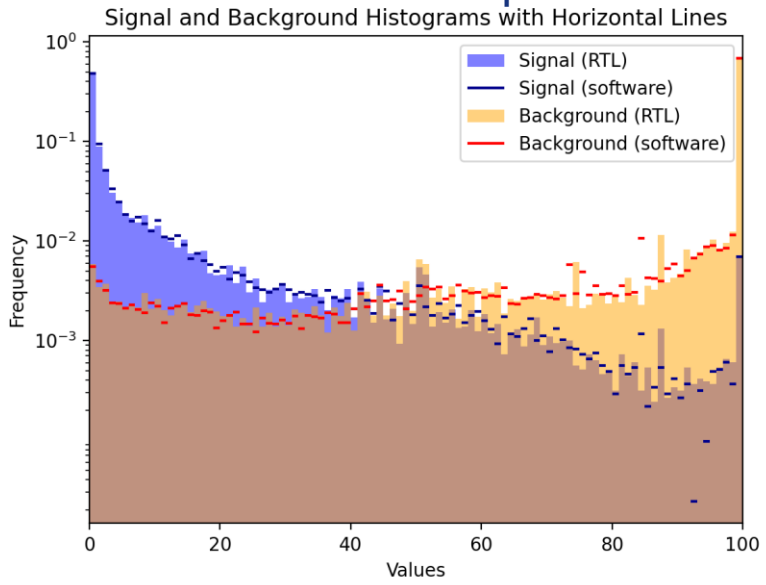
Delta track z (±1cm)

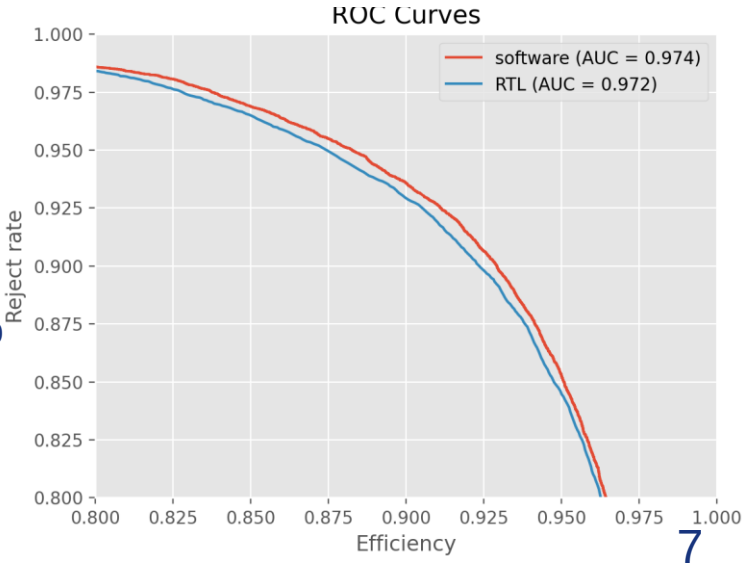


Delta track θ



Classifier output





- Latency:71clock=554.7ns; requirement: <600ns
- FPGA (VPK120: Versal premium w/0 AI-engine) resourse usage:
  - DSP: ~60%, LUT: ~65%, others<15%
- σ^{z0}=4.5cm,σ^{θ}=18°, Signal efficiency ~95%, Background rejection rate ~88%
- AUC do not get large drop comparing RTL and software simulation
- **Transform-based trigger in Hardware timing debugging phase**

GPU Parallelization of L3 Energy Trigger

POD (Plain Old Data)Transformation

- Problem:Complex class cannot be copied to GPU
- Reason：
  - FTSegment class contains pointer members (FTWire*,etc.) that are invalid on GPU.
- Solution: Flat POD transformation
  - Designed flattened pure numerical structs, extracted parameters,remove references, pointers.
  - batch-copied continuous memory
- Output
  - successfully transferring data the GPU and verifying the correctness of the results.

