GAP RT

INFN
Istituto Nazionale
di Fisica Nucleare

Università degli Studi di Ferrara
1391
Ex labore fructus

GPUs for Online Processing in Low-Level Trigger Systems

Massimiliano Fiorini
(Università di Ferrara and INFN Ferrara)

On behalf of the GAP Collaboration

3rd International Conference on
Technology and Instrumentation in Particle Physics
Amsterdam, 2-6 June 2014

GAP Project

- **GAP** (**G**PU **A**pplication **P**roject) for real-time in HEP and medical imaging is a 3 years project funded by the Italian Ministry of research, started in 2013
- Collaboration between INFN Sezione di Pisa, University of Ferrara and University of Roma "La Sapienza"
- Demonstrate the feasibility of using off-the-shelf computer commodities to accelerate real-time scientific computations
- Application in different fields:
  - High Energy Physics (low and high level triggers)
  - Medical Imaging (NMR, CT and PET)

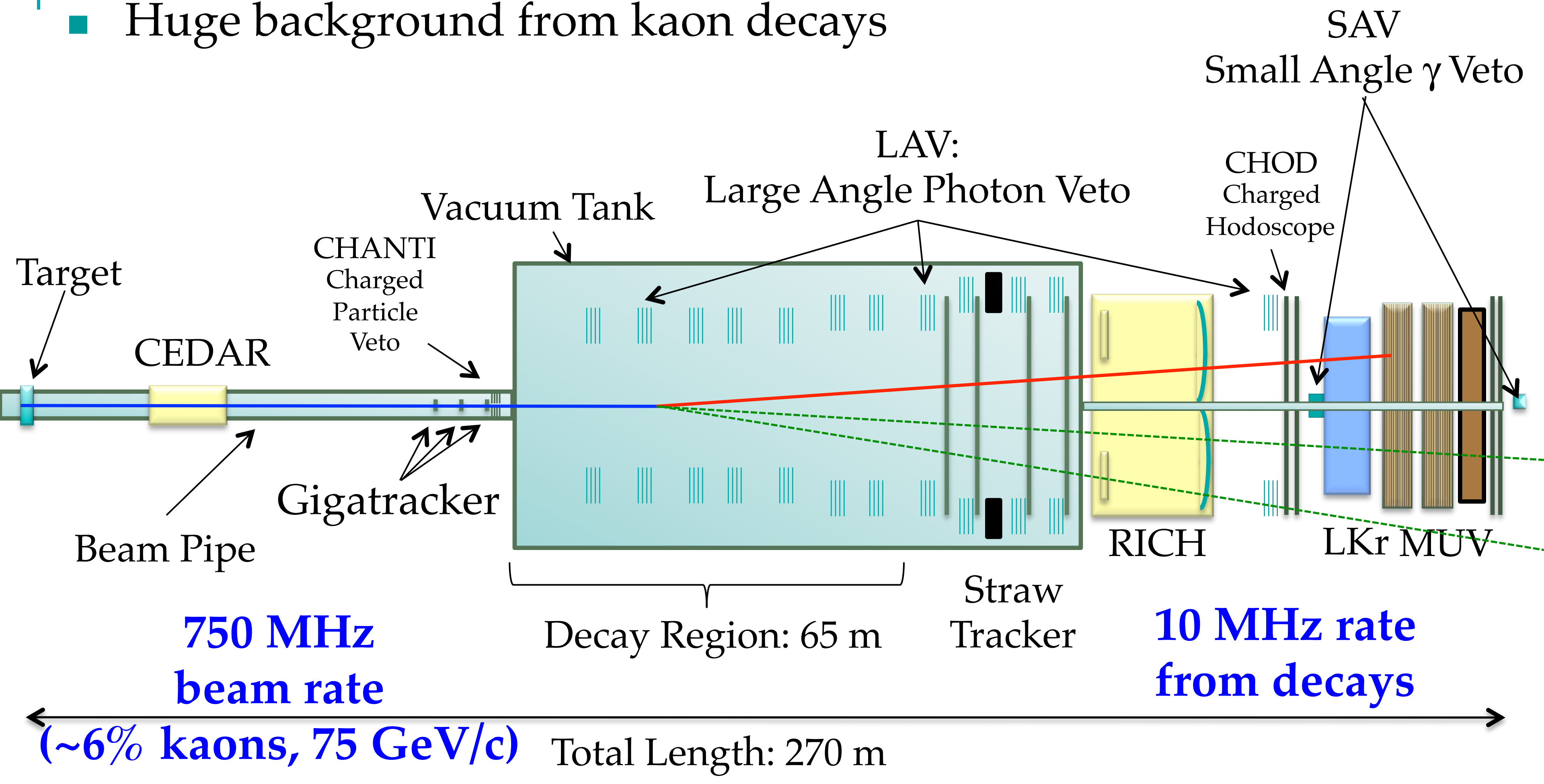
GAP Project

- **GAP** (**G**PU **A**pplication **P**roject) for real-time in HEP and medical imaging is a 3 years project funded by the Italian Ministry of research, started in 2013
- Collaboration between INFN Sezione di Pisa, University of Ferrara and University of Roma "La Sapienza"
- Demonstrate the feasibility of usin commodities to acce ns
- Appli
  - ysics (low and high level triggers)
  - Medical Imaging (NMR, CT and PET)

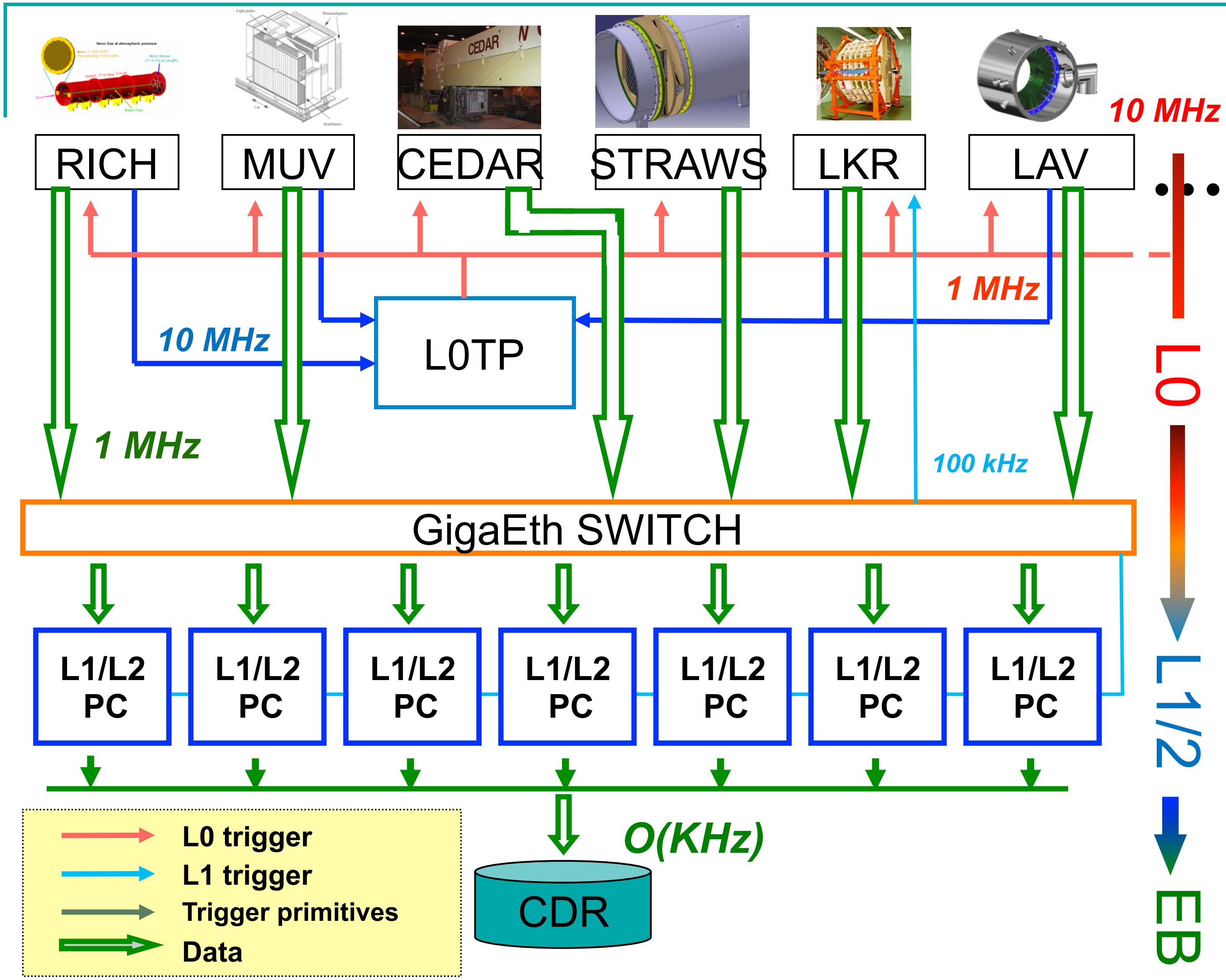
Poster: M. Fiorini "The GAP Project - GPU for Realtime Applications in High Level Trigger and Medical Imaging"

Physics case: NA62

- $K^+ \rightarrow \pi^+ \upsilon\upsilon$ decay ($BR \sim 8 \times 10^{-11}$)
- Huge background from kaon decays



Trigger and DAQ



RICH

MUV

CEDAR

STRAWS

LKR

LAV

10 MHz

1 MHz

10 MHz

L0TP

1 MHz

100 kHz

GigaEth SWITCH

L1/L2 PC

L1/L2 PC

L1/L2 PC

L1/L2 PC

L1/L2 PC

L1/L2 PC

L1/L2 PC

O(KHz)

CDR

L0

L1/2

EB

L0 trigger

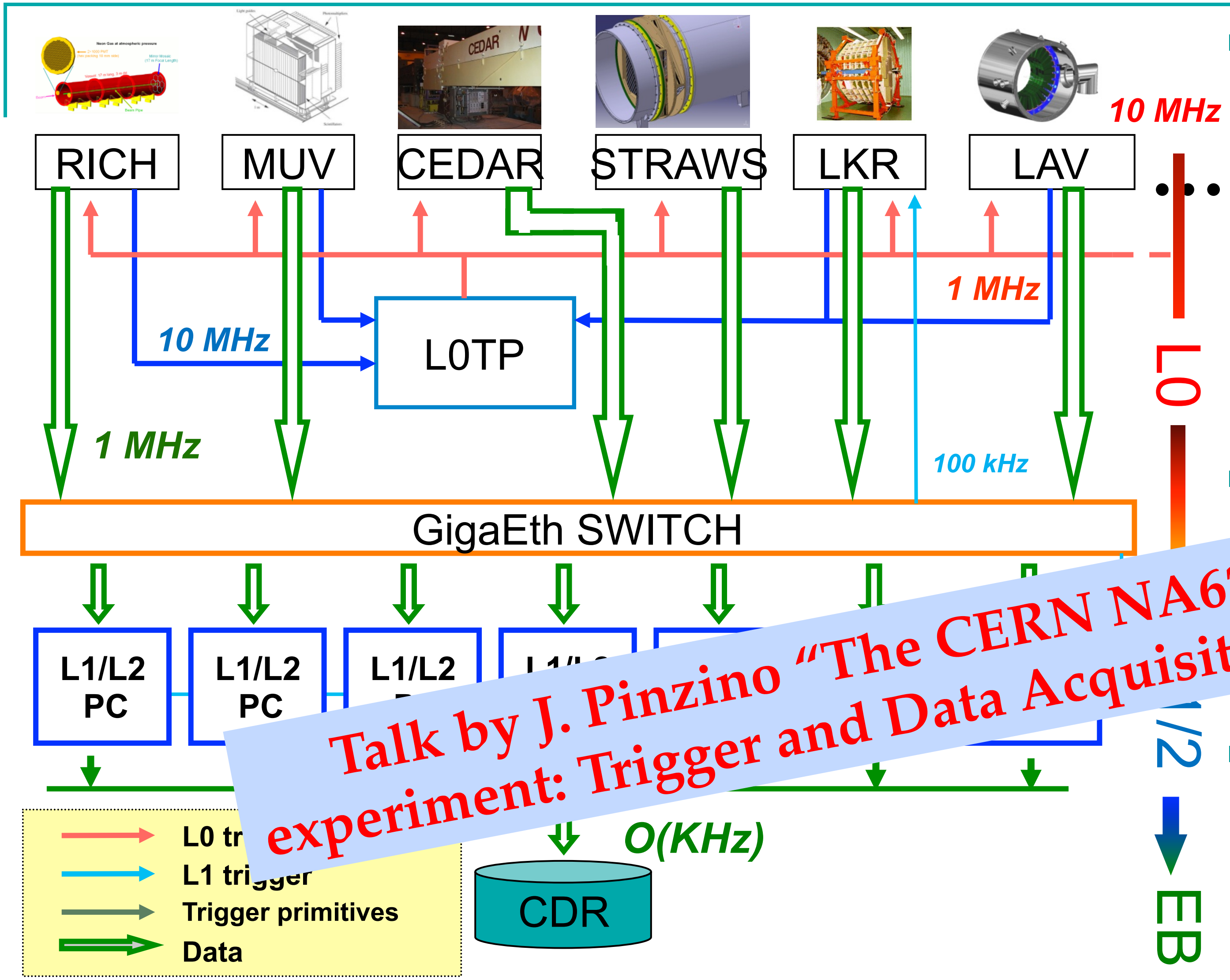
L1 trigger

Trigger primitives

Data

- **L0**: Hardware synchronous level
  - 10 MHz to 1 MHz, **1 ms max. latency**
  - Primitives (MUV, RICH, LAV, LKR)
- **L1**: Software level
  - “Single detector”, 1 MHz to 100 kHz
- **L2**: Software level
  - “Complete information”, 100 kHz to 10 kHz

