Use of GPUs for trigger applications in HEP experiments

Felice Pantaleo

(CERN PH-SFT)

DESY Computing Seminar

15/10/2012

Outline

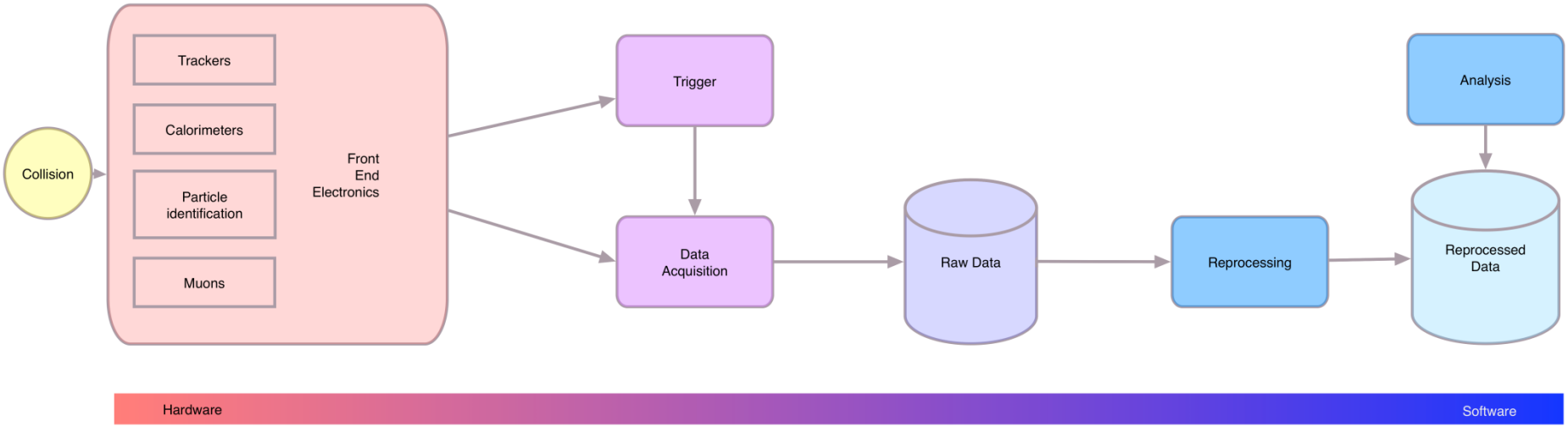
- Introduction
- Graphic Processing Units
- ALICE HLT
- NA62
  - RICH
  - CHOD
  - Tests
- Conclusion
- Reference

Introduction

Introduction

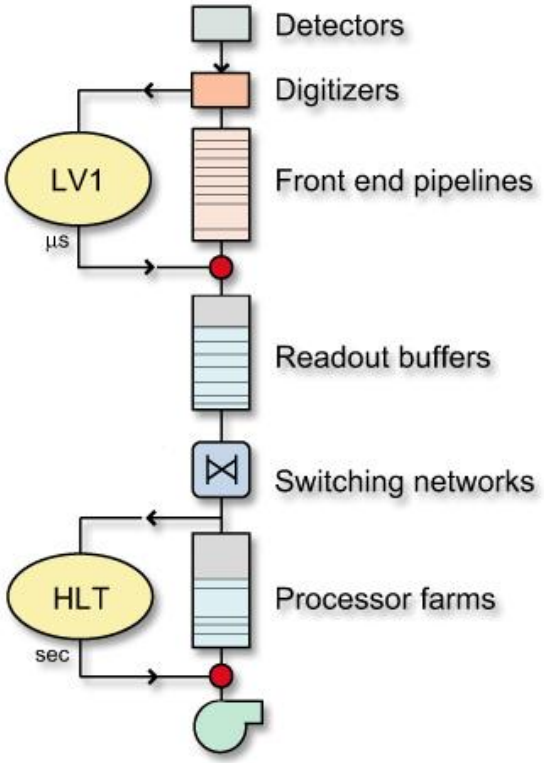
- High Energy Physics detectors needs
  - Processing large amounts of information
  - Very short time response
  - Complex structures, many sensors with topographic information
  - Efficient processing of data
- Huge benefits expected from the use of Multicore techniques
- Where are we ?

Generic Detector Structure



Generic Trigger Structure

40 MHz
Clock driven
Custom processors

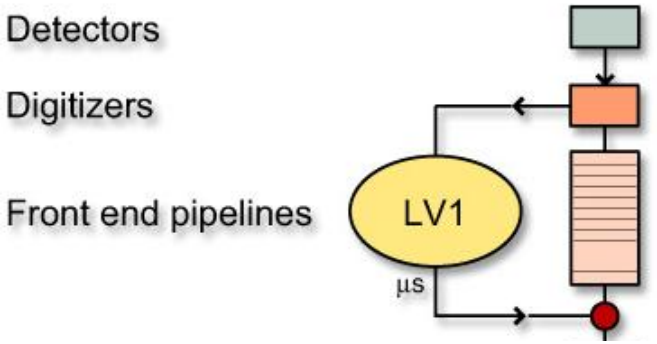


Low Level Trigger
NA62

100 kHz
Event driven
PC network

High-Level Trigger
ALICE

Low Level Trigger



- Time needed for decision $\Delta t_{dec} \approx 1$ ms
- Particle rate ≈ 10MHz
- Need pipelines to hold data
- Need fast response

- Backgrounds are huge
- High rejection factor

- Algorithms run on local, coarse data

- Ultimately, determines the physics

Parallel Opportunities

- On detector (mostly Cellular Automata)
  - Trackers
  - Calorimeters
- Trigger Systems
  - Event decision: one event per node + dispatcher
- The Grid

Current Proposals: steps beyond

Many GPU applications:

- Reconstruction of tomographies in the European Synchrotron Radiation Facility
- Integration of GPUs into the ITMS framework of data acquisition in fusion experiments
- Track reconstruction simulation for PANDA in FAIR
- NA62 Low-Level Trigger
- ALICE High-Level Trigger

HEP is a long lived science...

- Many algorithms in HEP have cores dating the seventies
- Mostly thought and devised as single threaded
- Current solution: one event-one core
- This solution shows saturation

Change of paradigm ?

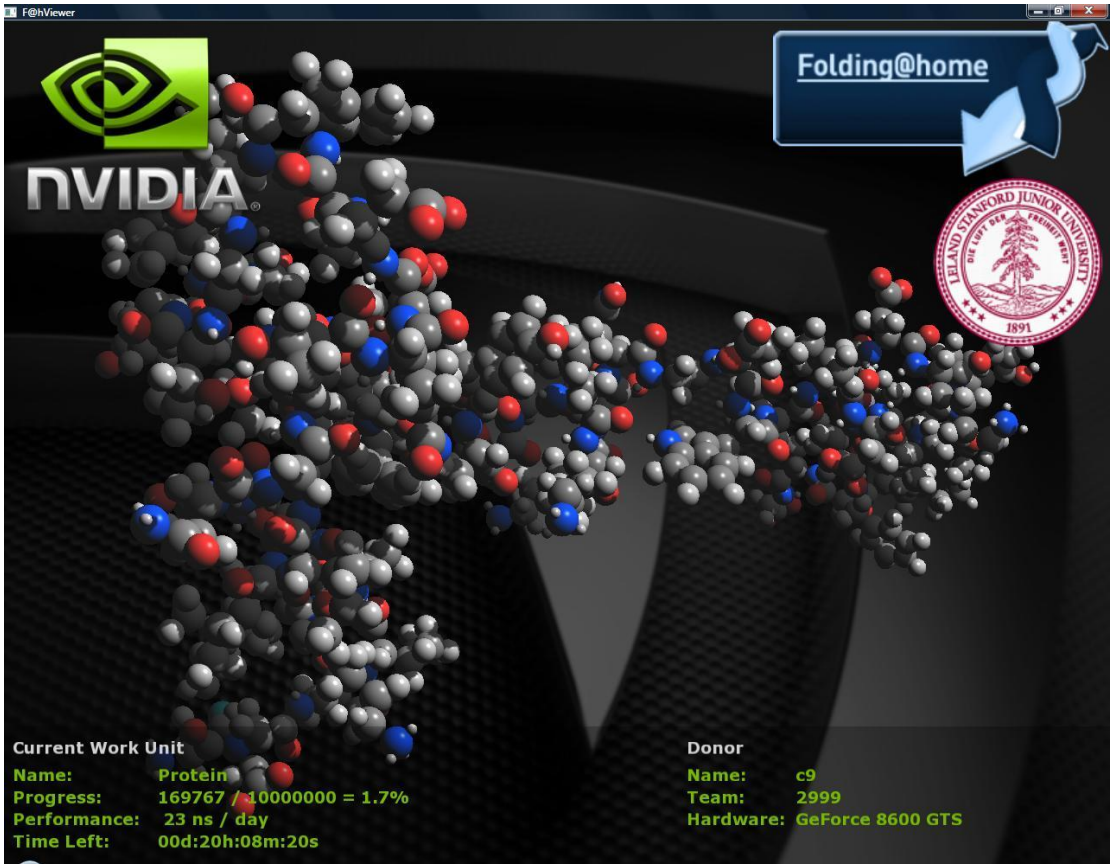
Opportunities for HEP experiments

- HEP characteristics and needs seem to fit perfectly both
  - Cellular techniques
  - Multi- Many- cores architectures
- Our feeling: a large number of opportunities ahead of us
- Further study is needed to evaluate the real benefits of these solutions.

Graphic Processing Units

GPUs (1/2)

- Exceptional raw power wrt CPUs
- Higher energy efficiency (NVIDIA Kepler PUE 1.05)
- Plug & Accelerate
- Massively parallel architecture
- Low Memory/core



GPUs (2/2)

GPUs were traditionally used for real-time rendering.

NVIDIA & AMD main manufacturers.

Architecture (Kepler)

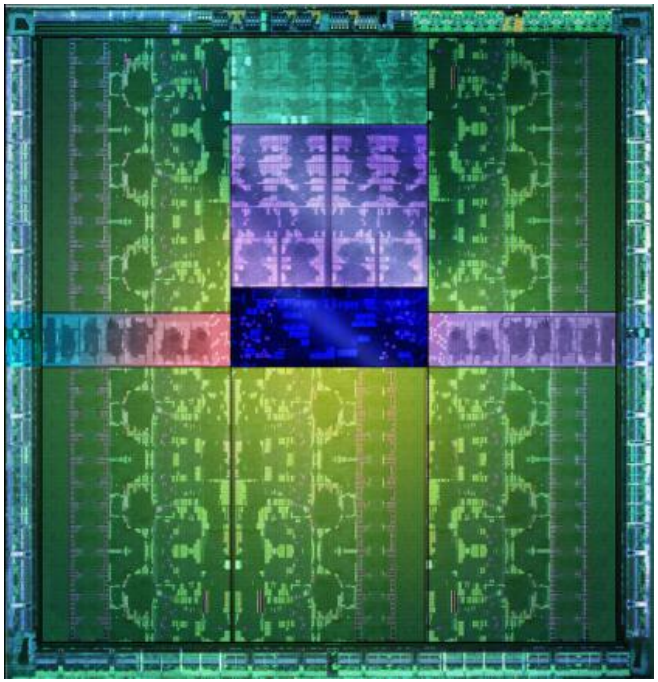
- 7.1B Transistors
- 15 SMX units
- > 1 TFLOP FP64
- 1.5 MB L2 Cache
- 384-bit GDDR5
- PCI Express Gen3

CUDA – Kepler Architecture

- SMX executes hundreds of threads concurrently.
- **SIMT** (Single-Instruction, Multiple-Thread)
- Instructions pipelined
- Thread-level parallelism
- Instructions issued in order
- No Branch prediction
- No speculative execution
- Branch predication

CUDA – Abstractions

1. Hierarchy of thread groups
2. Shared memory
3. Barrier synchronization



Fine-grained *data/thread* parallelism

Coarse-grained *data/task* parallelism

Instruction-level parallelism

ALICE High Level Trigger

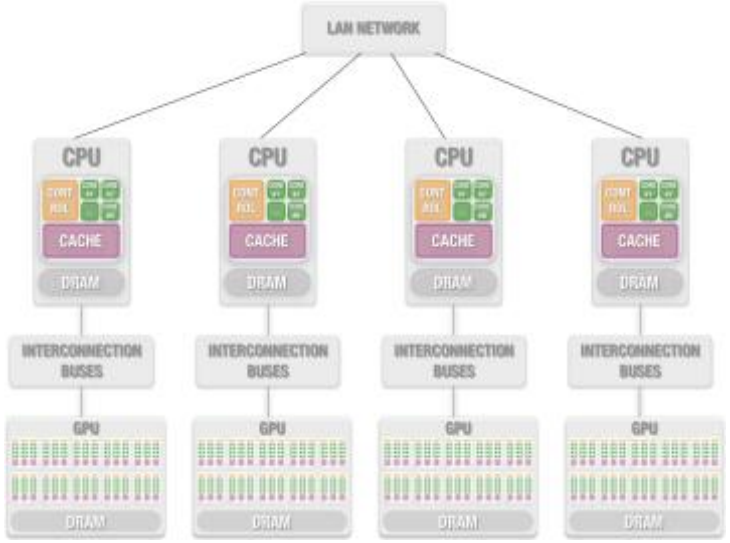
Thanks to D. Rohr (ALICE Collaboration)

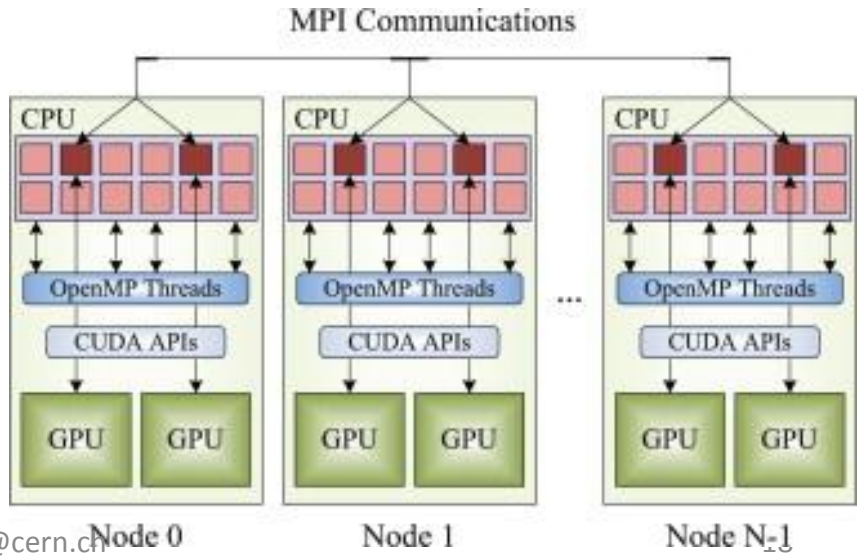
What it is happening in HPC

Use several platforms containing GPUs to solve one single problem

Programming challenges:

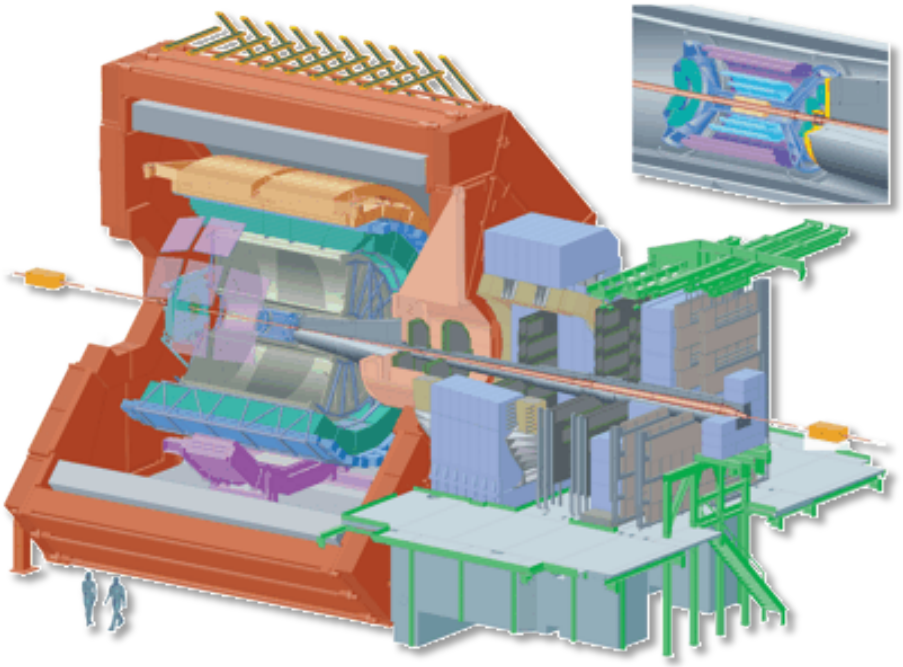
- Algorithm parallelization
- Perform computation in GPUs
- Execution in a distributed system where platforms have their own memory
- Network communication.





