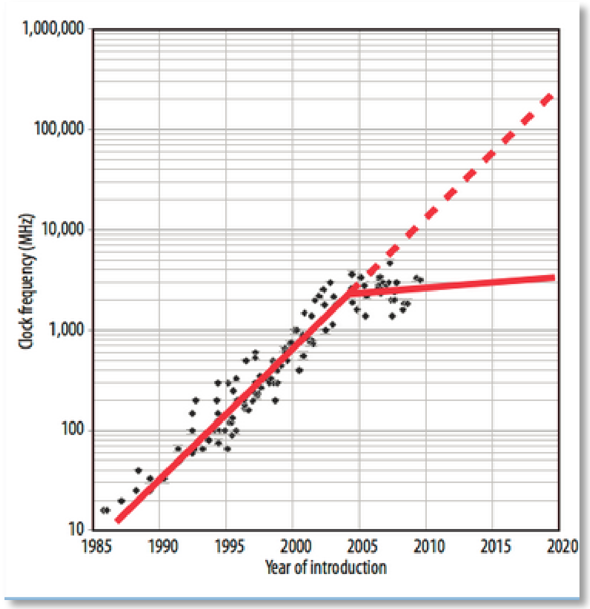
“Graphics Processing Units (GPU) for HEP trigger systems”

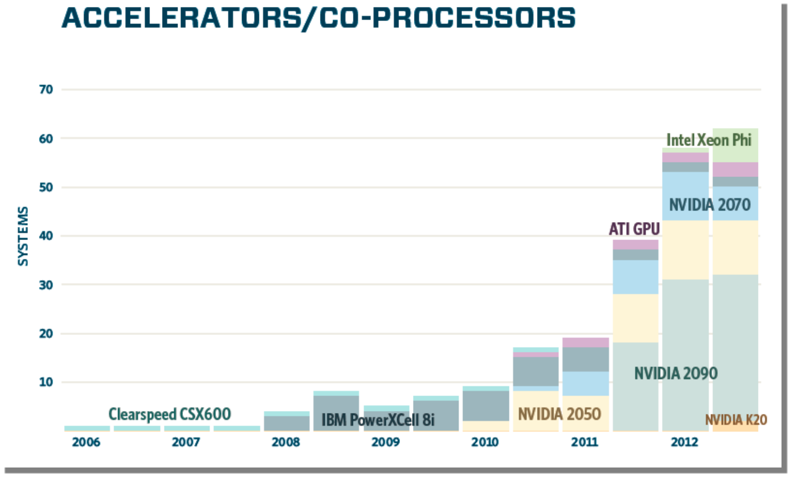
13th PisaMeeting on Advanced Detectors
La Biodola 28.5.2015

Gianluca Lamanna (INFN)
On behalf of GAP collaboration

What are the GPUs?

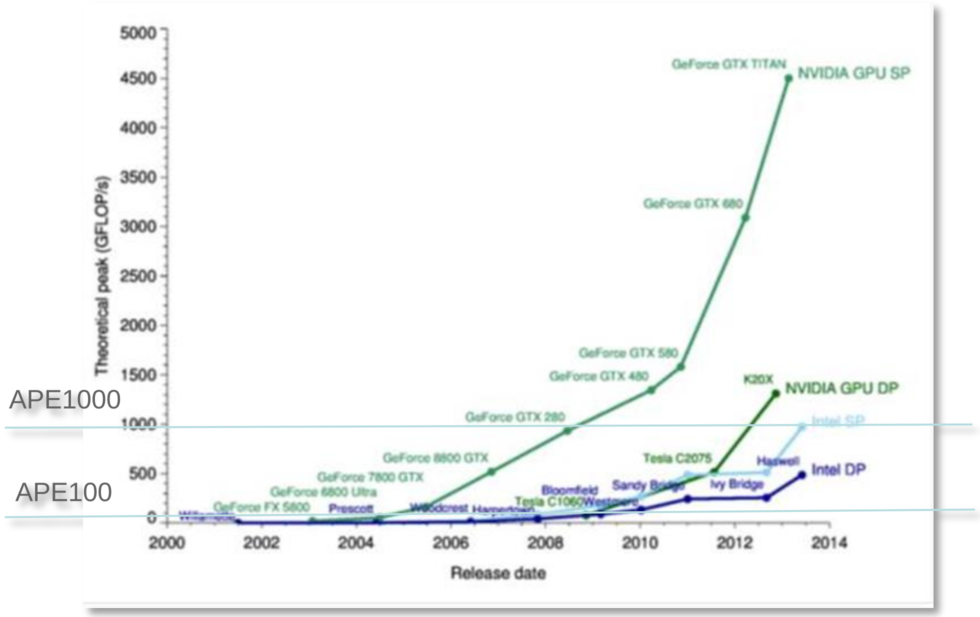
- The technical definition of a GPU is *"a single-chip processor with integrated transform, lighting, triangle setup/clipping, and rendering engines that is capable of processing a minimum of 10 million polygons per second."*
- The possibility to use the GPU for generic computing (GPGPU) has been introduced by NVIDIA in 2007 (CUDA)
- In 2008 OpenCL: consortium of different firms to introduce a multi-platform language for manycores computing.
- At the same time the Moore's law starts to show saturation: parallel computing is the solution!
- Nowadays several supercomputers are based on GPU.

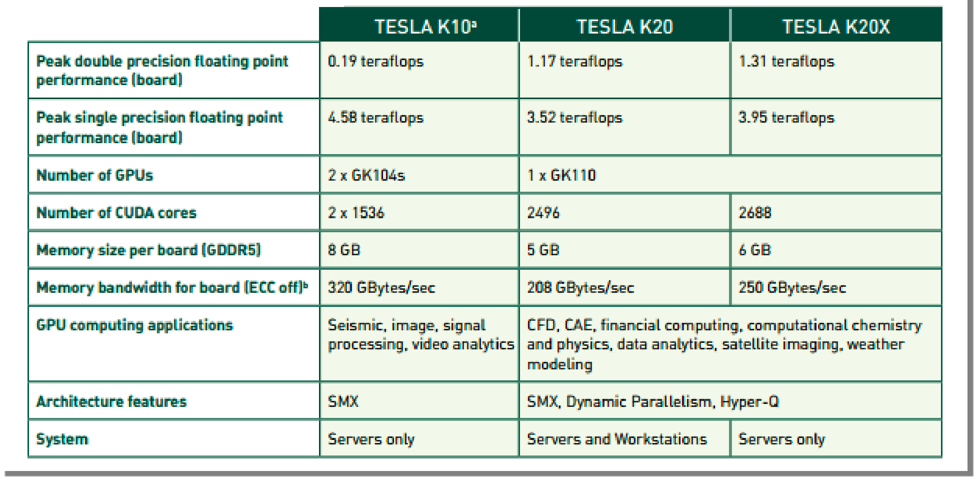




The GPUs are very powerful!

- The GPU is a way to cheat the Moore's law
  - SIMD parallel architecture
  - The PC no longer get faster, just wider.
- Very high computing power for «vectorizable» problems
  - Impressive derivative
  - almost a factor of 2 in each generation
- Continuous development
- Easy to have a desktop PC with teraflops of computing power, with thousand of cores.
- Several applications in HPC, simulation, scientific computing…





	TESLA K10[a]	TESLA K20	TESLA K20X
Peak double precision floating point performance (board)	0.19 teraflops	1.17 teraflops	1.31 teraflops
Peak single precision floating point performance (board)	4.58 teraflops	3.52 teraflops	3.95 teraflops
Number of GPUs	2 x GK104s	1 x GK110	
Number of CUDA cores	2 x 1536	2496	2688
Memory size per board (GDDR5)	8 GB	5 GB	6 GB
Memory bandwidth for board (ECC off)[b]	320 GBytes/sec	208 GBytes/sec	250 GBytes/sec
GPU computing applications	Seismic, image, signal processing, video analytics	CFD, CAE, financial computing, computational chemistry and physics, data analytics, satellite imaging, weather modeling	
Architecture features	SMX	SMX, Dynamic Parallelism, Hyper-Q	
System	Servers only	Servers and Workstations	Servers only

Next generation trigger

- Next generation experiments will look for tiny effects:
  - The trigger systems become more and more important
- Higher readout band
  - New links to bring data faster on processing nodes
- Accurate online selection
  - High quality selection closer and closer to the detector readout
- Flexibility, Scalability, Upgradability
  - More software less hardware

Different Solutions

- Brute force: PCs
  - Bring all data on a huge pc farm, using fast (and eventually smart) routers.
  - Pro: easy to program, flexibility; Cons: very expensive, most of resources just to process junk.
- Rock Solid: Custom Hardware
  - Build your own board with dedicated processors and links
  - Pro: power, reliability; Cons: several years of R&D (sometimes to re-rebuild the wheel), limited flexibility

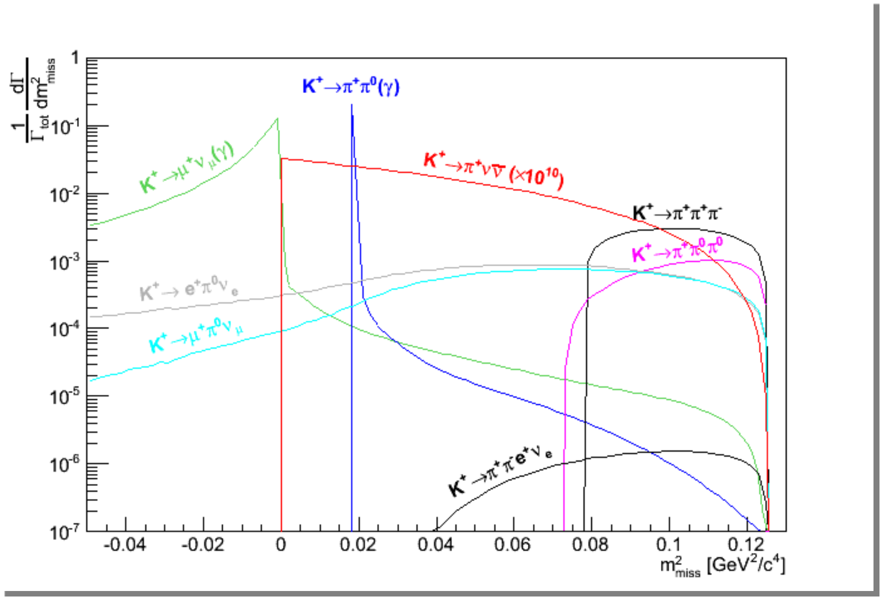
- Elegant: FPGA
  - Use a programmable logic to have a flexible way to apply your trigger conditions.
  - Pro: flexibility and low deterministic latency; Cons: not so easy (up to now) to program, algorithm complexity limited by FPGA clock and logic.
- Off-the-shelf: GPU
  - Try to exploit hardware built for other purposes continuously developed for other reasons
  - Pro: cheap, flexible, scalable, PC based. Cons: Latency

GPU in low level trigger

- Computing power: Is the GPU fast enough to take trigger decision at tens of MHz events rate?

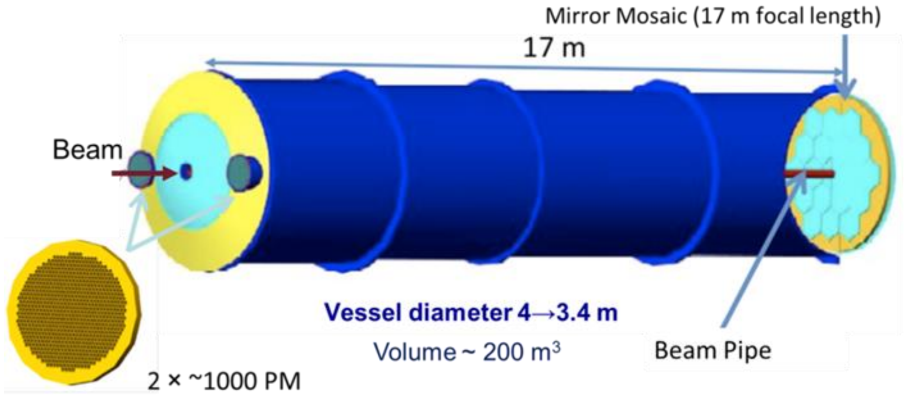
- Latency: Is the GPU latency per event small enough to cope with the tiny latency of a low level trigger system? Is the latency stable enough for usage in synchronous trigger systems?

