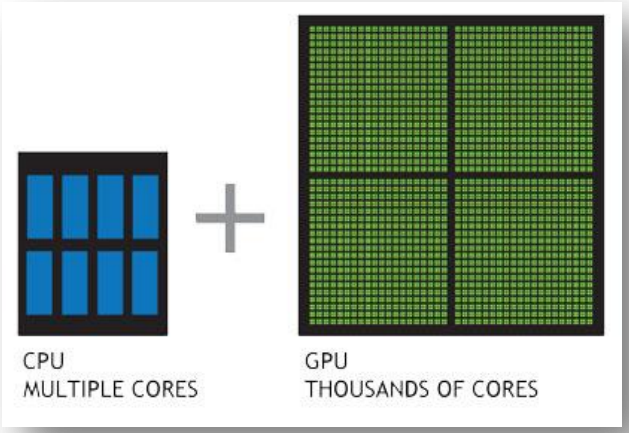
Graphics Processors for HEP trigger systems

Luca Pontisso (INFN)

on behalf of GAP and NaNet collaborations

VIENNA CONFERENCE ON INSTRUMENTATION 2016

Graphics Processing Unit (GPU)



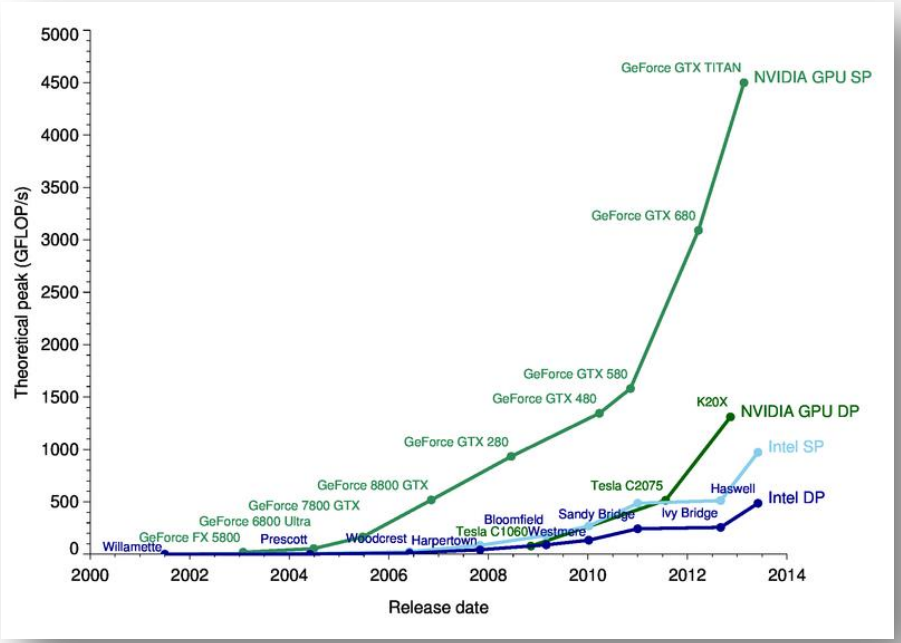
- Based on a massively parallel architecture
- Thousands of cores to process parallel workloads efficiently

- Less control units, many more ARITHMETIC LOGIC units.

Graphics Processing Unit (GPU)

- GPU's advanced capabilities were originally used primarily for 3D game rendering
- Since 2007 high-Level programming languages (CUDA, OpenCL) have been introduced
- Now this devices are largely deployed in General Purpose applications (GPGPU)

- Faster evolution with respect to traditional CPU
- Easy to have a desktop PC with teraflops of computing power, with thousands of cores.



GPUs in High Energy Physics: High Level Trigger systems

- HEP's High Level Trigger systems offer a complex environment in terms of rate, bandwidth and latency
- GPUs can be easily exploited in the higher software levels: being powerful computing devices, they can boost the capabilities of the processing system, thus allowing more complex computations
- CERN's experiments like ATLAS and CMS, for example, are implementing GPUs in their tracking algorithms

GPUs in High Energy Physics: Low Level Trigger systems

Their implementation is less obvious:

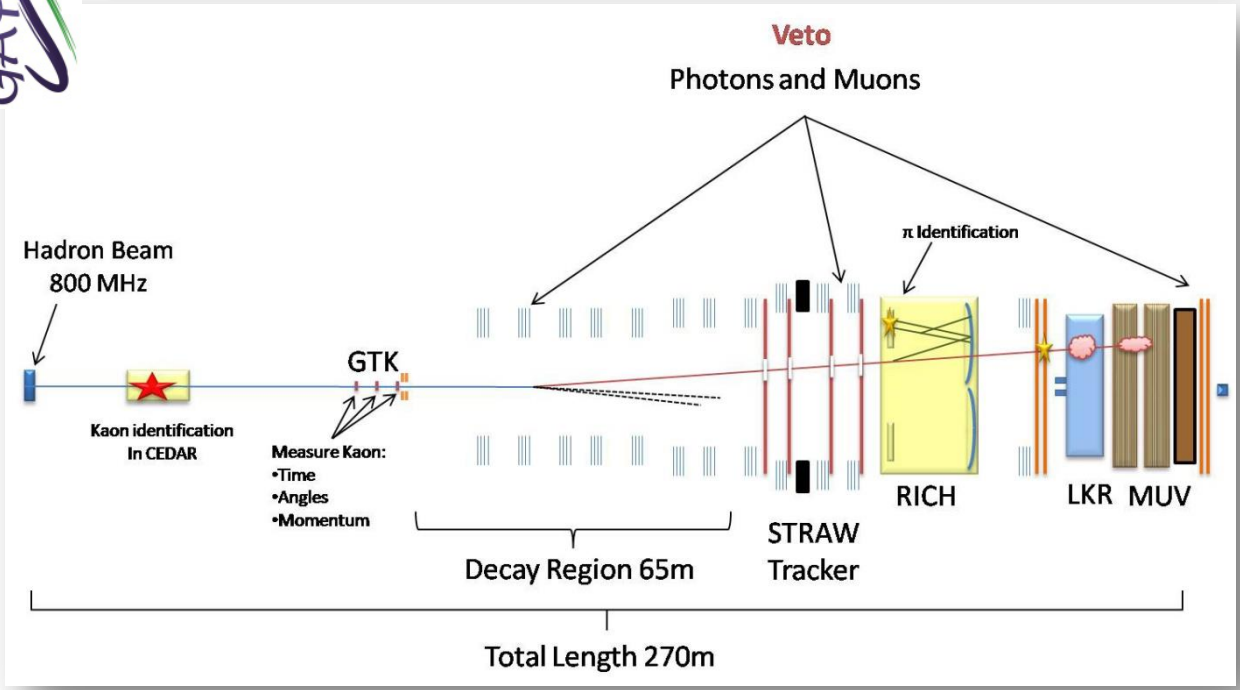
High rate events (tenths of MHz)

are GPUs fast enough in taking decisions?

Tiny latency

Is GPU processing latency small and stable enough for the given task?

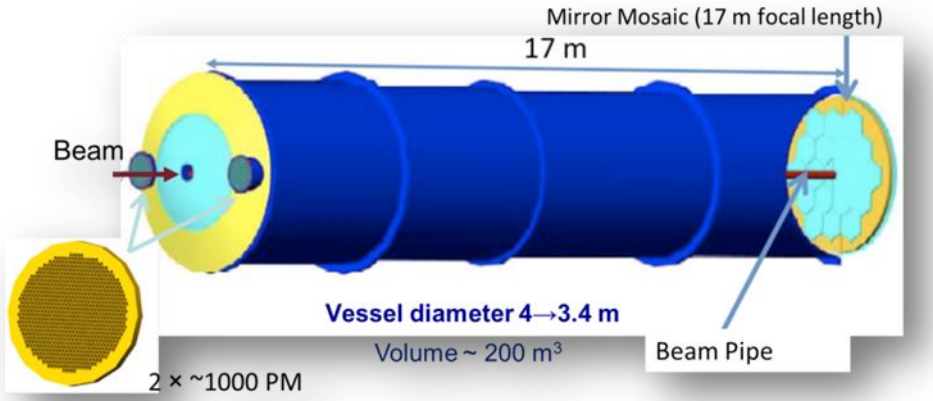
Physics case: Low Level Trigger system in NA62

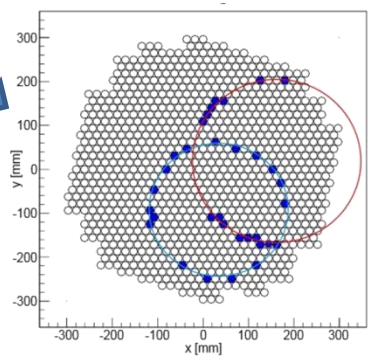


- **Measurement of ultra-rare decay**
  $K^+ \to \pi^+ \nu \bar{\nu}$
  (BR ~8×10^{-11})
- **Kaon decays in flight**
- **High intensity unseparated hadron beam (6% kaons)**