Trigger and DAQ

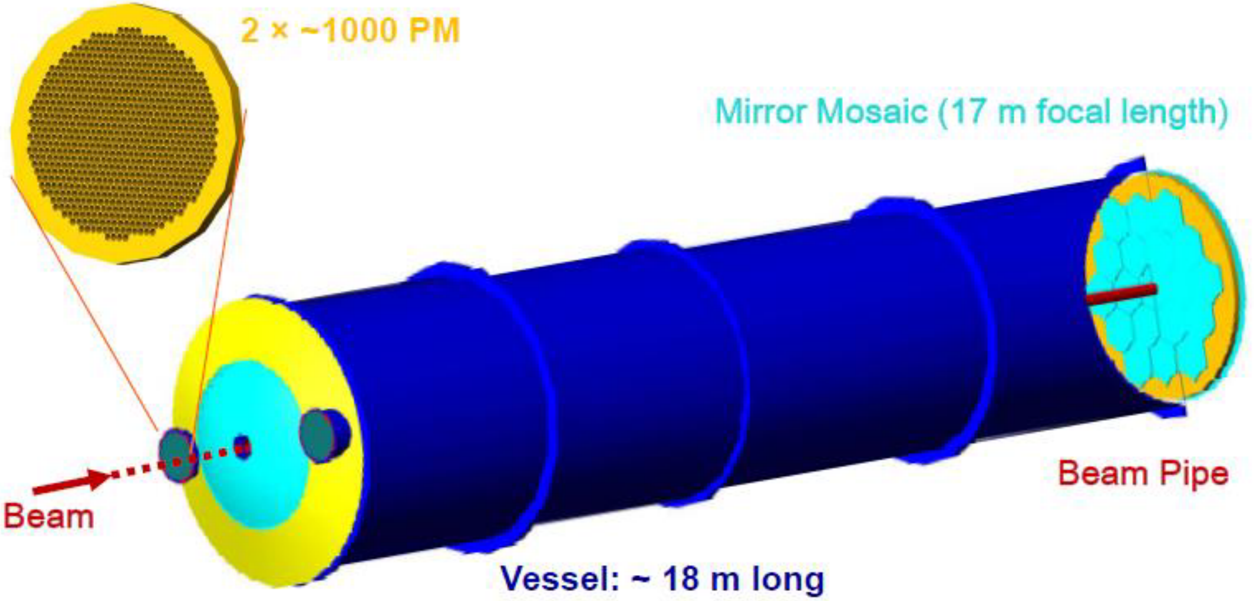


Talk by J. Pinzino "The CERN NA62 experiment: Trigger and Data Acquisition"

- **L0**: Hardware synchronous level
  - 10 MHz to 1 MHz, **1 ms max. latency**
  - Primitives (MUV, RICH, LAV, LKR)
- **L1**: Software level
  - "Single detector", 1 MHz to 100 kHz
- **L2**: Software level
  - "Complete information", 100 kHz to 10 kHz

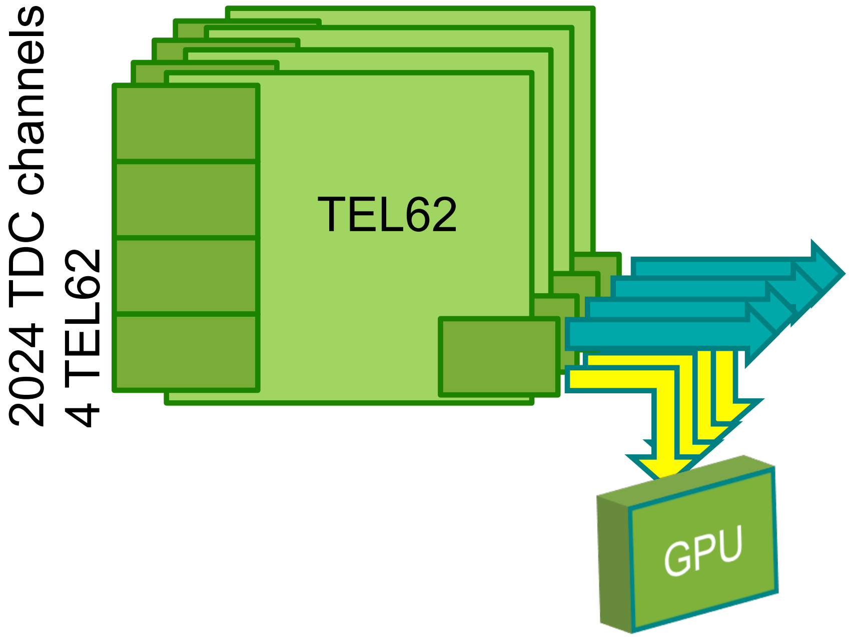
The RICH detector

- 17 m focal length, 3 m in diameter, filled with Ne at 1 atm
- Pion/muon separation in the range 15-35 GeV/c
- Time resolution ~70 ps
- Mis-identification 5×10-^{3}
- 2 spots of 1000 PMTs each



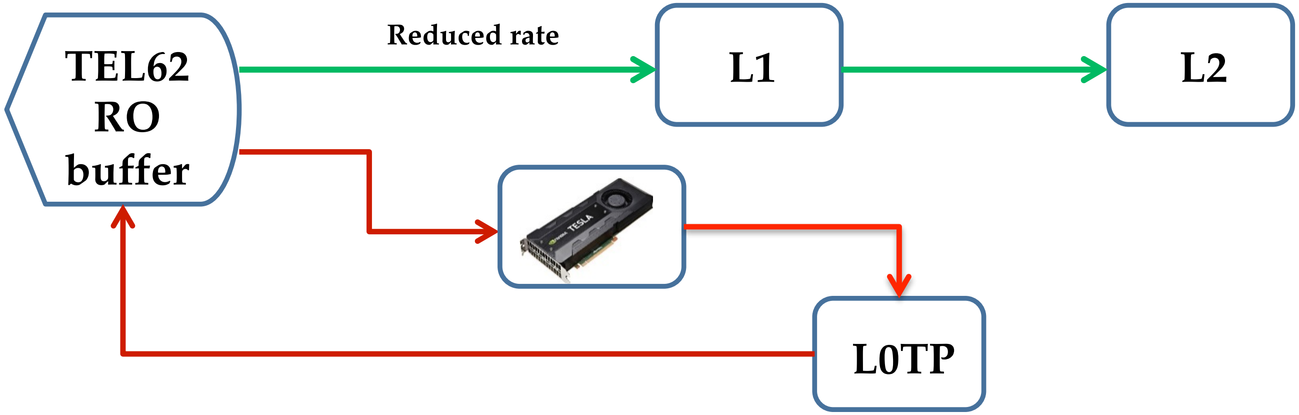
- 10 MHz events rate in the RICH (~20 hits/track)
  - Main contribution from kaon decays (~1 MHz from halo muons and pion decays)

The goal: GPU in L0 RICH



- 4 TEL62 for RICH detector
  - ❑ 8×1Gb/s links for data r/o
  - ❑ 4×1Gb/s trigger primitives
  - ❑ 4×1Gb/s GPU trigger
- Events rate: 10 MHz
- L0 trigger rate: 1 MHz
- Max Latency: 1 ms

- This is not the L0 trigger baseline version for the NA62 RICH detector



GPUs in Low Level Triggers

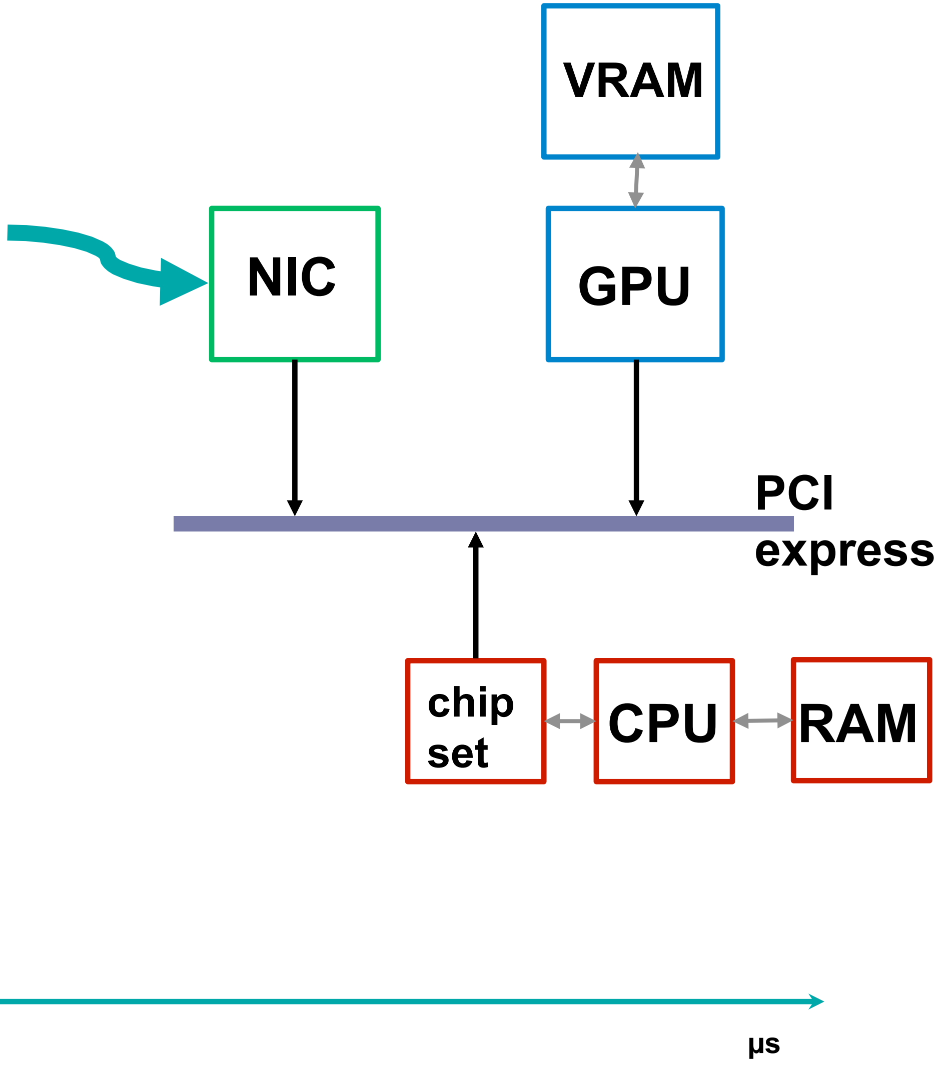
- Two main issues to be solved:

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

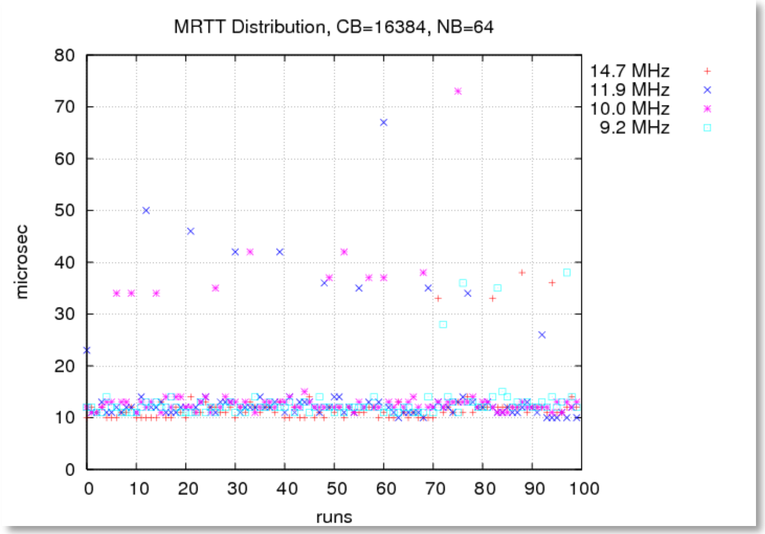
- **Computing power**
  - Is the GPU fast enough to take a trigger decision at tens of MHz events rate?

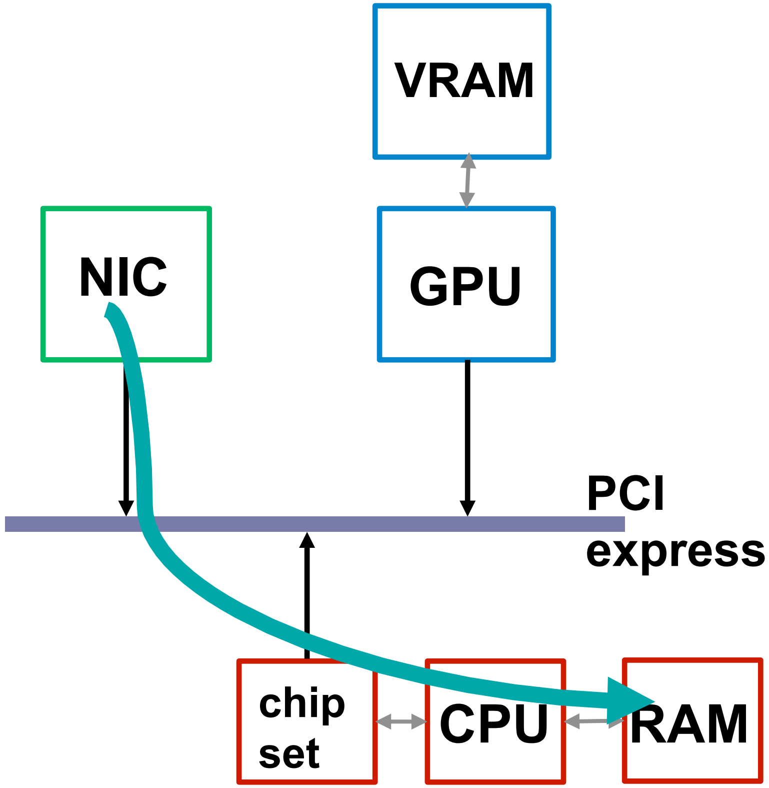
GPU Processing

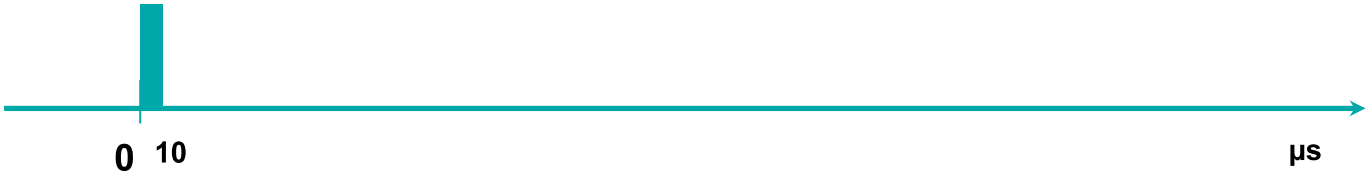
- Example: packet with 1404 B (few tens of events in NA62 RICH application)
- T=0