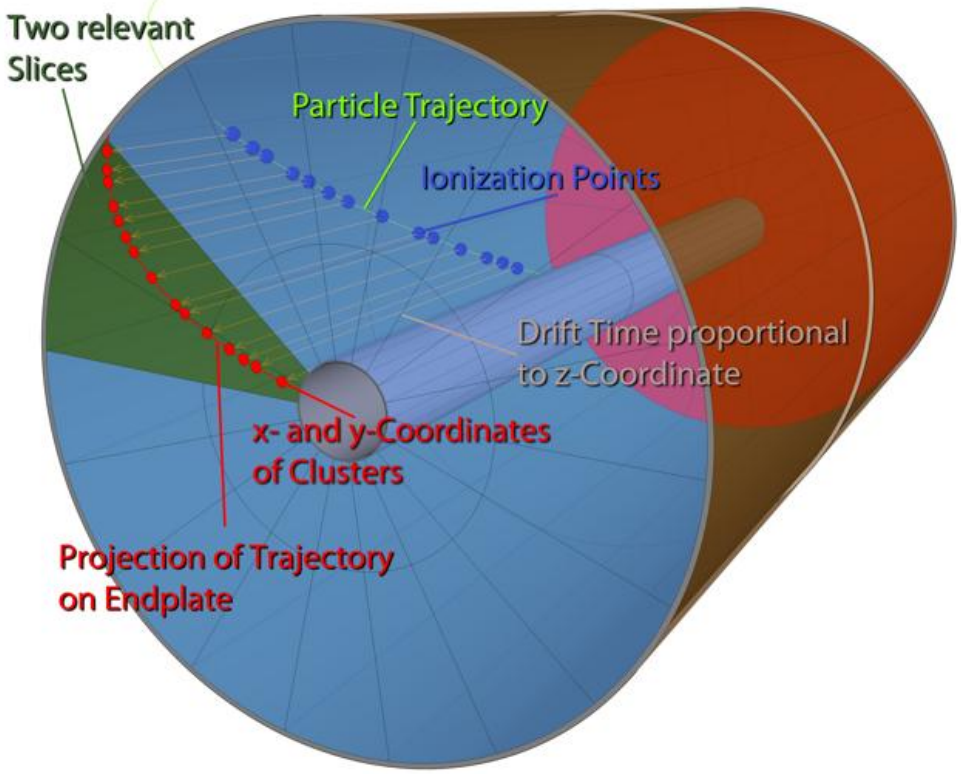
ALICE Detector

ALICE is one of the major four experiments of the Large Hadron Collider at CERN. It was specifically designed to study heavy ion collisions.



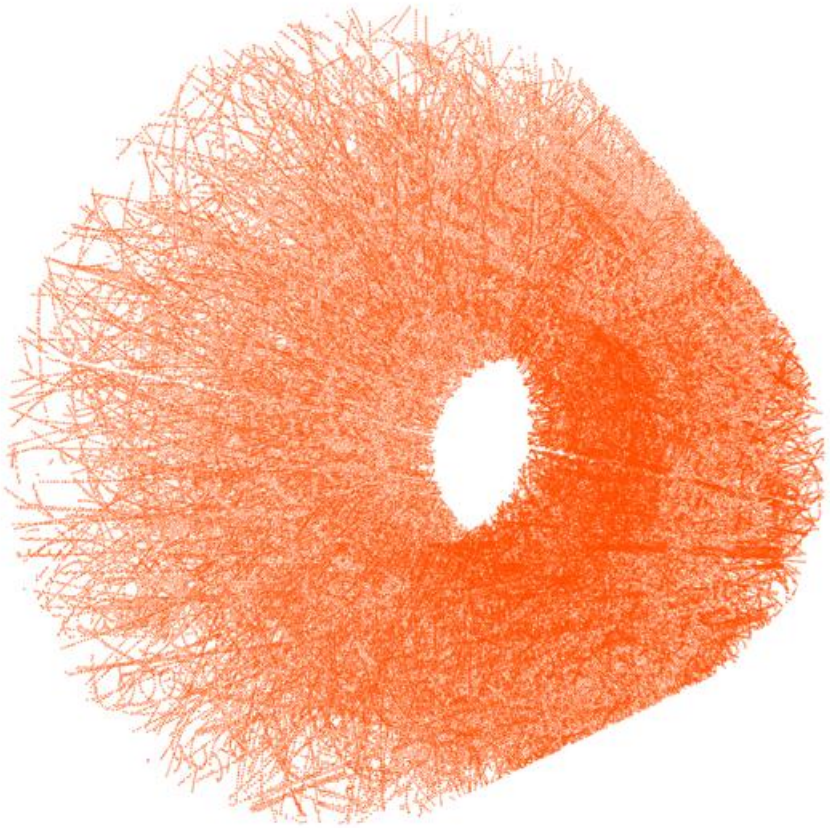
ALICE TPC

A Time Projection Chamber (TPC) is used to measure particle trajectories.



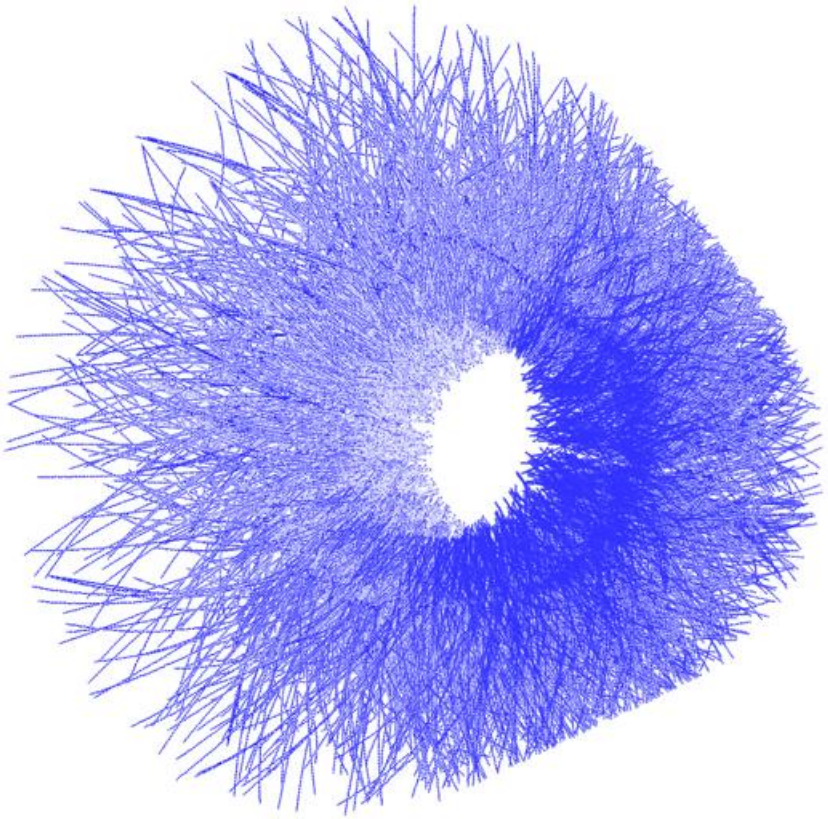
ALICE TPC

- TPC clusters of a heavy ion event



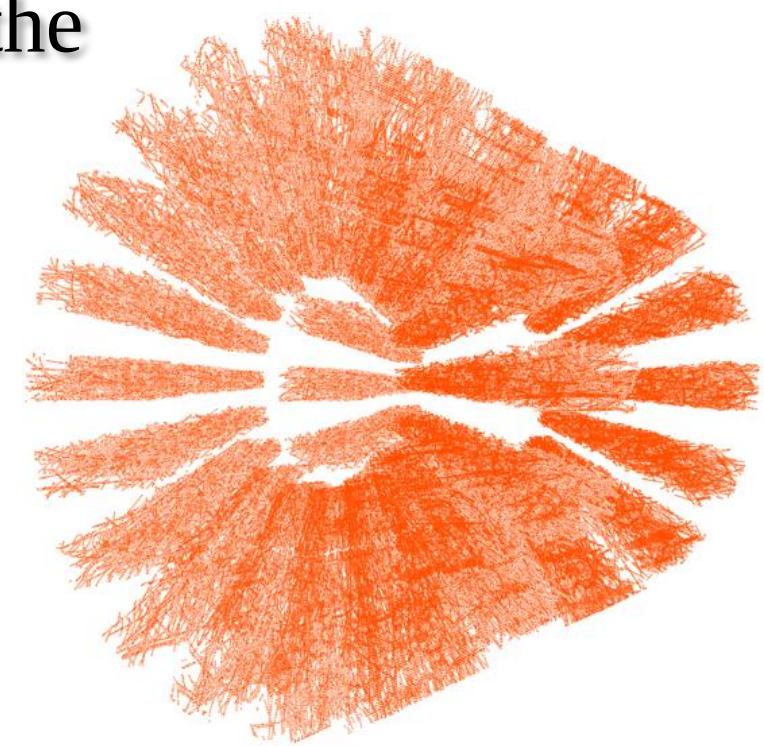
ALICE TPC

- Tracks reconstructed from the clusters



ALICE HLT

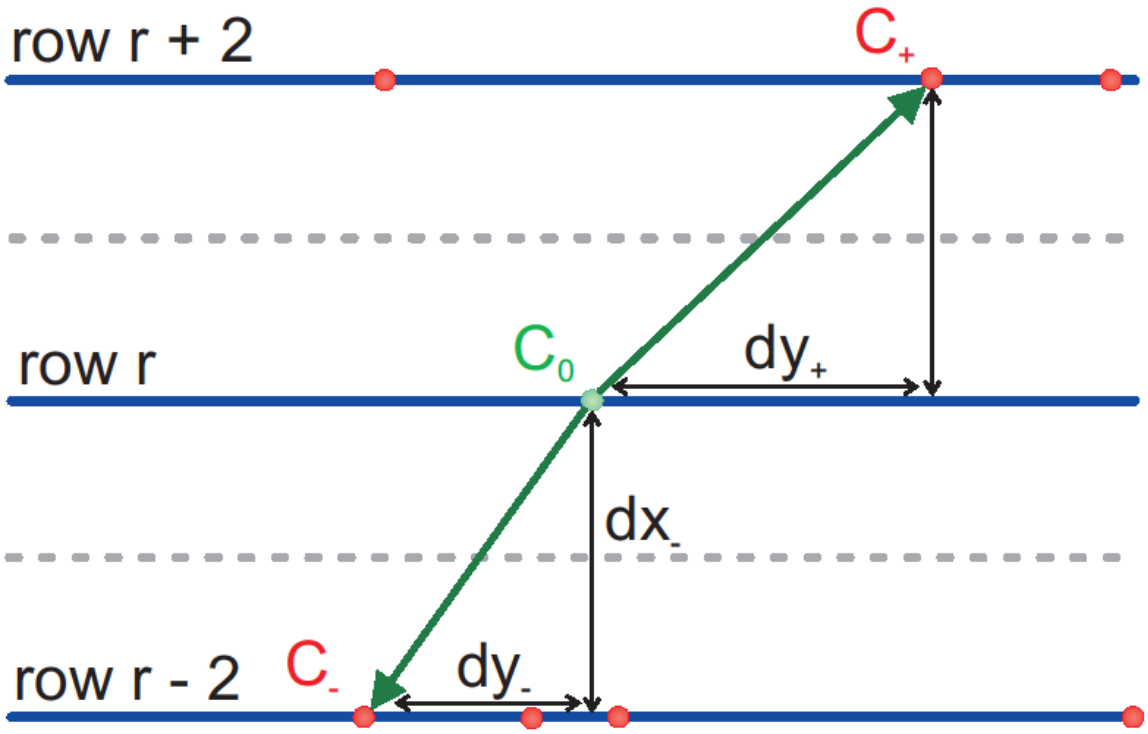
- Alice HLT tracker divides the TPC in slices and processes the slices individually.
- Track segments from all the slices are merged later



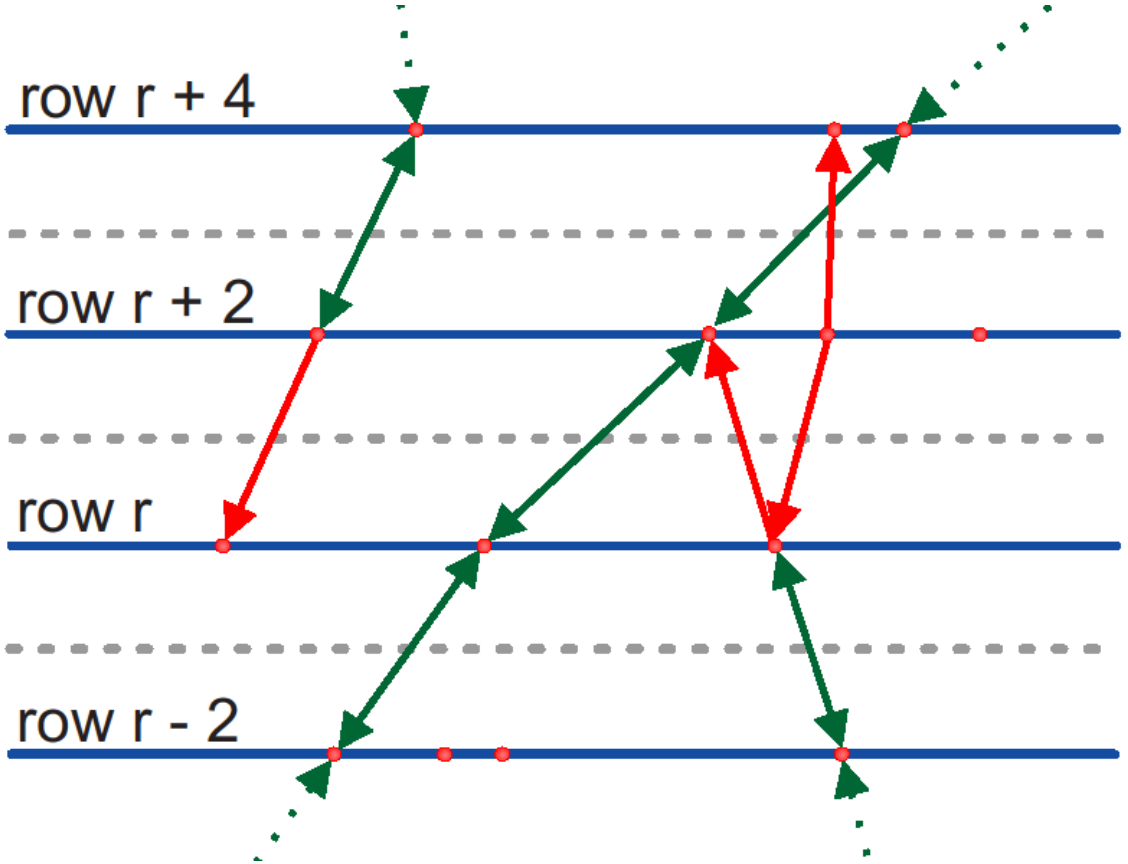
Tracking algorithm

Category of task	Name of task	Description on task
	(Initialization)	
Combinatorial part (Cellular automation)	I: Neighbors finding	Construct seeds (Track candidates)
	II: Evolution	
Kalman filter part	III: Tracklet construction	Fit seed, extrapolate tracklet, find new clusters
	IV: Tracklet selection	Select good tracklets, assign clusters to tracks
	(Tracklet output)	