- **L0 trigger: synchronous level must reduce rate from 10 MHz to 1 MHz**
  - **1 ms max. latency**

NA62 RICH Detector

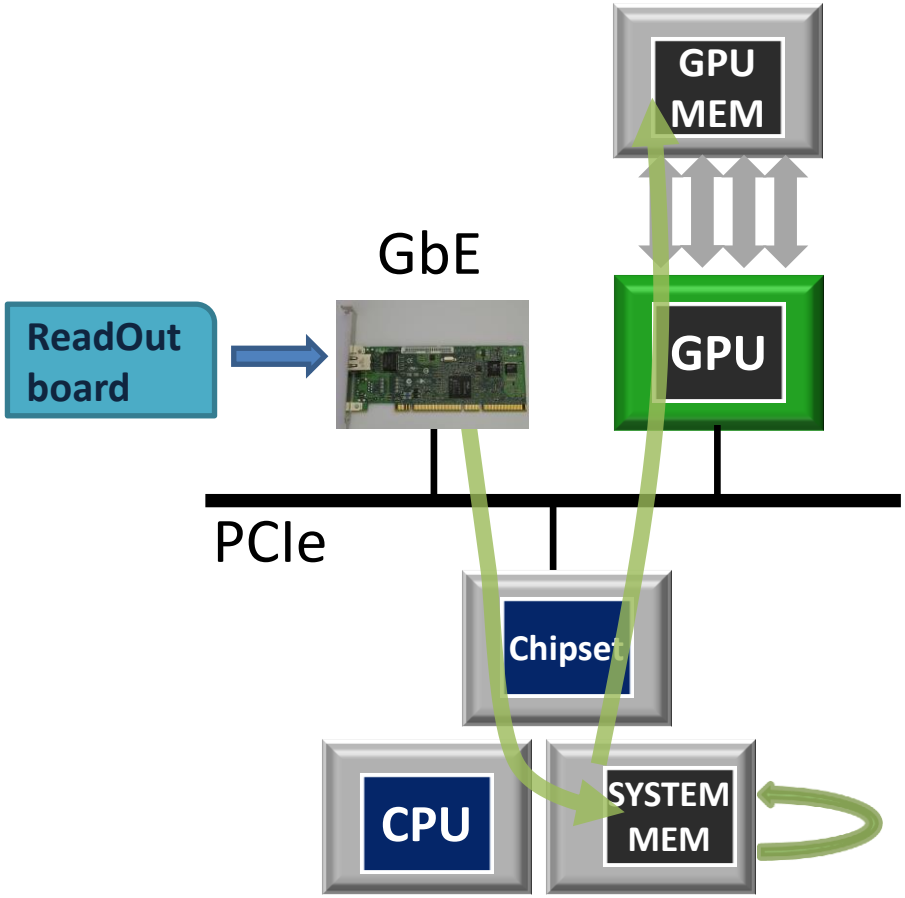




GPU based L0 trigger focused on ring reconstruction

- 17 m long, 3 m in diameter, filled with Ne at 1 atm
- Distinguish between pions and muons from 15 to 35 GeV
- The separation inefficiency between pions and muons is below 1%
- Time resolution: 70 ps
- 10 MHz events: about 20 hits per particle
- 2 spots of 1000 PMs each
  - 2 read-out boards for each spot (it is necessary a merging stage for the events)

GPU based trigger main problem: latency



Total latency dominated by double copy in Host RAM:

- **Data are copied from kernel buffer to destination buffer in user space**
- **Data are copied from CPU memory to GPU memory**

How to reduce data transfer time:

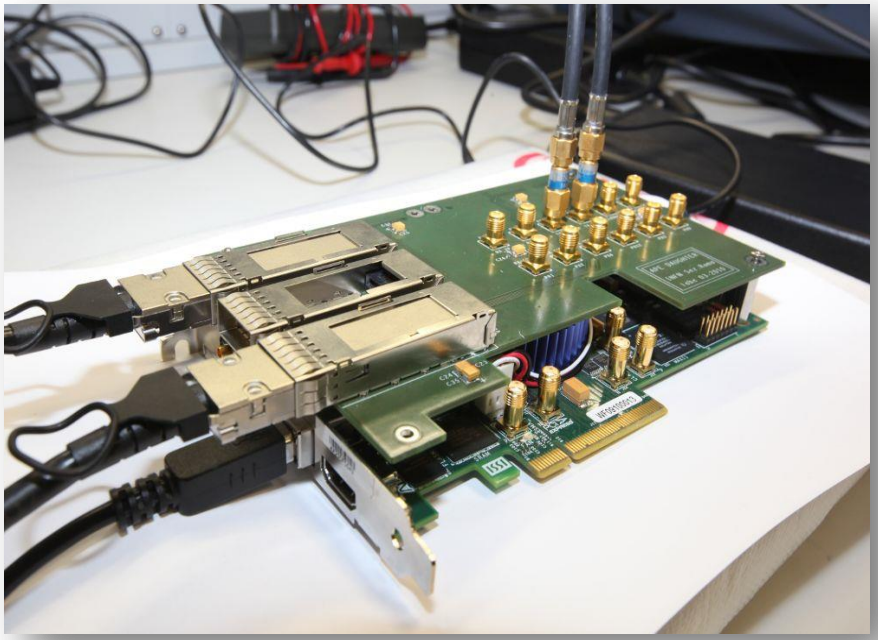
- **DMA (Direct Memory Access) from the network channel directly in GPU memory**
- **Custom management of NIC buffers**

NaNet: a PCIe NIC family for HEP

OBJECTIVES:

- Bridging the front-end electronics and the software trigger computing nodes.
- Supporting multiple link technologies and network protocols.
- Enabling a low and stable communication latency.
- Having a high bandwidth.
- Processing data streams from detectors on the fly.
- Optimizing data transfers with GPU accelerators.

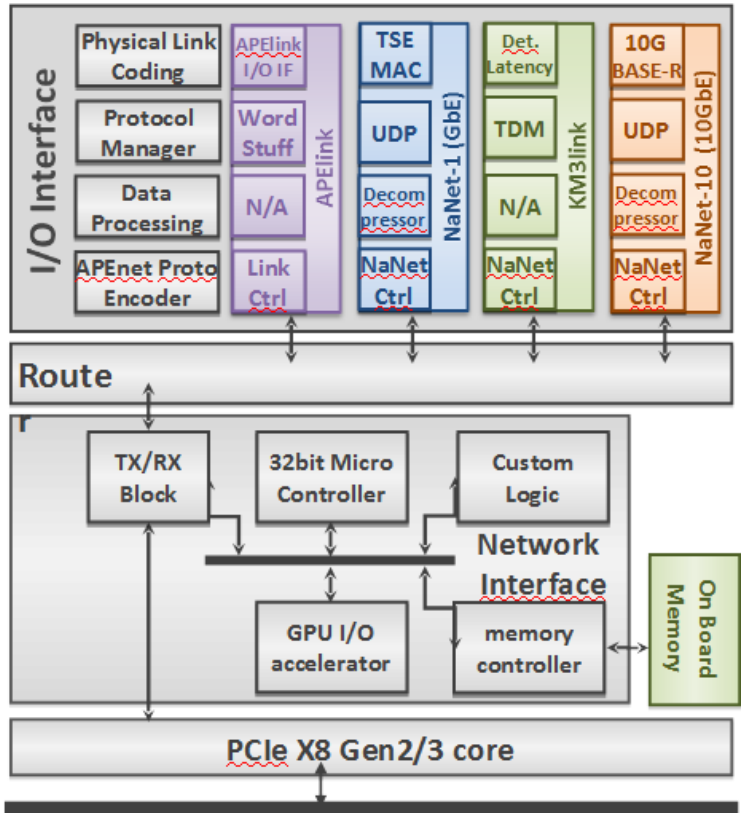
Developed at INFN Roma APE Lab



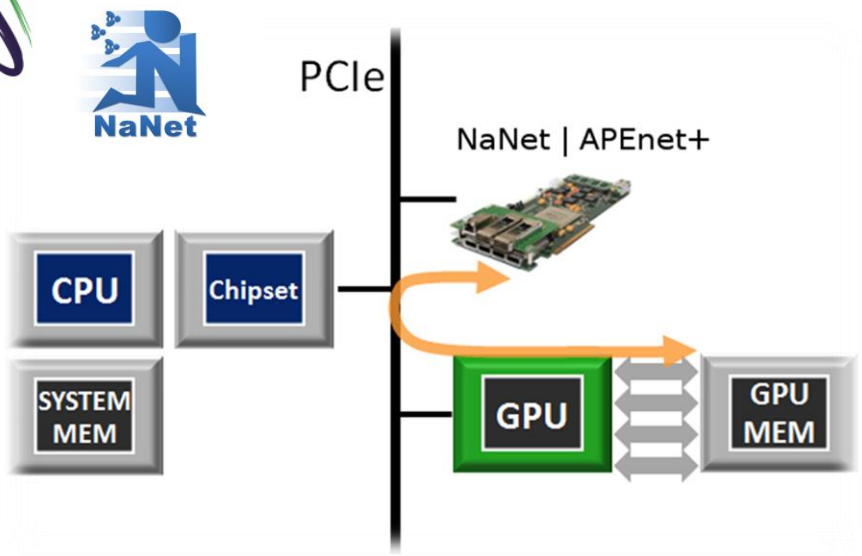
NaNet-1 is based on Altera Stratix IV dev board