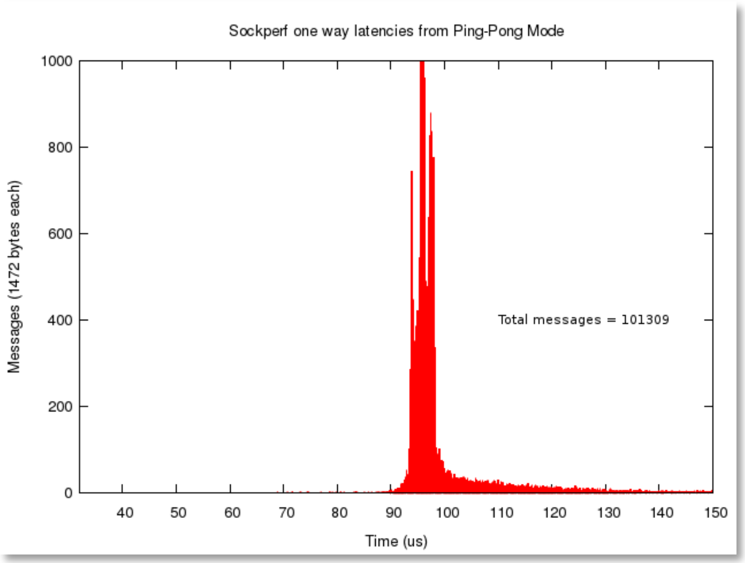
GPU Processing

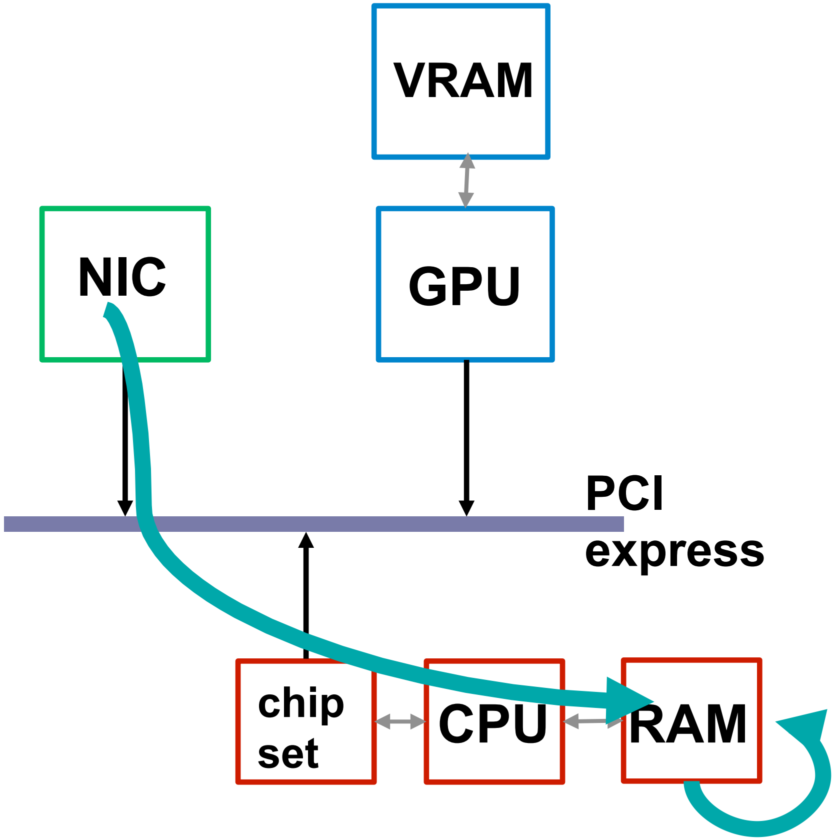


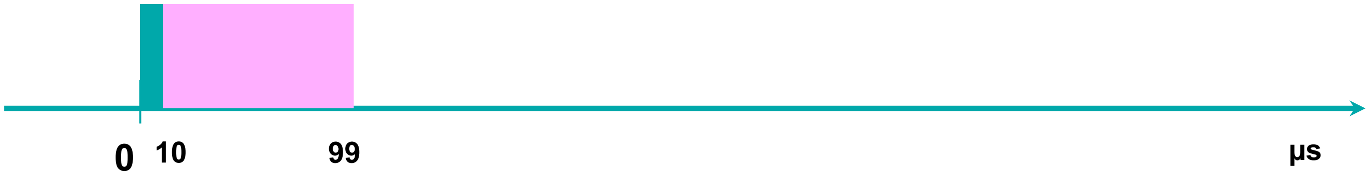




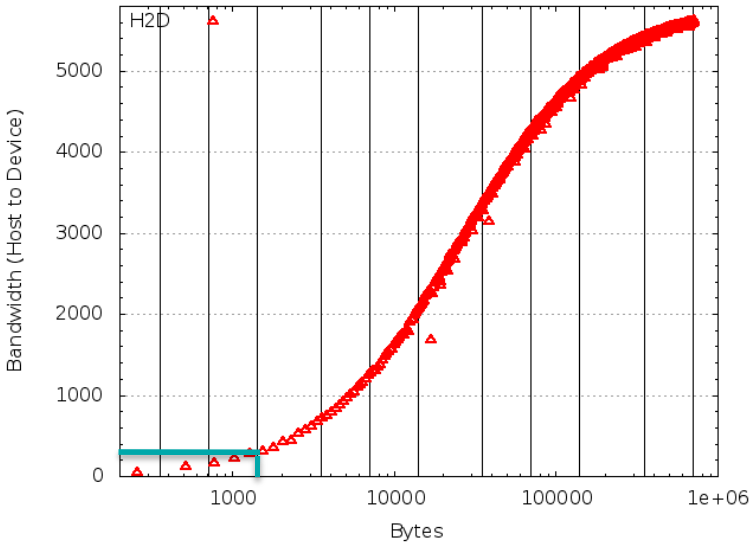
GPU Processing

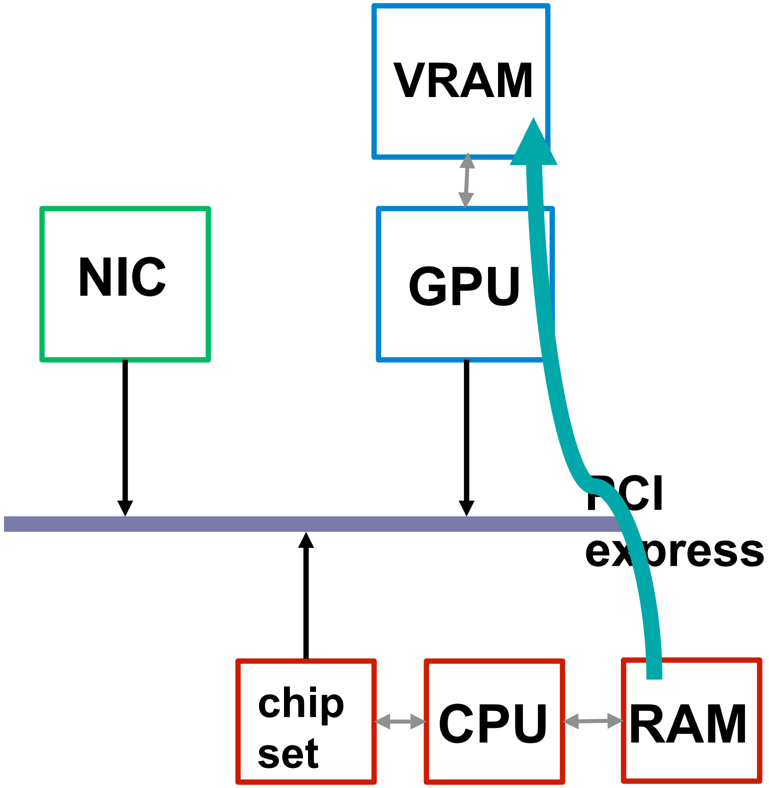


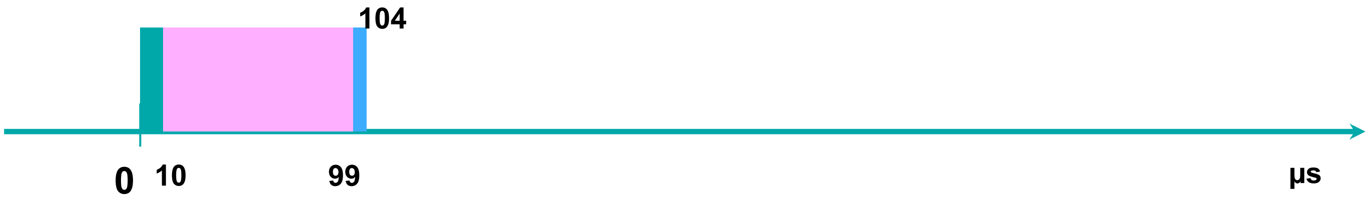




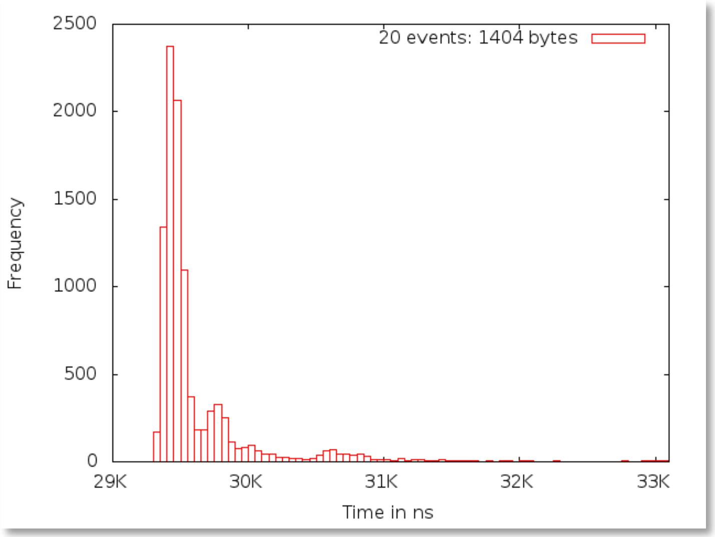
GPU Processing

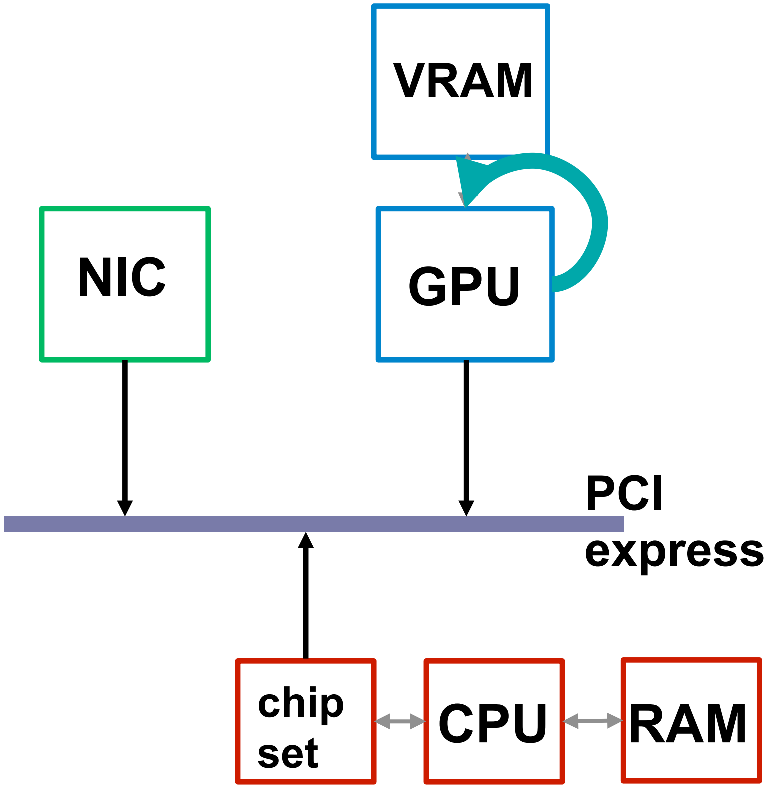


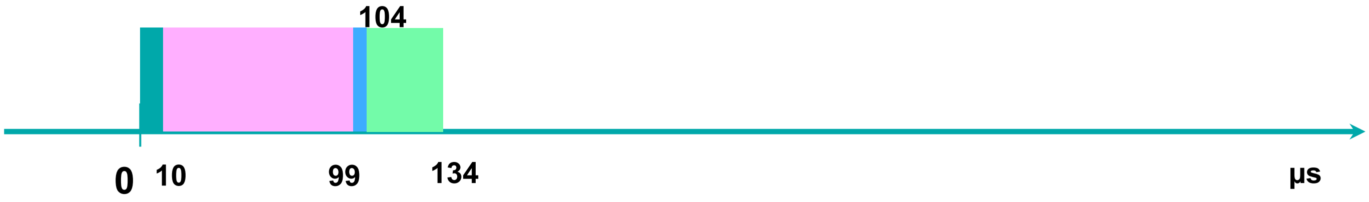




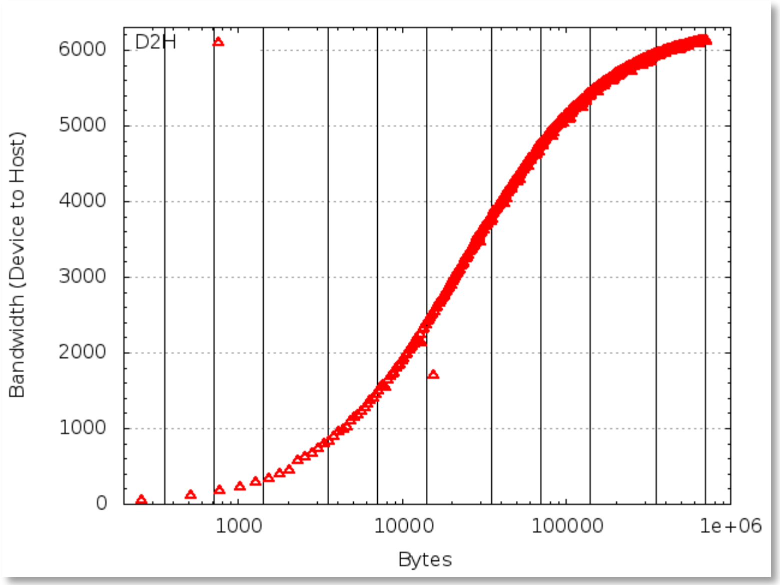
GPU Processing

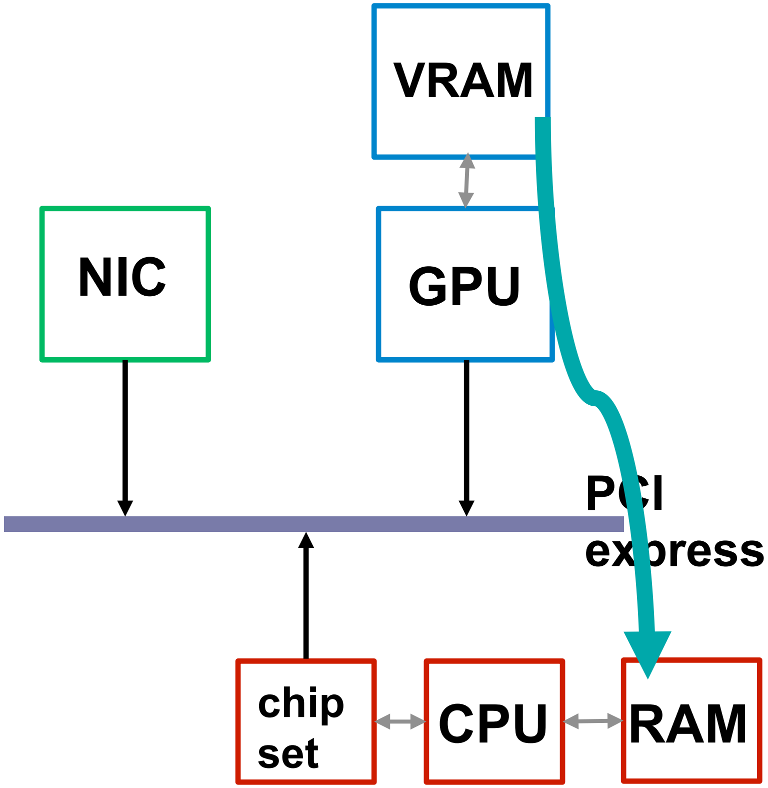


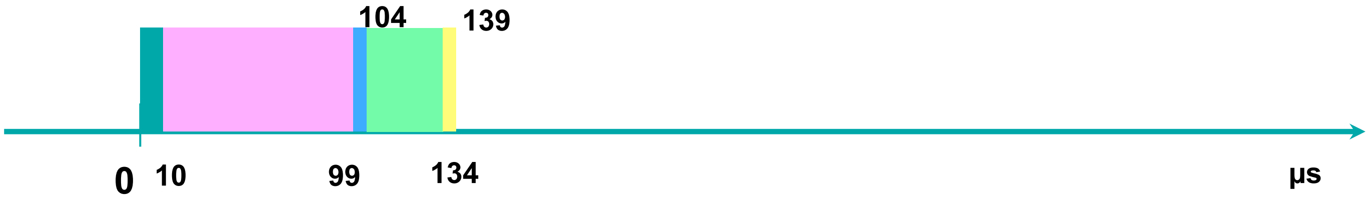




GPU Processing

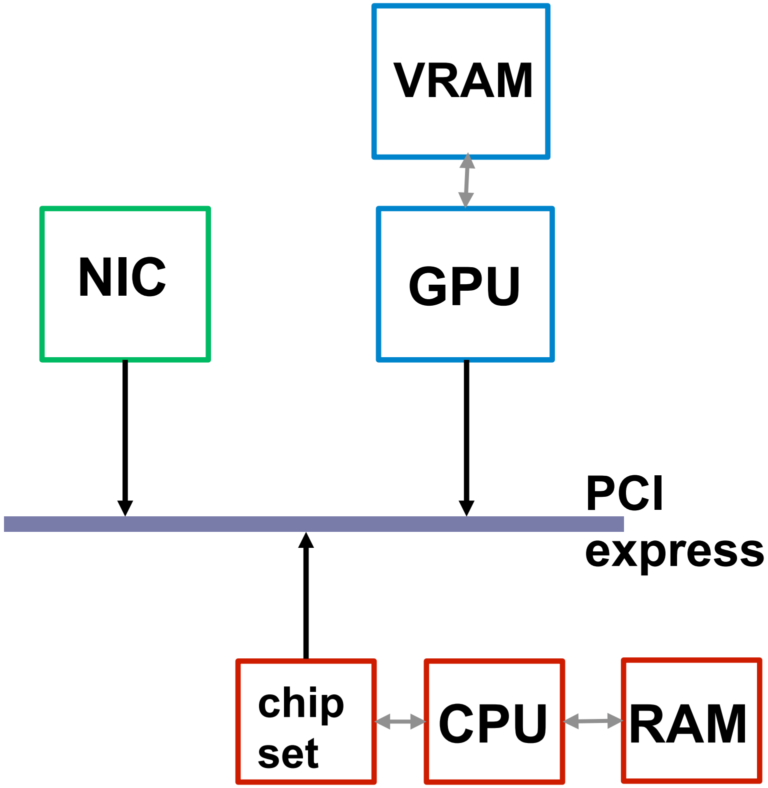


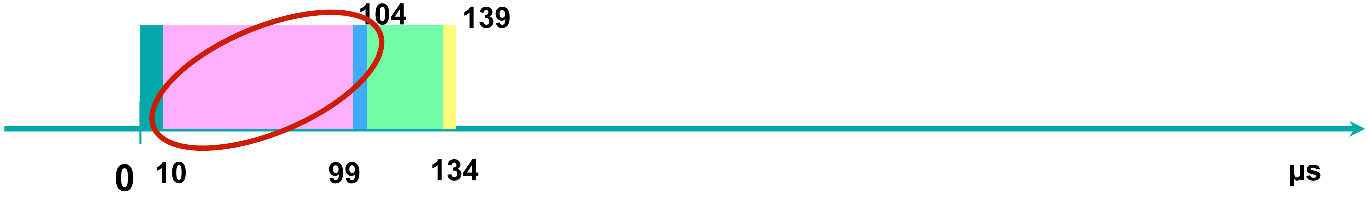




GPU Processing

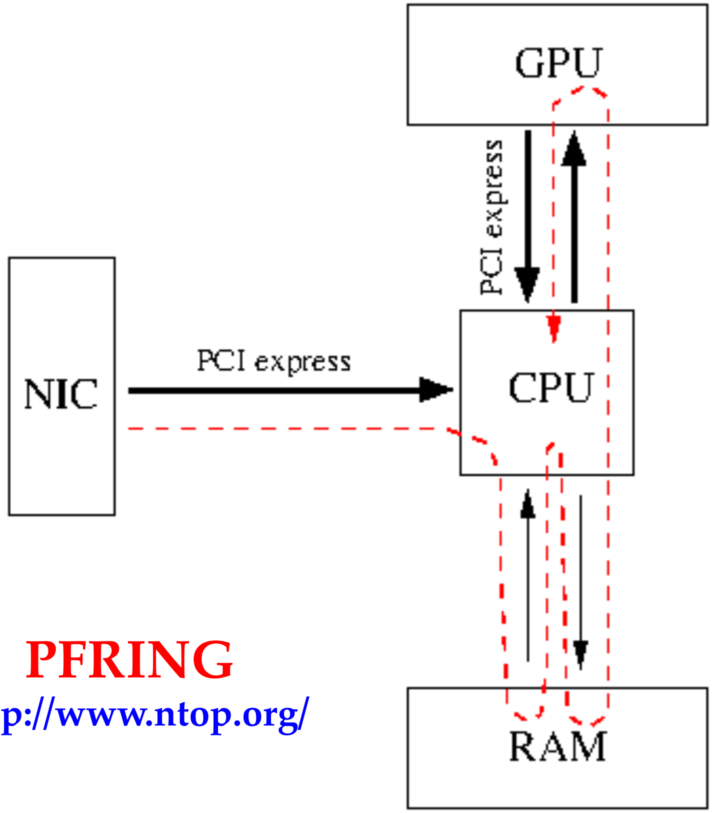
- Latency due to data transfer from the detector to the system is bigger than the latency due to GPU computing