Low Level Trigger: NA62 Test bench

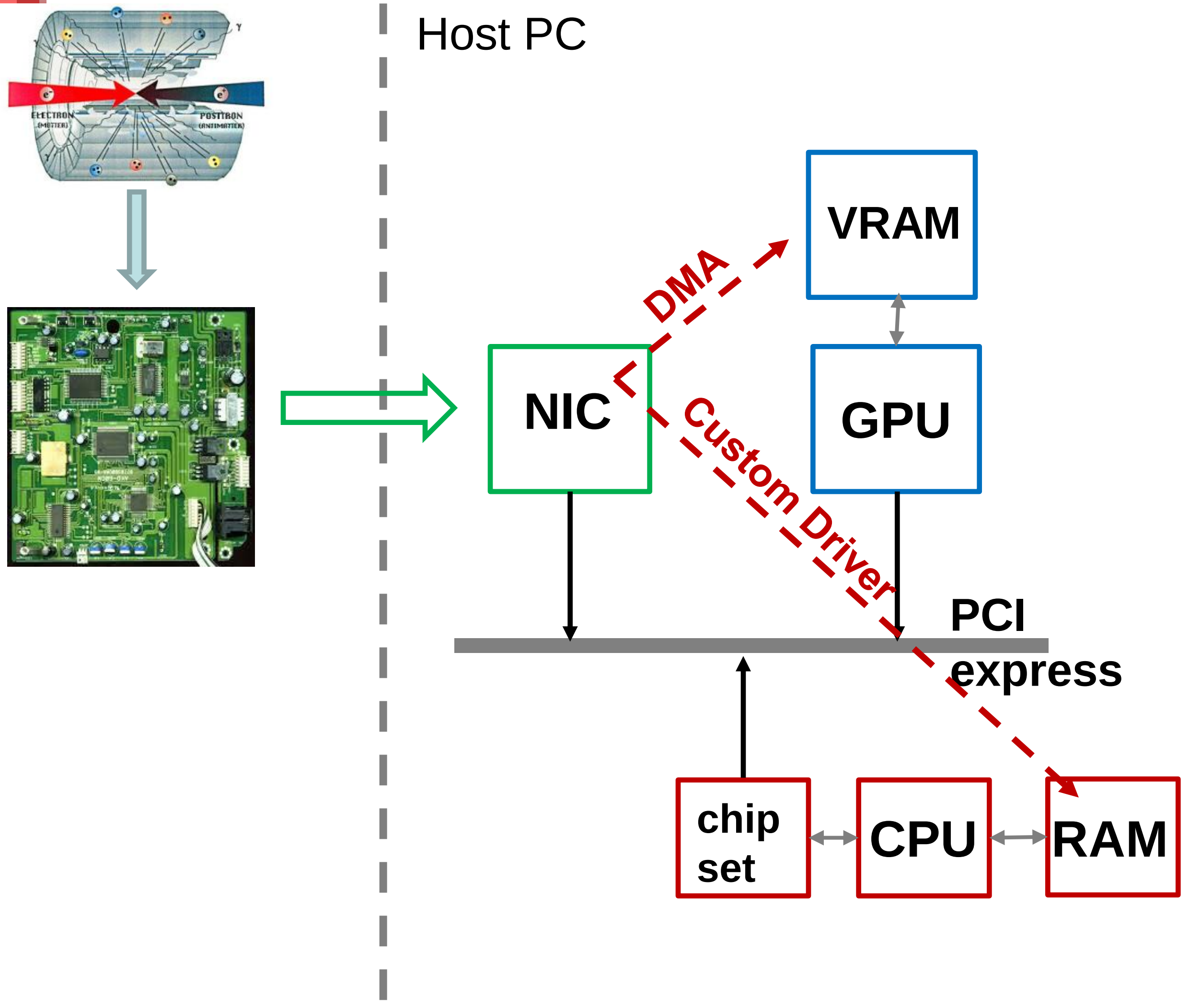


- Kaon decays in flight
  - High intensity unseparated hadron beam (6% kaons).
  - Event by event K momentum measurement.
- Huge background from kaon decays
  - ~10^8 background wrt signal
  - Good kinematics reconstruction.
  - Efficient veto and PID system for not kinematically constrained background.

- RICH:
  - 17 m long, 3 m in diameter, filled with Ne at 1 atm
  - Distinguish between pions and muons from 15 to 35 GeV
    - 2 spots of 1000 PMs each
    - Time resolution: 70 ps
    - MisID: 5×10^{-3}
    - 10 MHz events: about 20 hits per particle



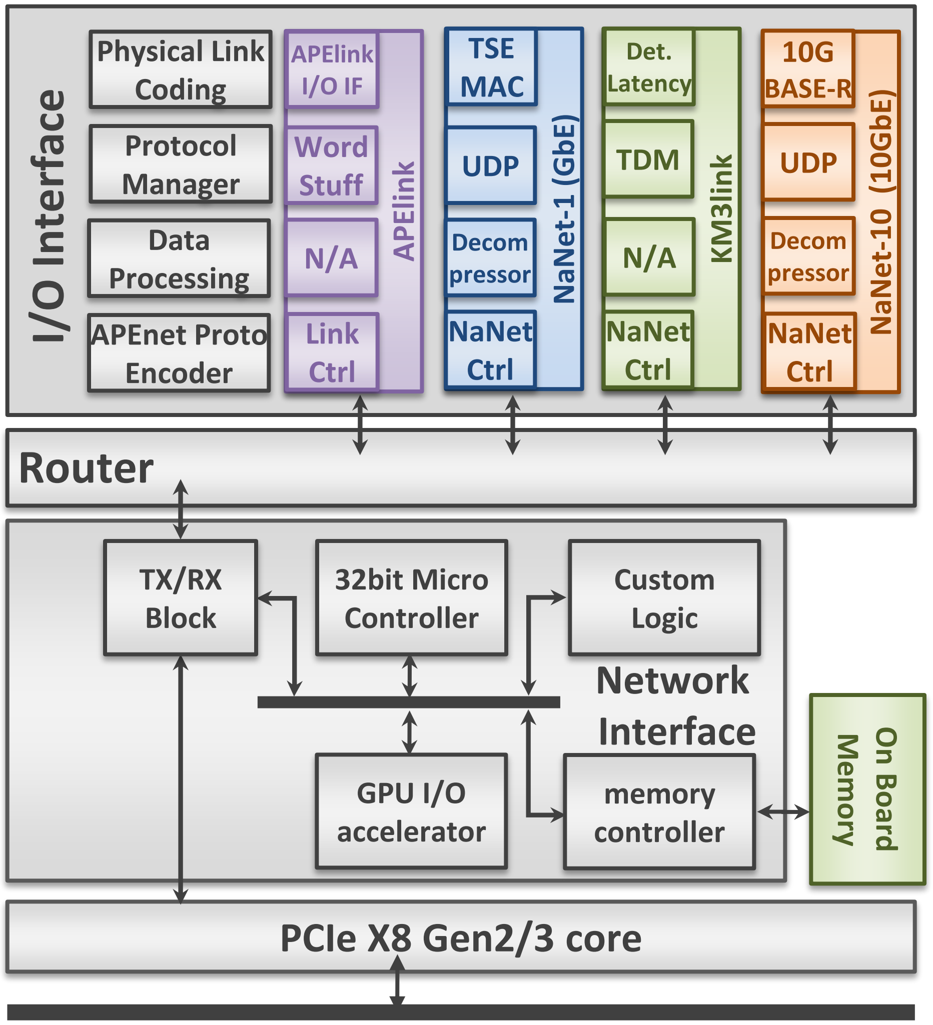
Latency: main problem of GPU computing



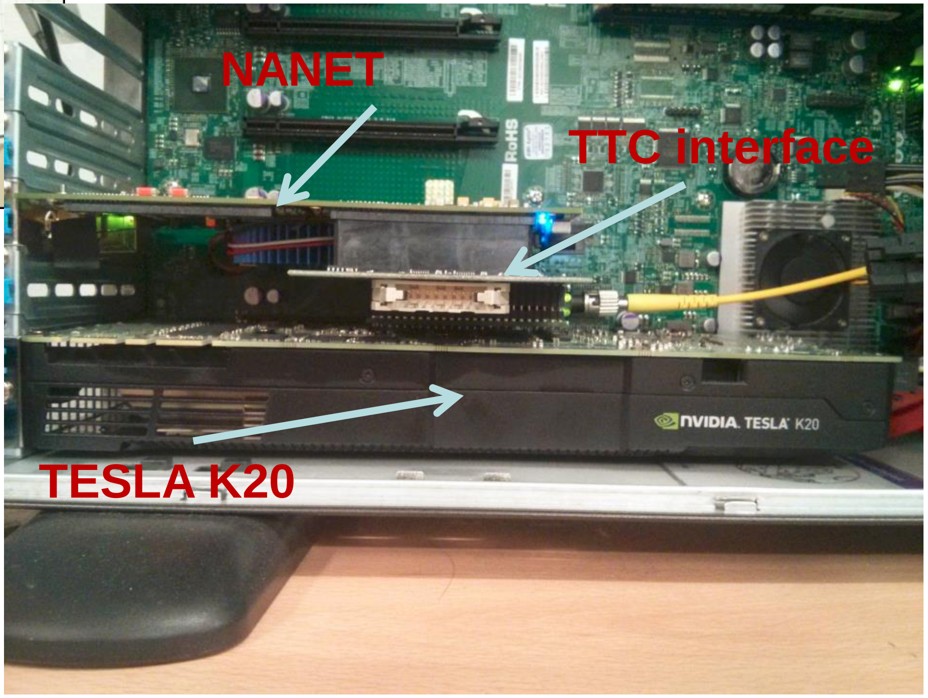
- Total latency dominated by double copy in Host RAM
- Decrease the data transfer time:
  - DMA (Direct Memory Access)
  - Custom manage of NIC buffers
- *"Hide"* some component of the latency optimizing the multi-events computing

Nanet board

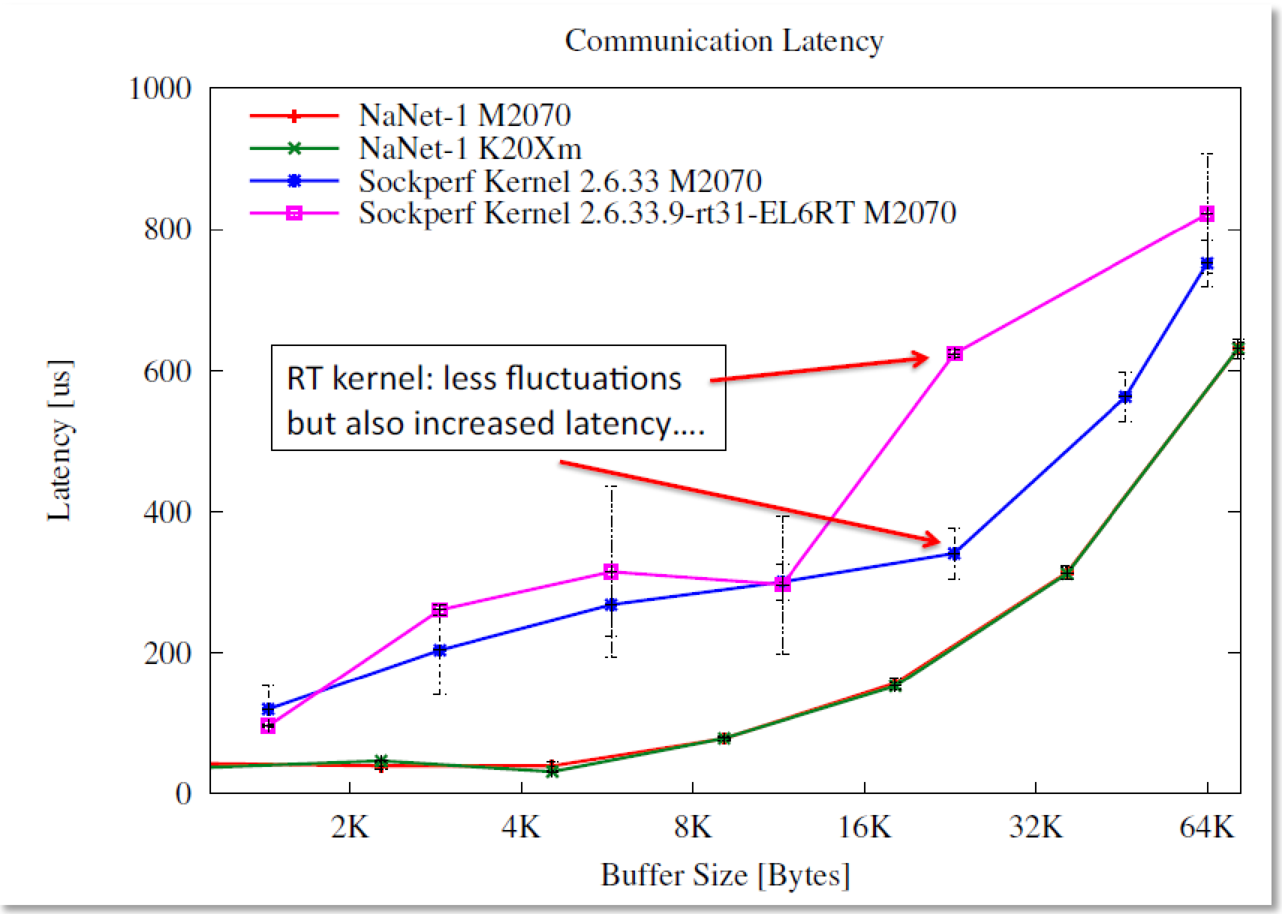
- Nanet: board based on the ApeNet+ card logic
  - PCIe interface with GPU Direct P2P/RDMA capability
  - Offloading of network protocol
  - Multiple link support
  - Use FPGA resources to perform on-the-fly data preparation