NaNet: modular architecture

- I/O Interface
  - Multiple link
  - Multiple network protocols
    - Off-the-shelf: 1GbE, 10GbE
    - Custom: APElink (34gbps/QSFP), KM3link
- Router
  - Dynamically interconnects I/O and NI ports
- Network Interface
  - Manages packets TX/RX from and to CPU/GPU memory
  - TLB & Nios II Microcontroller
    - Virtual memory management
- PCIe X8 Gen2/3



NaNet: GPUDirect and Software



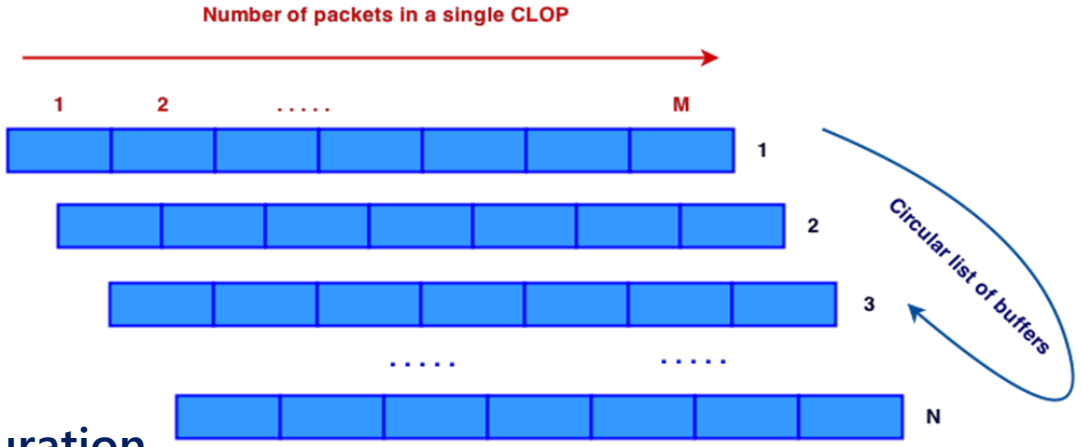
- GPUDirect allows direct data exchange on the PCIe bus with no CPU involvement
- No bounce buffers on host memory
- Zero copy I/O
- Buffers on GPU arranged in a circular list of persistent receiving buffers (CLOP)
- nVIDIA Fermi/Kepler/Maxwell

Host

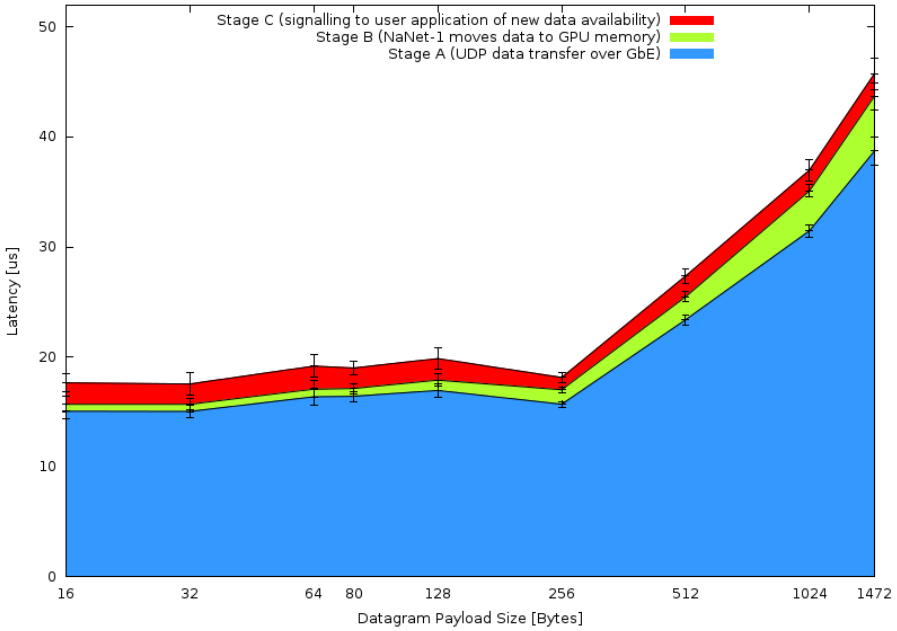
- Linux Kernel Driver
- User space Library (open/close, buf reg, wait recv evts, …)

Nios II Microcontroller

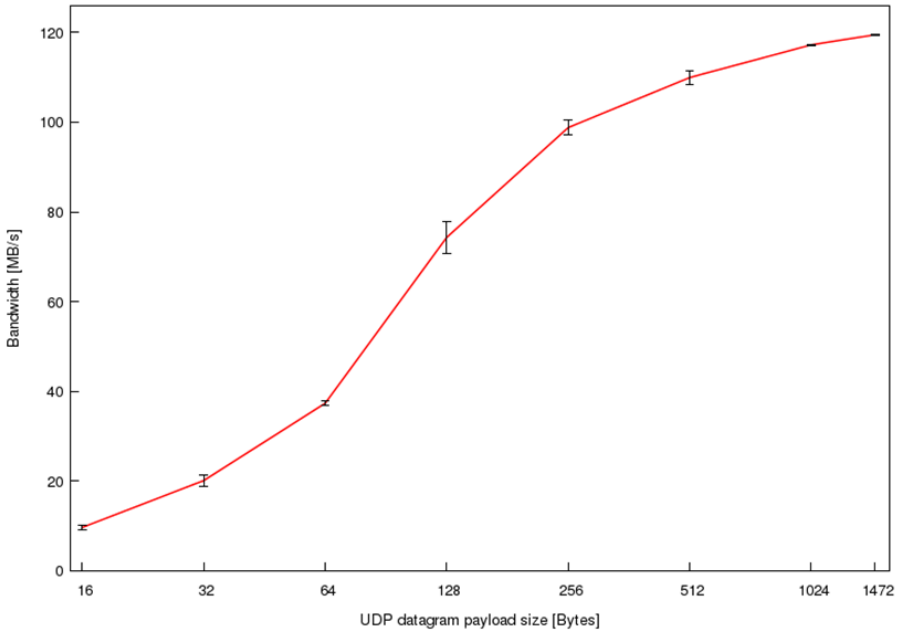
- Single process program performing System Configuration & Initialization tasks



NaNet: latencies and bandwidth



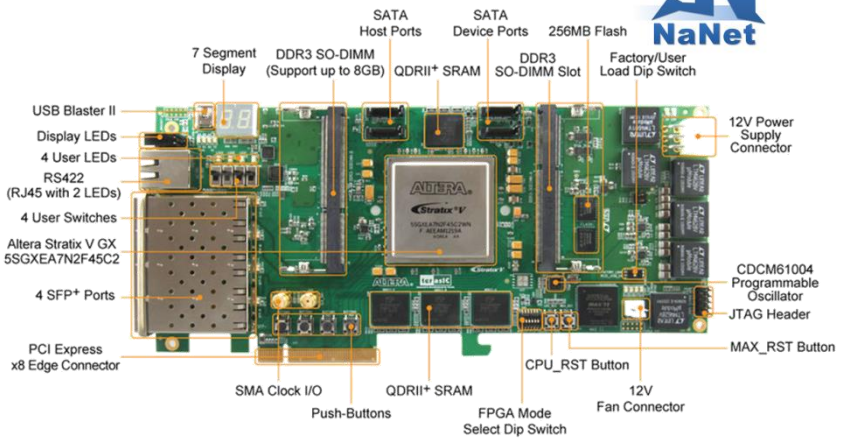
Latency measurements of data transfer to GPU memory