- It scales almost linearly (apart from the overheads) with the data size while the latency due to computing can be hidden exploiting the huge resources
- Communication latency fluctuations quite big (~50%)



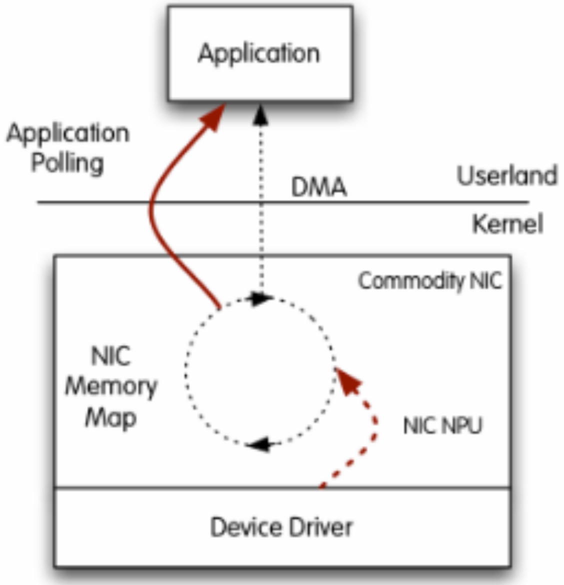


First solution: PFRING

- 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



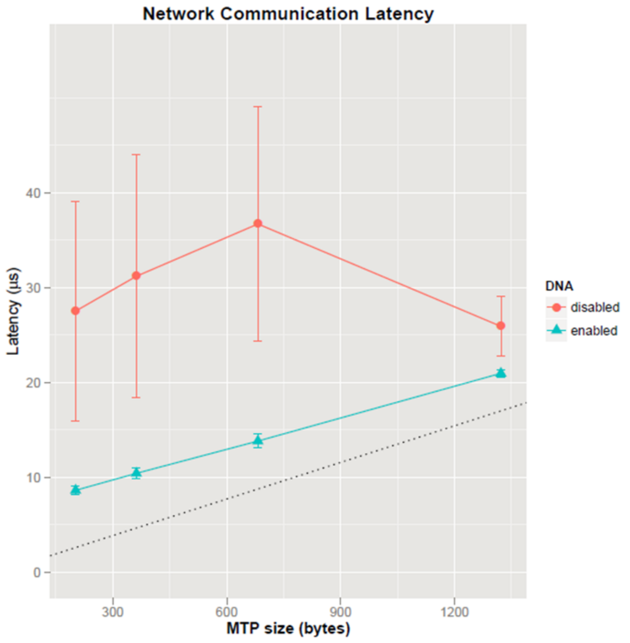
PFRING
http://www.ntop.org/

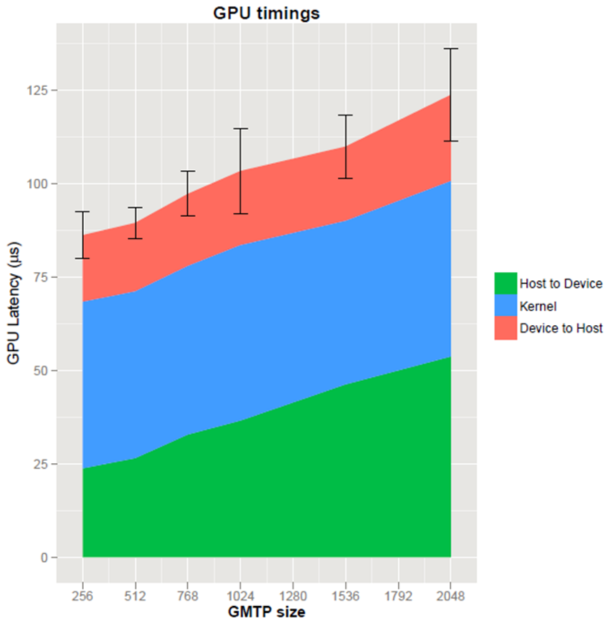


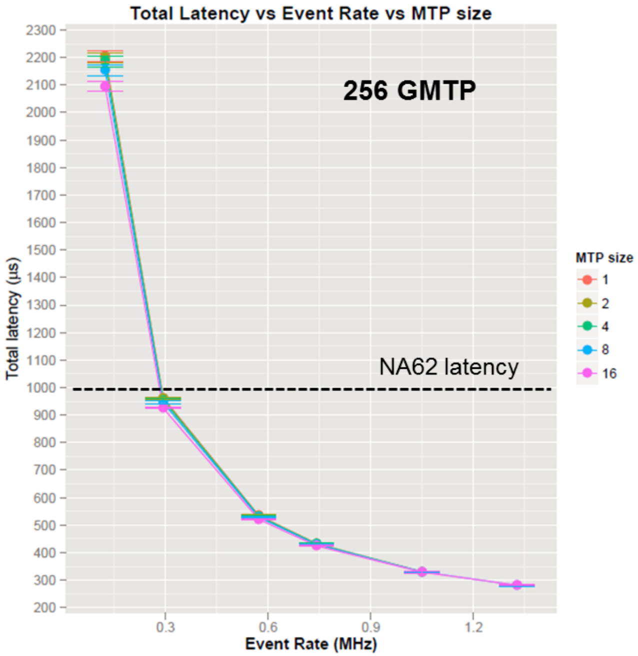
PF_RING with DNA
(Direct NIC Access) driver

Results: PFRING

- Latency reduced by a factor 3 and negligible fluctuations
- The total latency is given as a function of the number of events to buffer before the start of GPU computation
- For real application the "working point" depends on the events rate and event dimension