Nanet in NA62

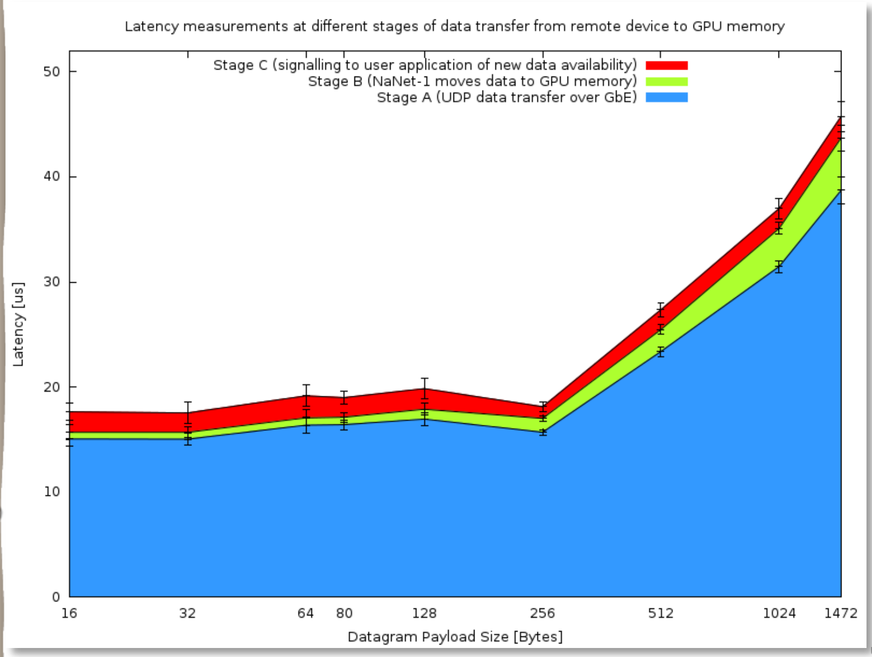


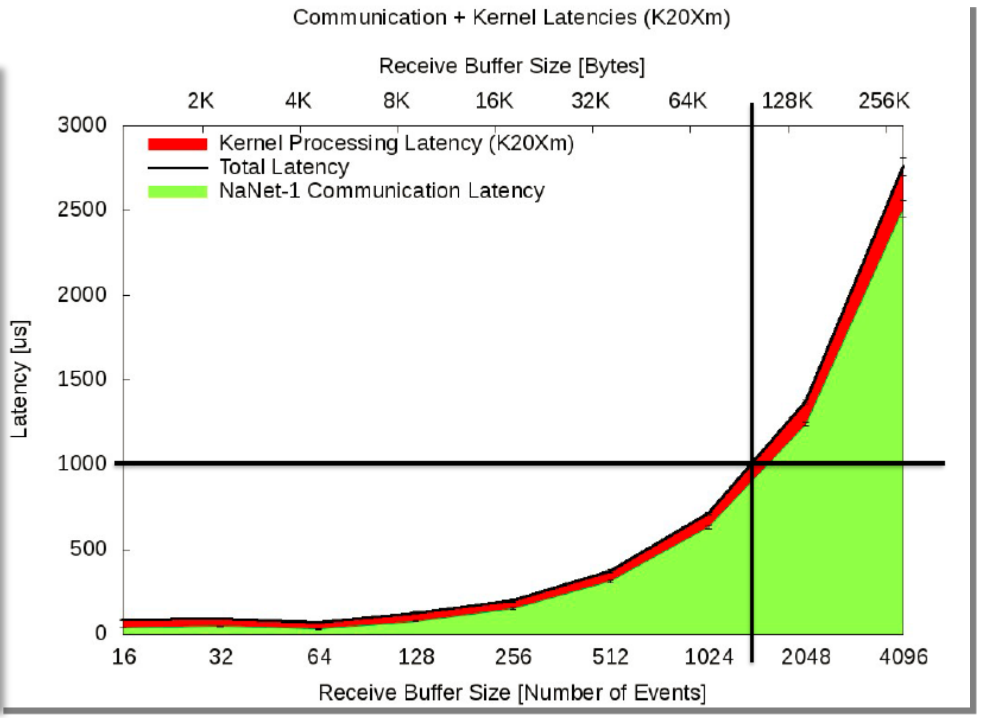
Latency performances



RT kernel: less fluctuations
but also increased latency….

Performances

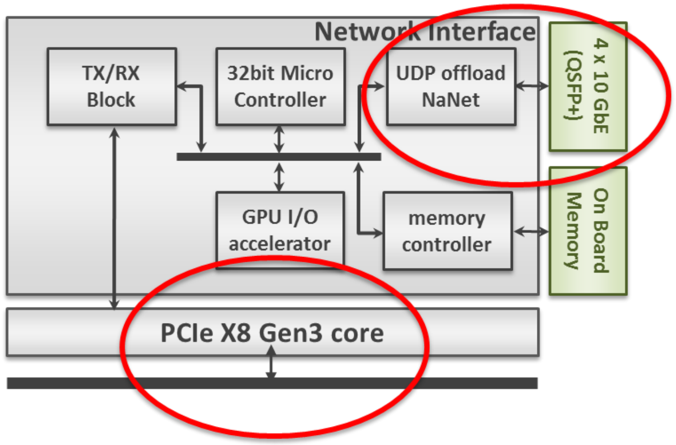


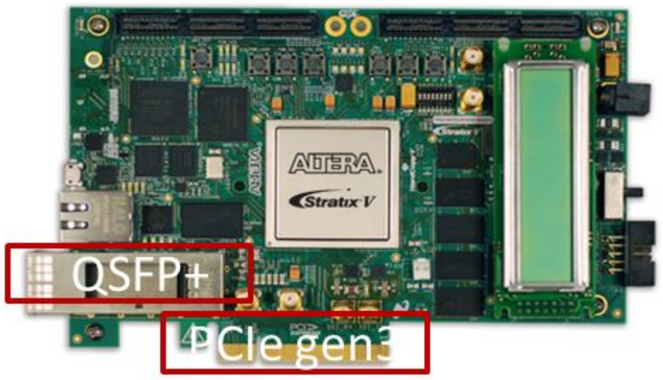


After NANET latency if fully dominated by GbE transmission.

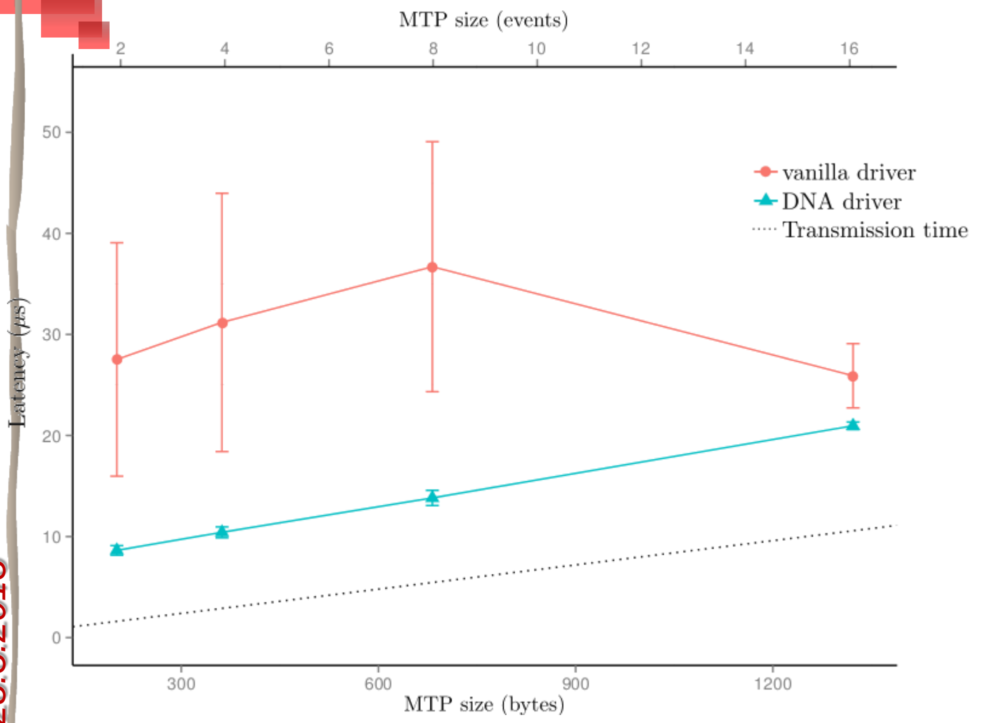
Nanet-10: the next generation

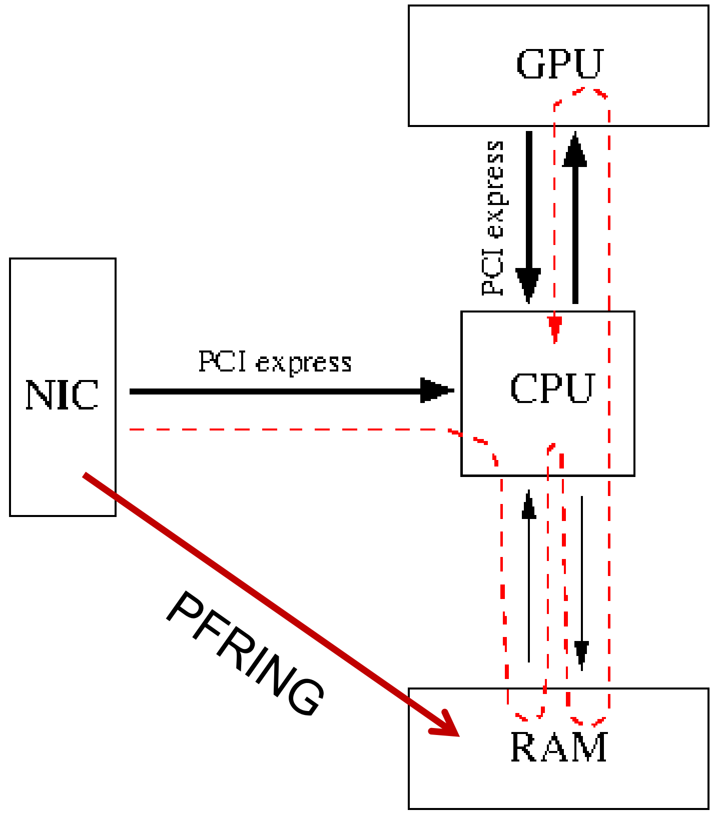
- Will be implemented on the Altera Stratix V dev board but porting on cheaper board (Terasic TR5-F40W) is possible
  - Four 10 GbE SFP+ ports
  - PCIe Gen3 (8 GB/s)
  - Faster embedded Altera transceivers (up to 14.1 Gbps)
  - hardened 10GBASE-R PCS
  - Ready in July





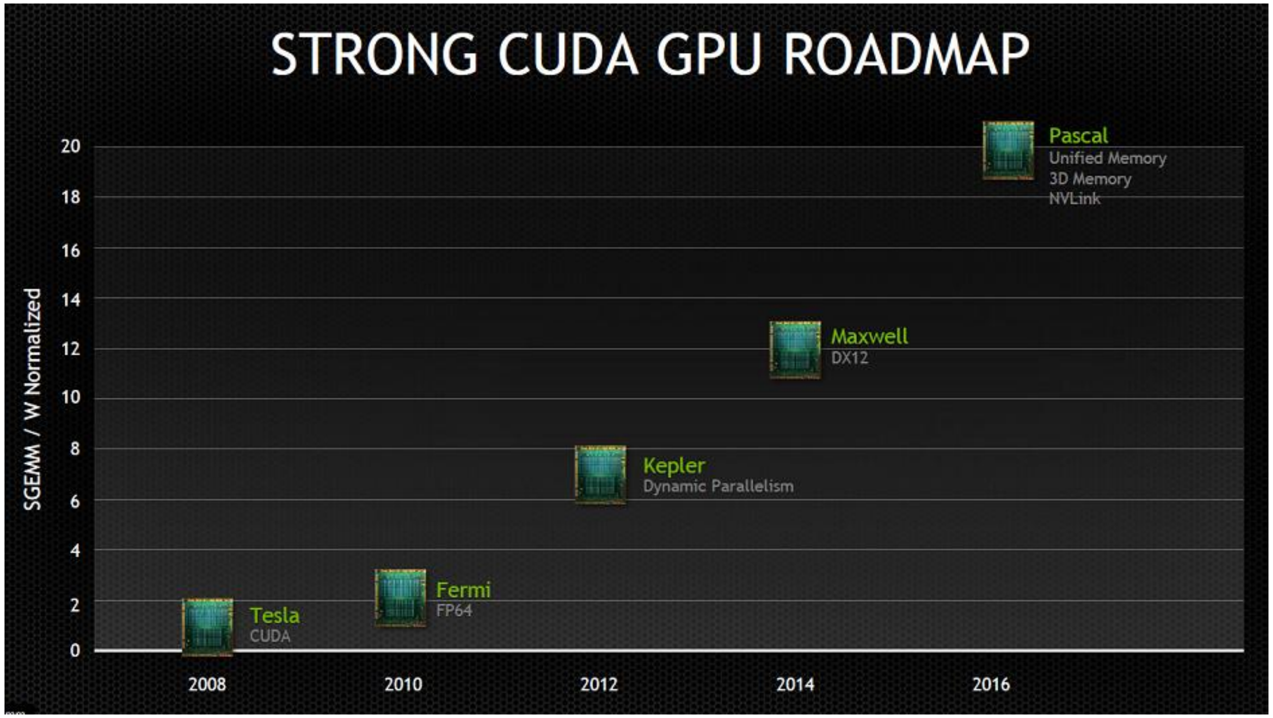
PFRING





- Special driver for direct access to NIC buffer
- Data are directly available in userland
- Double copy avoided
- Pros: No extra HW needed; Cons: Pre-processing on CPU

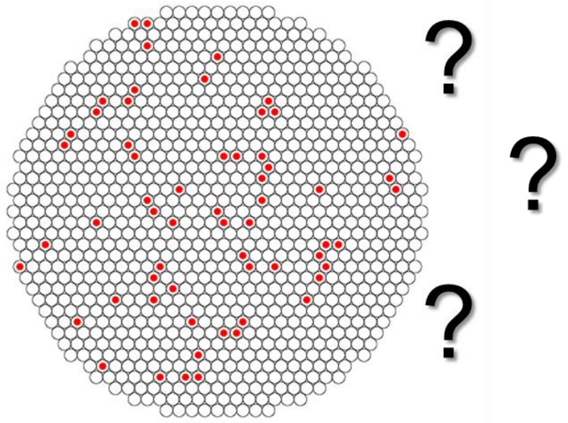
The Future



- Big effort from NVIDIA to reduce latency
  - NVLINK: 80-200 GB/s data transfer between CPU and GPU (CPU->RAM is max 75 GB/s in DDR4)
  - Unified memory: reduce overhead for data transfer.