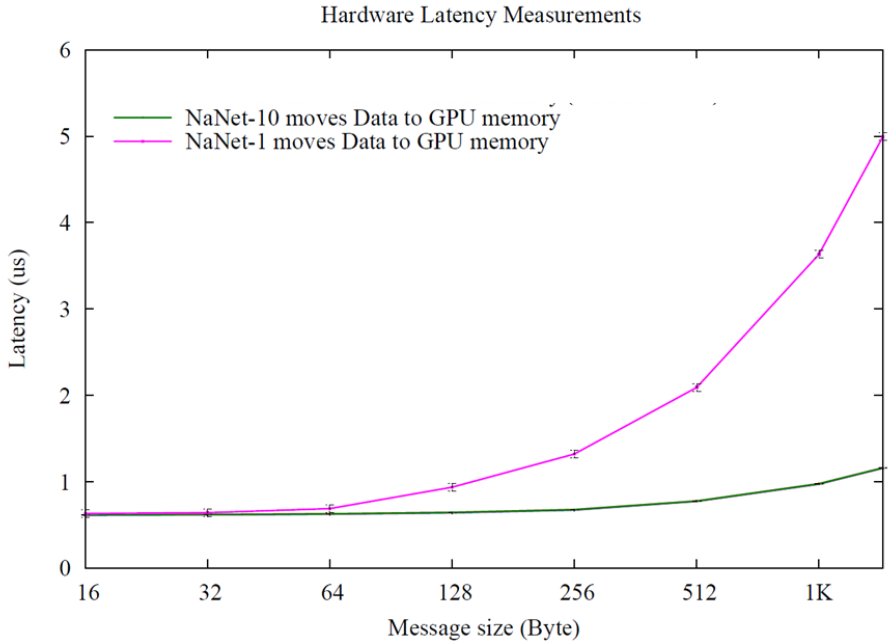


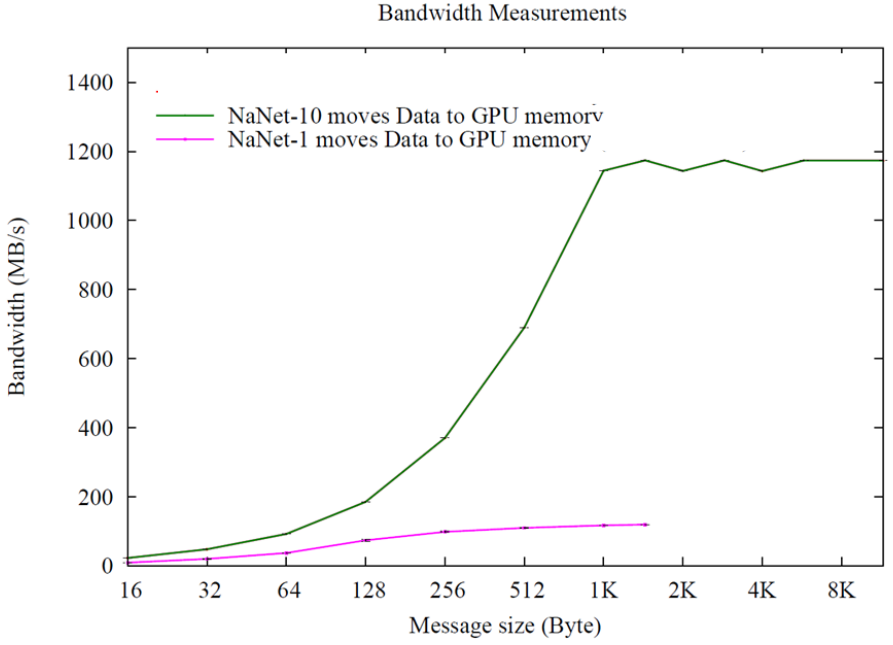
NaNet-1 GbE bandwidth of data transfer to GPU memory

NaNet-10: the next generation

- ALTERA Stratix V dev board
- PCIe x8 Gen3 (8 GB/s)
- 4 SFP+ ports (Link speed up to 10Gb/s)
- Implemented on Terasic DE5-NET board
- GPUDirect /RDMA capability
- UDP offloads supports

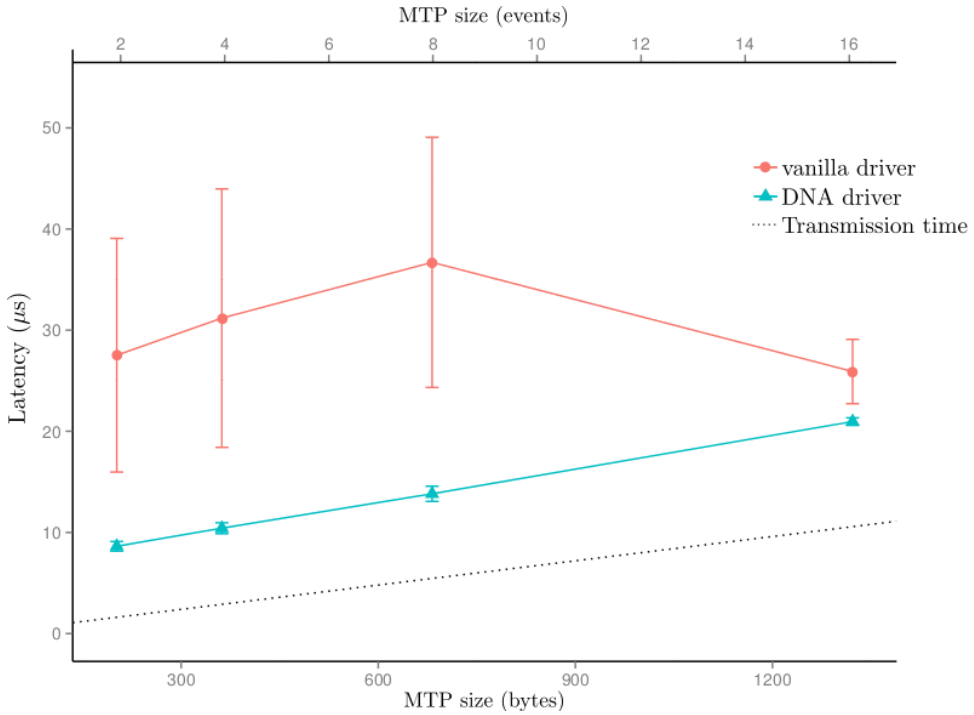




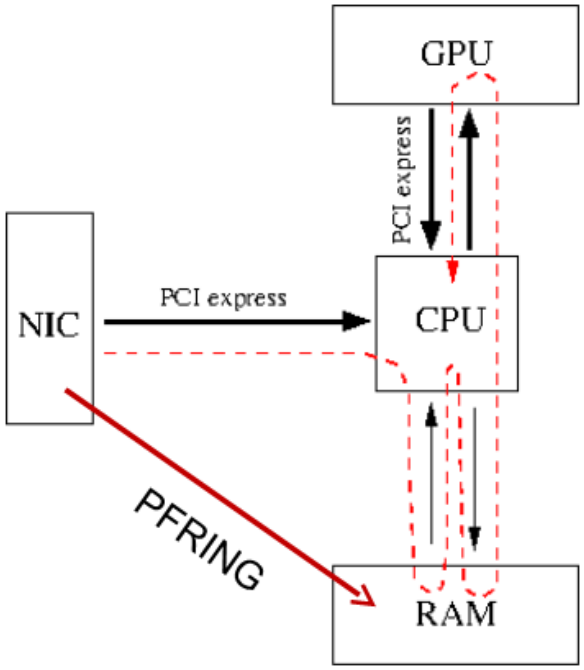


PFRING

PFRING DNA (Direct NIC Access) is a way to map NIC memory to userland so that there is no additional packet copy besides the DMA transfer done by the NIC



Communication latency to CPU memory



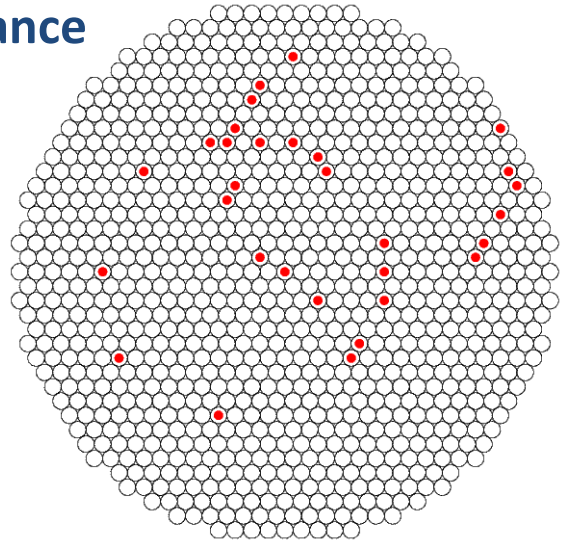
Pros: No extra HW needed

Cons: Pre-processing on CPU

L0 RICH trigger algorithm

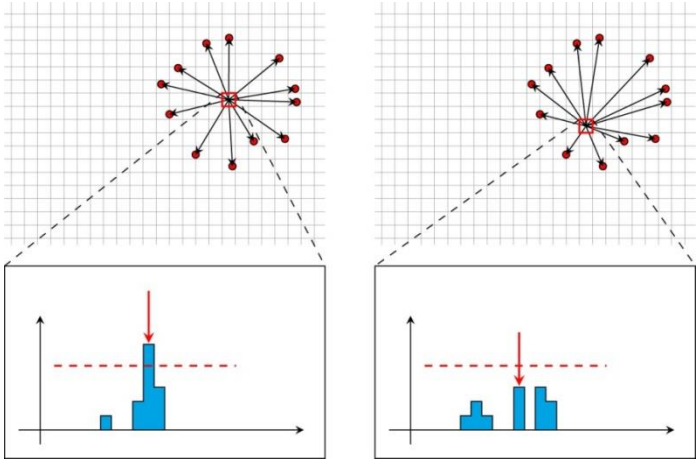
Requirements for an on-line RICH reconstruction algorithm:

- **Trackless**
  - No information from the tracker
  - Difficult to merge information from many detectors at L0
- **Multi-rings**
  - Many-body decay in the RICH acceptance
- **Fast**
  - Non-iterative procedure
  - Events rate at ~10 MHz
- **Low latency**
  - Online (synchronous) trigger
- **Accurate**
  - Offline resolution required



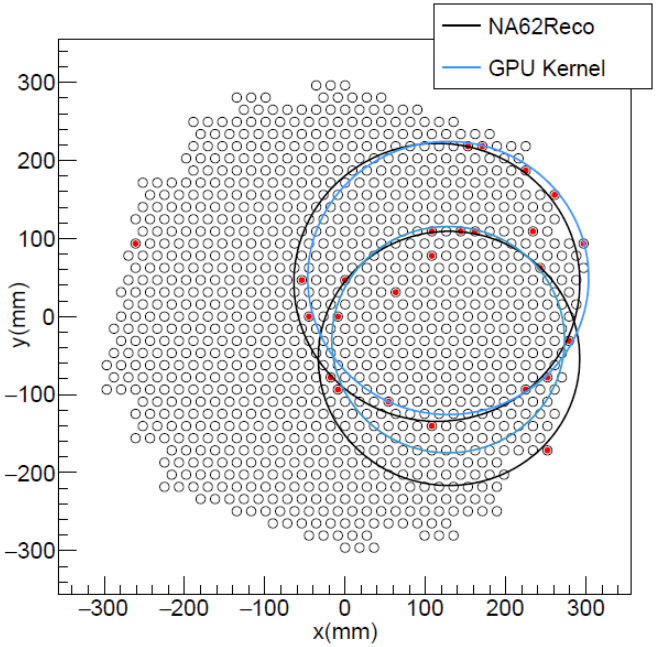
Histogram: another pattern recognition algorithm

- XY plane divided into a grid
- An histogram is created with distances from these points and hits of the physics event
- Rings are identified looking at distance bins whose contents exceed a threshold value



Pros: naturally mapped on the GPU threads grid

Cons: memory limited, performances depending on number of hits



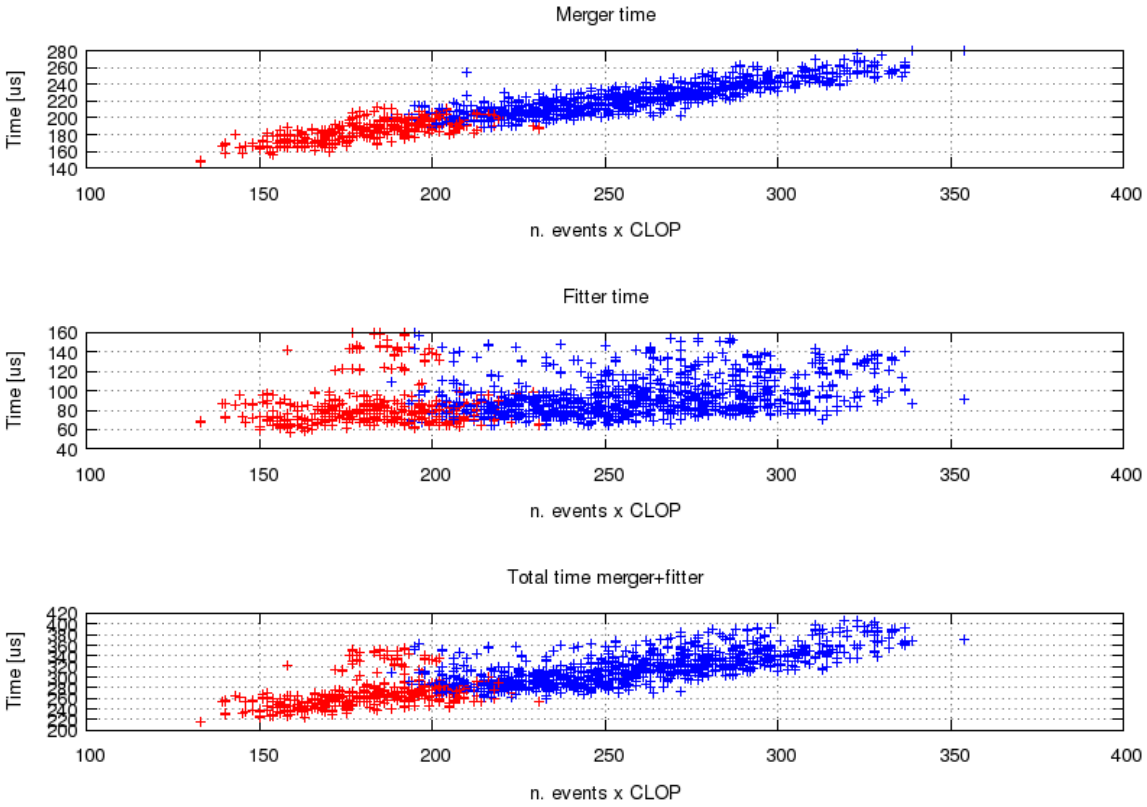
Good quality of online ring fitting

Still room for improvement: work in progress

NaNet: results with histogram kernel

Testing the GPU based L0 trigger chain

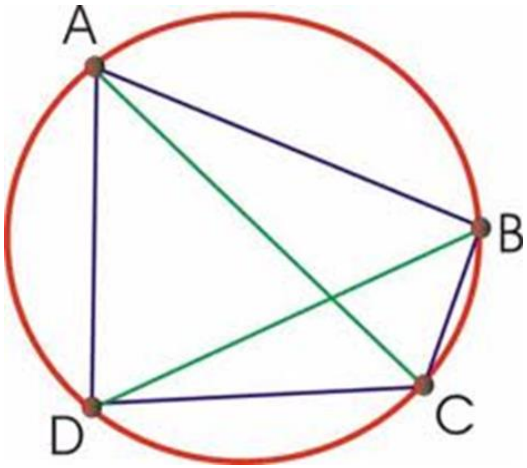
- Sending real data from NA62 2015 RUN
- NaNet-1 board
- GPU NVidia C2050
- Merging events in GPU
- Kernel histogram



Almagest: a new multi-ring algorithm

Based on Ptolemy's theorem:

"A quadrilater is cyclic (the vertex lie on a circle) if and only if is valid the relation: AD*BC+AB*DC=AC*BD"



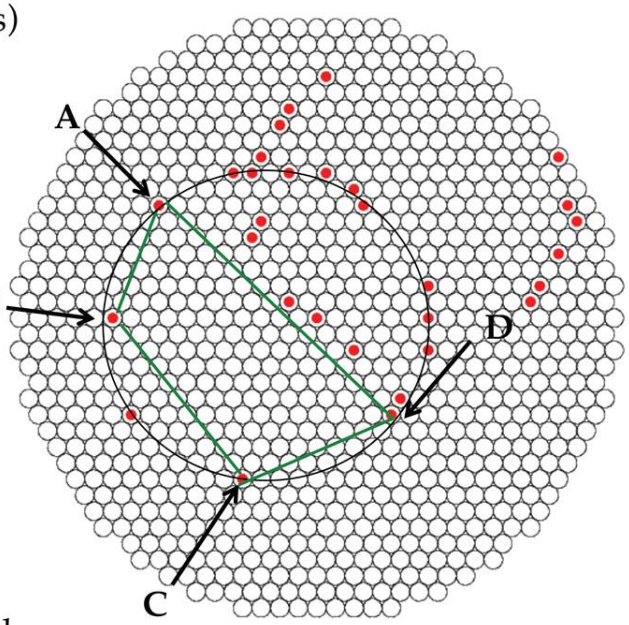
Almagest: a new multi-ring algorithm

i) Select a *triplet* (3 starting points)

ii) Loop on the remaining points: if the next point does not satisfy the Ptolemy's condition then reject it