Target Algorithm: L3 Energy Trigger Principle

- Core Logic :Traverse all ECL clusters, sum total energy
- CPU bottleneck: Linear latency growth with cluster number
- (~46ms for 10M clusters)

Parallelization & Optimization Strategy

- Block-based parallel reduction with shared memory
- Pre-allocated memory,Pinned memory optimization

Metric	CPU Serial	Optimized GPU
kernel time	46ms	0.163ms
total processing time	46ms	49.18ms

Test Environment:

GPU: NVIDIA Ampere architecture GPU, compute capability sm_86

Test data: 10 million ECL clusters, each with 1GeV energy, trigger threshold 5 million GeV

Definition in Level3

Hits to **Segments** → Segment Reconstruction

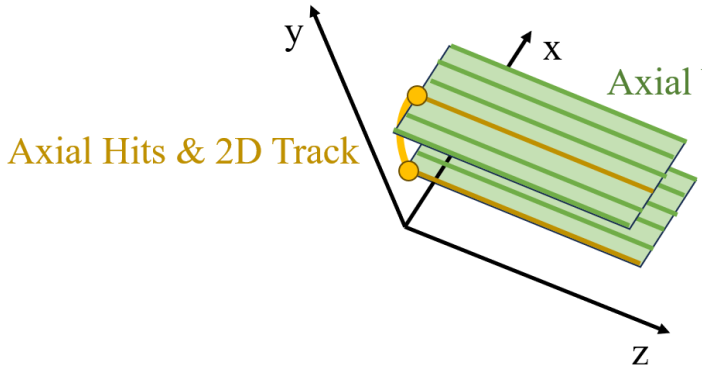
Link Segments in **Axial SLs** → Axial Segment Linking

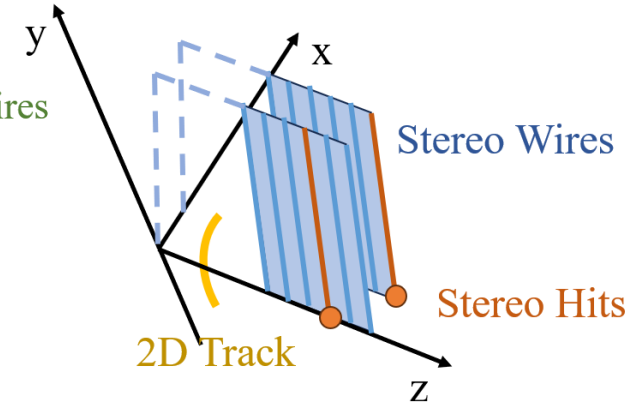
Hits in Linked Segments to Simulated **2D Track & Collision Point** (with dr, ϕ, κ) → 2D R-Phi Fitting

Link Segments in Stereo SLs with Simulated **2D Track** → Stereo Segment Linking

Simulate **3D Track** (with dr, κ, dz) → 3D S-Z Fitting

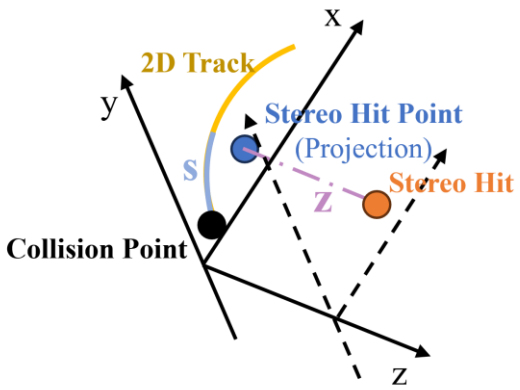
- What is **Wire Hit**?

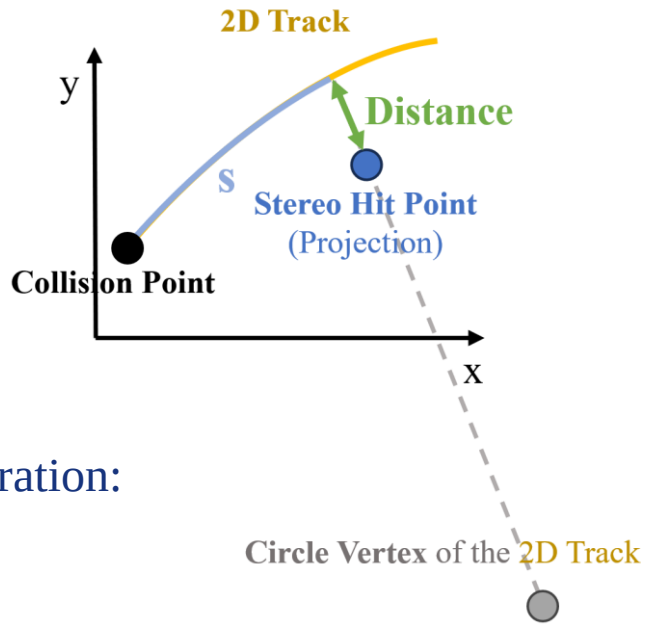




- What is s of **Stereo Hit**?