Neighbors Finding

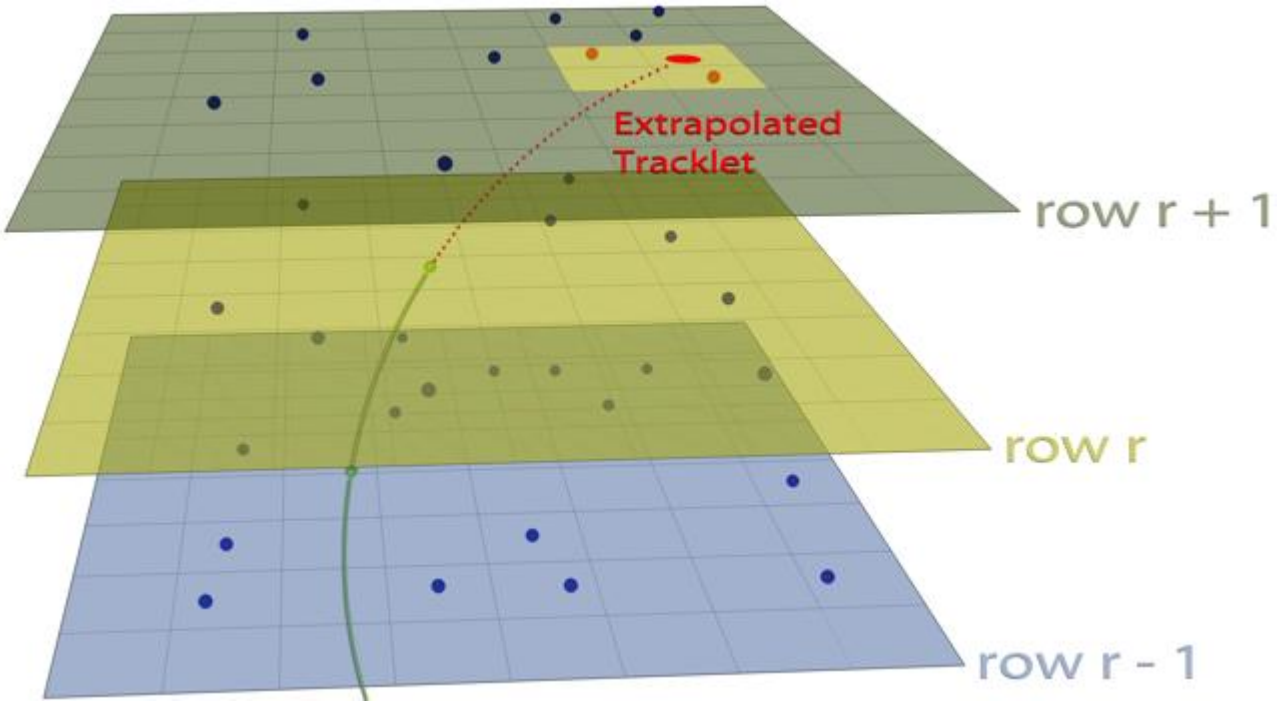


Evolution step

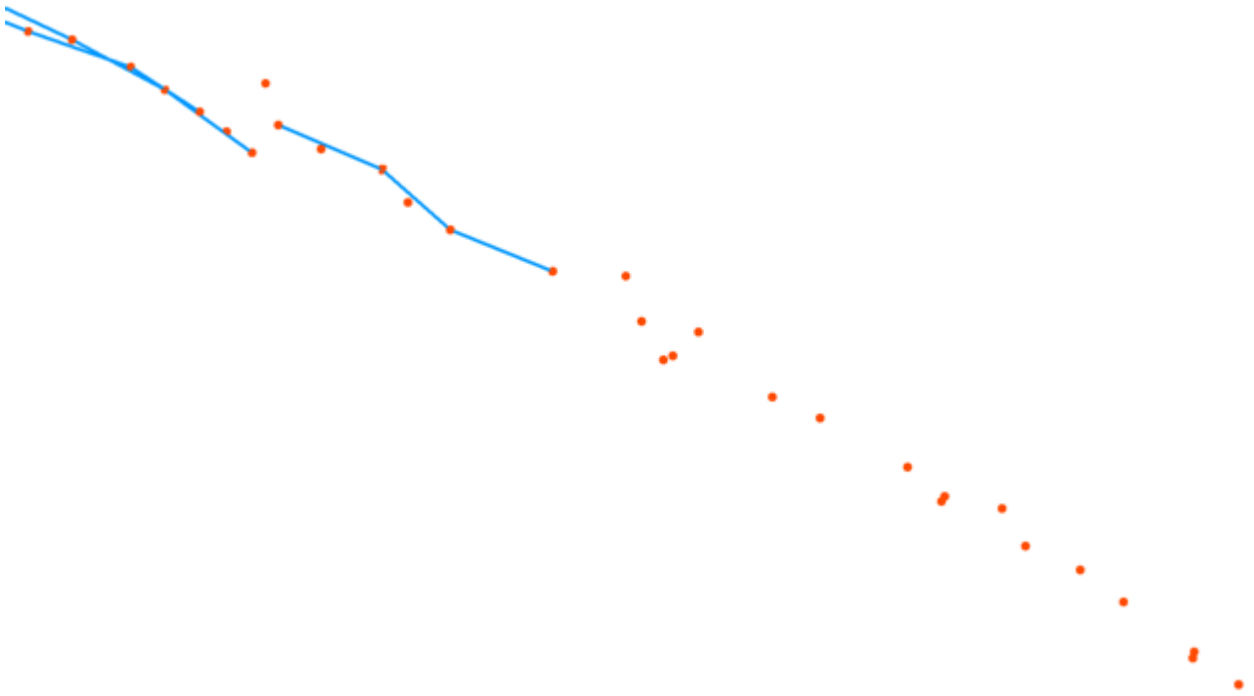


Tracklet reconstruction

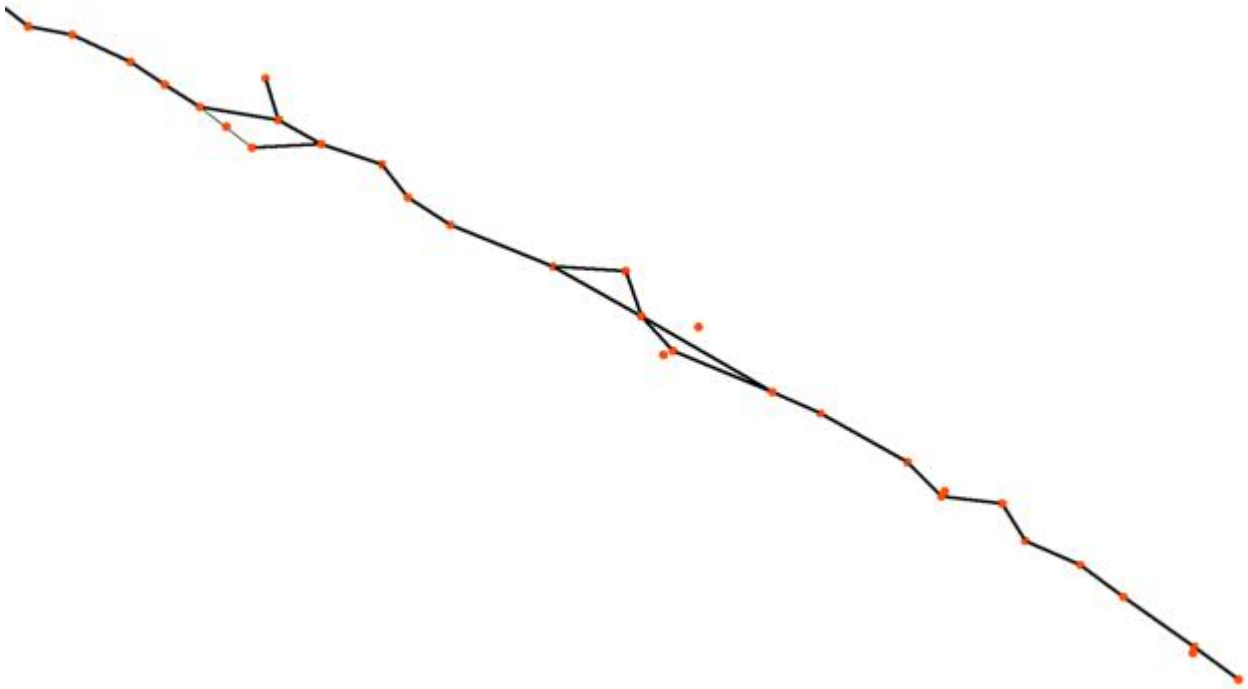
- The algorithm looks for clusters close to extrapolation point