iii) If the point satisfy the Ptolemy's condition then consider it for the fit

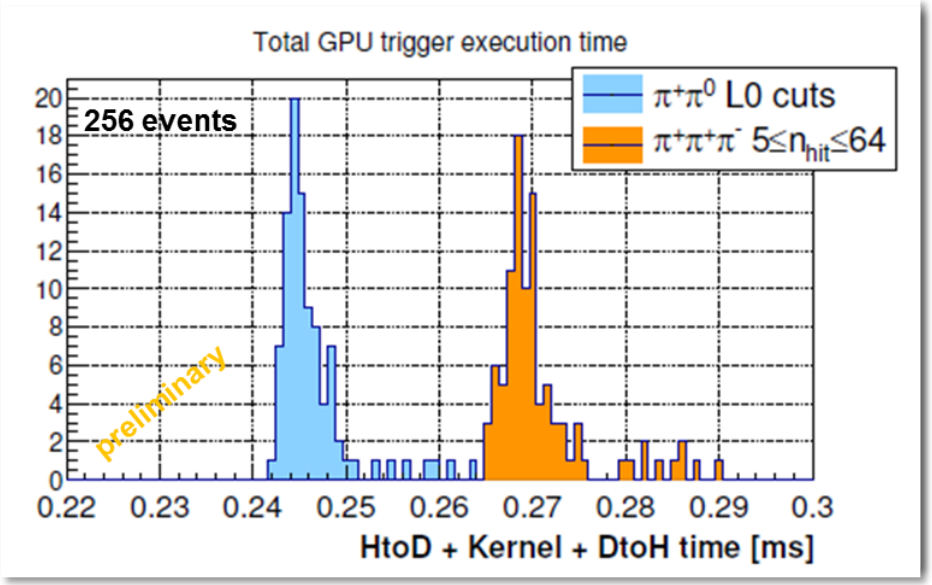
iv) …again…

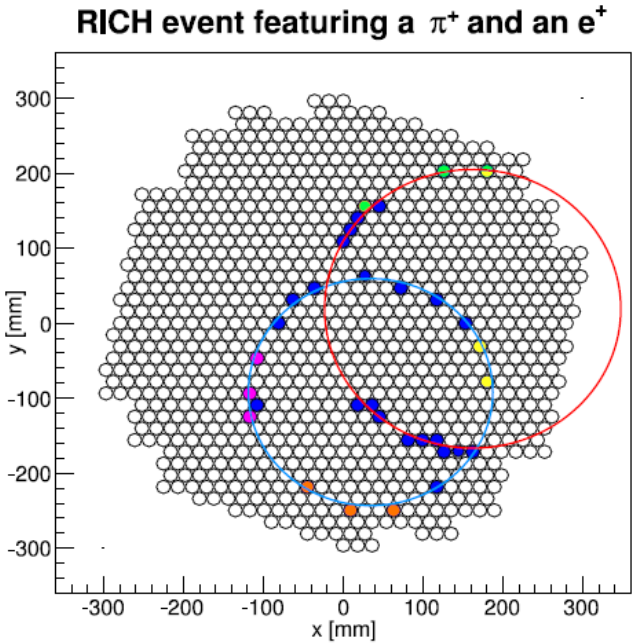


This algorithm exposes two levels of parallelism:

- **Several triplets run in parallel**
- **Several events at the same time**

Almagest: some results





- **Preliminary: efficiency greater than 80% (performances depending on the number of rings)**
- **About 1us per event for multi-ring**
- **Test on single C2070 board**

Conclusions

- To match the required latency in Low Level triggers, it is mandatory that data coming from the network must be copied to GPU memory avoiding bouncing buffers on host.
  - A working solution with the NaNet-1 board has been realized and tested on the NA62 RICH detector.
  - The GPU-based L0 trigger with the new board NaNet-10 will be implemented during the next NA62 Run starting on April 2016.
- Multi-ring algorithms such as Almagest and Histogram are implemented on GPU.

  There is still room for improvement -> Optimize computing kernels
- Finalizing and testing the solution based on the PFRING driver

R. Ammendola [(b)], A. Biagioni[(a)],
S. Chiozzi[(d)], A. Cotta Ramusino[(d)],
S. Di Lorenzo[(f,g)] , R. Fantechi[(f)],
M. Fiorini[(d,e)], O. Frezza[(a)],
G. Lamanna[(c)], F. Lo Cicero[(a)],
A. Lonardo [(a)], M. Martinelli[(a)],
I. Neri[(d)], P.S. Paolucci[(a)],
E. Pastorelli[(a)], R. Piandani[(f)],
L. Pontisso[(f)], F. Simula[(a)],
M. Sozzi[(f,g)], P. Vicini[(a)].

[(a)] INFN Sezione di Roma
[(b)] INFN Sezione di Roma Tor Vergata
[(c)] INFN - Laboratori Nazionali di Frascati
[(d)] INFN Sezione di Ferrara
[(e)] Università di Ferrara
[(f)] INFN Sezione di Pisa
[(g)] Università di Pisa

Thank You

SPARE SLIDES

GPGPU deployment in ATLAS High Level Trigger

ATLAS Trigger and TDAQ

Three upgrade phases expected in the future of LHC: the **detector occupancy** for the ATLAS experiment is expected to be **2-3 times higher**.
Trigger system needs to be upgraded to maintain or improve current performances.

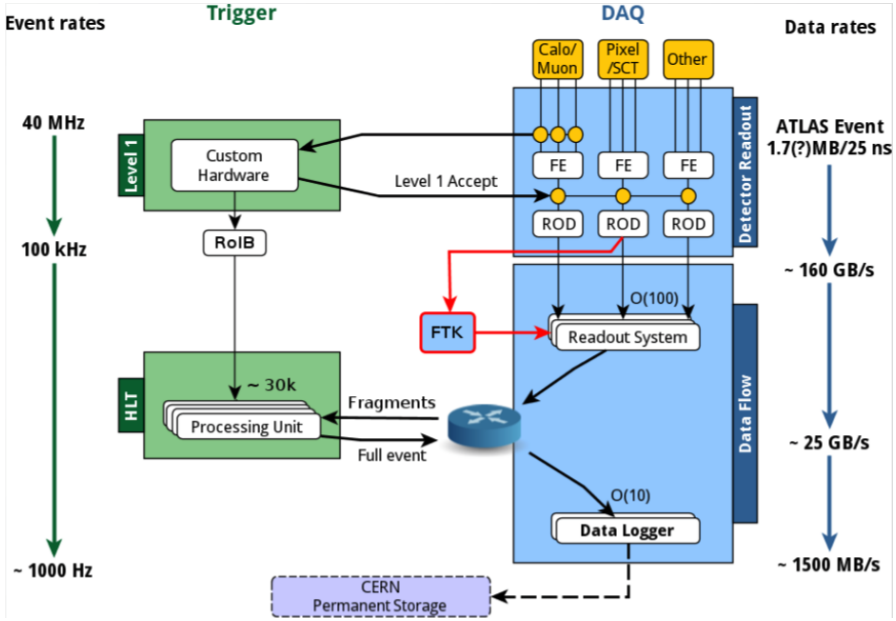
	Run 1		LS1 - Phase-0	Run 2	LS2 - Phase-I	Run 3	LS3 - Phase-II	Run 4
	2011	2012		2015-17		2019-21		2023-
Center of mass Energy $\sqrt{s}$ (TeV)	7	8		13-14		14		14
Luminosity ($cm^{-2}s^{-1}$)	8×10^{33}			1×10^{34}		2×10^{34}		5×10^{34}
Bunch spacing (ns)	50			25		25		25
Number of interaction/event, $< \mu >$	10	20		~ 27		$\sim 55 - 80$		~ 140
Total Integrated luminosity (fb^{-1})	25			~ 100		~ 300		~ 3000

ATLAS Trigger system selects 1 interesting event out of 40000 at a rate of 40MHz.
Organized in 2 levels:

- **L1 - hardware based**: coarse granularity, simplified calibrations, provides seeds for next level
- **HLT - software based**: runs on a farm of CPUs under a customized offline framework, can access full detector information

Investigating the **deployment of GPGPUs in the ATLAS HLT**, optimizing throughput per cost unit and power consumption.
Several **algorithm** and **sub-detector** under study:

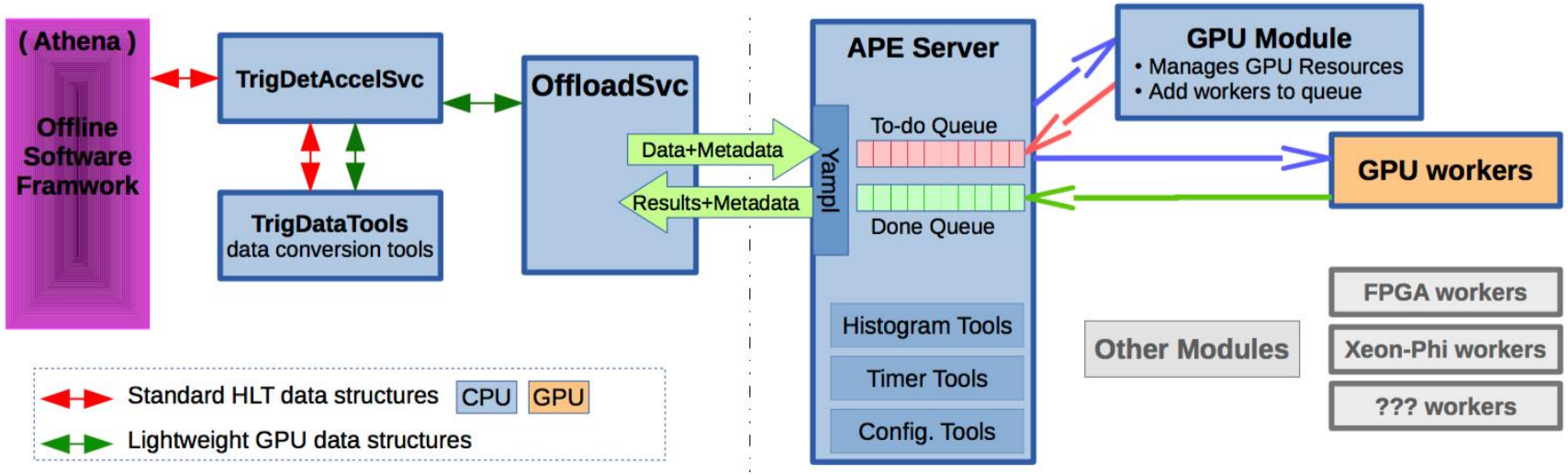
- Inner Detector tracking
- Calorimetric jet clustering
- Muon track finding



GPGPU deployment in ATLAS High Level Trigger

GPU acceleration framework

Implemented a **client-server architecture** to offload and process HLT data:
different modules for each algorithm/subdetector, accepts multiple clients, includes monitoring tools



Client side

- HLT **algorithm requires offloading** to AccelSvc
- AccelSvc:
  - **converts data** through TrigDataTools
  - adds metadata and requests offload (OffloadSvc)
  - **returns the result** to requesting algorithms

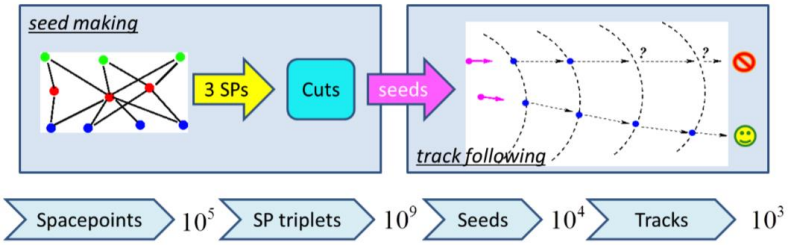
Server side

- Manager handles **communication** and **scheduling**:
  - links requests to corresponding modules
  - executes items in the queue and send results back
- Module: **manages GPU resources** and create work items requested
- Worker: **runs algorithms** over data and **prepares results**

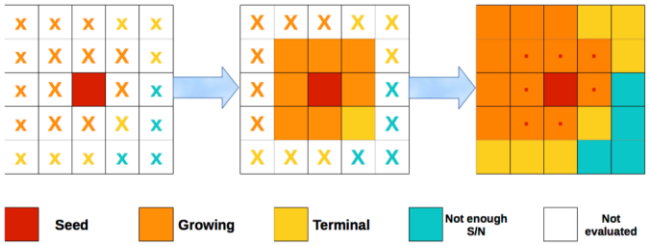
GPGPU deployment in ATLAS High Level Trigger

Algorithm performances

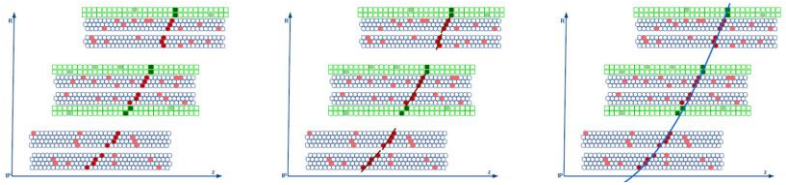
Inner Detector tracking - most time consuming part, dependent on detector occupancy.
- raw-data decoding and hits clustering
- hit triplets formation based on hit-pair seeds
- track following through the detector

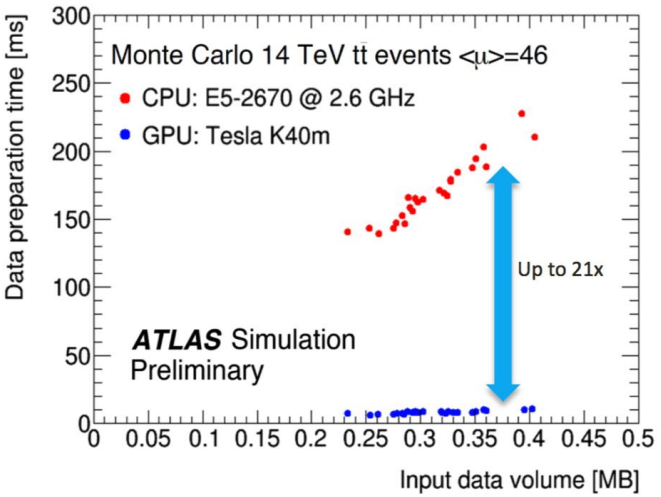


Calorimeter clustering - based on cellular automaton algorithm.
- starts from high S/N seed cells and growing cells
- evolve clusters until system is stable



Muon track finding - based on Hough Transform.
- voting kernels translate hit information to Hough Space
- maxima detection to determine track parameters





Preliminary result: **up to 20x faster** for **Inner Detector tracking.**

Setup working smoothly and **ready for different hardware configuration**.

Measurements ongoing, **further results expected soon**.

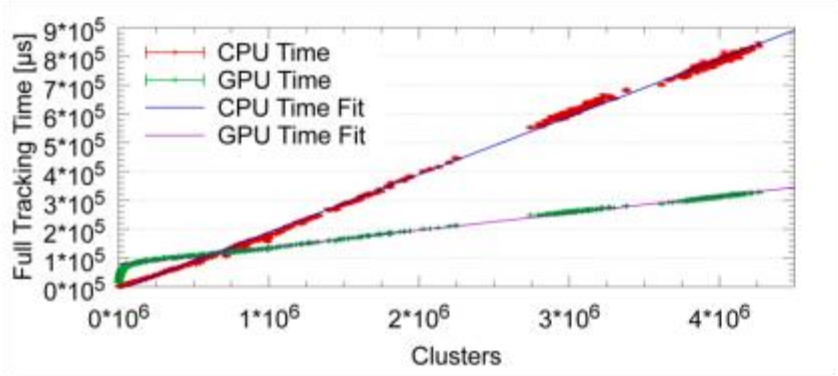
Other experiments

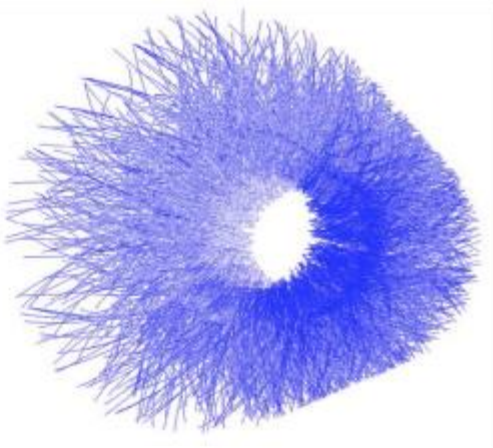
Several other experiments are studying GPUs for online data selection.
Not exhaustive list:

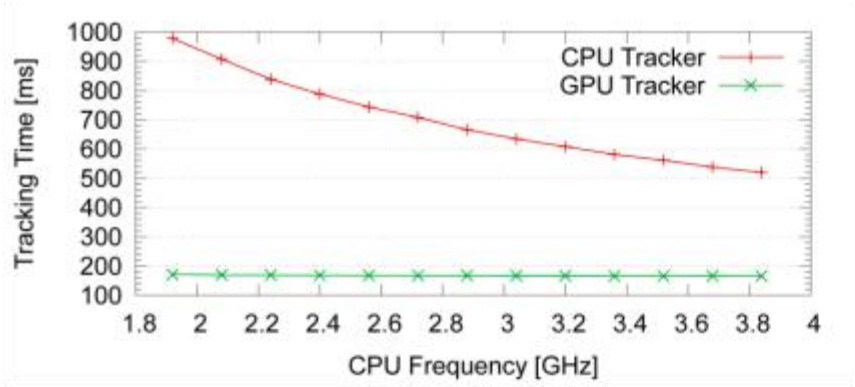
- Alice: A part of the online TPC reconstruction is already running on GPU
- LHCb: is considering GPU (and other accelerators)
- CMS: GPU to help in cluster splitting, for calorimetric jet trigger.
- Panda, CBM, Star, Mu3e, KM3, …