Ring fitting

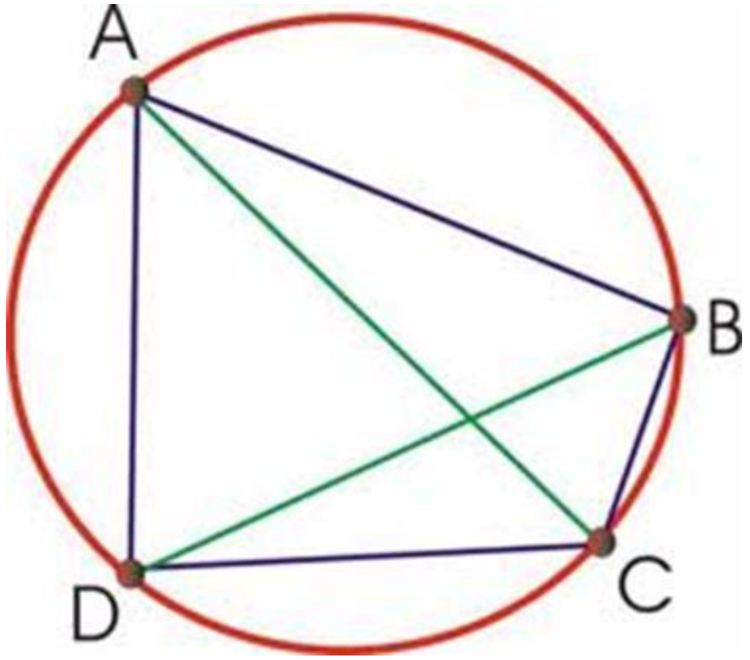
- Trackless
  - no information from the tracker
  - Difficult to merge information from many detectors at L0
- Fast
  - Not iterative procedure
  - Events rate at levels of tens of MHz
- Low latency
  - Online (synchronous) trigger
- Accurate
  - Offline resolution required



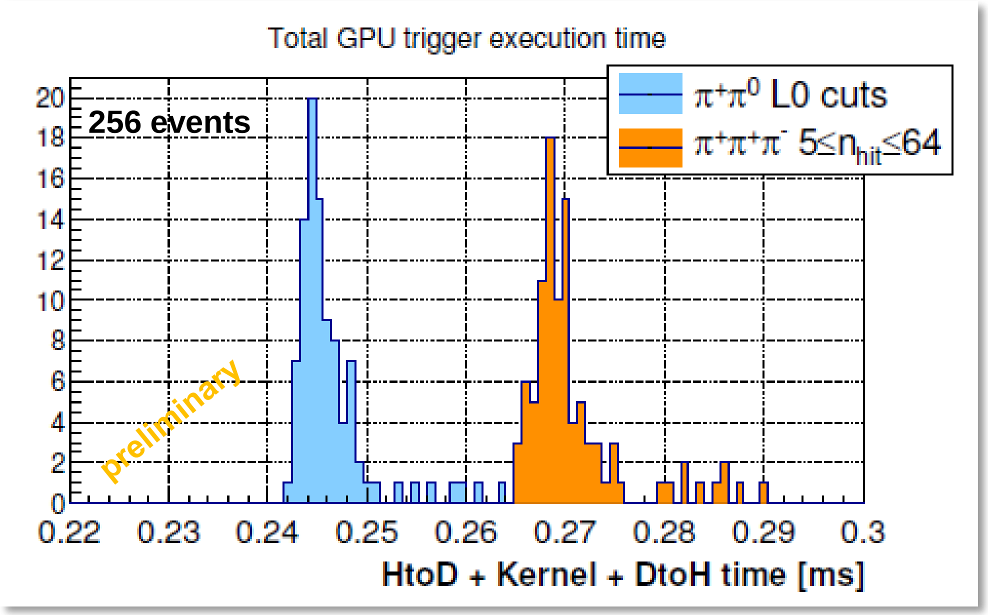
- Multi rings on the market:
  - With seeds: Likelihood, Constrained Hough, …
  - Trackless: fiTQun, APFit, possibilistic clustering, Metropolis-Hastings, Hough transform, …

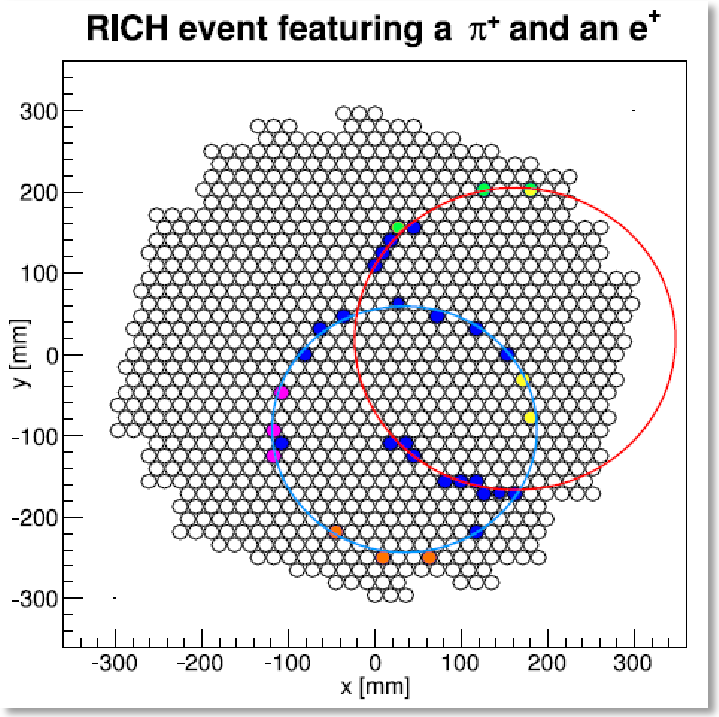
Almagest: multi-ring identification

- New algorithm (Almagest) based on Ptolemy's theorem: *"A quadrilateral is cyclic (the vertex lie on a circle) if and only if is valid the relation: AD*BC+AB*DC=AC*BD "*
- Select a triplet and check if all the other points lie on the same ring by checking the Ptolemy's theorem
- Design a procedure for parallel implementation



Almagest: performances

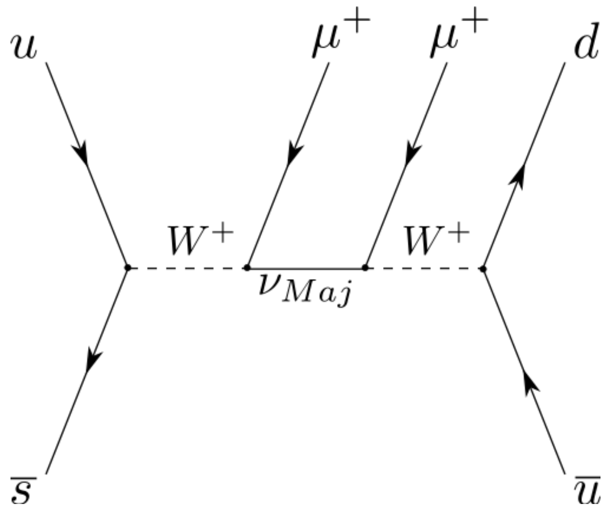


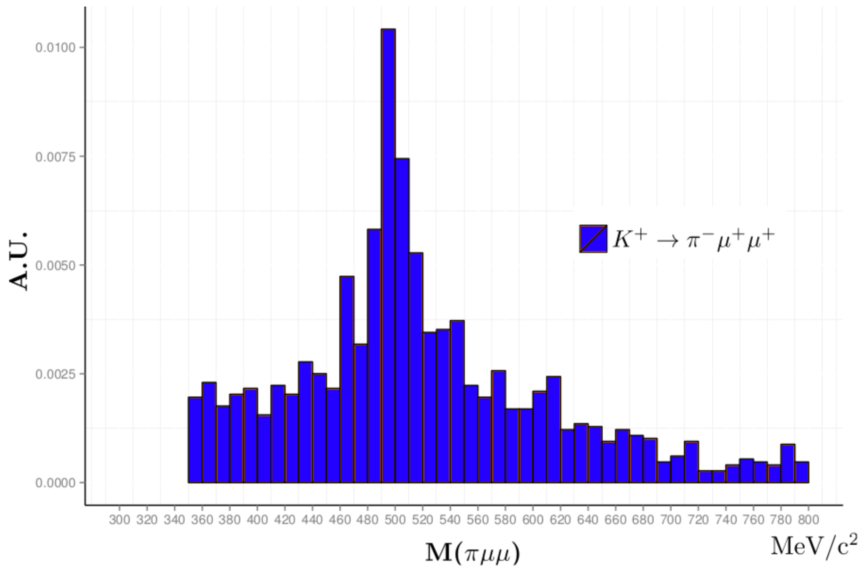


- Preliminary: efficiency greater than 80% (depends on the number of rings)
- About 1us per event for multi-ring
- Test on single C2070 board

Planning

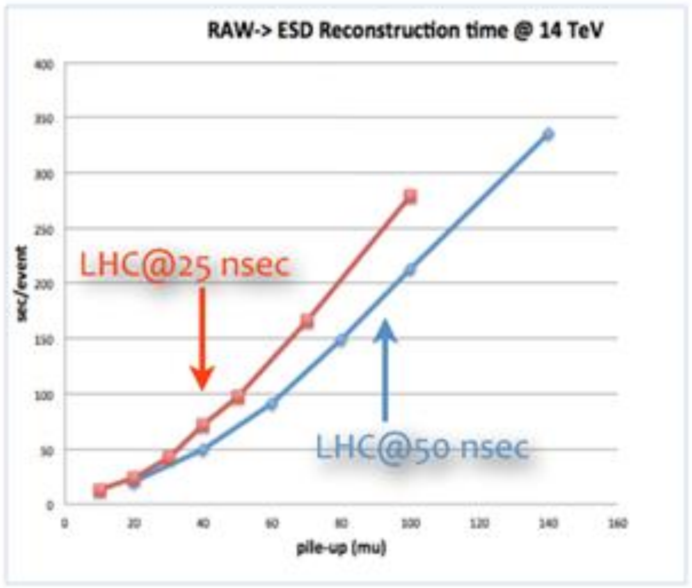
- NA62 physics run will start at middle of June
- Prototype with NANET-1 and PFRING in parasitic mode
- Possibility to extend NA62 physics program (with higher data collection efficiency)
- For example: online tagging of BR($K^+ \rightarrow \pi^- \mu^+ \mu^+$) < 1.1×10^{-9} @ 90% CL





HLT with GPU

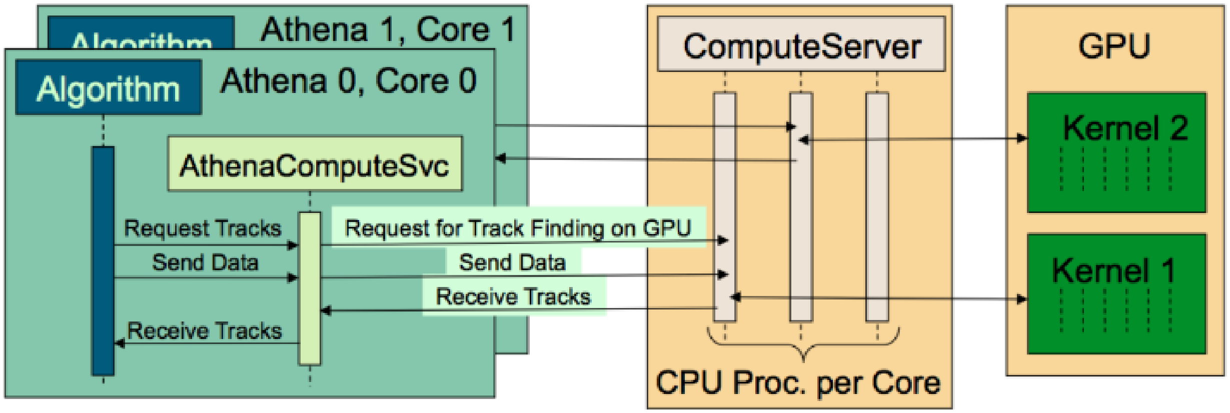
- HLT is a *"natural"* place where to use GPU
- The increasing in LHC luminosity and in the number of overlapping events poses new challenges to the trigger system, and new solutions have to be developed for the fore coming upgrades



  - A simple increase of the threshold can reduce signal efficiency drastically
  - More resolution and more complex reconstruction in HLT capabilities
- Reconstruction complexity and computing time scales with number of hits/tracks
  - Higher throughput means increase network and CPU capabilities
  - Parallel computing is the solution

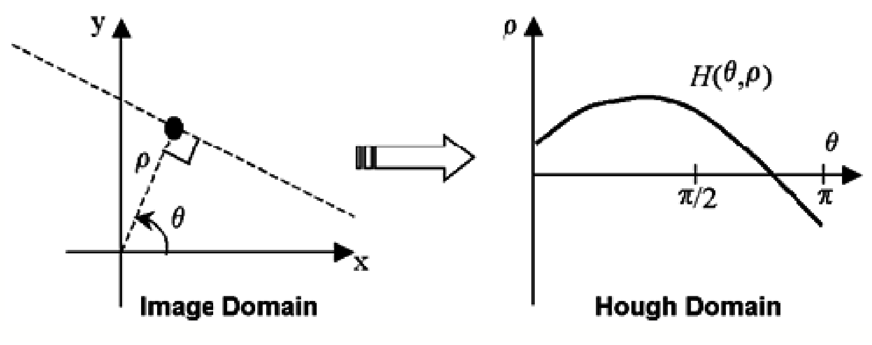
Physics Case: ATLAS

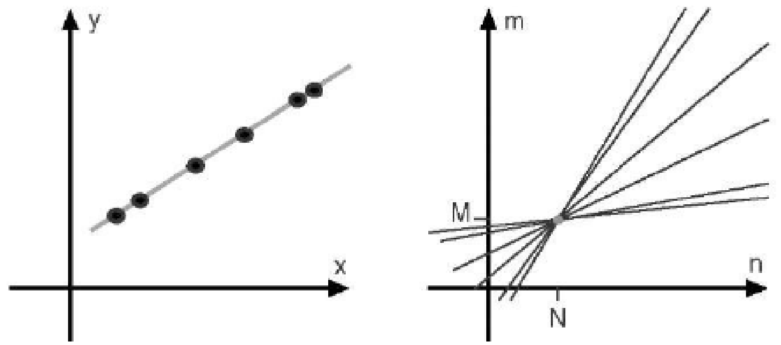
- Coordinated activity to build a Demonstrator in ATLAS
  - How does a HLT based on CPU+GPU servers compare with a CPU only solution?
  - Ready for the end of the year
- Several algorithms could benefit from GPU acceleration:
  - Tracking
  - Calo
  - Muons
- Integration of GPU with the existing Athena framework by using an additional layer (APE client/server)