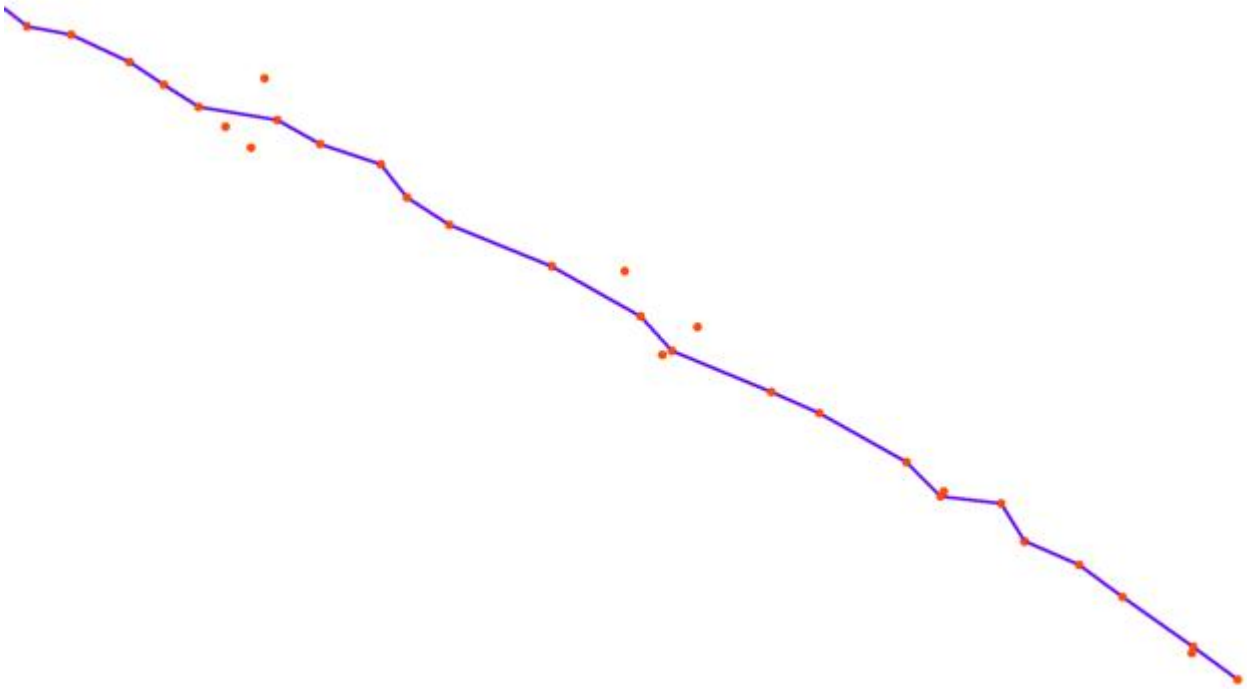
Evolution step



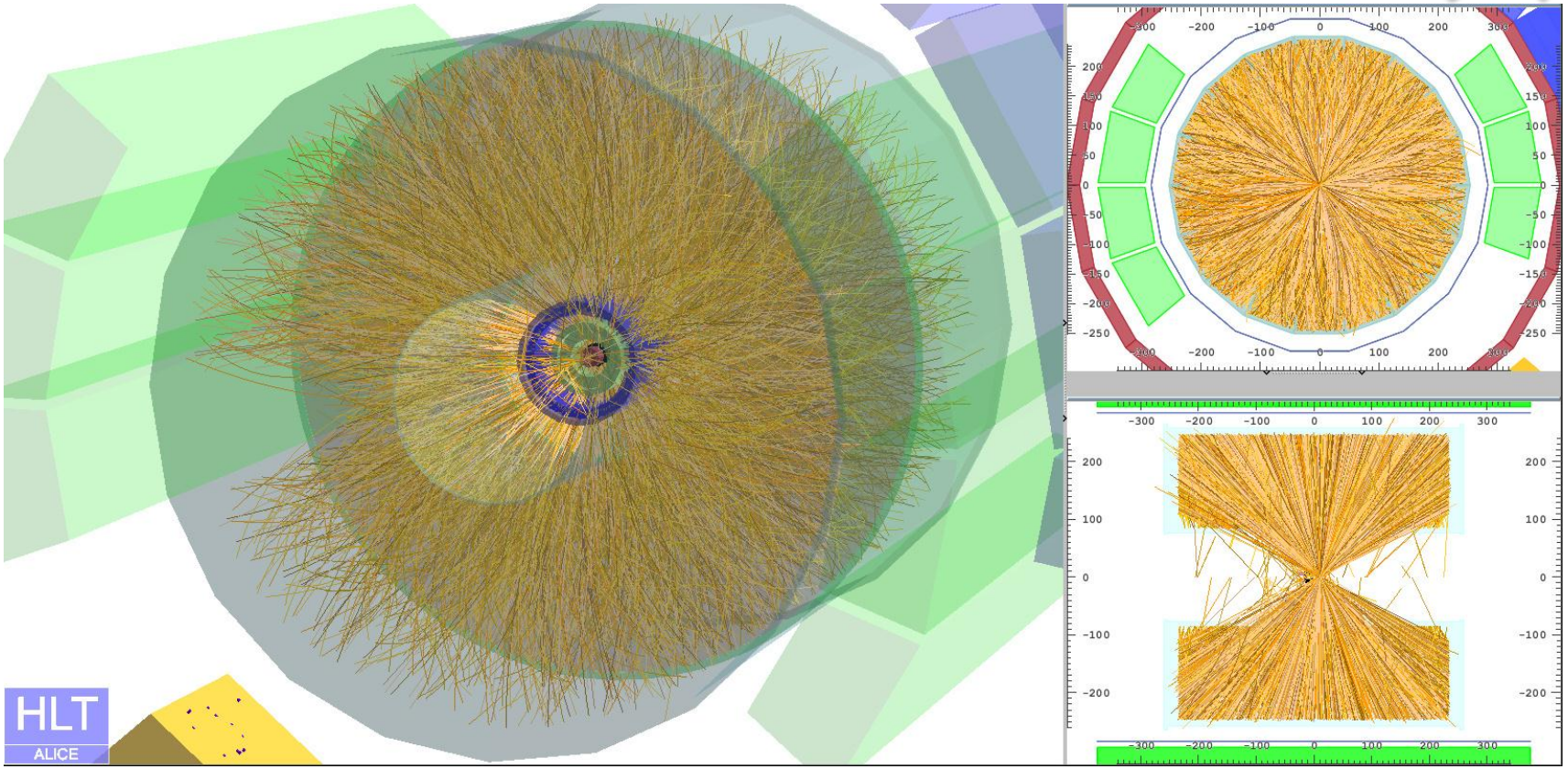
Tracklet construction



Tracklet selection



ALICE GPU Tracker

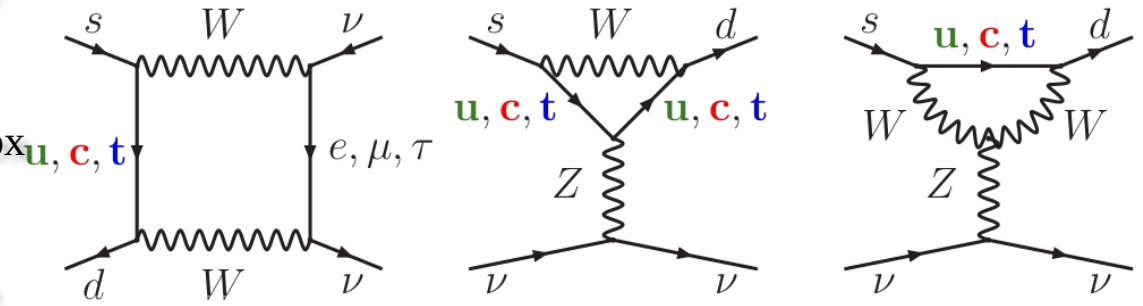


Screenshot of ALICE Online-Event-Display during first physics-fill with active GPU Tracker.

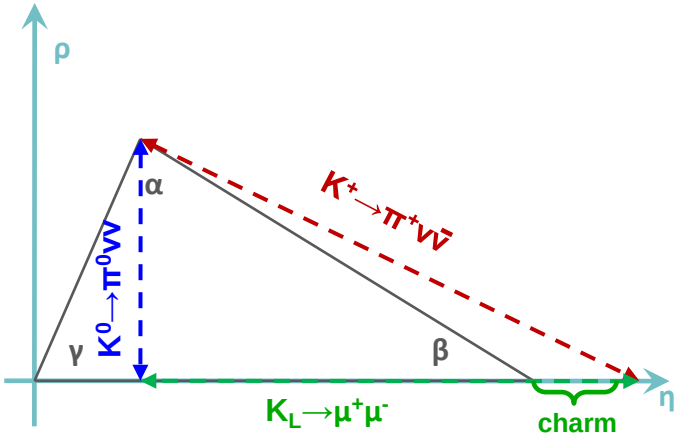
NA62

$K^+ \rightarrow \pi^+ \nu\nu$ in the Standard Model

- FCNC process forbidden at tree level
- Short distance contribution dominated by Z penguins and box diagrams
- Negligible contribution from u quark, small contribution from c quark
- Very small BR due to the CKM top coupling $\rightarrow \lambda^5$

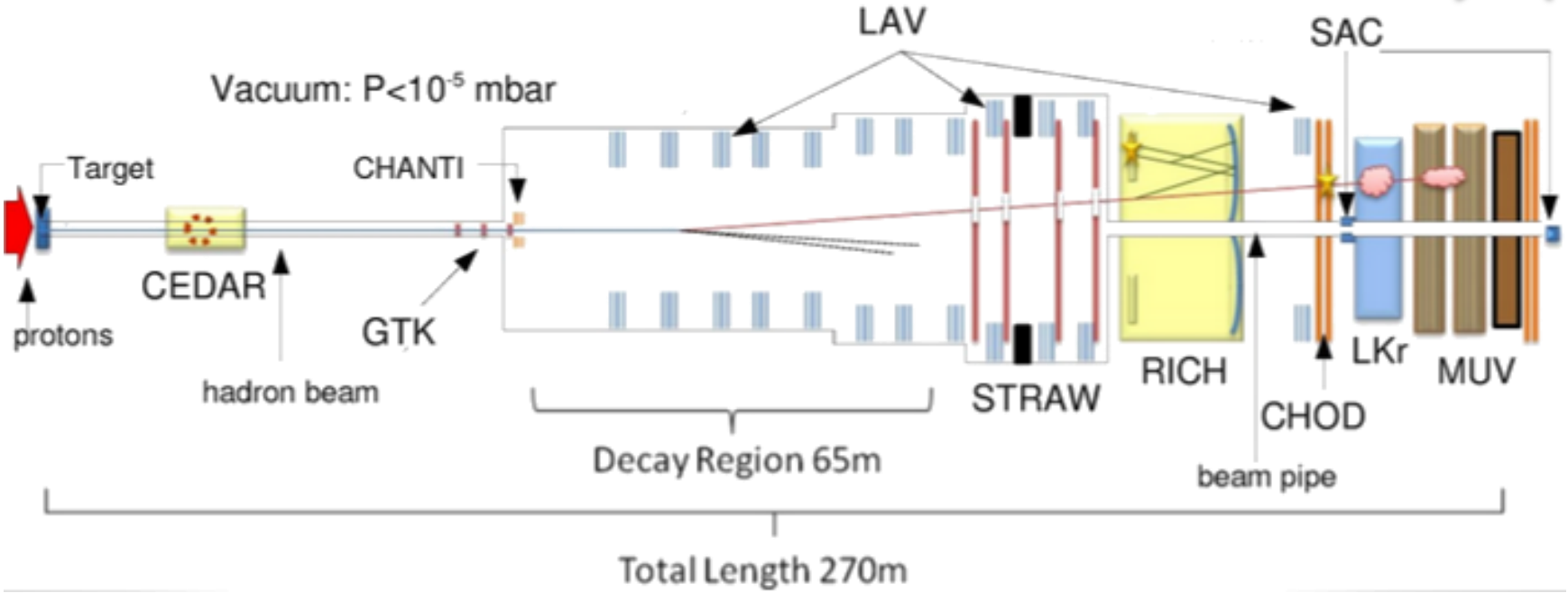


- Amplitude well predicted in SM (measurement of V_{td}) [see E.Stamou]
- Residual error in the BR due to parametric uncertainties (mainly due to charm contributions): ~7%
- Alternative way to measure the Unitarity Triangle with smaller theoretical uncertainty



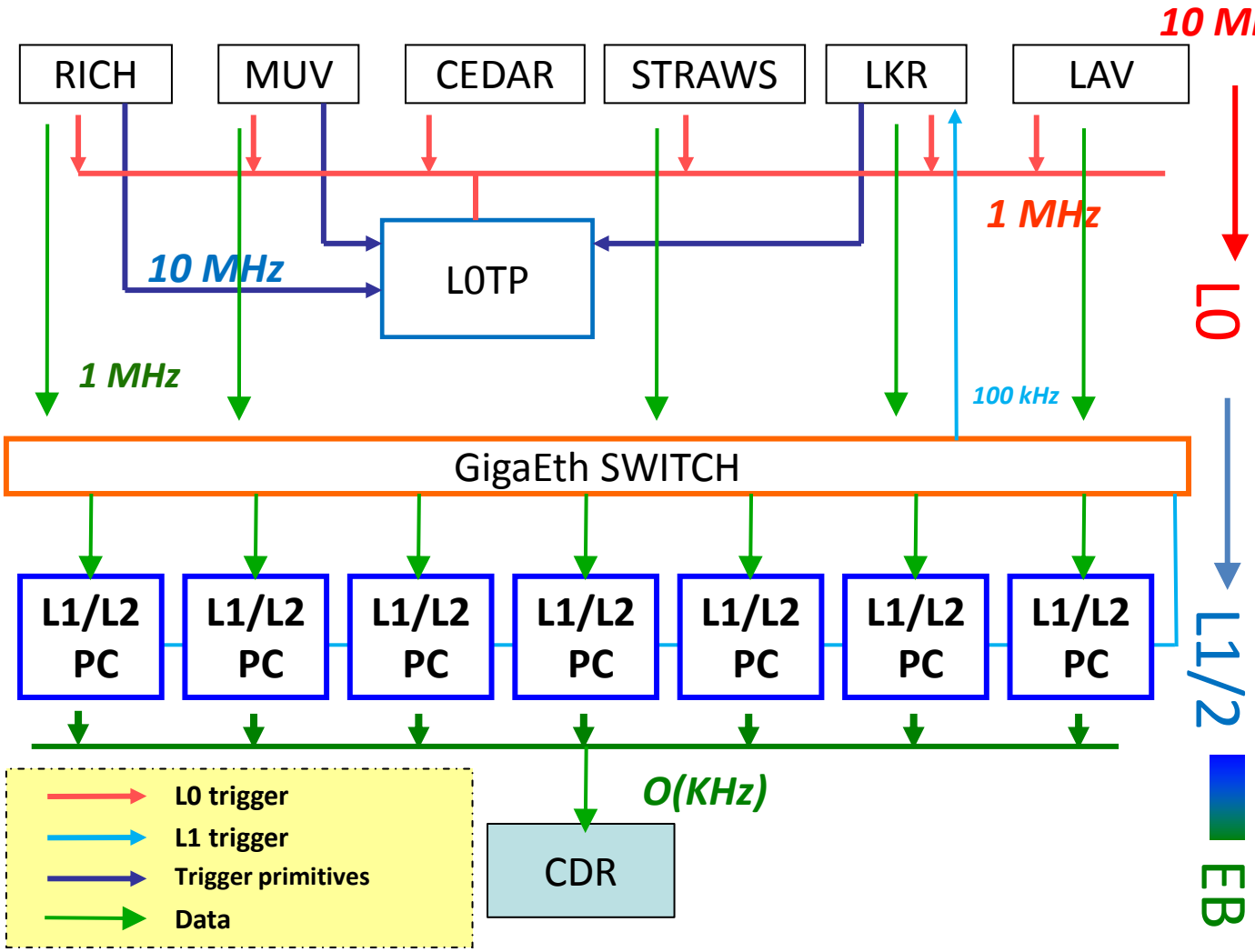
	G_{SD}/G	Irr. theory err.	BR x 10^{-11}
$K_L \rightarrow \pi\nu\nu$	>99%	1%	3
$K^+ \rightarrow \pi^+\nu\nu$	88%	3%	8
$K_L \rightarrow \pi^0 e^+ e^-$	38%	15%	3.5
$K_L \rightarrow \pi^0 \mu^+ \mu^-$	28%	30%	1.5

Experimental Technique



- Kaon decay in-flight from an unseparated 75 GeV/c hadron beam, produced with 400 GeV/c protons from SPS on a fixed berilium target
- ~800 MHz hadron beam with ~6% kaons
- The pion decay products in the beam remain in the beam pipe
- **Goal:** measurement of **O(100)** events in two years of data taking with % level of systematics
- Present result (E787+E949): 7 events, total error of ~65%.

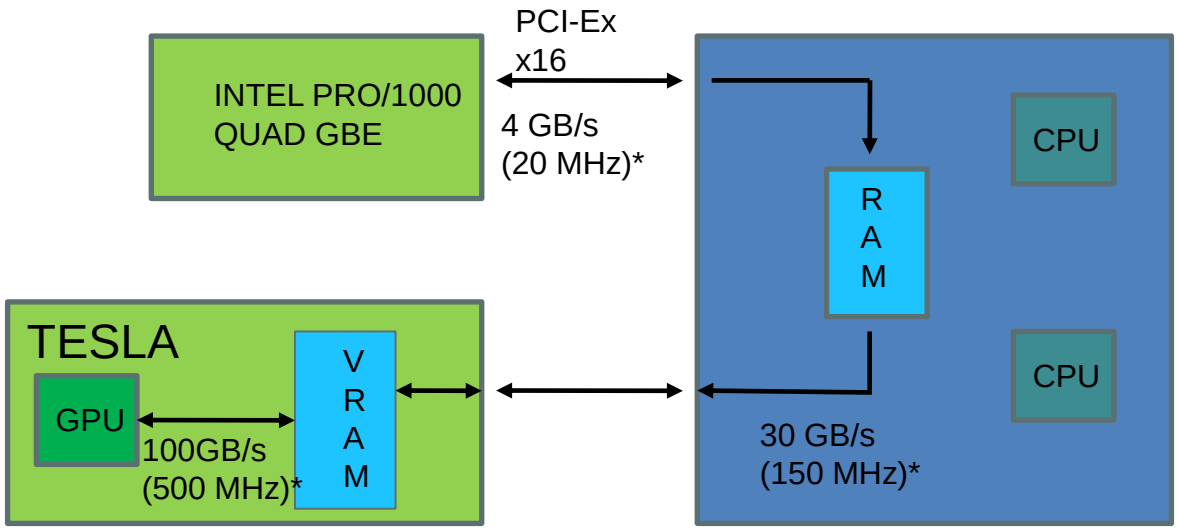
NA62 Trigger



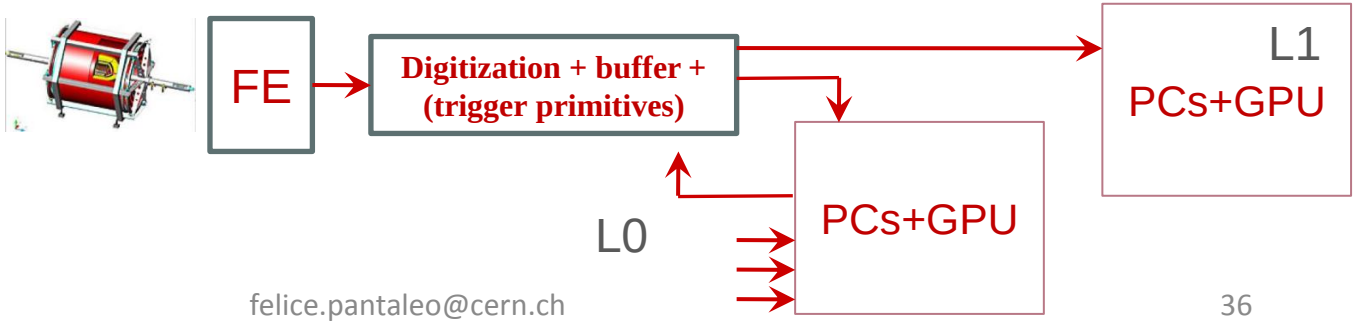
L0: Hardware synchronous level. 10 MHz to 1 MHz. Max latency 1 ms.
L1: Software level. "Single detector". 1 MHz to 100 kHz
L2: Software level. "Complete information level". 100 kHz to few kHz.