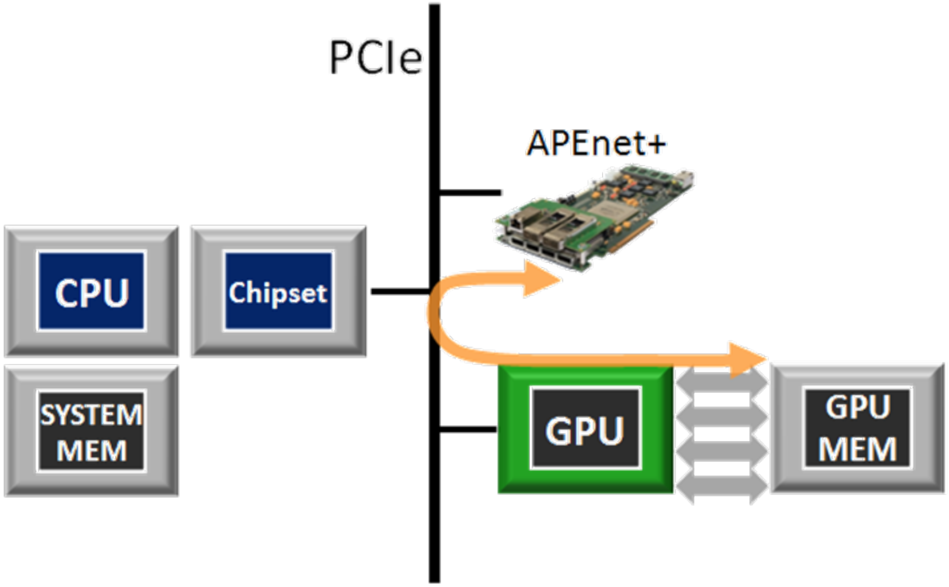


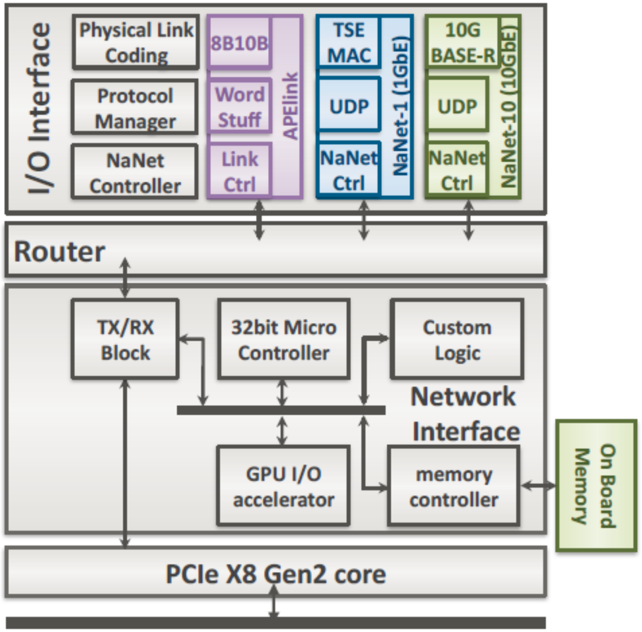


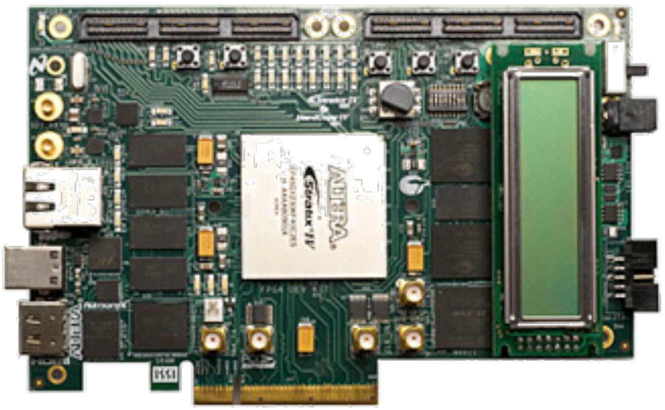


Second solution: NANET

- NANET is an FPGA-based NIC that has GPUDirect RDMA capabilities





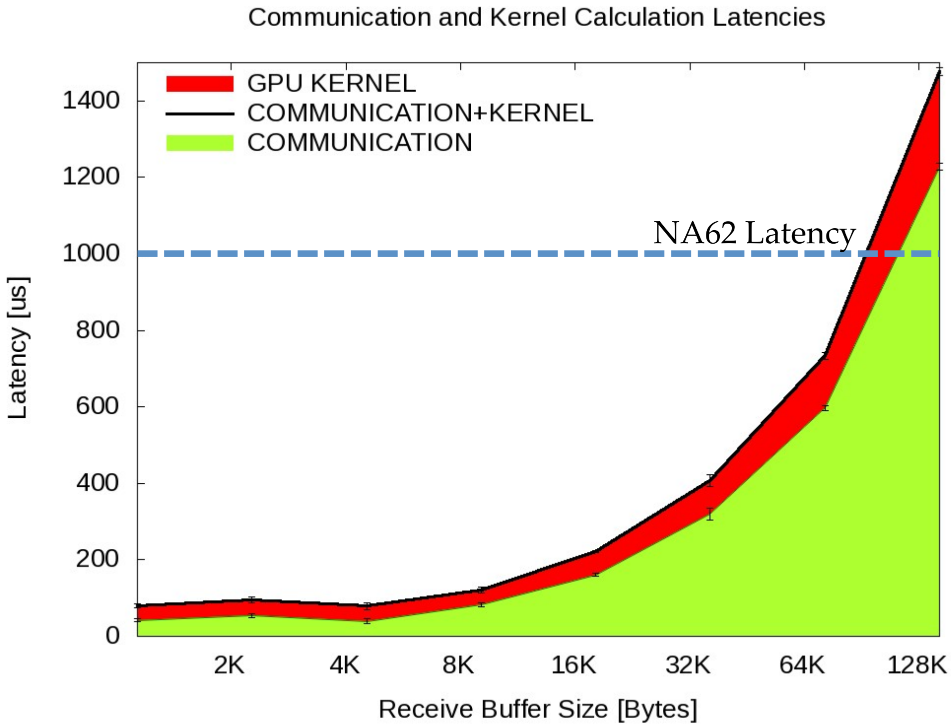


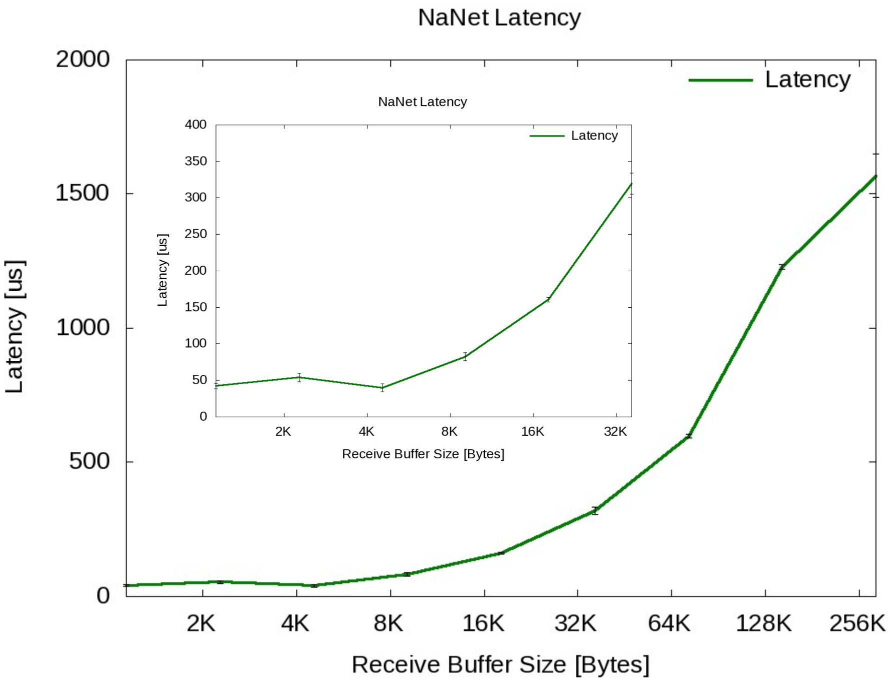
APEnet Rome Group

R. Ammendola *et al.*, *JINST* 9 C02023, 2014

Results: NANET

- Variable size buffers sent to the GPU memory
  - Data sent from GbE NIC from the same PC hosting NANET



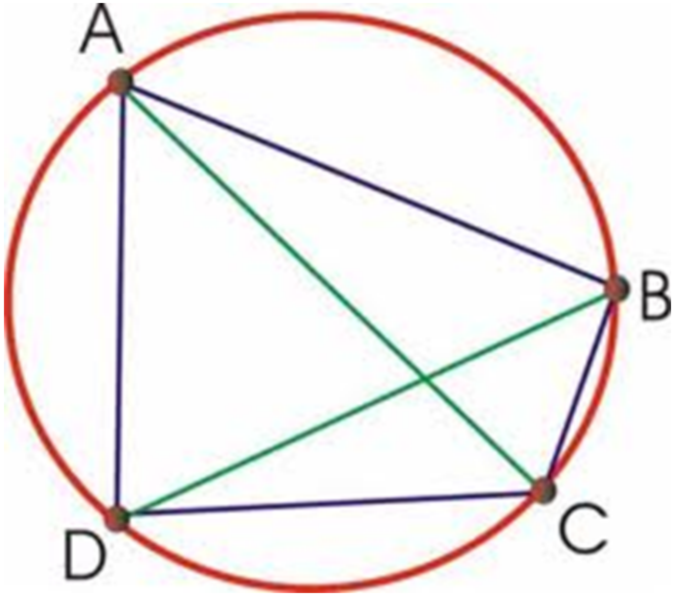


- GPU kernel for pattern matching is then executed
  - Maximum buffer size limited to <90 KBytes

L0 RICH trigger algorithm

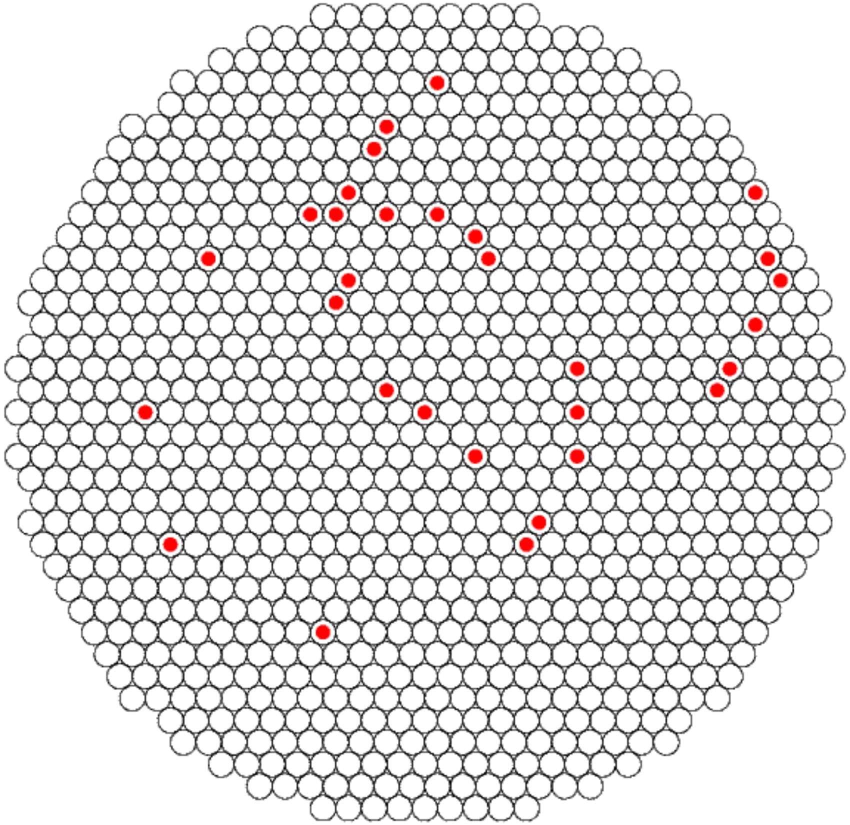
- Requirements for an on-line RICH reconstruction algorithm:
- Trackless
  - No information from the tracker
  - Difficult to merge information from many detectors at L0
- Multi-rings
  - Many-body decay in the RICH acceptance
- Fast
  - Non-iterative procedure
  - Events rate at a level of ~10 MHz
- Low latency
  - Online (synchronous) trigger
- Accurate

Almagest



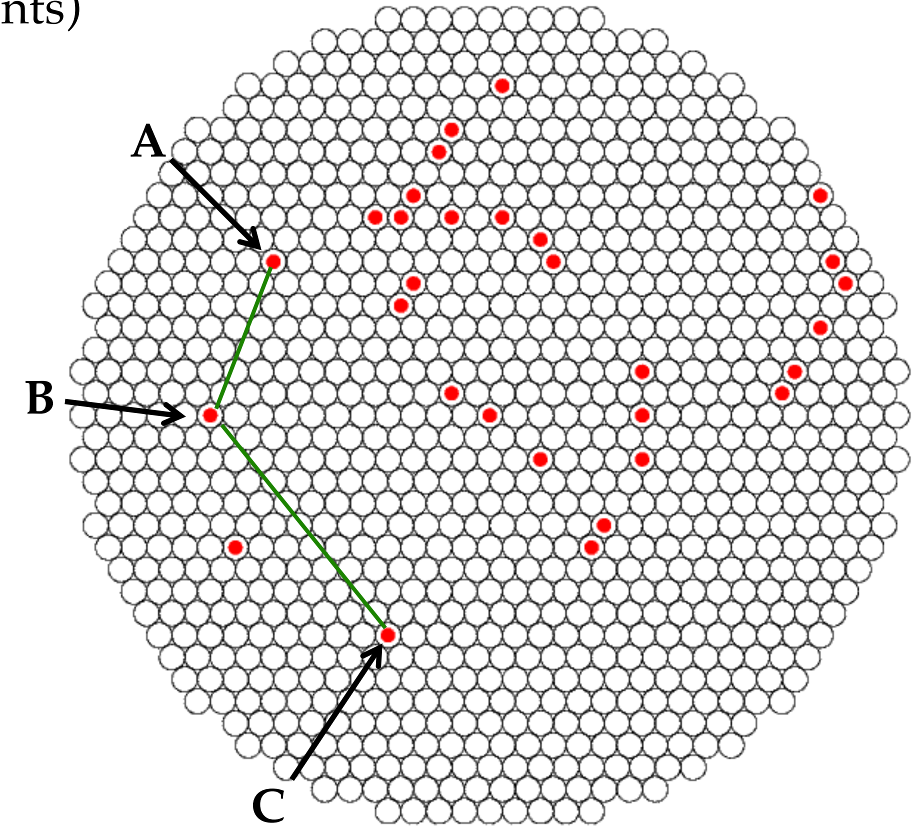
- New algorithm (Almagest) based on Ptolemy’s theorem: *“A quadrilater is cyclic (the vertex lie on a circle) if and only if is valid the relation: AD*BC+AB*DC=AC*BD “*
- Design a procedure for parallel implementation

Almagest: example



Almagest: example

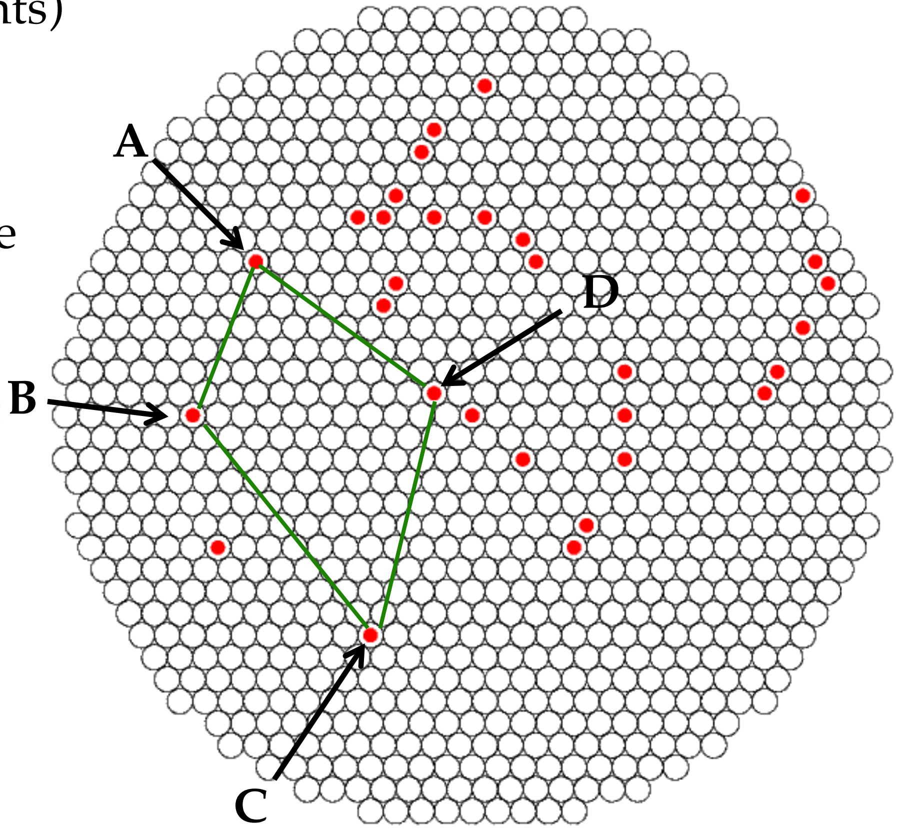
i) Select a *triplet* (3 starting points)