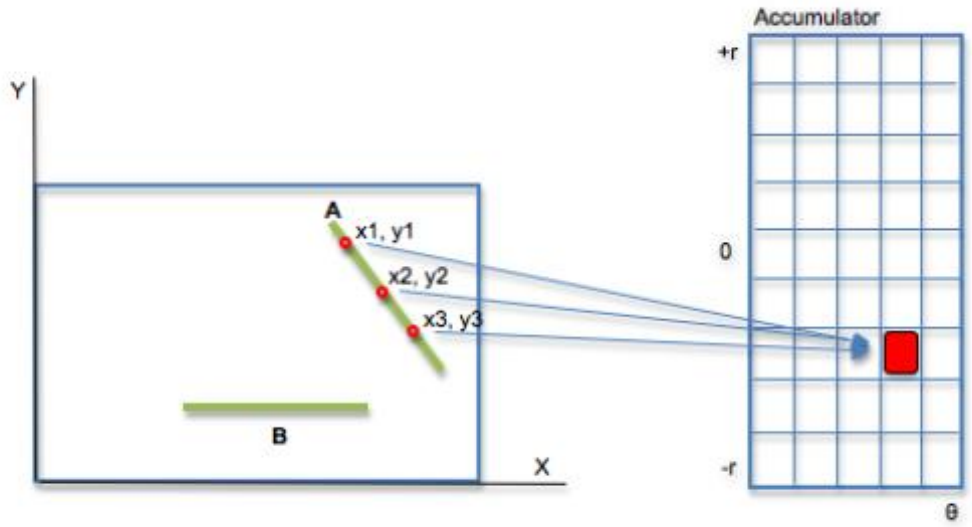
The Muon algorithm

- Parallelized Hough transform to identify segments in the muons spectrometer (most computing demanding part in the muon algorithm)

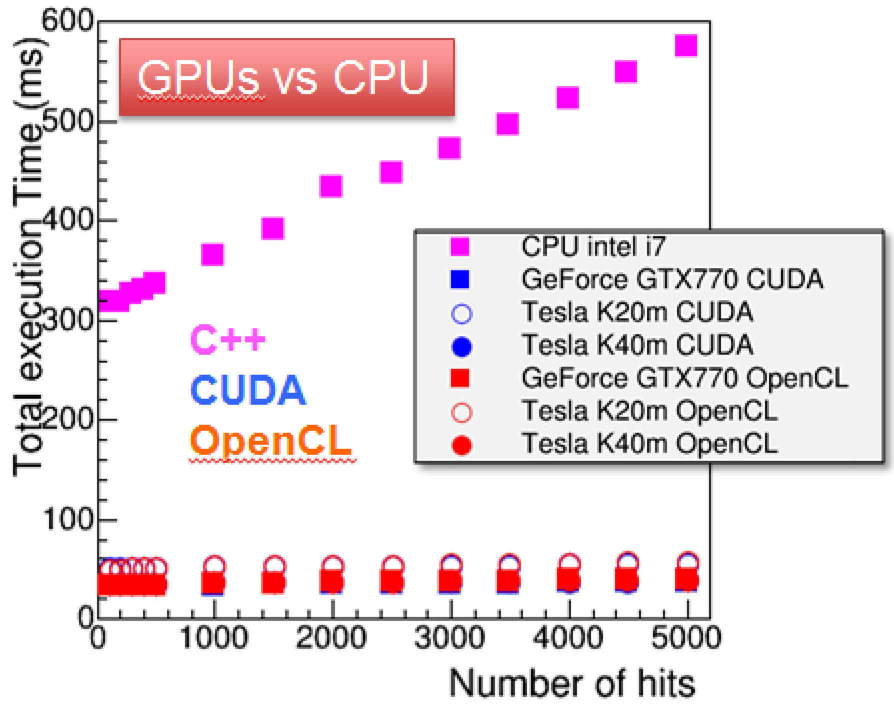


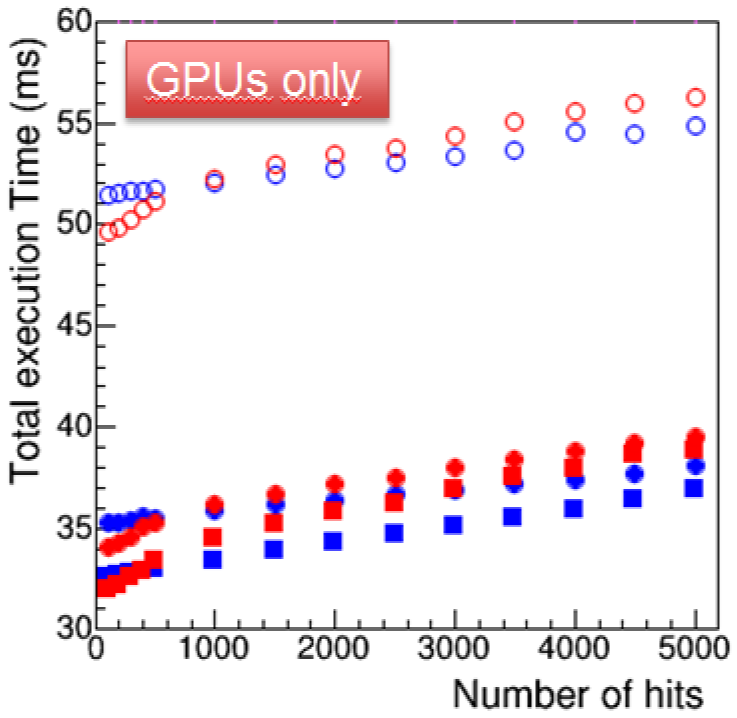


- The Hough transform projects points in curves in the parameter space.
- The tracks is identified through a voting procedure in the accumulator space.



Example of tracking





[Rinaldi (INFN BO) CCR15]

- Standalone simulation of a typical central tracker detector
- Hough transform
- Factor x15 gain with respect to CPU version

Other experiments

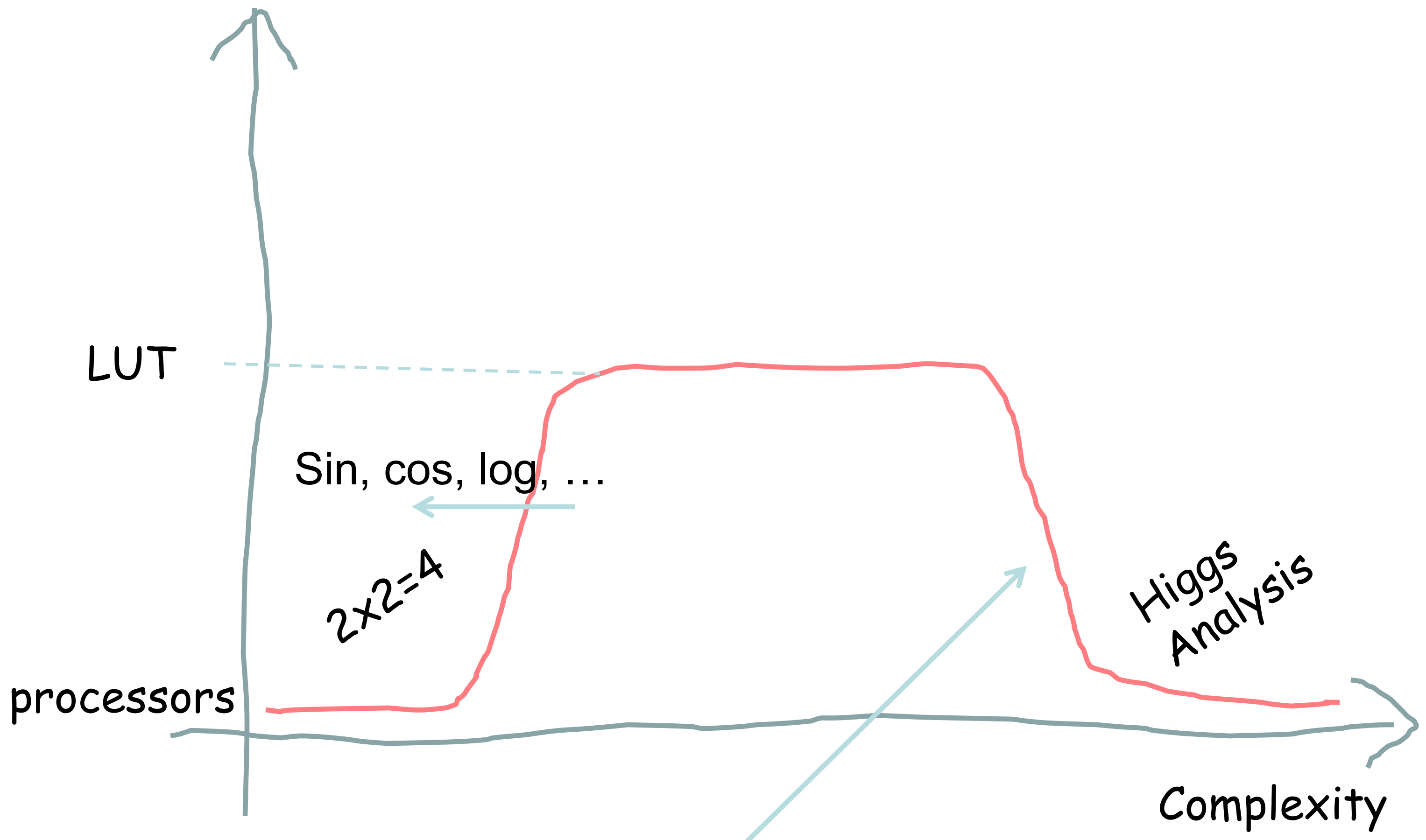
- Several other experiments are studying GPUs for online data selection.
- Not exhaustive list:
  - Alice: A part of the online TPC reconstruction is already running on GPU
  - LHCb: is considering GPU (and other accelerators)
  - CMS: GPU to help in cluster splitting, for calorimetric jet trigger.
  - Panda, CBM, Star, Mu3e, KM3, …

Conclusions

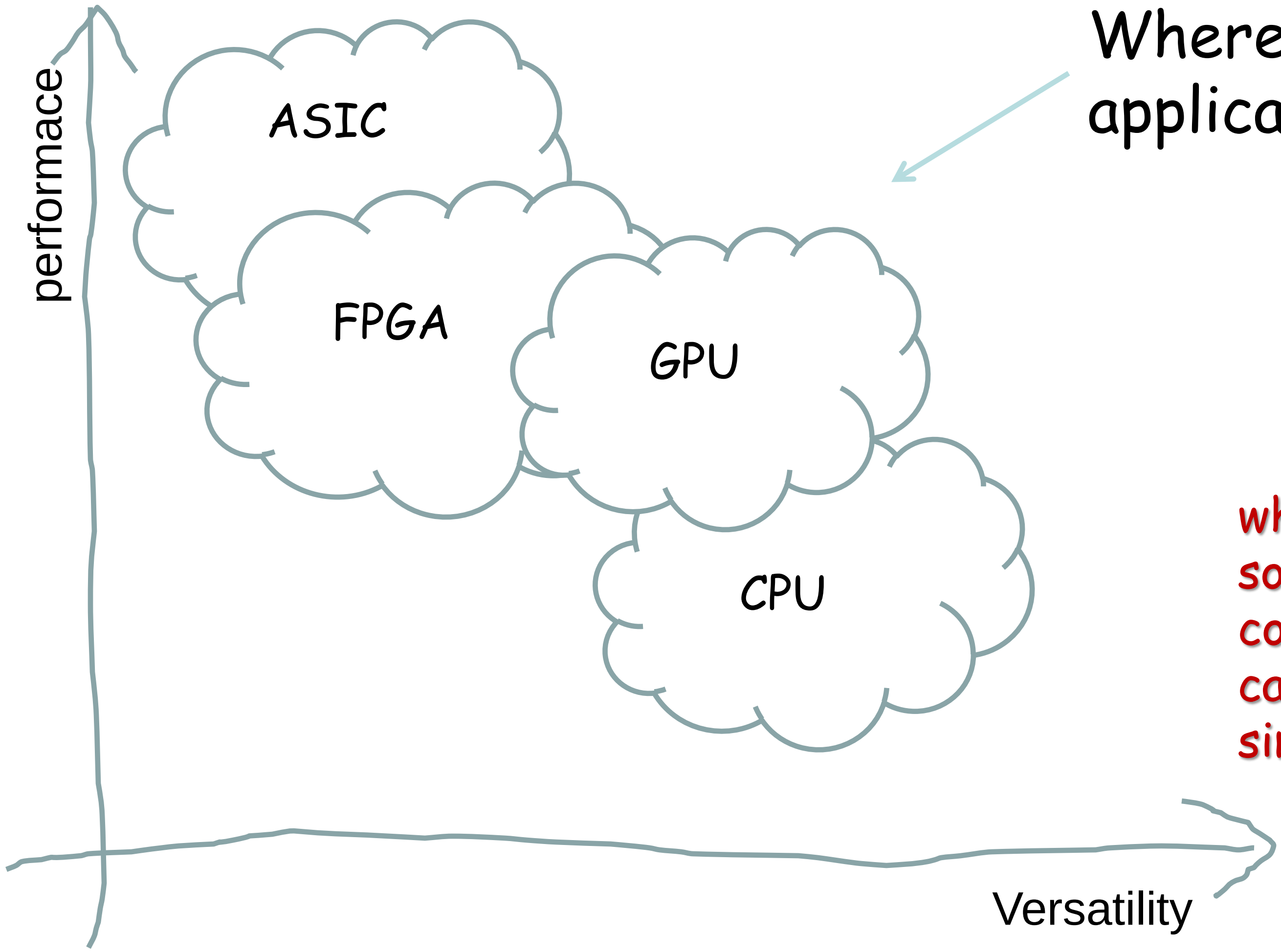
- The use of GPU for scientific computing is becoming commonplace
  - The future will be necessarily parallel!!!
- Several possible uses in HEP: data analysis, Monte Carlo, …
- The GPU in the trigger could give several advantages, but the processing performances should be carefully studied (IO, Latency, Throughput)
- Several experiments are thinking about to use GPU in the trigger in future (both in Lower and Higher levels):
  - Upgrade: ATLAS, LHCb, ALICE (already used GPU in run1), …
  - NA62, PANDA, CBM, STAR, …
- GPUs are flexibles, scalable, powerful, ready to use, cheap and take advantage of continuous development for other purposes: they are a viable alternative to other expensive and less powerful solution.