GPU as a Level 0 Trigger

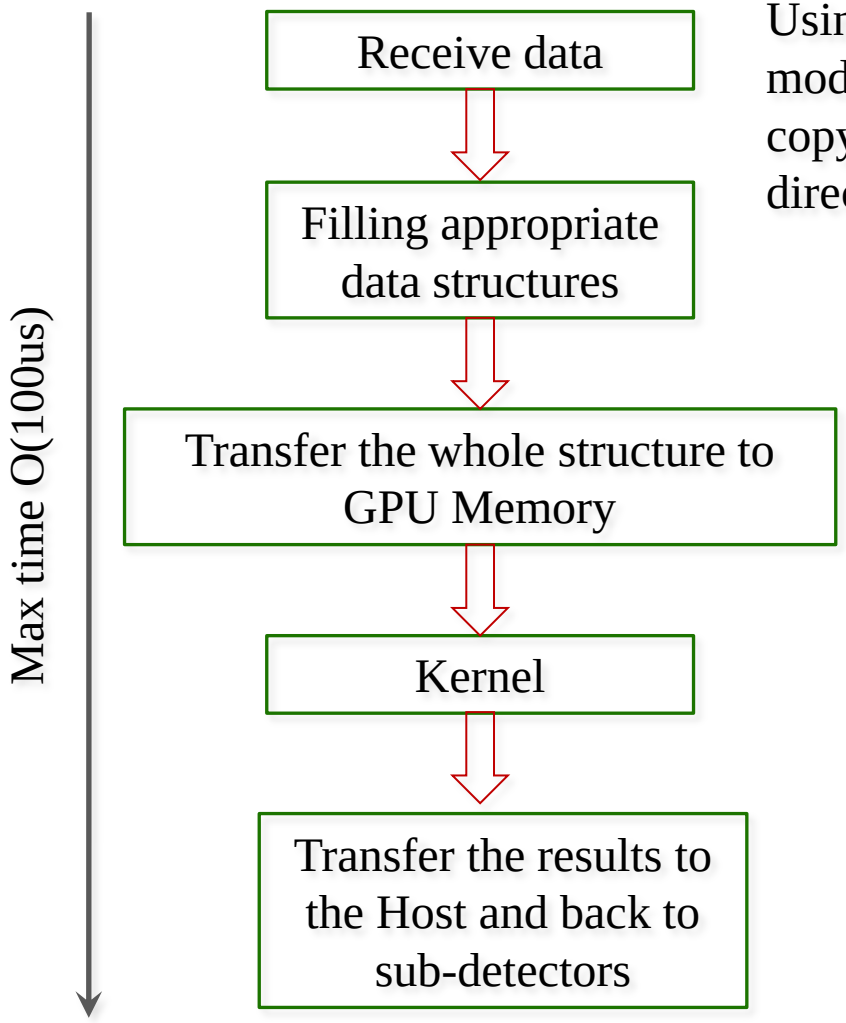
- The idea: exploit GPUs to perform high quality analysis at trigger level
- GPU architecture: massive parallel processor SIMD
- "Easy" at L1/2, challenging at L0
- Real benefits: increase the physics potential of the experiment at very low cost!
- Profit from continuative developments in technology for free (Video Games,…)



"quasi-triggerless" with GPUs



Data Flow

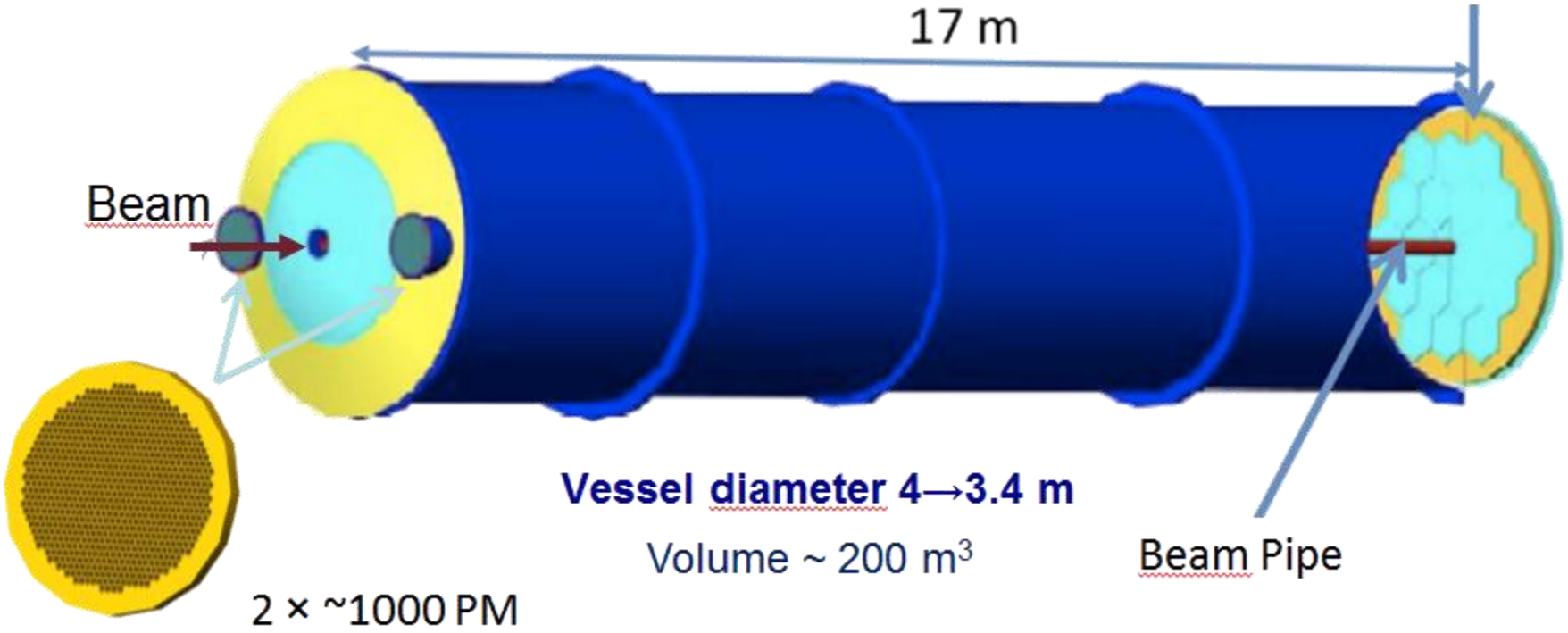


Using a "zero copy" intel modded driver, we are able to copy data from the NIC directly to the userland

exploiting concurrency

NA62 RICH Level0 Trigger

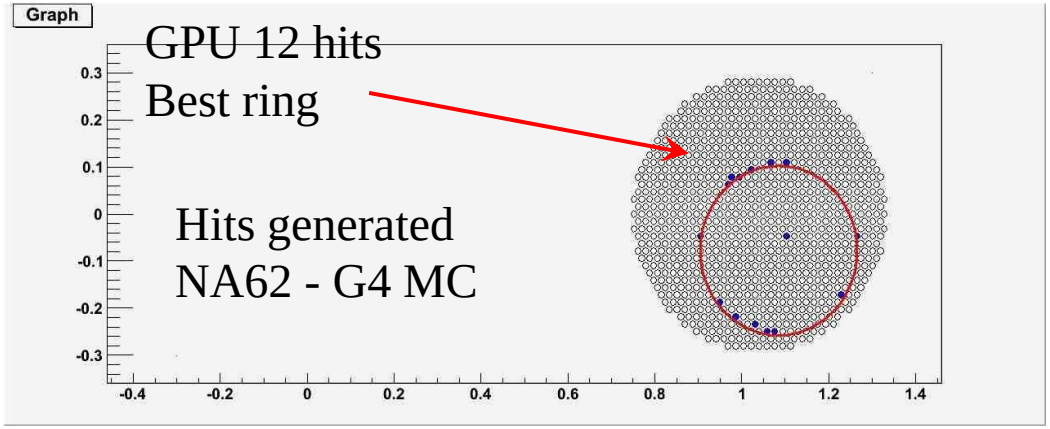
RICH

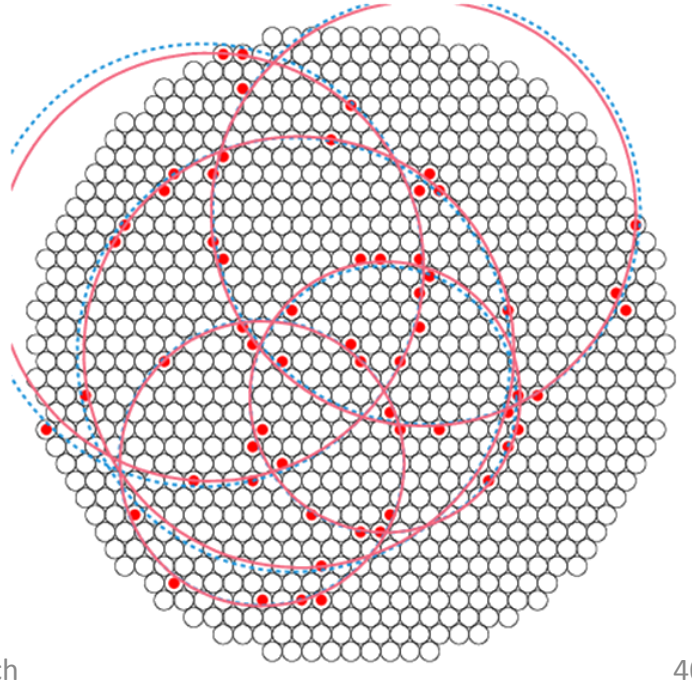


- ~17 m RICH
- 1 atm Neon
- Light focused by two mirrors on two spots equipped with ~1000 PMs each (pixel 18 mm)
- 3s p-m separation in 15-35 GeV/c, ~18 hits per ring in average
- ~100 ps time resolution, ~10 MHz events rate
- Time reference for trigger

Ring Reconstruction

- Natively built for pattern recognition problems
- First attempt: ring reconstruction in RICH detector.





It's a **pilot project**, very promising R&D!

Crawford Algorithm

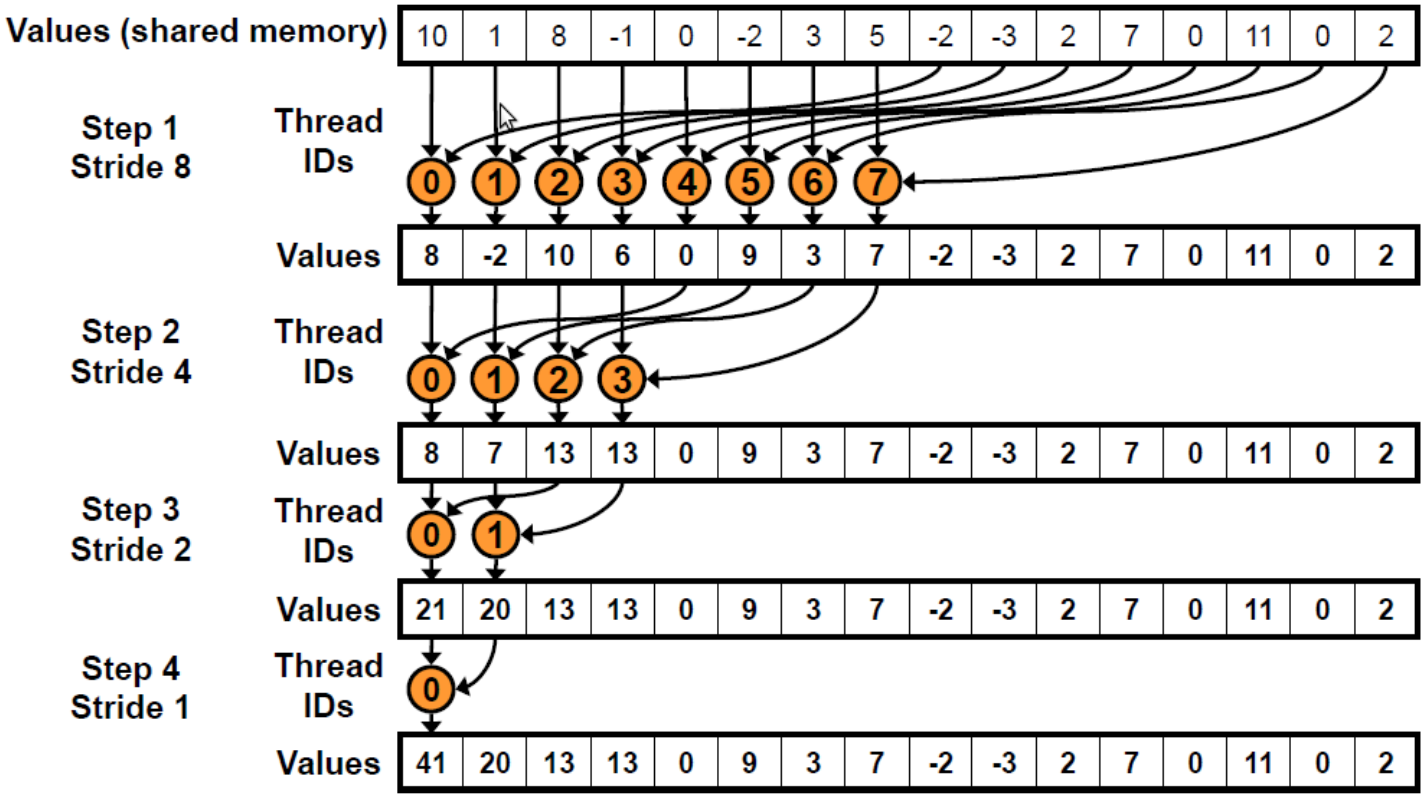
Consider a circle of radius R, centered in $(x_0, y_0)$and a list of points (x_i, y_i).
The following relations exist:

$$x_0{}^2 + y_0{}^2 - R^2 = \frac{1}{N}\{2x_0 \sum x_i + 2y_0 \sum y_i - \\ - \sum x_i^2 - \sum y_i^2\}. \qquad (1)$$

$$x_0\{\sum x_i^2 - \frac{(\sum x_i)^2}{N}\} + y_0\{\sum x_i y_i - \\ - \frac{\sum x_i \sum y_i}{N}\} = \frac{1}{2}\{\sum x_i^3 + \sum x_i y_i^2 \\ - \sum x_i \frac{\sum x_i^2 + \sum y_i^2}{N}\}, \qquad (2)$$