Almagest: example

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

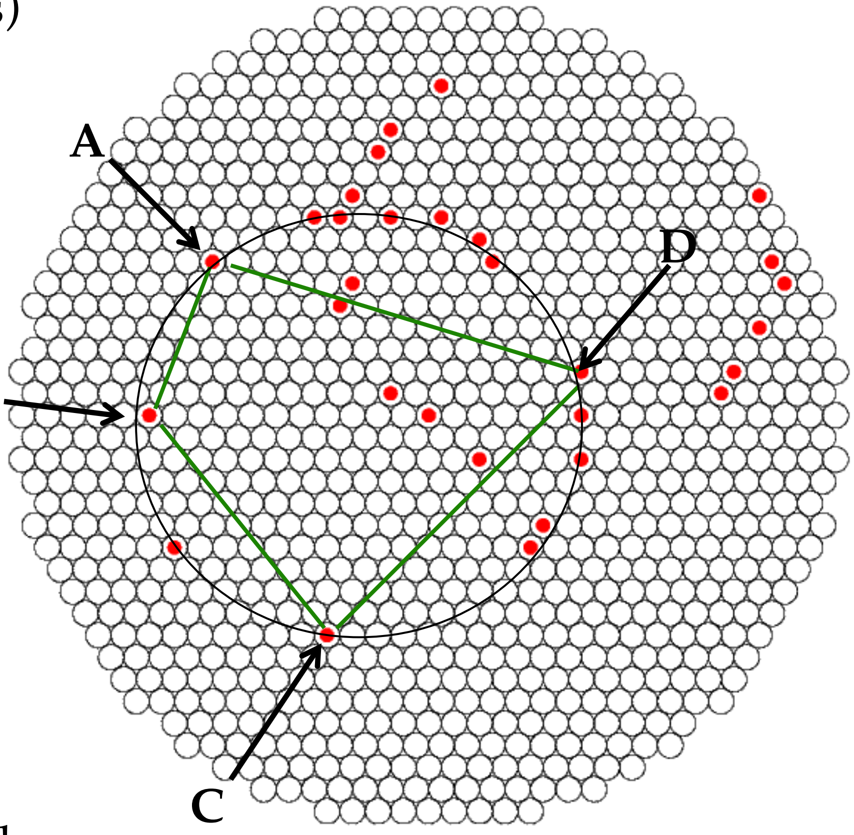


Almagest: example

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

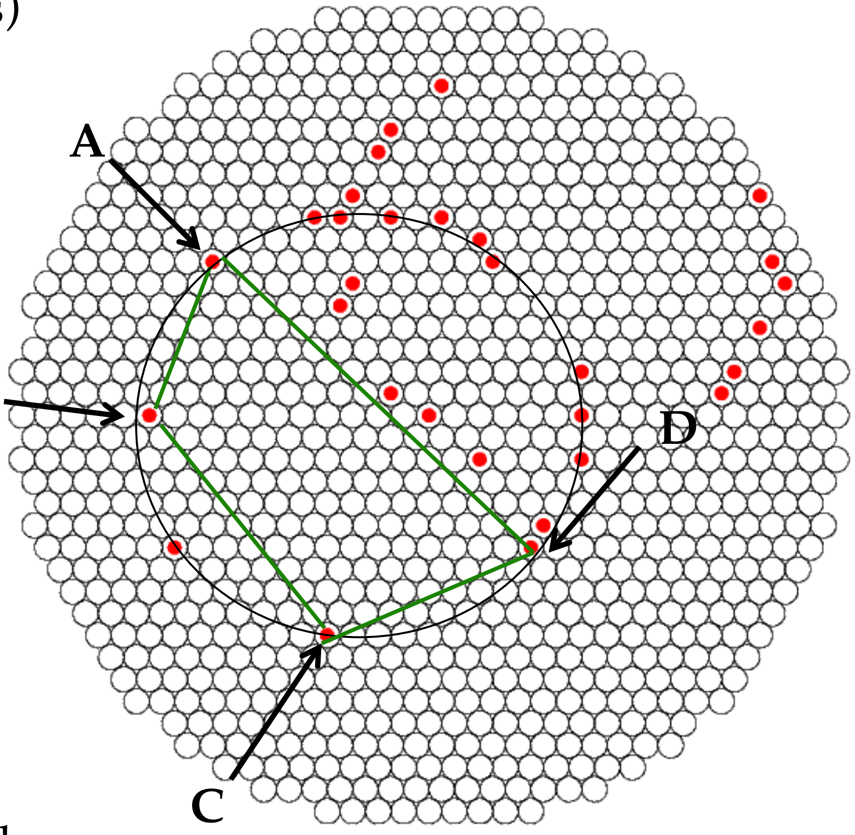


Almagest: example

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



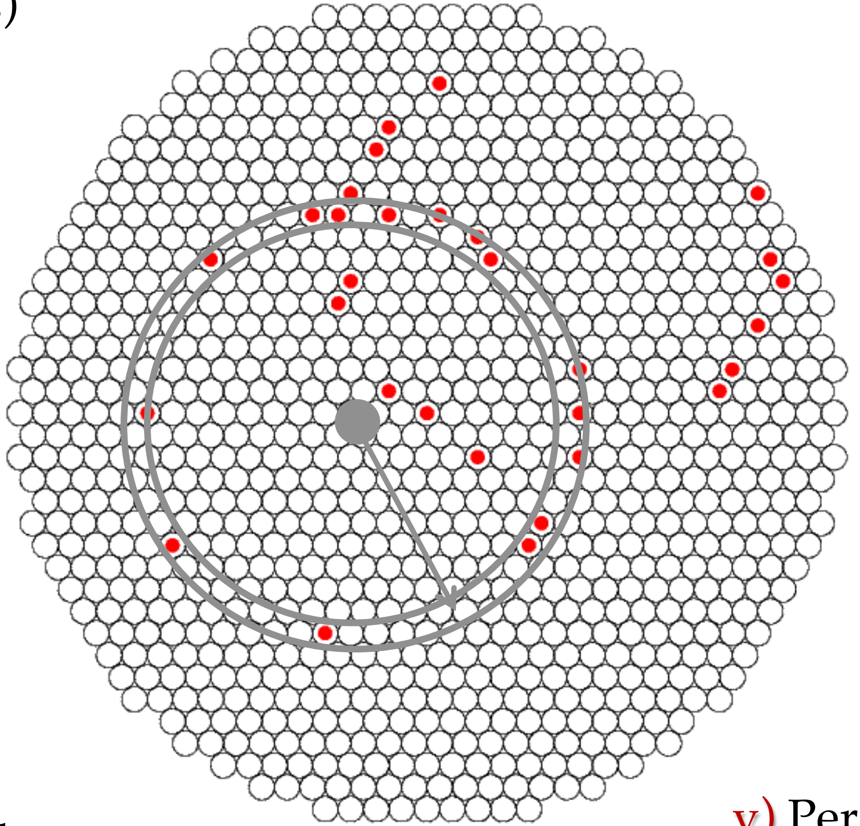
iv) …again…

Almagest: example

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…

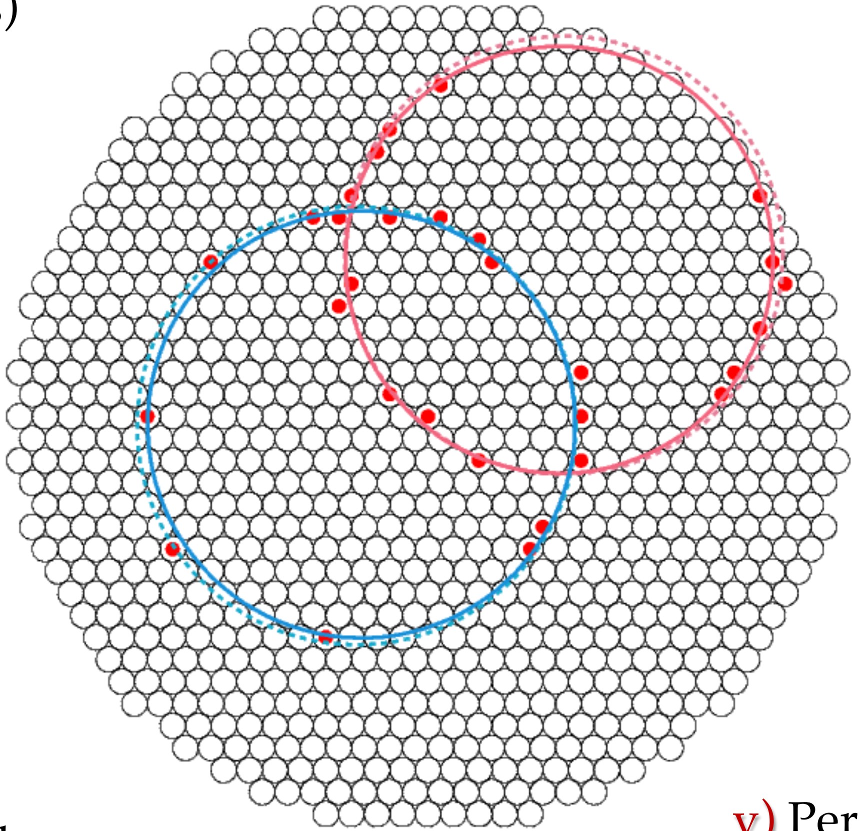
v) Perform a single ring fit

Almagest: example

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

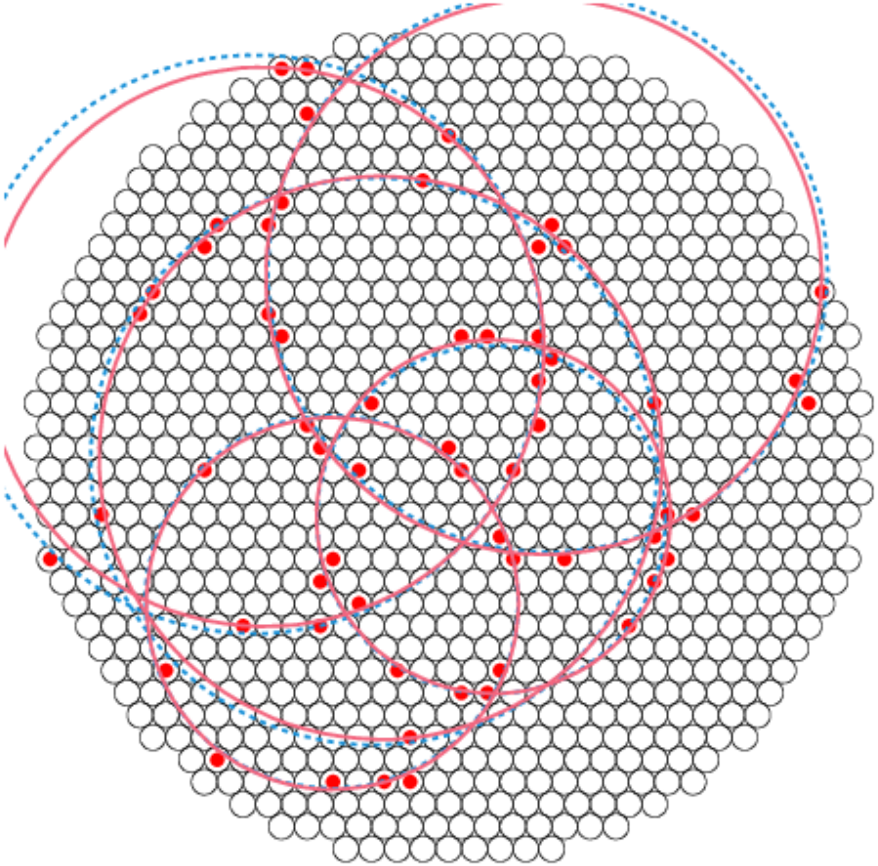


iv) …again…

v) Perform a single ring fit

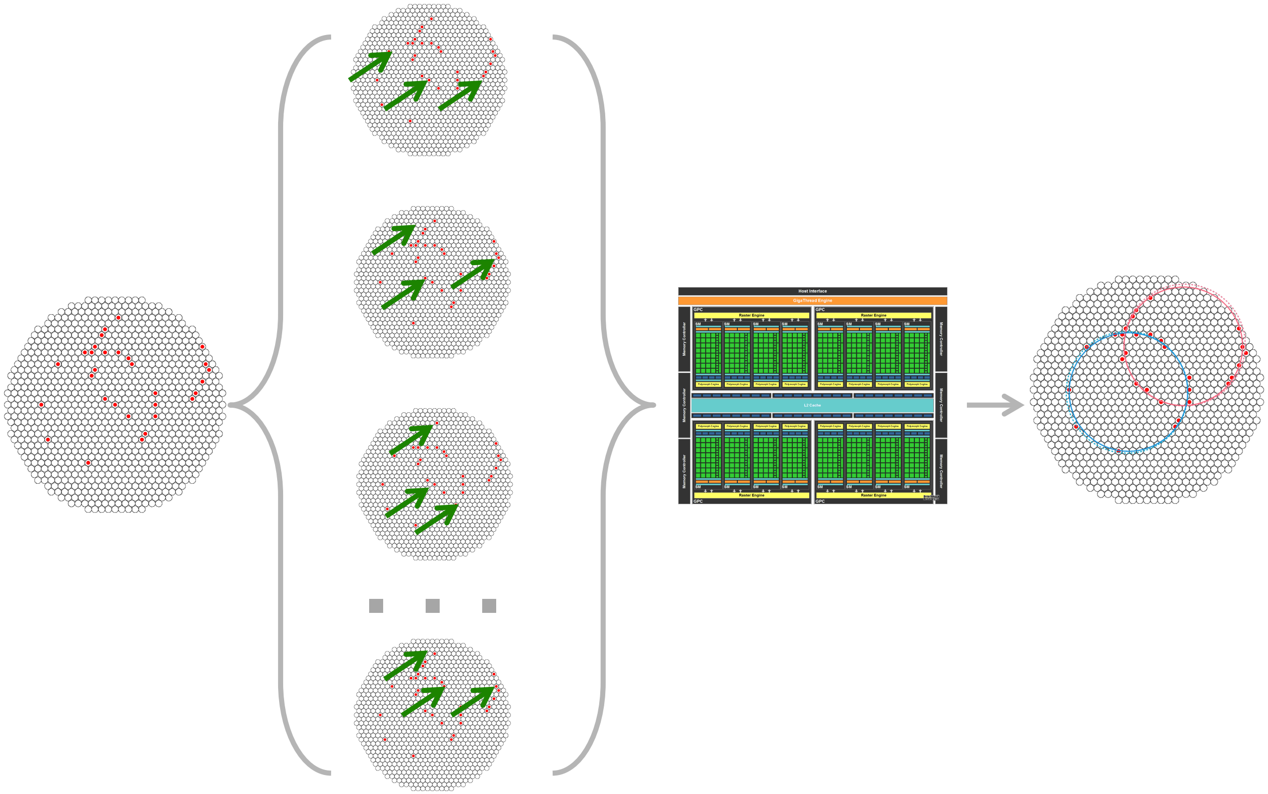
vi) Repeat by excluding the already used points