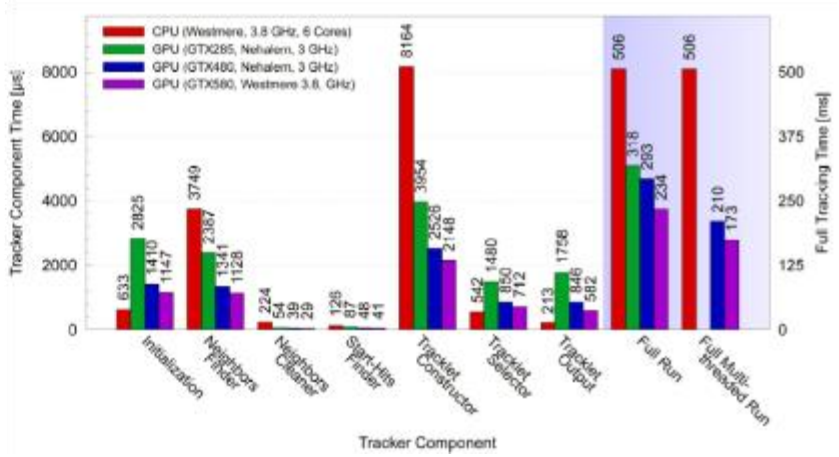
Alice: HLT TPC online Tracking

- 2 kHz input at HLT, $5x10^7$ B/event, 25 GB/s, 20000 tracks/event
- Cellular automaton + Kalman filter
- GTX 580



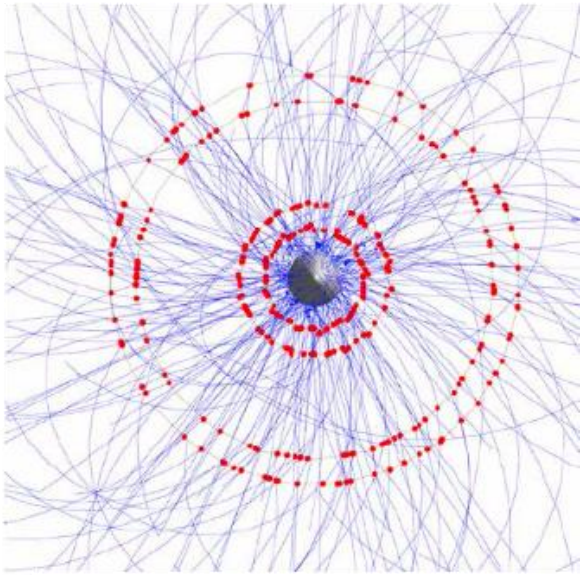


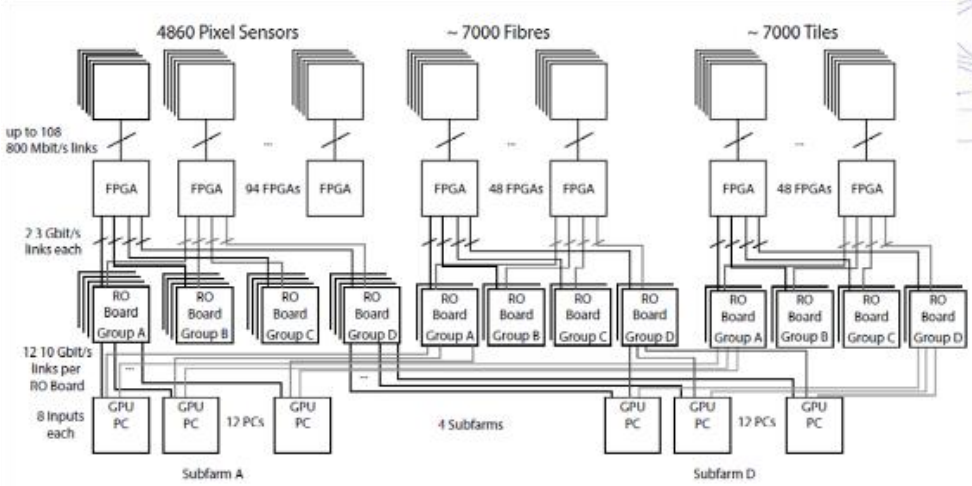


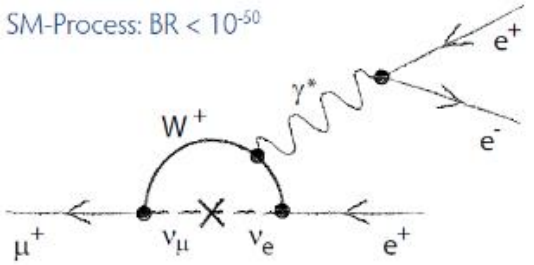


Mu3e

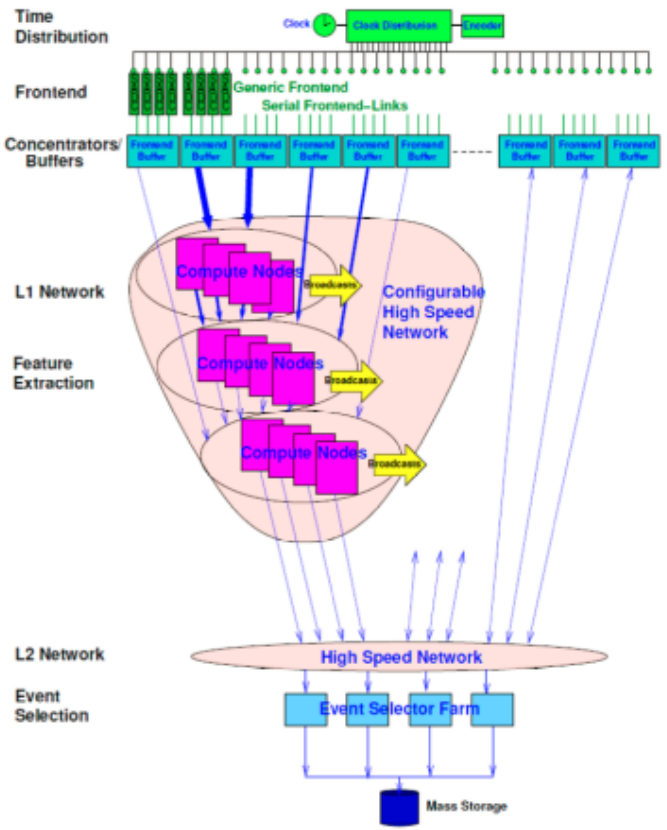
- Possibly a "trigger-less" approach
- High rate: 2×10^9 tracks/s
- >100 GB/s data rate
- Data taking will start >2016







Panda



- 10^7 events/s
- Full reconstruction for online selection: assuming 1-10 ms → 10000 – 100000 CPU cores
- Tracking, EMC, PID,…
- First exercise: online tracking
- Comparison between the same code on FPGA and on GPU: the GPUs are 30% faster for this application (a factor 200 with respect to CPU)

	CPU (ms)	GPU (ms)	Improvement	Occupancy	Notes
total runtime (without Z-Analysis)	117138	590	199		
startUp()	0.25	0.0122	20	2%	runs (num_points) times
setOrigin()	0.25	0.0119	21	25%	runs (num_points) times
clear Hough and Peaks (memset on GPU)	3	0.0463	65	100%	runs (num_points) times
conformalAndHough()	73	0.8363	87	25%	runs (num_points) times
findPeaksInHoughSpace()	51	0.497	103	100%	runs (num_points) times
findDoublePointPeaksInHoughSpace()	4	0.0645	62	100%	runs (num_points) times
collectPeaks()	4	0.066	61	100%	runs (num_points) times
sortPeaks()	0.25	0.0368	7	2%	runs (num_points) times
resetOrigin()	0.25	0.0121	21	25%	runs (num_points) times
countPointsCloseToTrackAndTrackParams()	22444	0.9581	23426	33%	runs once
collectSimilarTracks()	4	2.3506	2	67%	runs once
collectSimilarTracks2()				2%	runs once
getPointsOnTrack()	0.25	0.0187	13	33%	runs (num_tracks) times
nullifyPointsOfThisTrack()	0.25	0.0106	24	33%	runs (num_tracks) times
clear Hough space (memset on GPU)	2	0.0024	833	100%	runs (num_tracks) times
secondHough()	0.25	0.0734	3	4%	runs (num_tracks) times
findPeaksInHoughSpaceAgain()	290	0.2373	1222	66%	runs (num_tracks) times
collectTracks()	0.25	0.0368	7	2%	runs (num_tracks) times