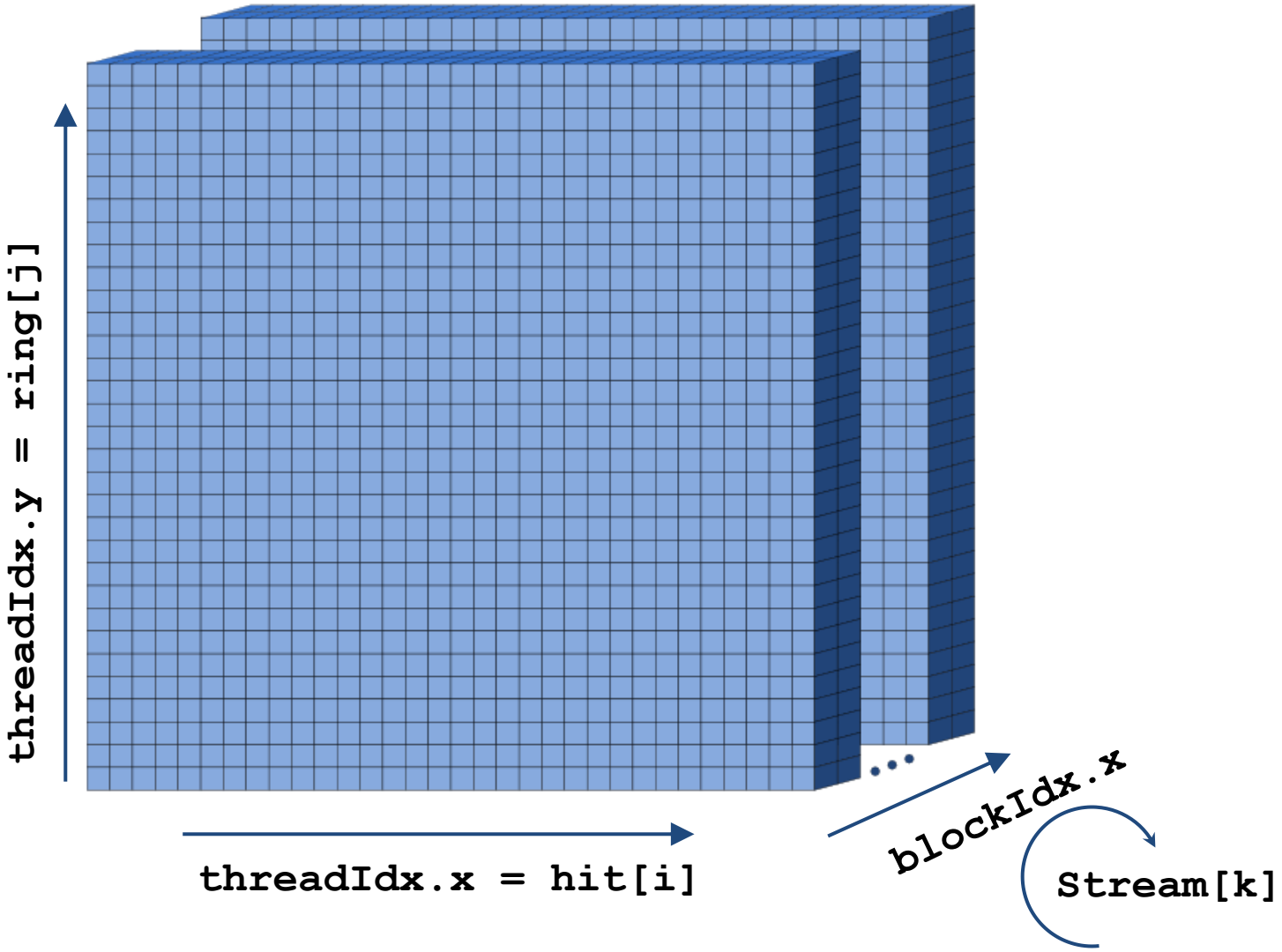
$$x_0\{\sum x_i y_i^2 - \frac{\sum x_i \sum y_i}{N}\} + y_0\{\sum y_i^2 - \\ - \frac{\sum y_i^2}{N}\} = \frac{1}{2}\{\sum x_i^2 y_i + \sum y_i^3 - \\ - \sum y_i \frac{\sum x_i^2 + \sum y_i^2}{N}\}. \qquad (3)$$

Reduction



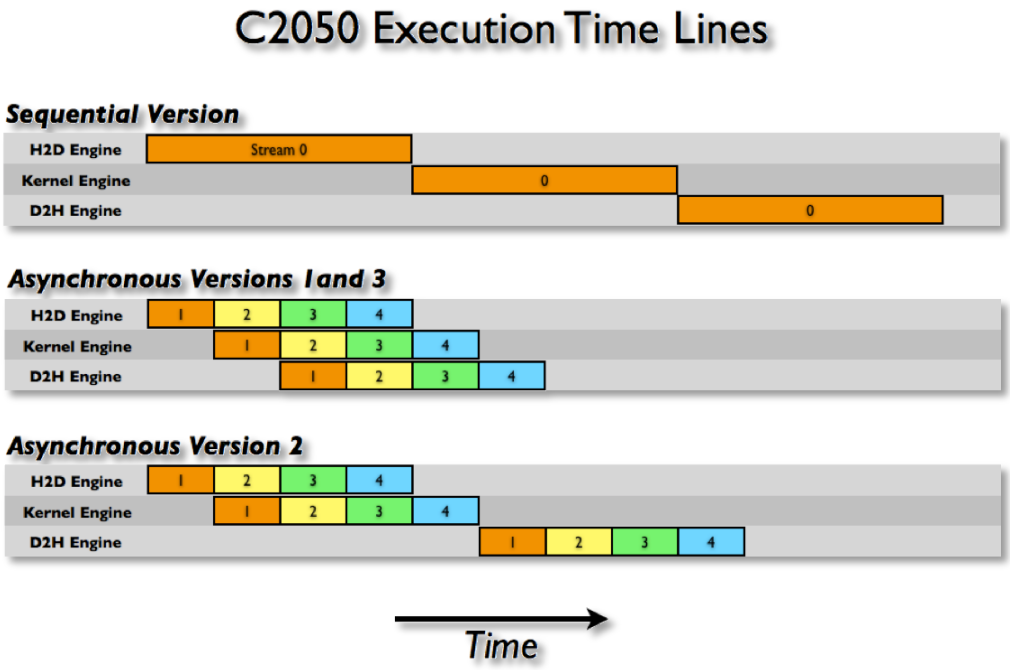
- One reduction kernel is called per block, giving an array of results (one for each event)
- Must use sequential addressing instead of interleaved addressing to avoid Shared Memory bank conflicts
- Time complexity is O(logN), cost is O(N*logN): not cost efficient
- Brent's theorem (algorithm cascading) suggests O(N/logN) threads:
  - Each thread does O(logN) sequential work
  - All O(N/logN) threads cooperate for O(logN) steps
  - New cost = O(N/logN * logN) = O(N)

GPU grid organization



Stream Scheduler

- Exploit the instruction-level parallelism (i.e. pipelining streams)
- This is usually done by interlacing one stream instructions with another stream ones
- This cannot be done in real-time without the introduction of other **unknown** latencies
- CPU thread-level parallelism is the solution



NA62 RICH Tests

Hardware configuration (1/2)

First Machine

- GPU: NVIDIA Tesla C2050
  - 448 CUDA cores @ 1.15GHz
  - 3GB GDDR5 ECC @ 1.5GHz
  - CUDA CC 2.0 (Fermi Architecture)
  - PCIe 2.0 (effective bandwidth up to ~5GB/s)
  - CUDA Runtime v4.2, driver v295.20 (Feb '12)

- CPU: Intel® Xeon® Processor E5630 (released in Q1'10)
  - 2 CPUs, 8 physical cores (16 HW-threads)

- SLC6, GNU C compiler v4.6.2

Hardware configuration (2/2)

Second Machine

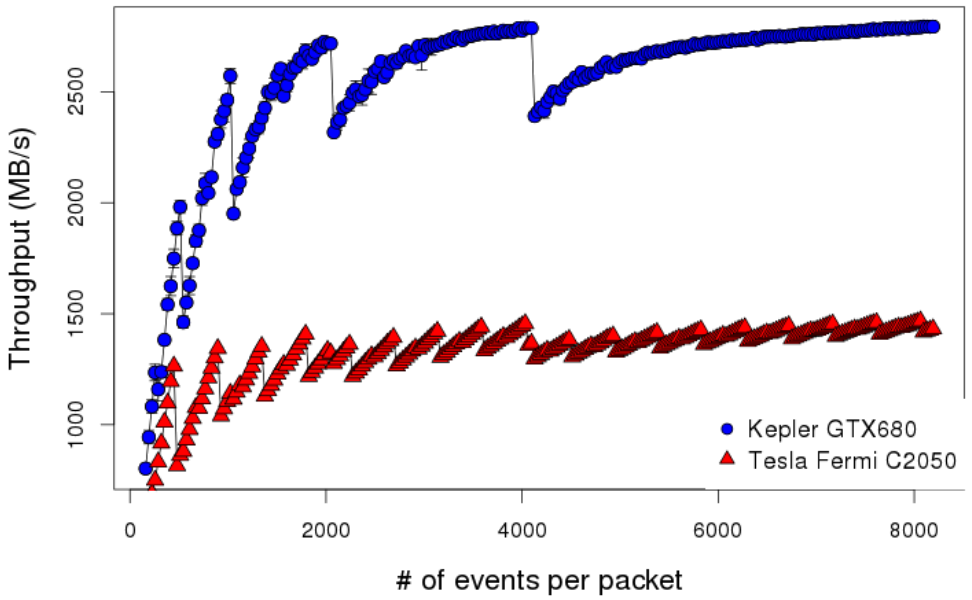
- GPU: NVIDIA GTX680
  - 1536 CUDA cores @ 1.01GHz
  - 2GB GDDR5 ECC @ 1.5GHz
  - CUDA CC 3.0 (Kepler Architecture)
  - PCIe 3.0 (effective bandwidth up to ~11GB/s)
  - CUDA Runtime v4.2, driver v295.20 (Feb '12)
- CPU: Intel® Ivy Bridge Processor i7-3770 (released in Q2 '12)
  - 1 CPUs, 4 physical cores (8 hw-threads) @3.4GHz
- Fedora 17, GNU C compiler v4.6.2

Results - Throughput

The throughput behaviour for a varying number of events inside a packet is a typical many-core device behaviour:

- constant time to process a varying number of events, activating more SMs as the packet size increases
- discrete oscillations due to the discrete nature of the GPU
- saturation plateau (**1.4GB/s** and **2.7GB/s**)

The right choice of packet dimension is not unique.
It depends on the maximum latency we don't want to exceed and on the input rate of events.

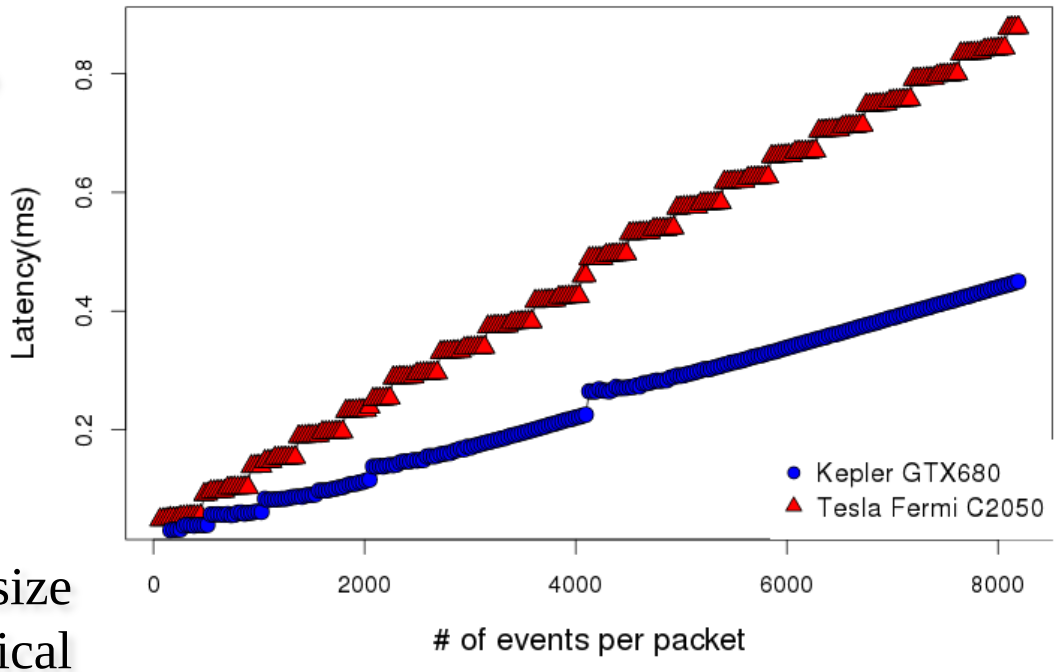


Considering that the maximum rate per sub-detector (@10MHz particles rate) for NA62 experiment is ~500MB/s, I would consider the throughput test **PASSED**

Results - Latency

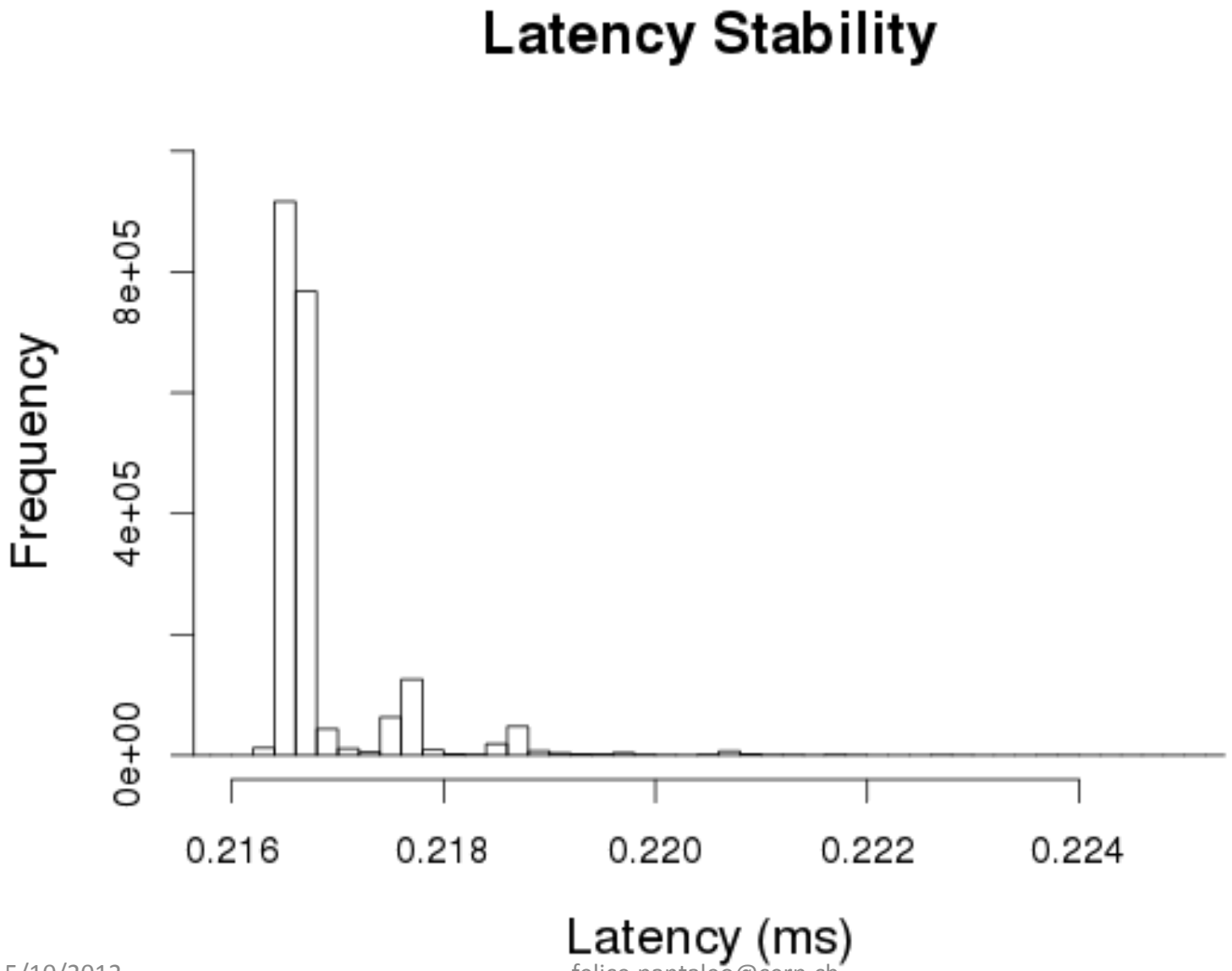
Latency pretty stable wrt event size.