SPARES

Computing vs LUT



Where is this limit?
It depends …
In any case the GPUs
aim to shrink this space



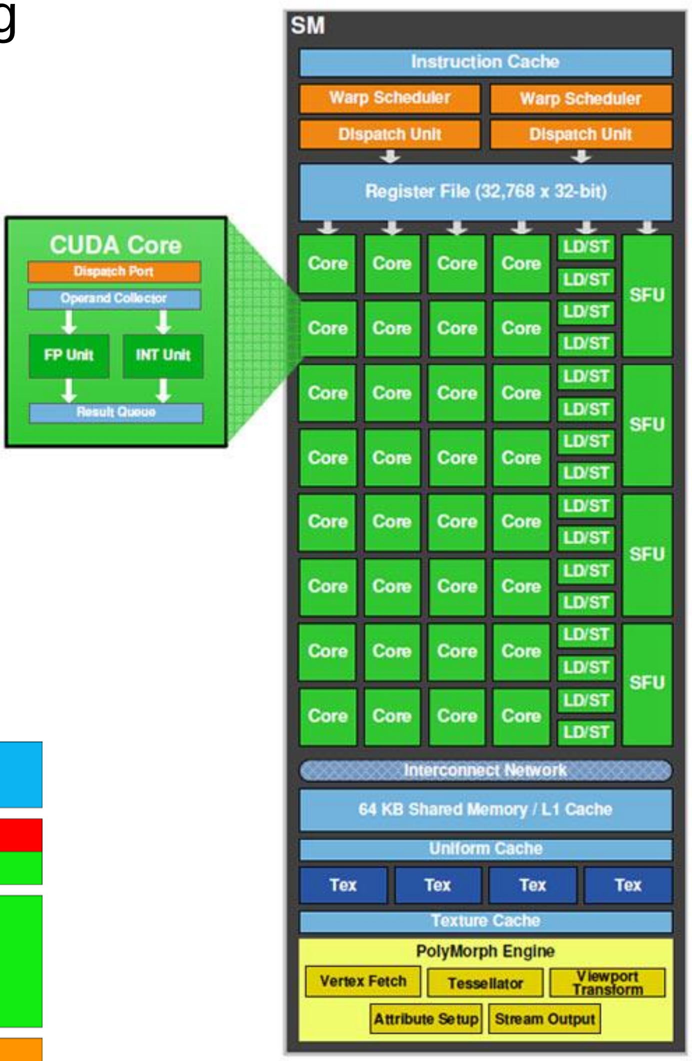
Where is your application?

why would I do something in such a complicated way if I can just make it simple?

General purpose or dedicated hardware??? → It depends on the application→ i.e. memory speed vs processor speed → GPUs are a good "compromise" …fill the GAP

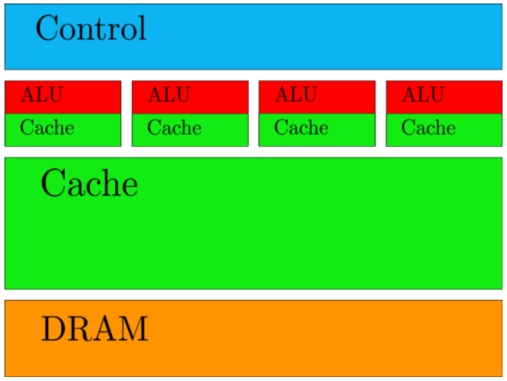
Different architecture

- SMX executes kernels (aka functions) using hundreds of threads concurrently.
- SIMT (Single-Instruction, Multiple-Thread)
- Instructions pipelined
- Thread-level parallelism
- Instructions issued in order
- No Branch prediction but Branch predication



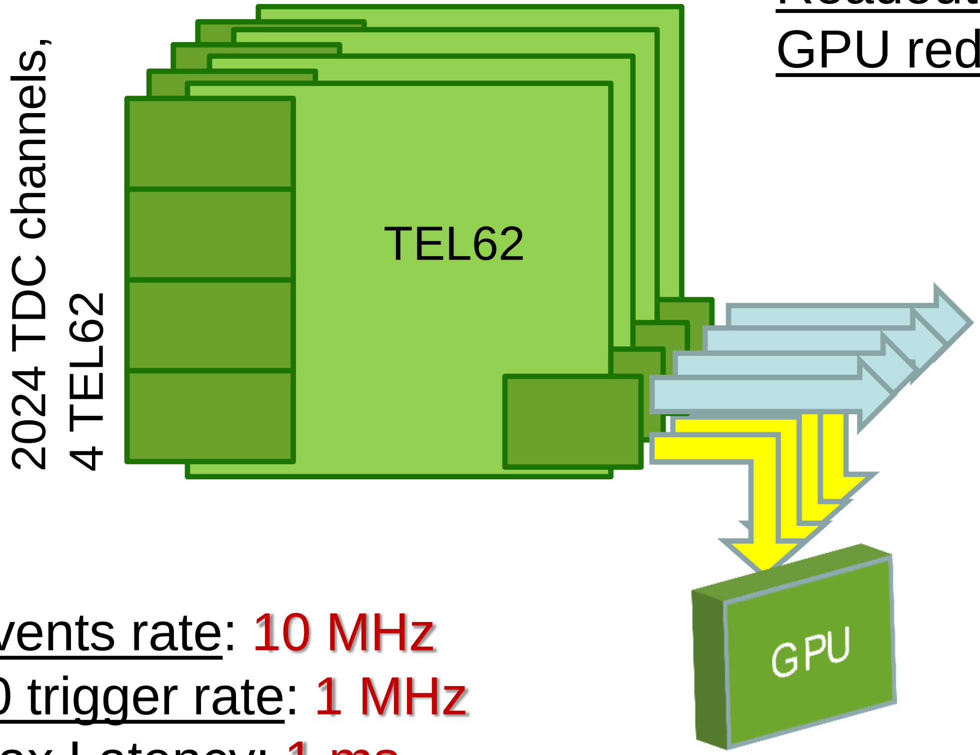
Streaming Multiprocessor (SM)

- Large caches
- ILP
- Brach prediction
- Few powerful ALU
- Short pipeline



CPU

NA62 GPU trigger system



Readout event: 1.5 kb (1.5 Gb/s)
GPU reduced event: 300 b (3 Gb/s)

8x1Gb/s links for data readout
4x1Gb/s Standard trigger primitives
4x1Gb/s GPU trigger

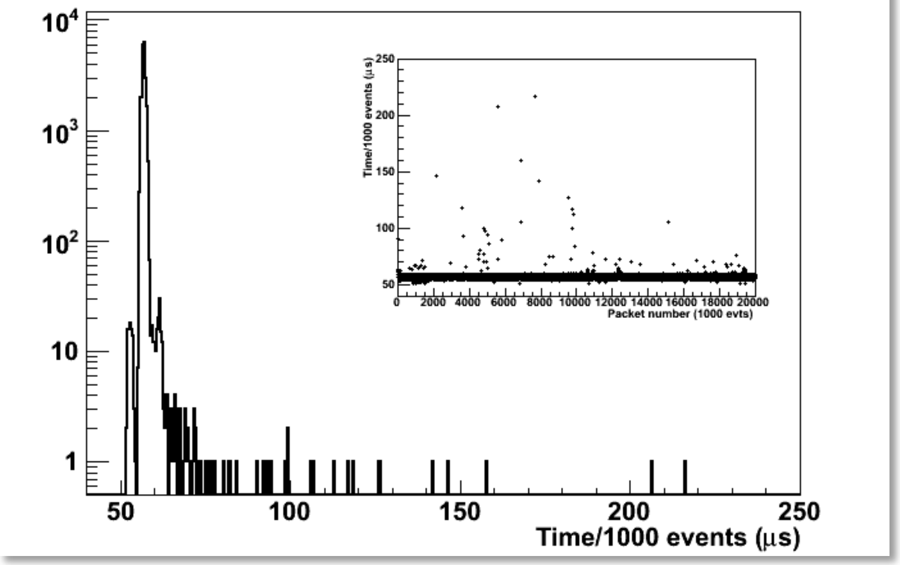
GPU NVIDIA TITAN:

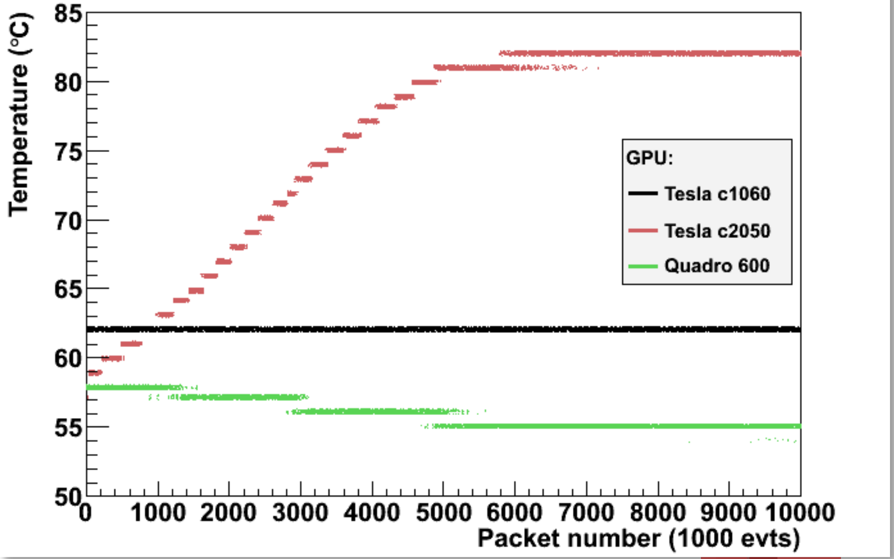
- 2688 cores
- 4.5 Teraflops
- 6GB VRAM
- PCI ex.gen3
- Bandwidth: 288 GB/s

Events rate: 10 MHz
L0 trigger rate: 1 MHz
Max Latency: 1 ms
Total buffering (per board): 8 GB
Max output bandwidth (per board): 4 Gb/s

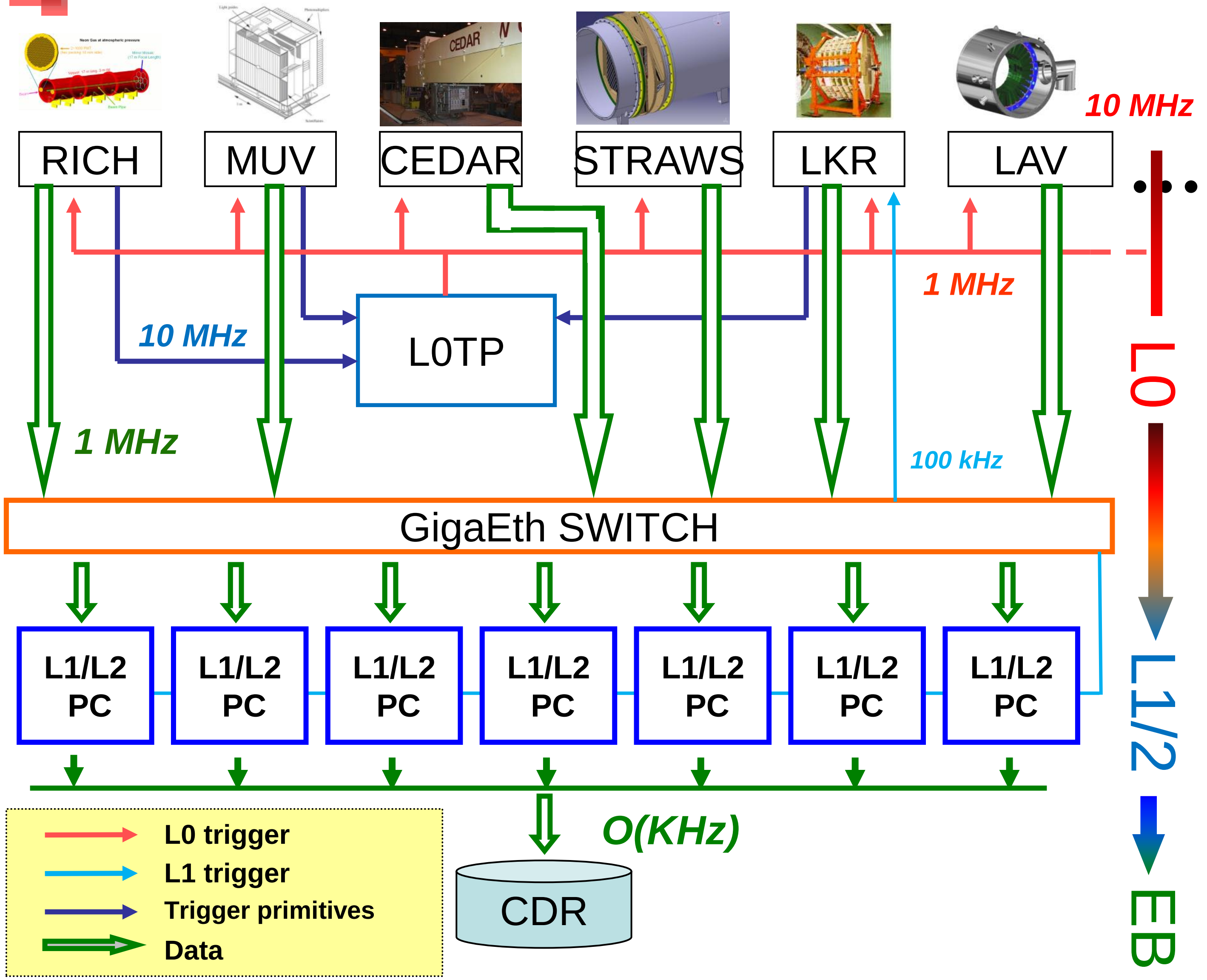
Processing time stability

- The stability of the execution time is an important parameter in a synchronous system
- The GPU (Tesla C1060, MATH algorithm) shows a "quasi deterministic" behavior with very small tails.
- The GPU temperature, during long runs, rises in different way on the different chips, but the computing performances aren't affected.