Arc from **Projection** point on $r-\phi$ **& Rec-Vertex(Rec-Collision)**





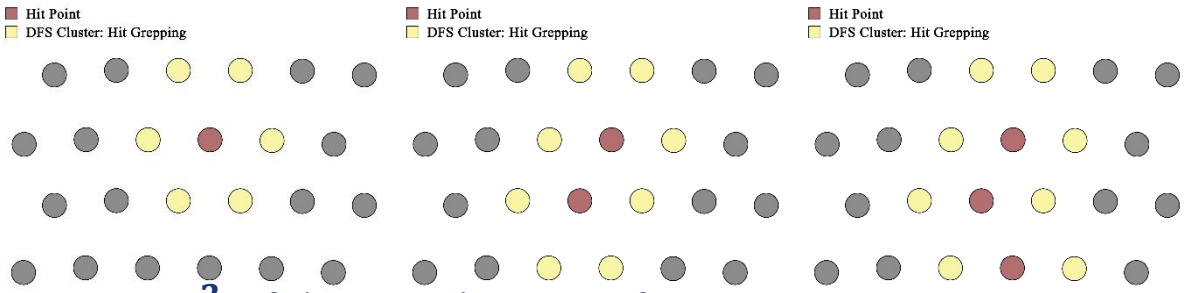
Standards of **Consistency** for comparation:

- s & 2D Track: **Distance**
- s / z & **2D Track** / **Total z**

Hits to **Segments**

Segment Reconstruction

DFS Clustering



Link Segments in **Axial SLs**

Axial Segment Linking

Min χ^2 of the Consistency of Segments' κ

Hits in Linked Segments to Simulated **2D Track & Vertex Point** (with dr, ϕ, κ)

2D R-Phi Fitting

Algebraic Circle Fit (Katayama Method) in $r-\phi$ Plane

Weights related to the **Drift Distance** obtained from the **TDC** via the **x–t** relationship

Link Segments in **Stereo SLs** with Simulated **2D Track**

Stereo Segment Linking

Winner (Consistency of $s-z$)

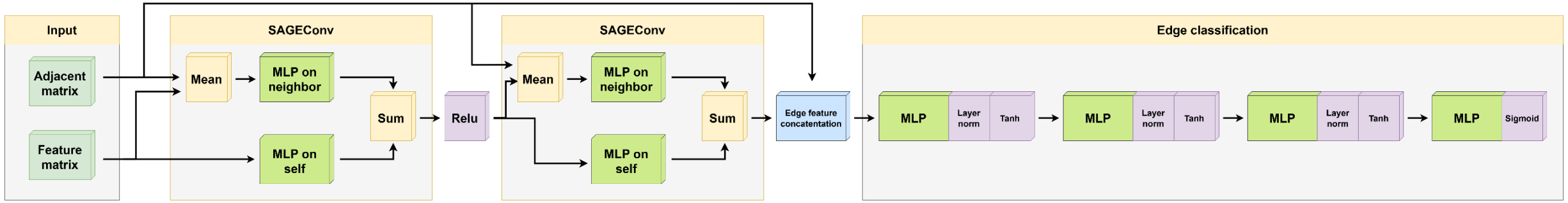
Simulate **3D Track** (with dr, κ, dz)

3D S-Z Fitting

Weighted Least-squares (WLS) Line Fit in $s-z$ Plane

Weights related to the **Drift Distance** obtained from the **TDC** via the **x–t** relationship

Latency



ILA CLK

Receiver interface

Event for test	ILA CLK frequency	SAGEConv1 latency	SAGEConv2 latency	Edge classification	Total latency
210 nodes 1712 edges	100 MHz	526us	1.2ms	6us * 1712	12ms