- A lower number of event inside a package is better to achieve a low latency.
- A larger number of event guarantees a better performance and a lower overhead.



The choice of the packet size depends on the technical requirements.

Results - Latency Stability



NA62 CHOD Level0 Trigger

CHOD

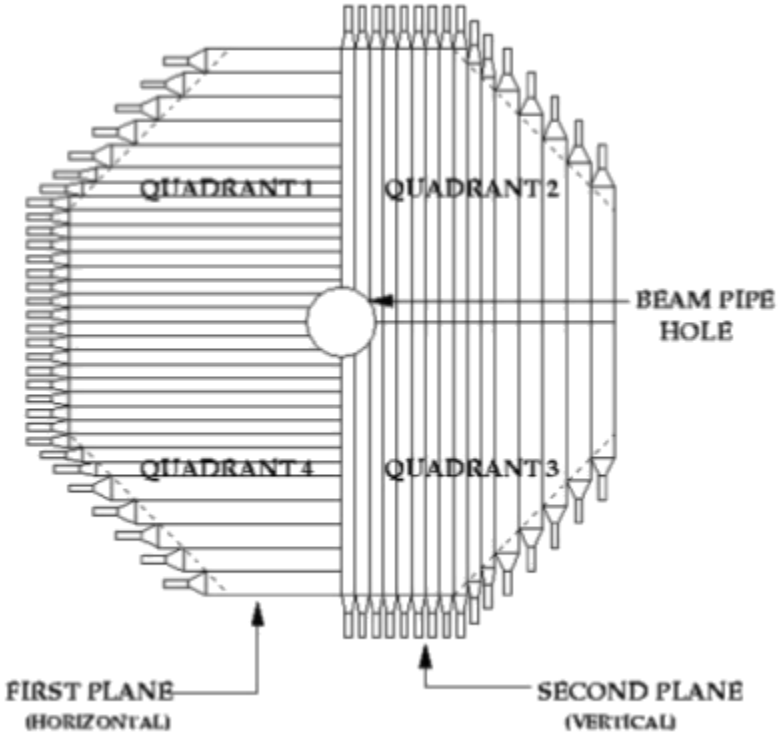
Opportunity to run test on a running sub-detector.

Aim: Applying time corrections in real-time can **improve time resolution.**

Divide et Impera:

- Real-Time-Ready data structures receiver implemented.
- Fast communication between Network Interface buffers and Host Memory (DNA driver) implemented
- A throughput/latency efficient kernel implemented

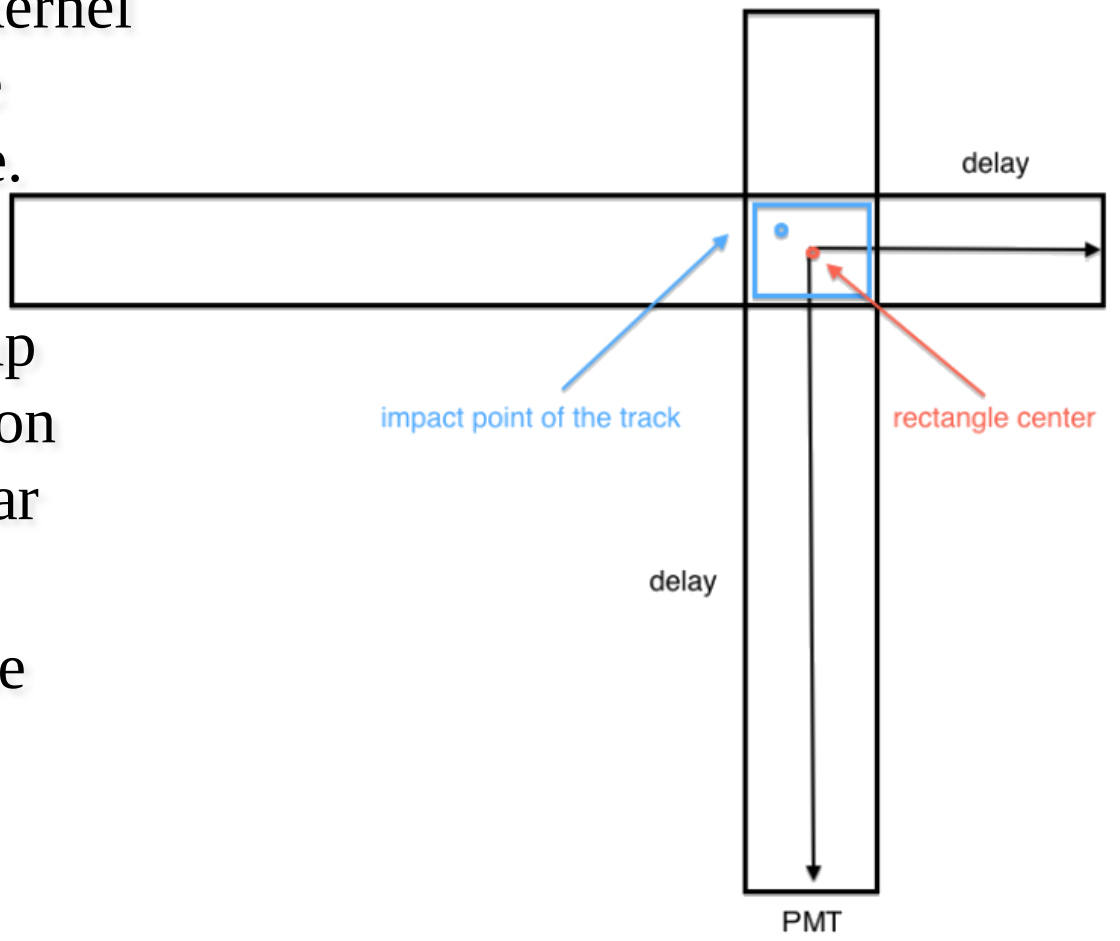
Integrate the single unit tests



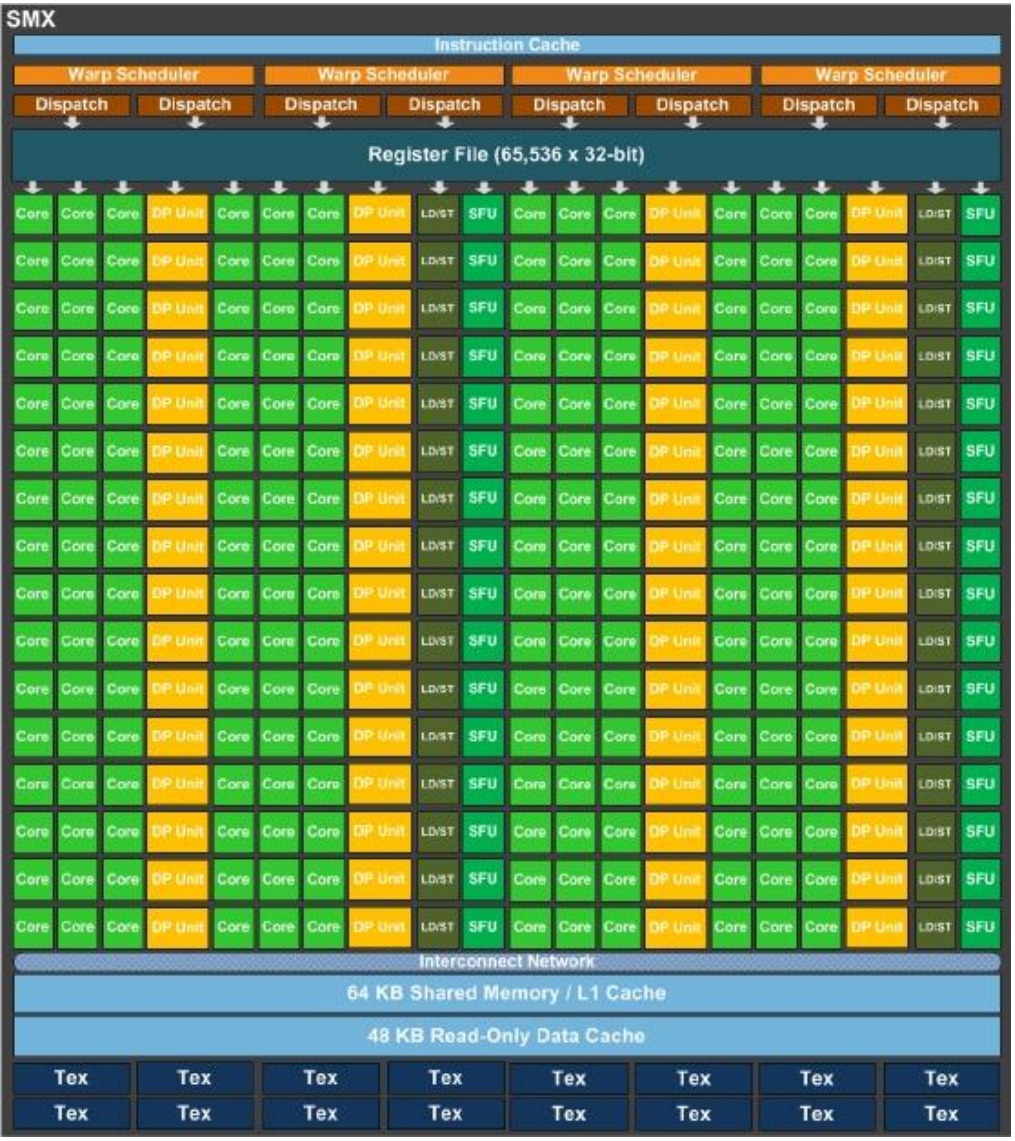
Due to the slabs length, all the times measured by PMTs need to be corrected.

CHOD – Kernel

- Assuming the conditions of the detector to be stable, the kernel is an increment by a value taken from a lookup table.
- Each element of the lookup table contains the correction specific of each rectangular zone.
- Each zone correction is the central point one.



CUDA Kepler Architecture



Investigation on which memory to use to store this matrix:

Global memory (read and write)

- Slow, but now with cache
- L1 cache designed for spatial re-usage, not temporal (similar to coalescing)
- It benefits if compiler detects that all threads load same value (*LDU* PTX ASM instruction, load uniform)

Texture memory

- Cache optimized for 2D spatial access pattern

Constant memory

- Slow, but with cache (8 kb)

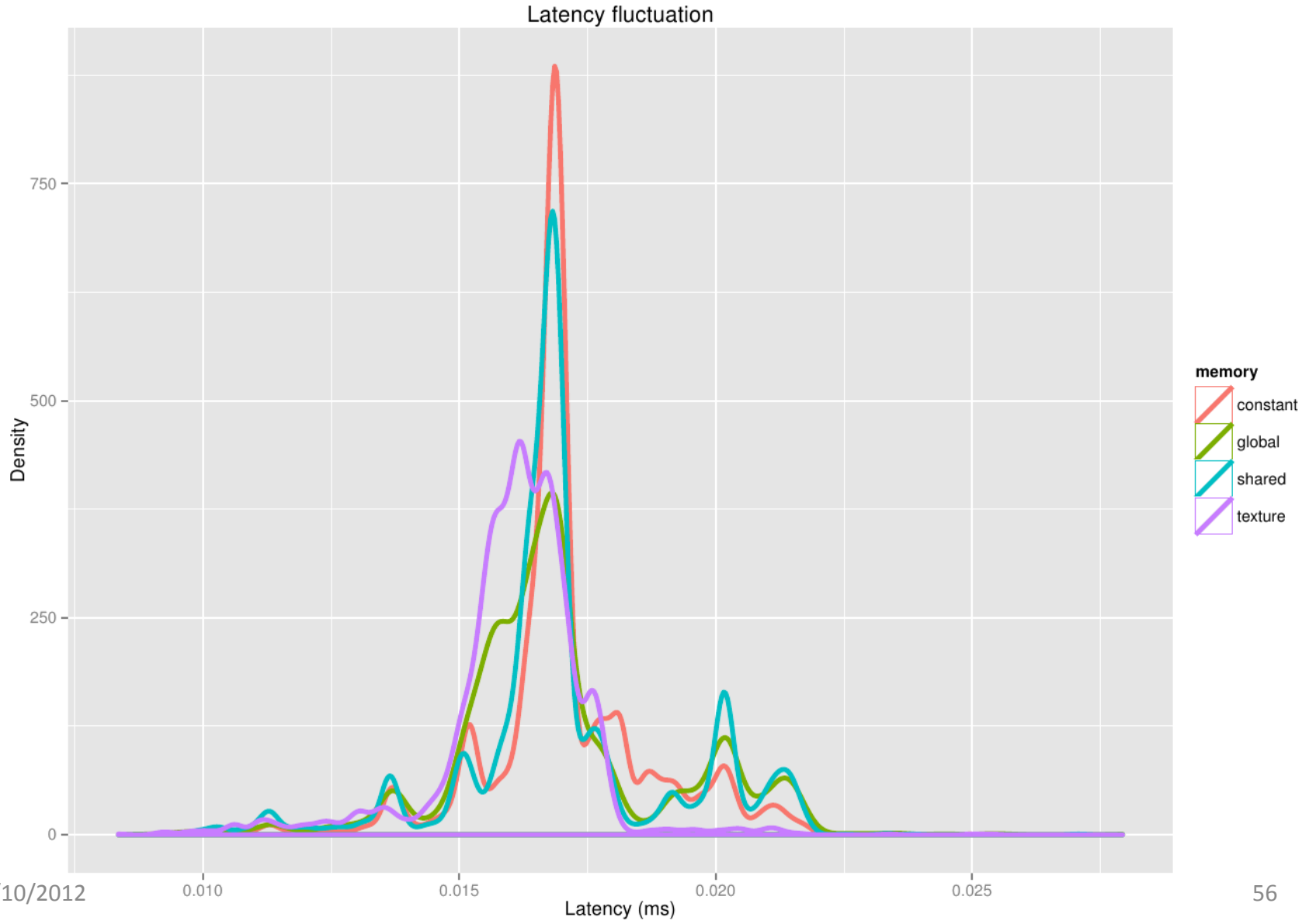
Shared memory (48kB per SMX)