The NA62 TDAQ system

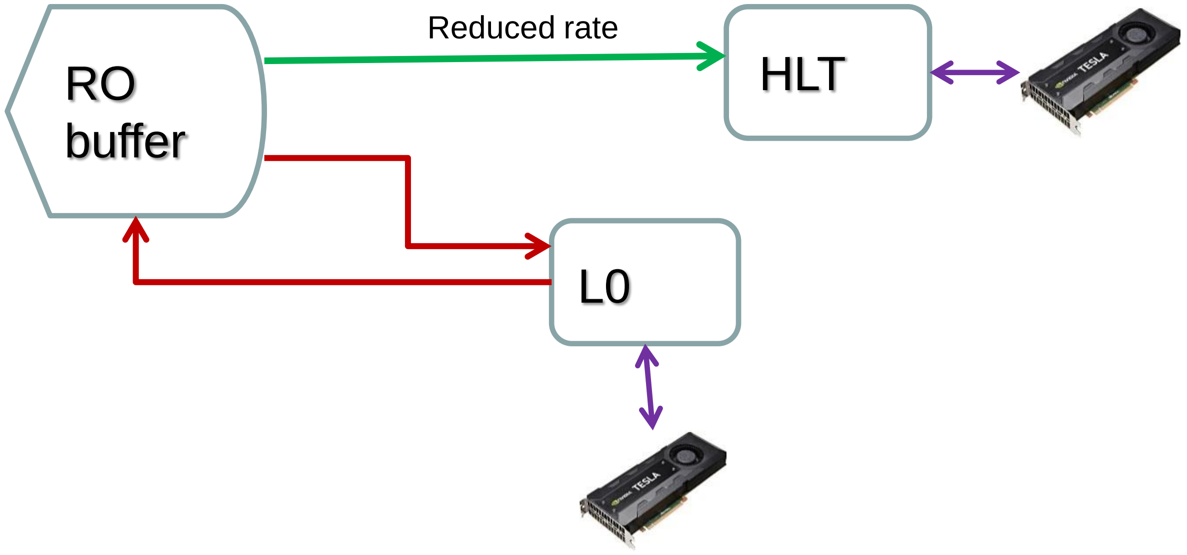


L0: Hardware synchronous level. 10 MHz to 1 MHz. Max latency 1 ms.

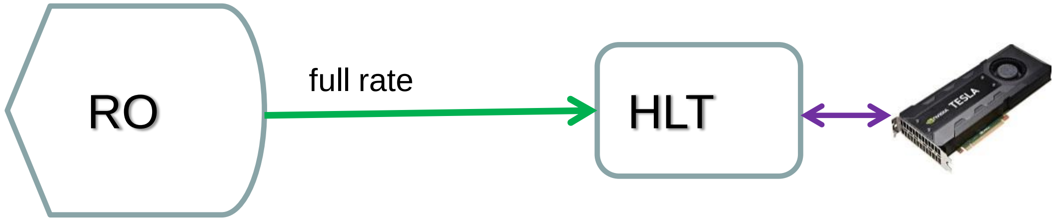
L1: Software level. "Single detector". 1 MHz to 100 kHz

L2: Software level. "Complete information level". 100 kHz to few kHz.

GPU: where?

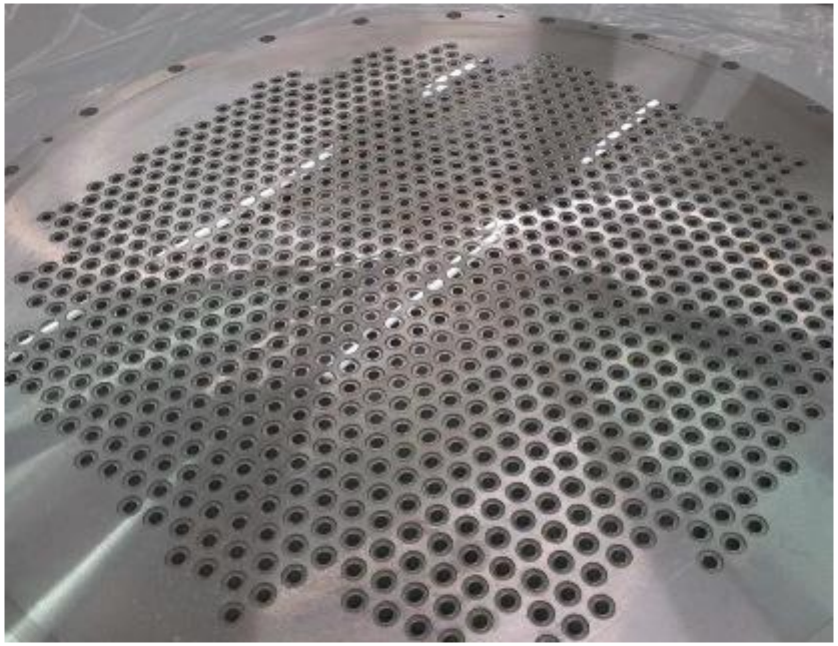


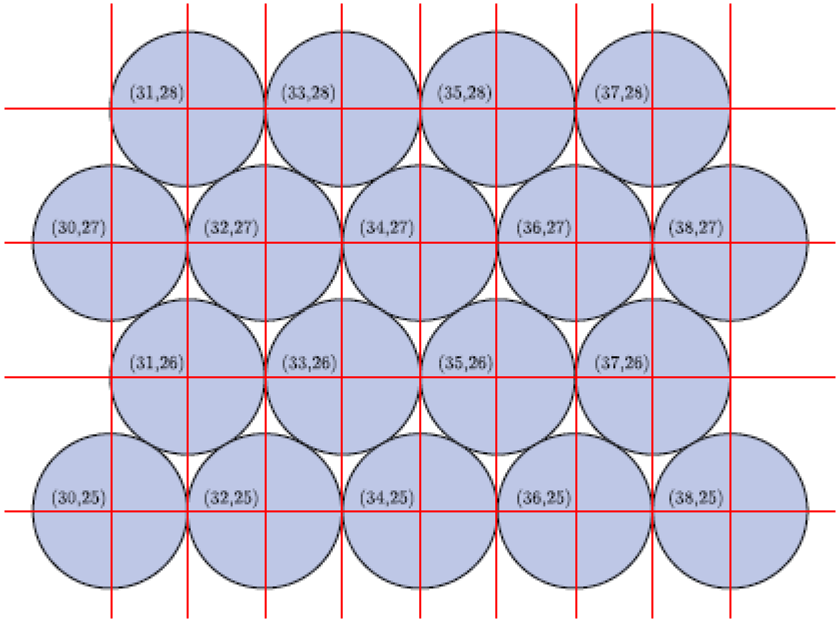
«classical trigger»



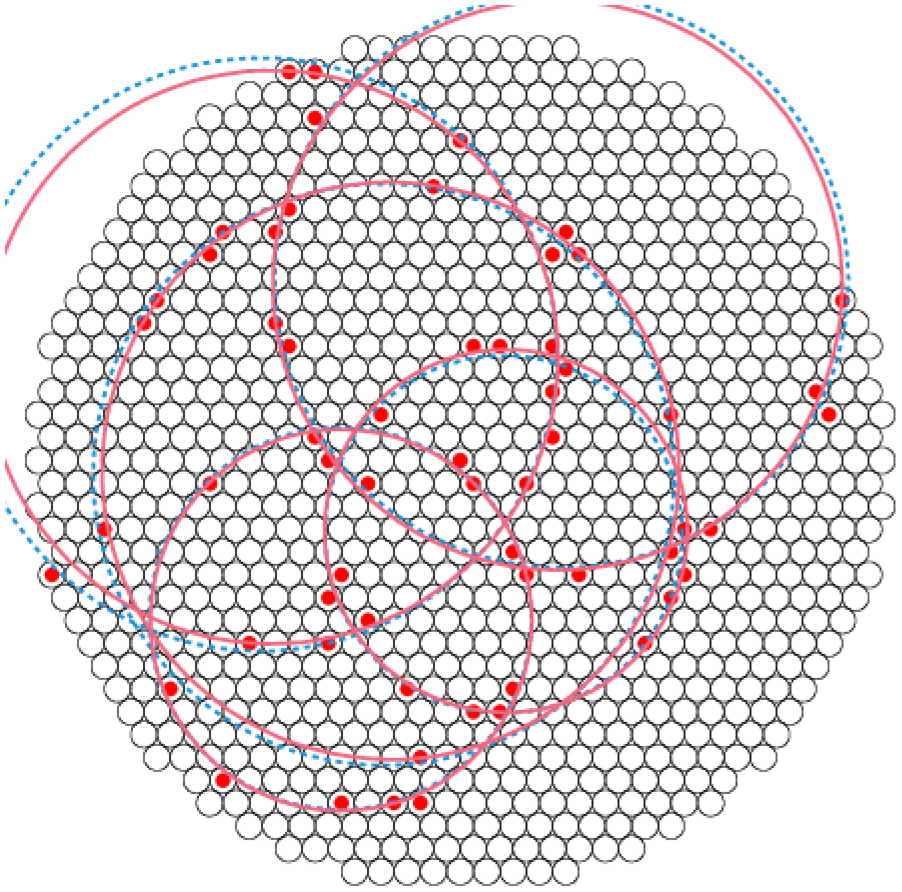
«triggerless»