A more complicated example



Almagest on GPU

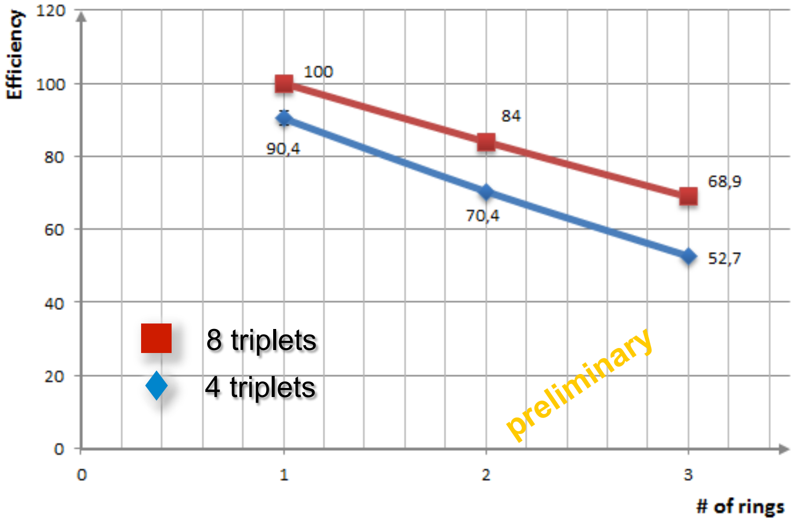
- Very high parallelism
  - Huge number of computing cores (>2000)
  - Huge memory bandwidth

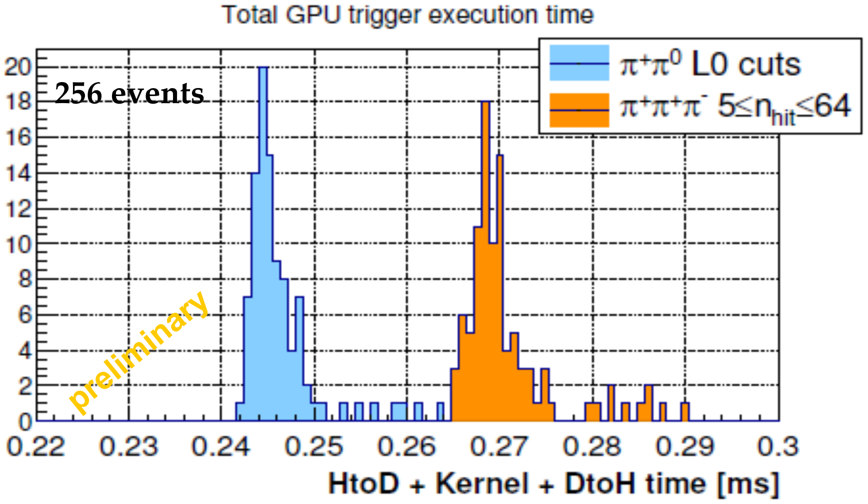


- Two levels of parallelism
  - Several triplets run in parallel
  - Several events at the same time

Almagest: implementation

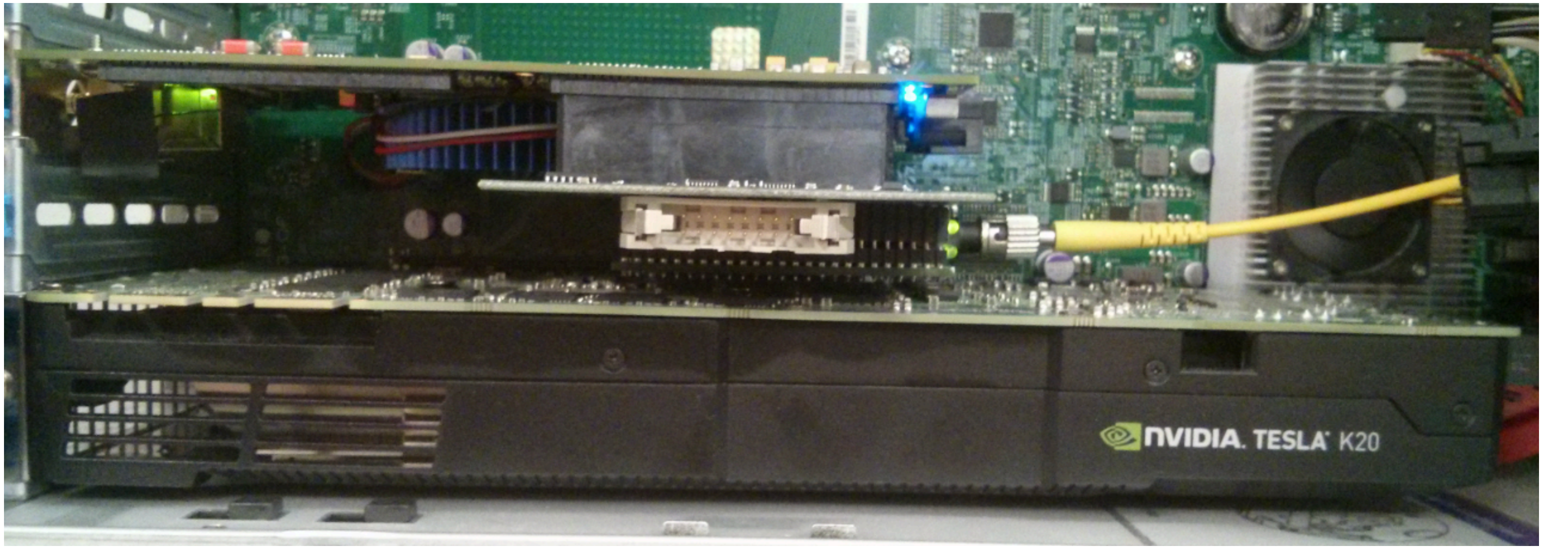
- Tests on NVIDIA Tesla K20 GPU
- Total computing time order **a few μs** per event (on single GPU)
- Good efficiency (using 8 triplets)
  - Room for improvement
- Further tests ongoing to study noise immunity, bias, efficiency a function of the number of hits, etc…





Next steps

- Receive the TTC stream (timing and trigger) from the experiment
  - ❑ TTC interface board recently produced
- Integration in the NA62 Trigger and DAQ system
  - ❑ Dry runs in summer
  - ❑ Parasitic test during NA62 experimental run in October



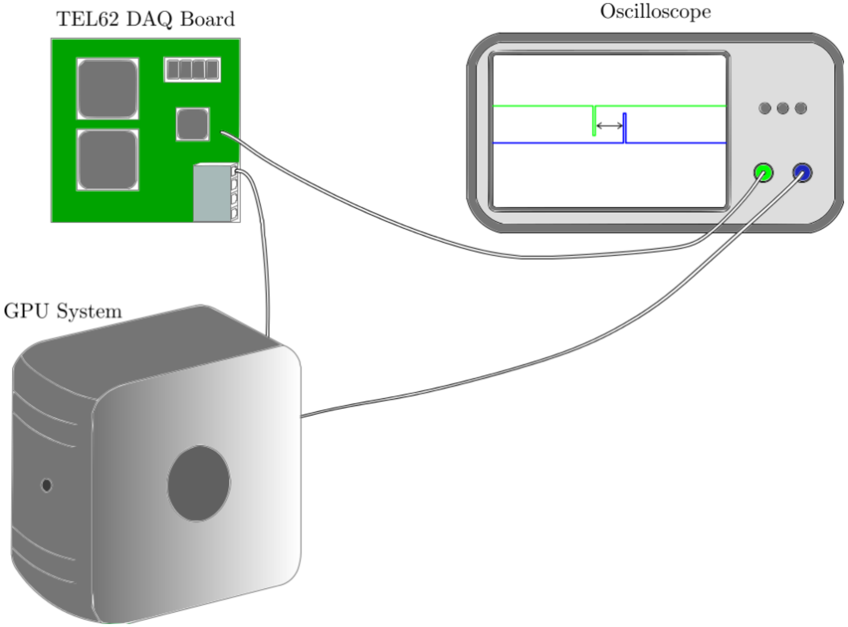
Conclusions

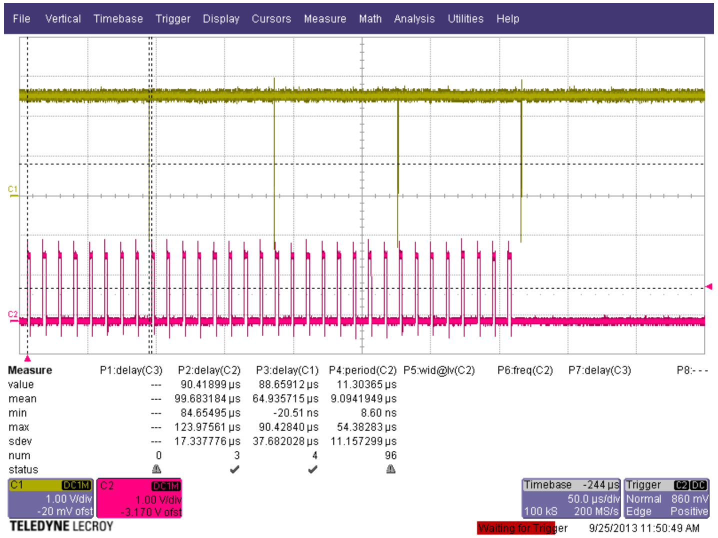
- The use of GPUs in HEP trigger systems could give several advantages, but processing performances and latencies should be carefully studied
  - Data transfer is the dominant contribution
- Construction of a demonstrator L0 processor for the NA62 RICH is under way
  - Cherenkov rings pattern recognition within the total L0 latency of 1 ms seems possible
- Integration with the NA62 Trigger and DAQ system
  - Dry runs in summer 2014
  - Parasitic data taking during NA62 experimental run starting October 2014

SPARES

Latency measurement

- Not easy to measure (with high precision) the latency between two systems running with uncorrelated clocks
- Use an oscilloscope to have a common time reference
  - Use the start of the packet in the TEL62 as “start” and the “computation done” in the PC as “stop”





Measure	P1:delay(C3)	P2:delay(C2)	P3:delay(C1)	P4:period(C2)	P5:wid@lv(C2)	P6:freq(C2)	P7:delay(C3)	P8:- - -
value	---	90.41899 µs	88.65912 µs	11.30365 µs				
mean	---	99.683184 µs	64.935715 µs	9.0941949 µs				
min	---	84.65495 µs	-20.51 ns	8.60 ns				
max	---	123.97561 µs	90.42840 µs	54.38283 µs				
sdev	---	17.337776 µs	37.682028 µs	11.157299 µs				
num	0	3	4	96				
status	⚠	✓	✓	⚠				

C1 DC1M 1.00 V/div -20 mV ofst

C2 DC1M 1.00 V/div -3.170 V ofst

Timebase -244 µs 50.0 µs/div 100 kS 200 MS/s

Trigger C2 DC Normal 860 mV Edge Positive

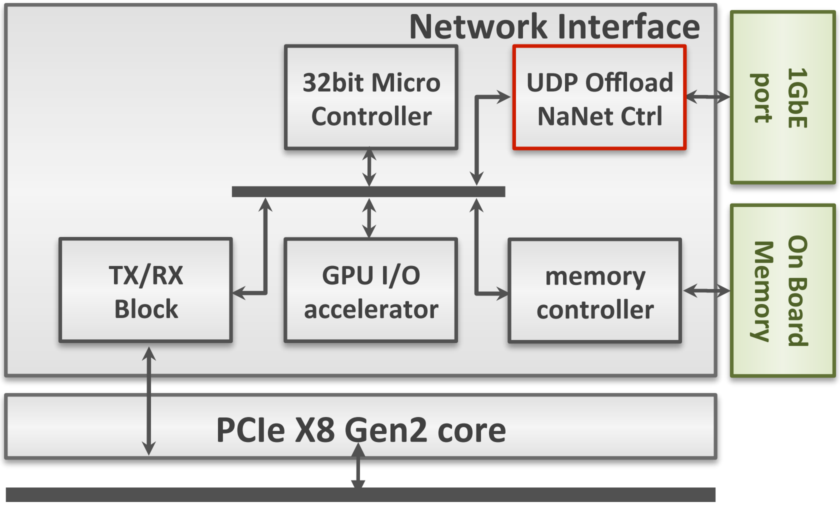
Waiting for Trigger 9/25/2013 11:50:49 AM

NANET (1)

- An **FPGA** based PCIe 8x gen 2 board derived from the **APEnet+ 3D NIC** design, able to directly inject an UDP data stream from an external **GbE** source into the memory of a Fermi- or Kepler-class NVIDIA GPU exploiting GPUDirect RDMA capabilities
- PCIe P2P protocol between Nvidia Fermi/Kepler devices and APEnet+
- First non-NVIDIA device (2012) supporting **GPUDirect RDMA**
  - No bounce buffers on host. APEnet+ can target GPU memory with no CPU involvement
  - **Latency reduction for small messages**

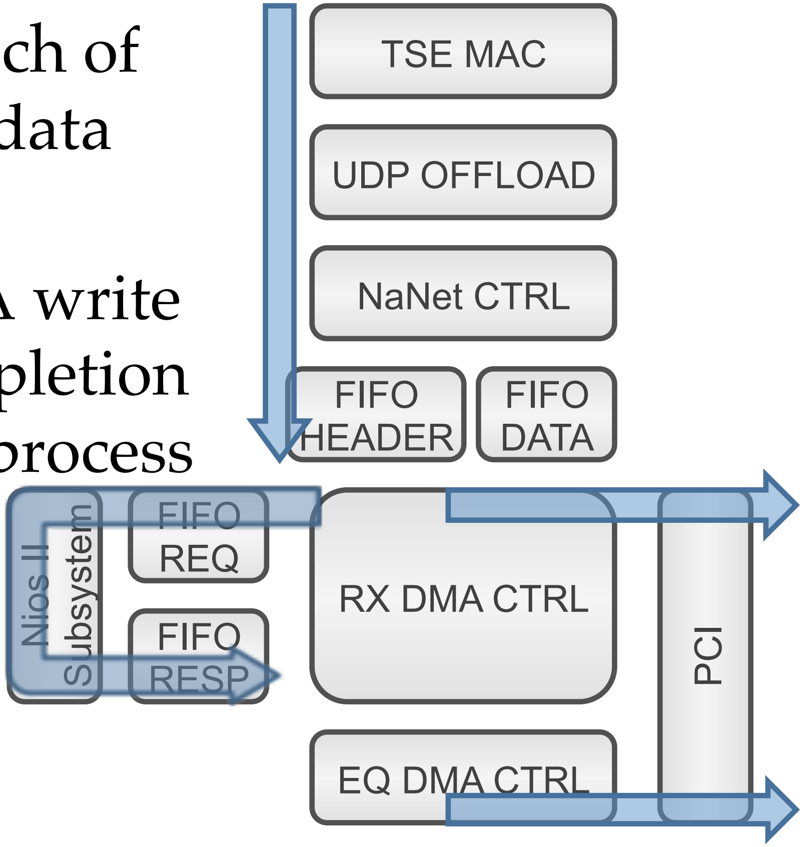
NANET (2)