- Fast, but slightly different rules for bank conflicts now

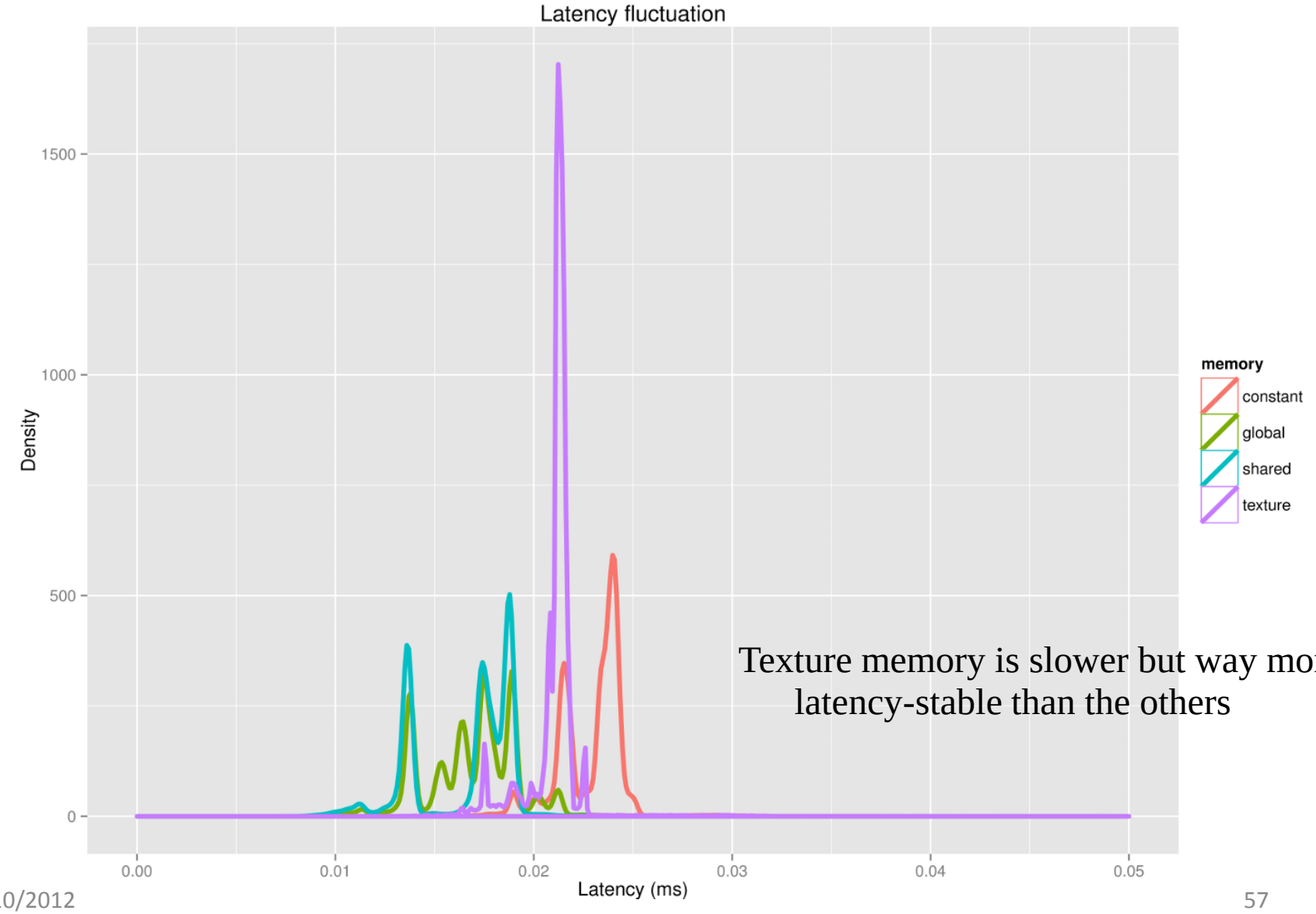
Registers (65536 32-bit registers per SMX)

NA62 CHOD Tests

Kernel 8x8

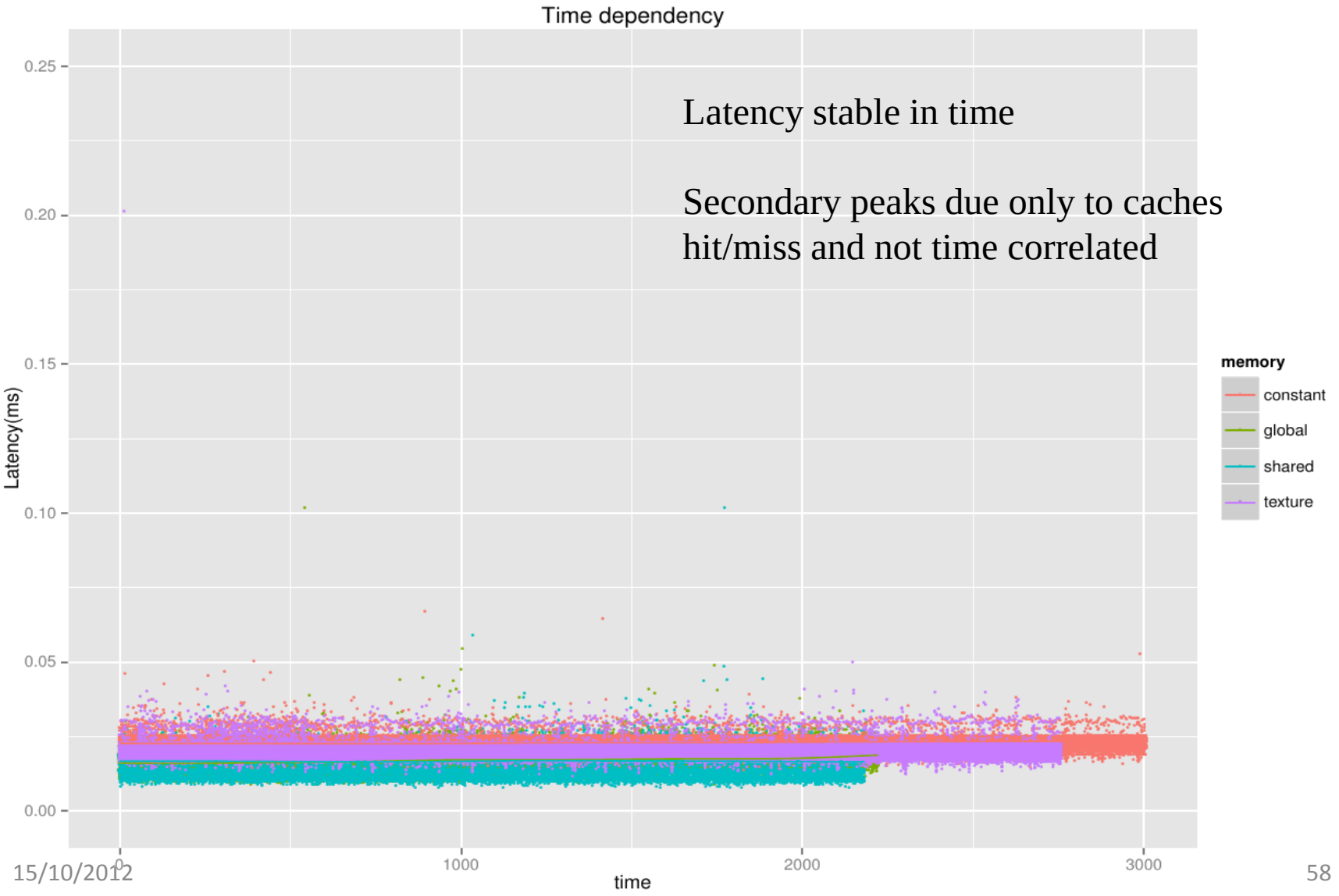

Latency fluctuation
Density
750
500
250
0
0.010
0.015
0.020
0.025
Latency (ms)
memory
constant
global
shared
texture

Kernel 64x64



Texture memory is slower but way more latency-stable than the others

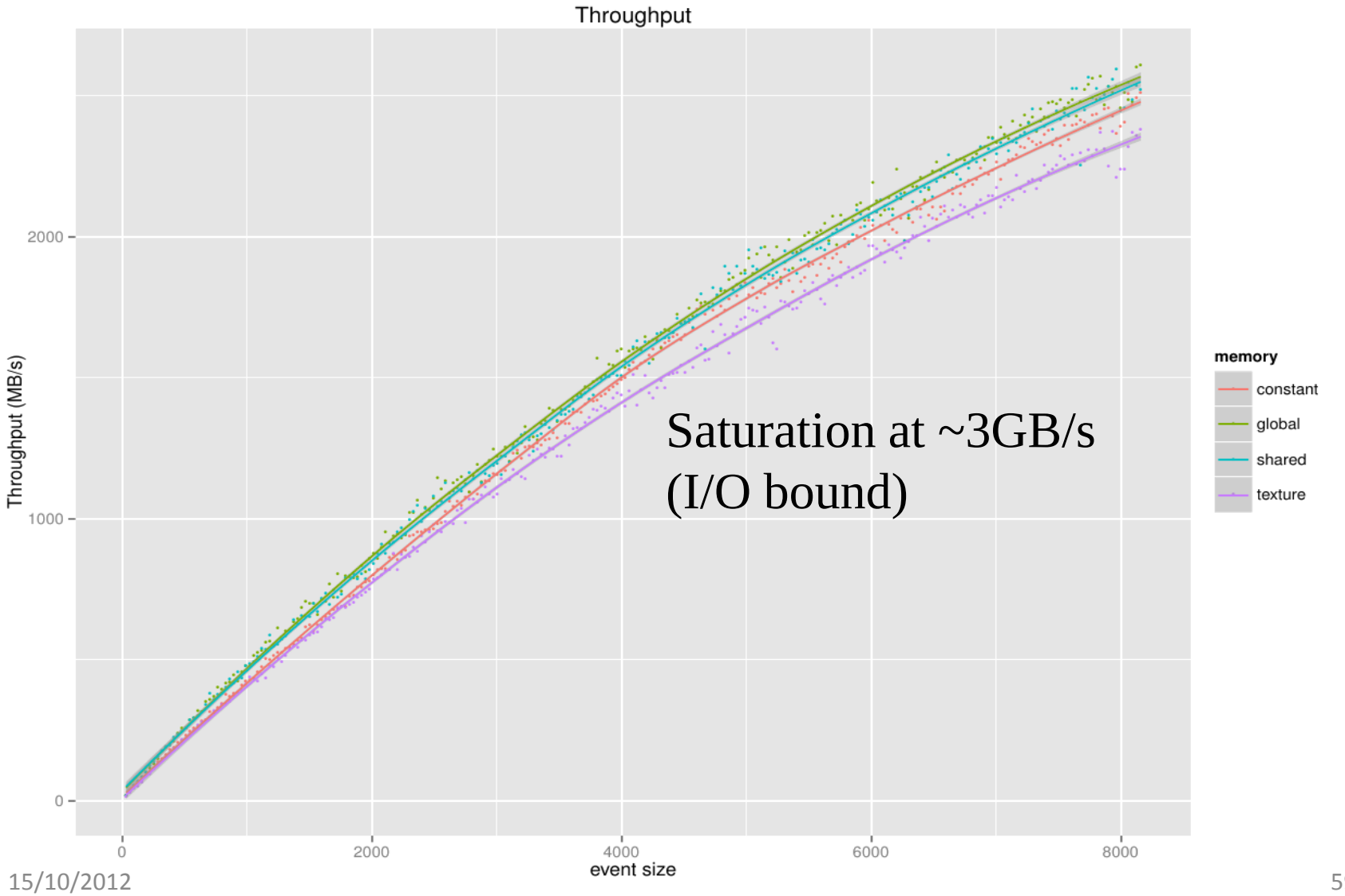
Time dependency



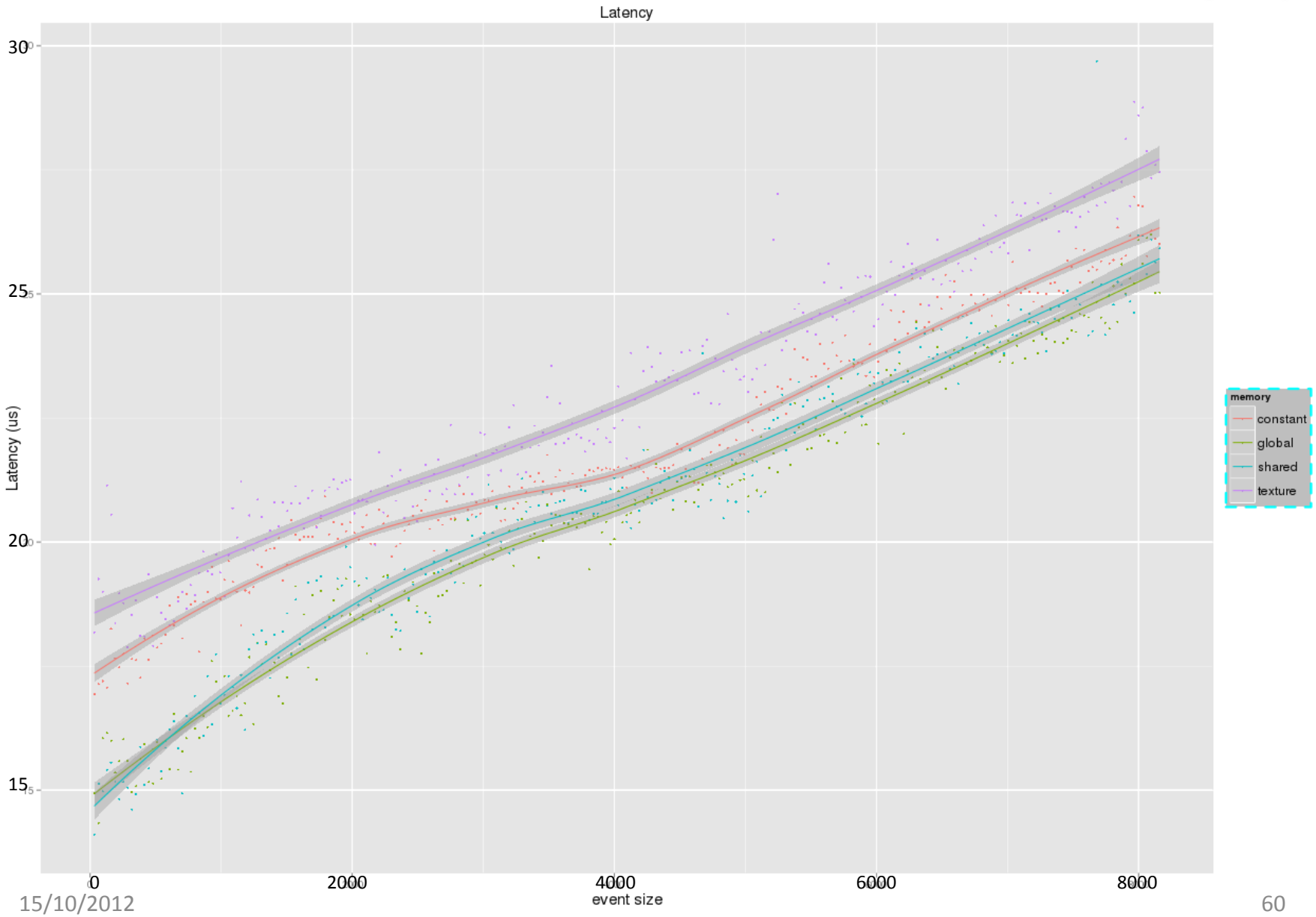
Latency stable in time

Secondary peaks due only to caches hit/miss and not time correlated

Throughput


Throughput
Throughput (MB/s)
2000
1000
0
0
2000
4000
6000
8000
event size
memory
constant
global
shared
texture
Saturation at ~3GB/s
(I/O bound)

Latency


Latency
Latency (us)
30
25
20
15
0
2000
4000
6000
8000
event size
memory
constant
global
shared
texture

Conclusion

- I will test a complete system in the first NA62 technical run in November.
- GPUs seem to represent a good opportunity, not only for analysis and simulation applications, but also for more “hardware” jobs.
- Replacing custom electronics with fully programmable processors to provide the maximum possible flexibility is a reality not so far in the future.

Conclusion

- GPU now mature to be used in HEP in particular in dedicated triggers.
- Already in use in HLT in ALICE for TPC reconstruction
- First test as Level0 trigger processor in NA62 next month

Questions?

Reference

Real-Time Use of GPUs in NA62 Experiment

F. Pantaleo et al., 13th International Workshop on Cellular Nanoscale Networks and their Applications (CERN-PH-EP-2012-260)

ALICE HLT TPC Tracking on GPUs

D. Rohr et al., CNNA2012

Many-core processors and GPU opportunities in Particle Detectors

X. Vilasís Cardona et al. CNNA2012