Flange geometry

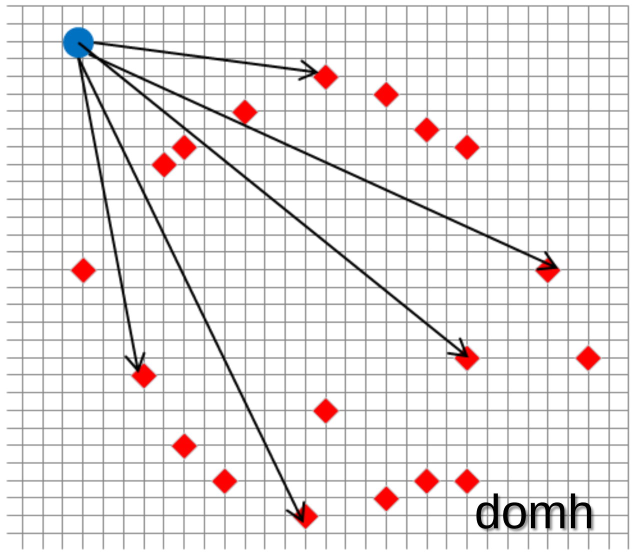


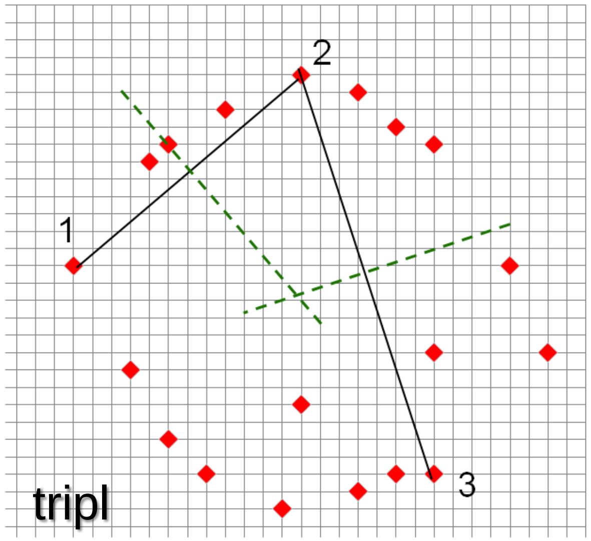


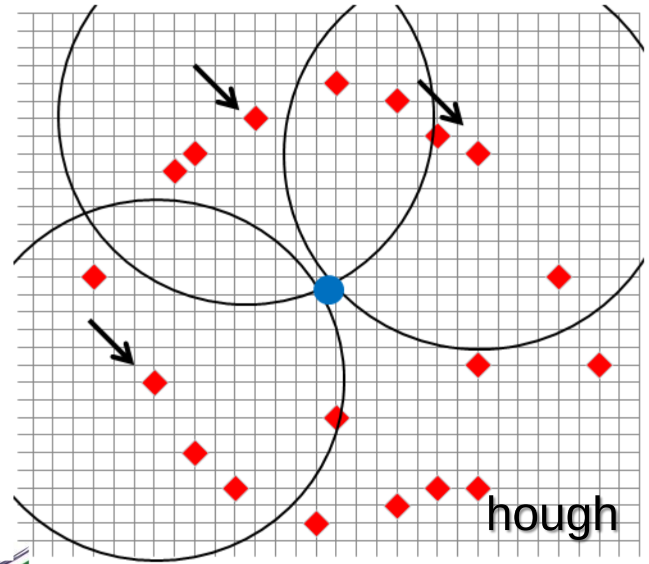
A more “complicated” example

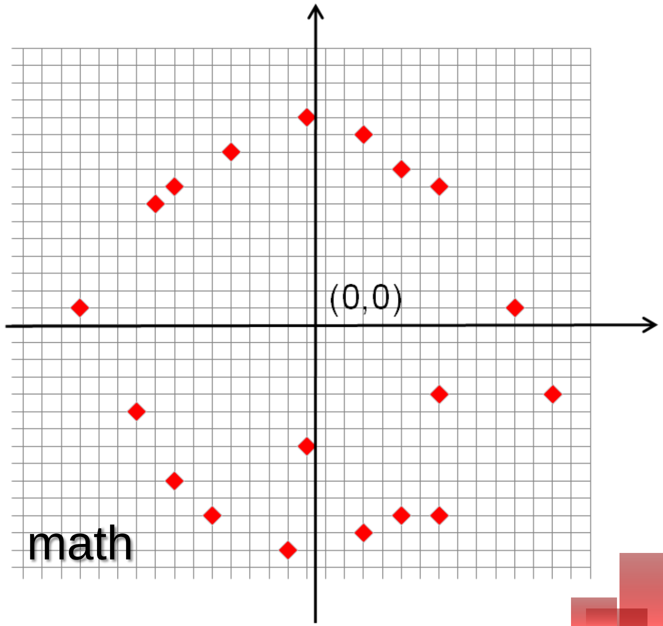


Single ring



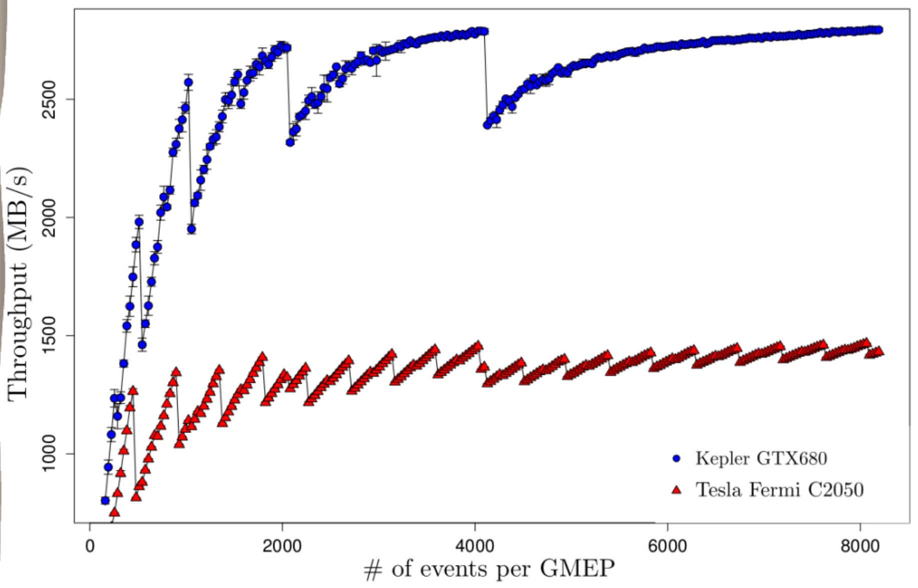


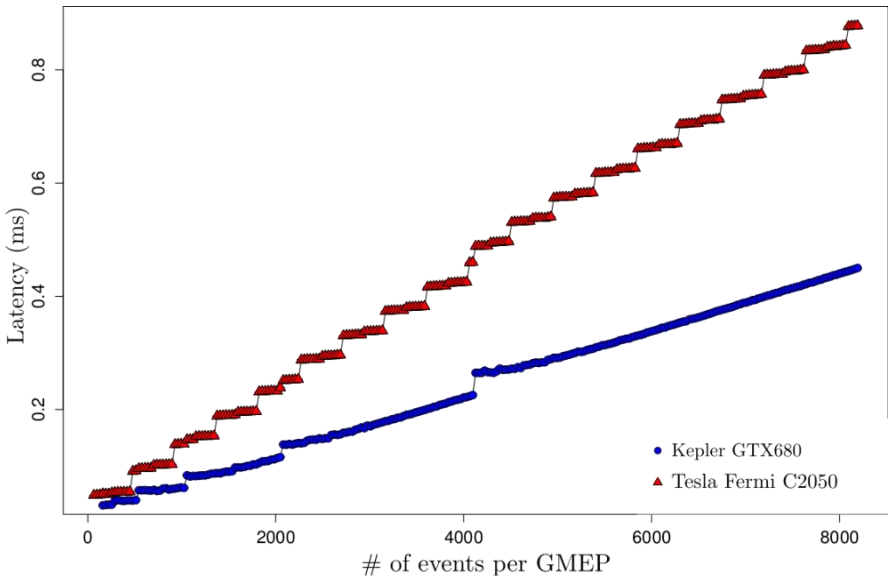




GAP RT

Single Ring Fit – Throughput & Latency



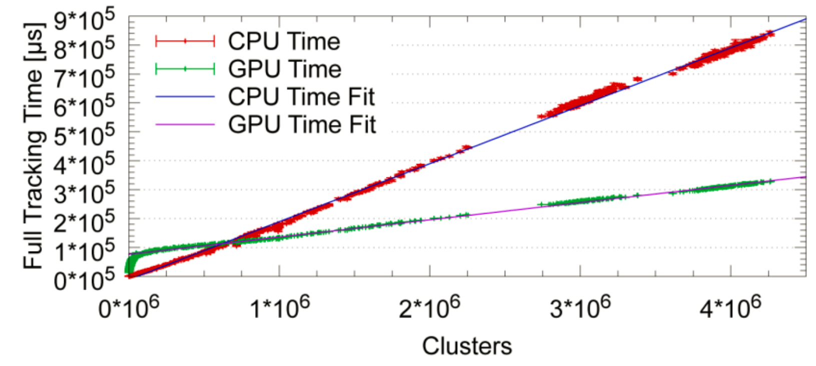


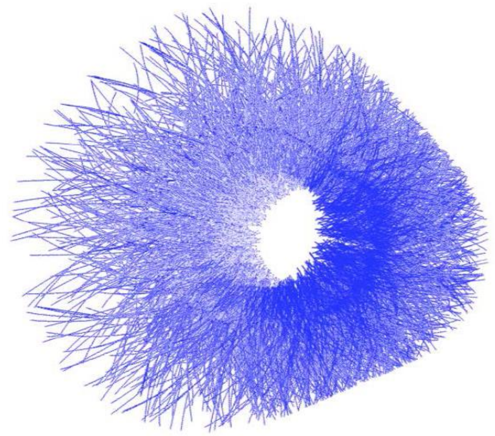
"Fast online triggering in high-energy physics experiments using GPUs" Nucl.Instrum.Meth.A662:49-54,2012

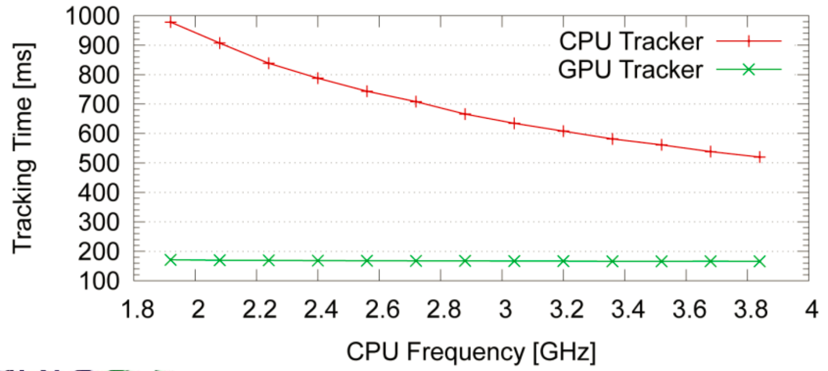
- discrete oscillations due to the discrete nature of the GPU
- saturation plateau (1.4 GB/s and 2.7 GB/s)
- A lower number of events inside the buffer is better to achieve a low latency
- A larger number of events guarantees a better performance and a lower overhead
- The choice of the buffer size is a compromise

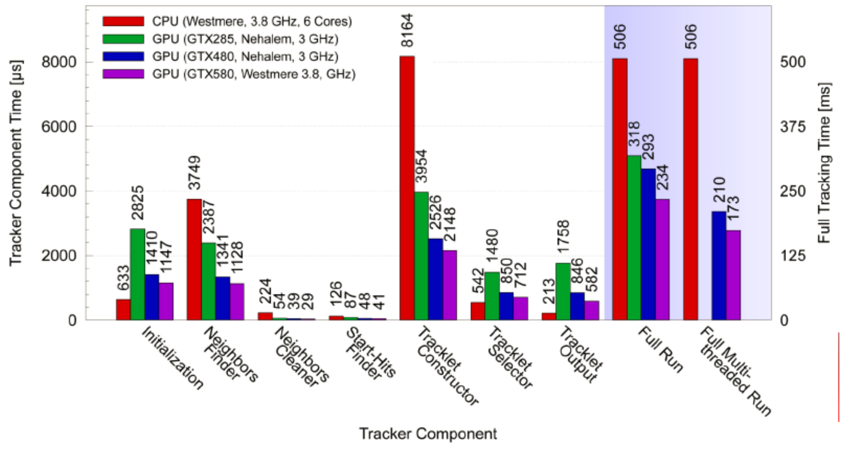
Alice: HLT TPC online Tracking

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



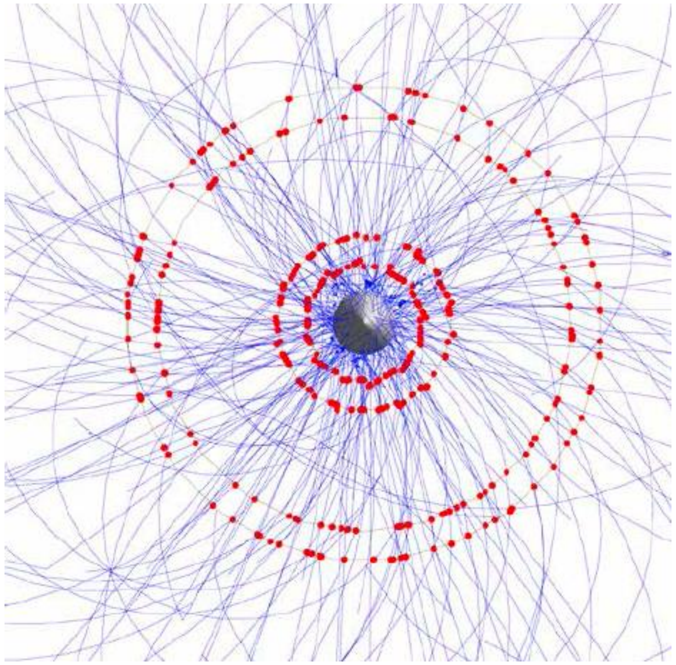


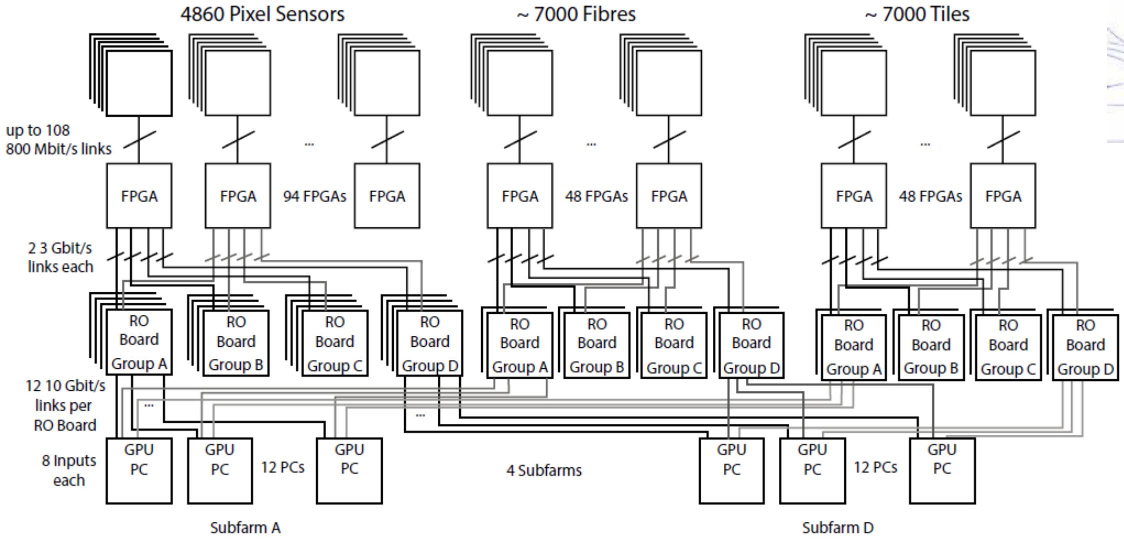


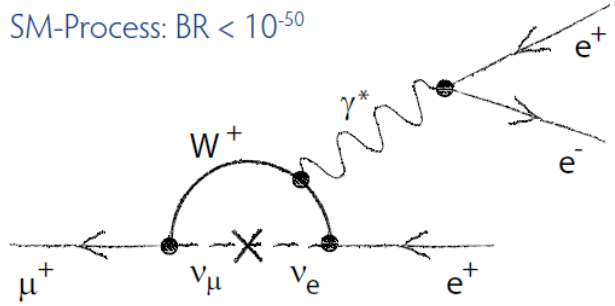


Mu3e

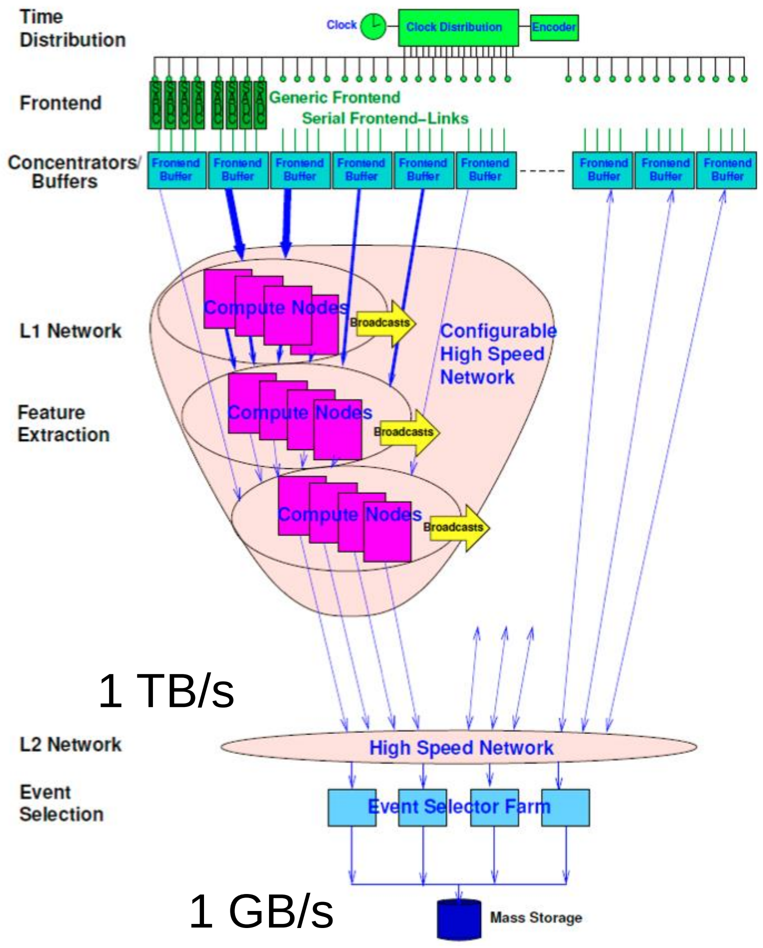
- Possibly a "trigger-less" approach
- High rate: 2×10^9 tracks/s
- >100 GB/s data rate
- Data taking will start >2016







PANDA



- 10^7 events/s
- Full reconstruction for online selection: assuming 1-10 ms → 10000 – 100000 CPU cores
- Tracking, EMC, PID,…
- First exercice: 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