- **NaNet**: reduce comm. latency and its fluctuations to meet real-time constraints
- UDP offload collects data coming from the Altera Triple-Speed Ethernet Megacore (TSE MAC) and redirects UDP packets into a hardware processing data path
- NaNet Controller encapsulates the UDP payload in a newly forged APEnet+ packet and sends it to the RX Network Interface logic
- RX DMA CTRL manages CPU/GPU memory write process



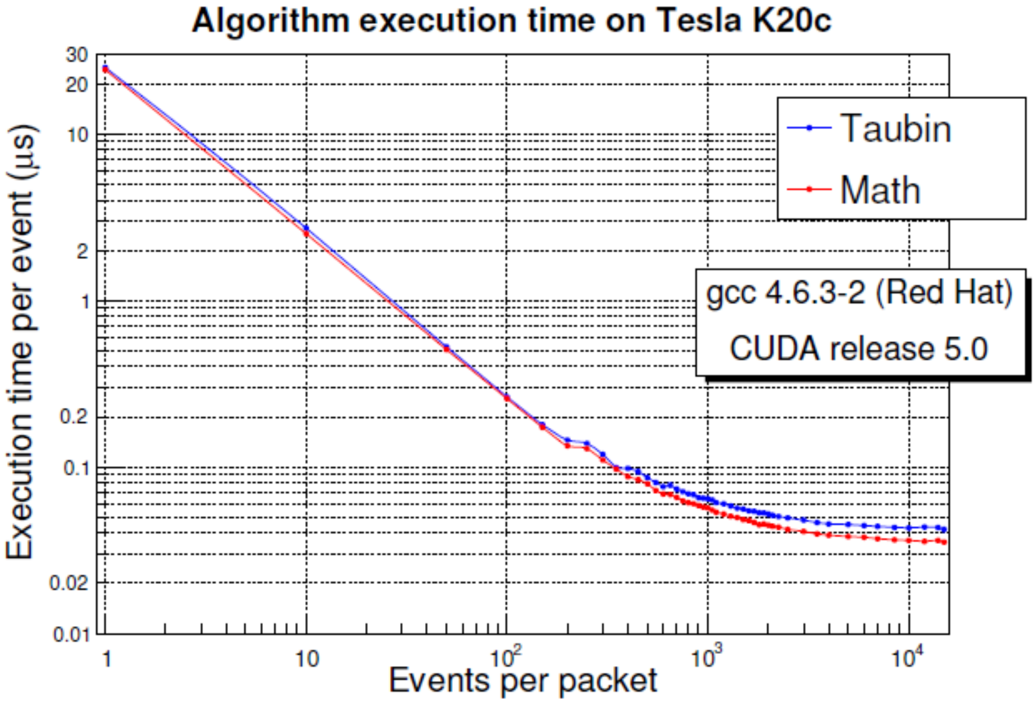
NANET (3)

- Nios II handles all the details pertaining to buffers registered by the application to implement a zero-copy approach of the RDMA protocol (OUT of the data stream)
- EQ DMA CTRL generates a DMA write transfer to communicate the completion of the CPU/GPU memory write process
- A Performance Counter is used to analyze the latency of the GbE data flow inside the NIC



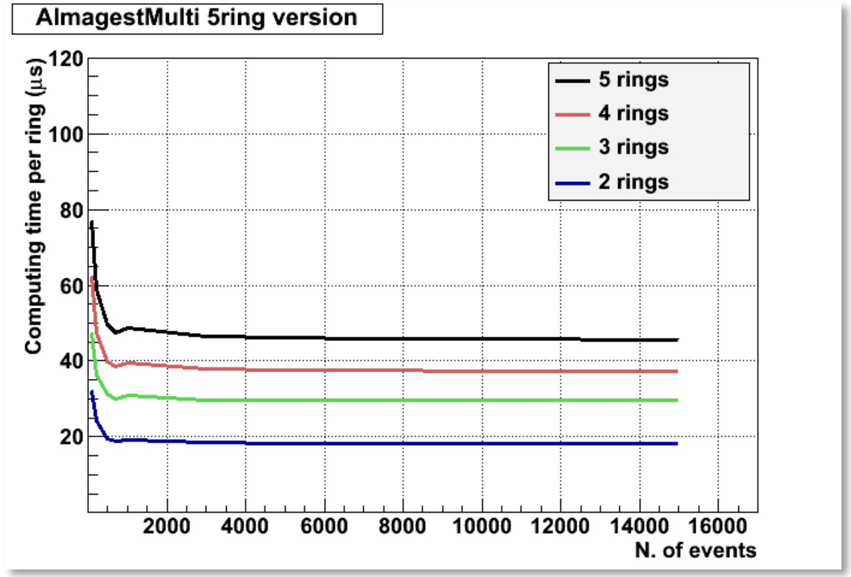
Single ring

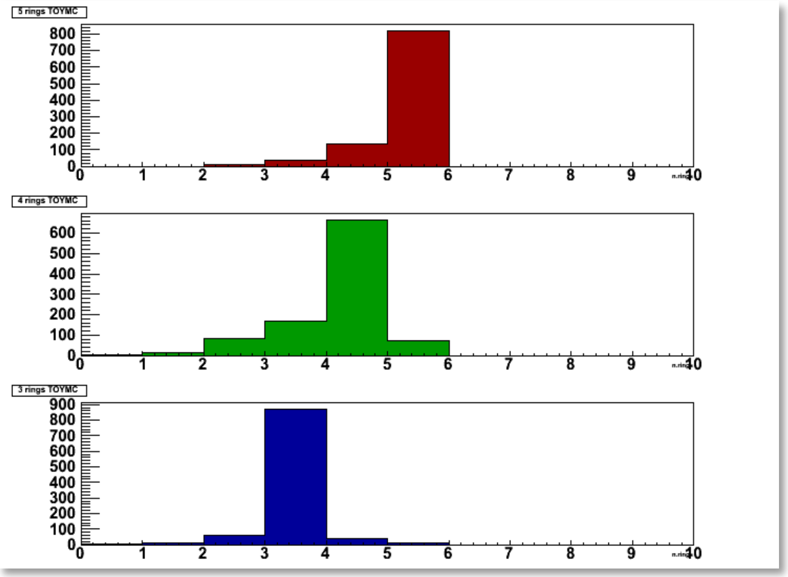
- Crowford method (“math”):
  - Translate in the center of mass
  - Least square minimization → linear
- Taubin method:
  - More efficient: minimize the bias introduced by the Kasa related methods (minimization of simple algebraic distance)
  - Resolution slightly better (on identified rings)
- The difference of computing time on the GPU is at level of 10 ns per event



N(=hits) triplets

- Number of triplets equal to the number of hits.
- Relatively high efficiency.
- Computing time depends on number of rings (different number of GPU cores per events)
- Results on TESLA C1060 (240 cores, less than 1 Tflops)
- Room for optimization

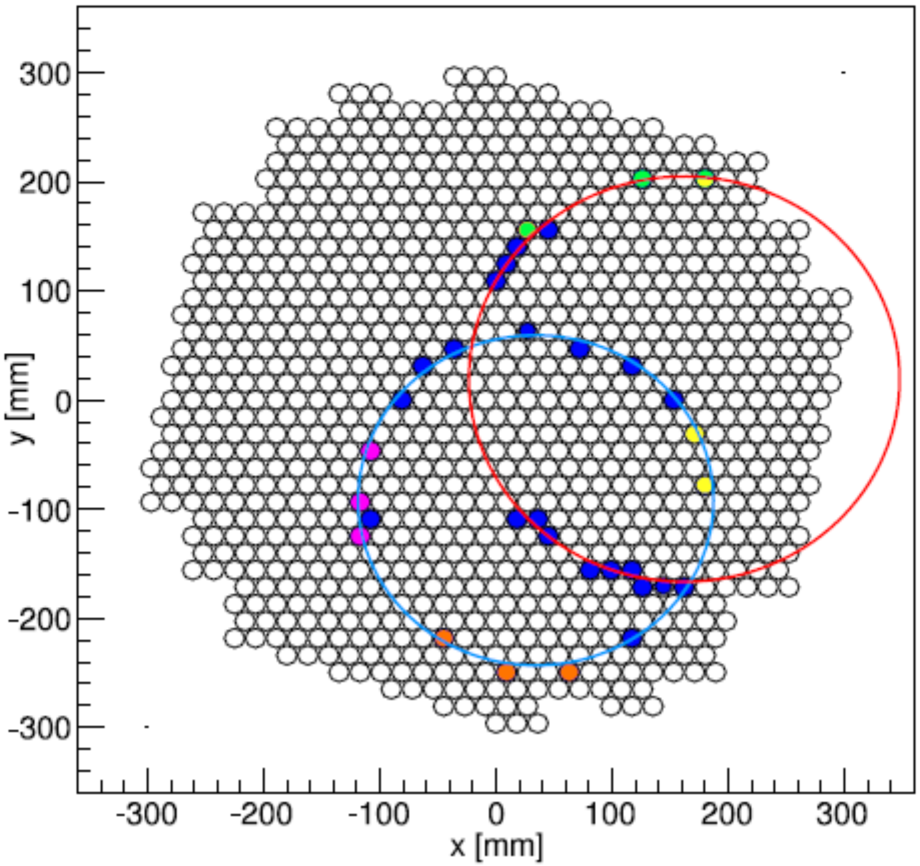




4 selected triplets

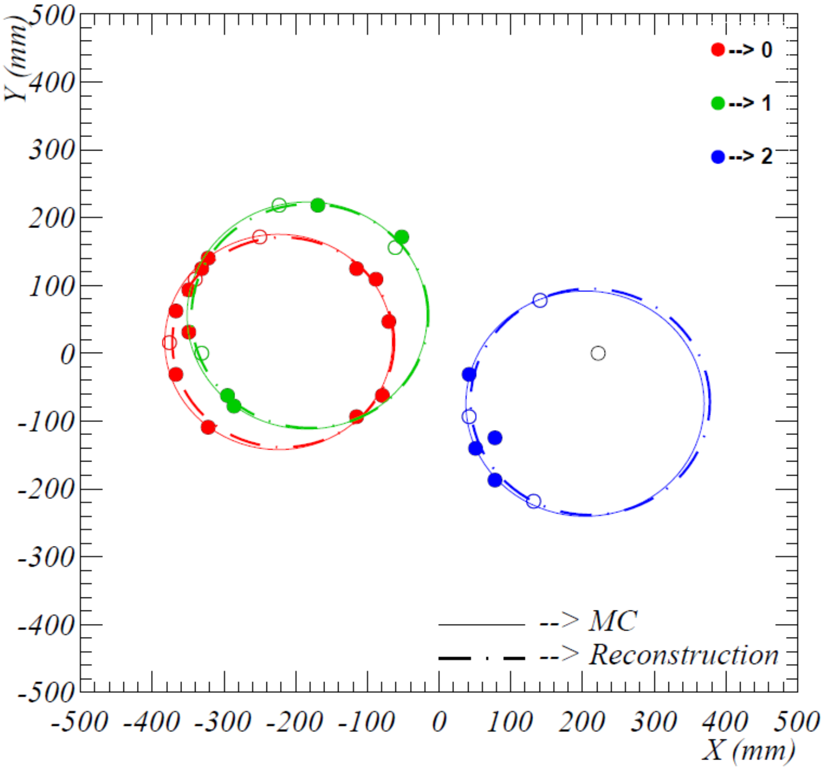
- Only 4 triplets per event are used: left, right, up and down
- Further cuts to avoid too close hits

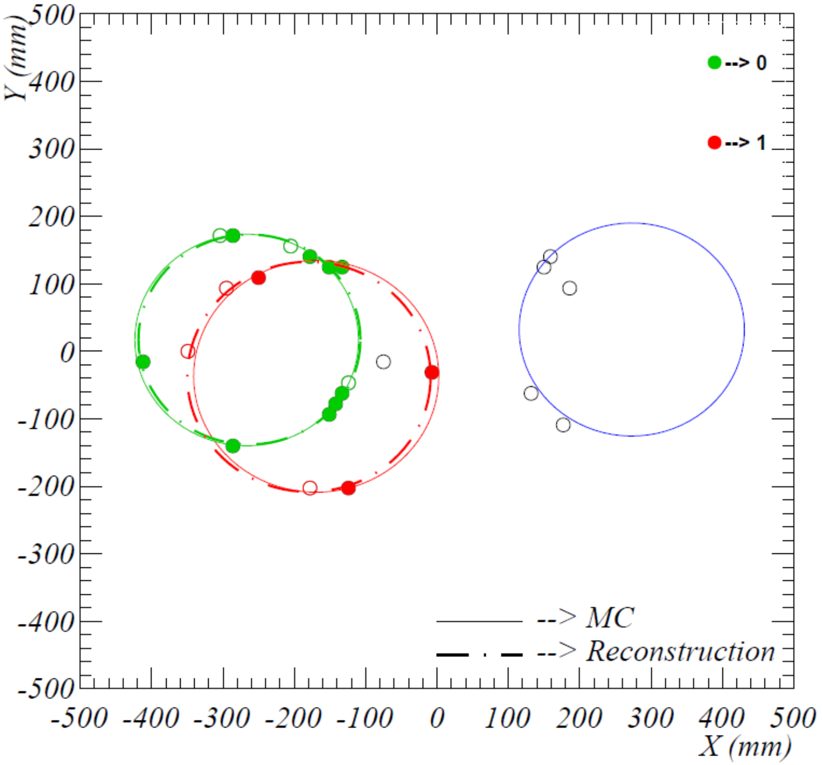
RICH event featuring a π^+ and an e^+



4 selected triplets

- Stability with small noise (studies are ongoing)
- Inefficiency due to the order in choosing the rings.
- Dependence on the cuts to define the triplets.


Y (mm)
X (mm)
● --> 0
● --> 1
● --> 2
--> MC
--> Reconstruction


Y (mm)
X (mm)
● --> 0
● --> 1
--> MC
--> Reconstruction