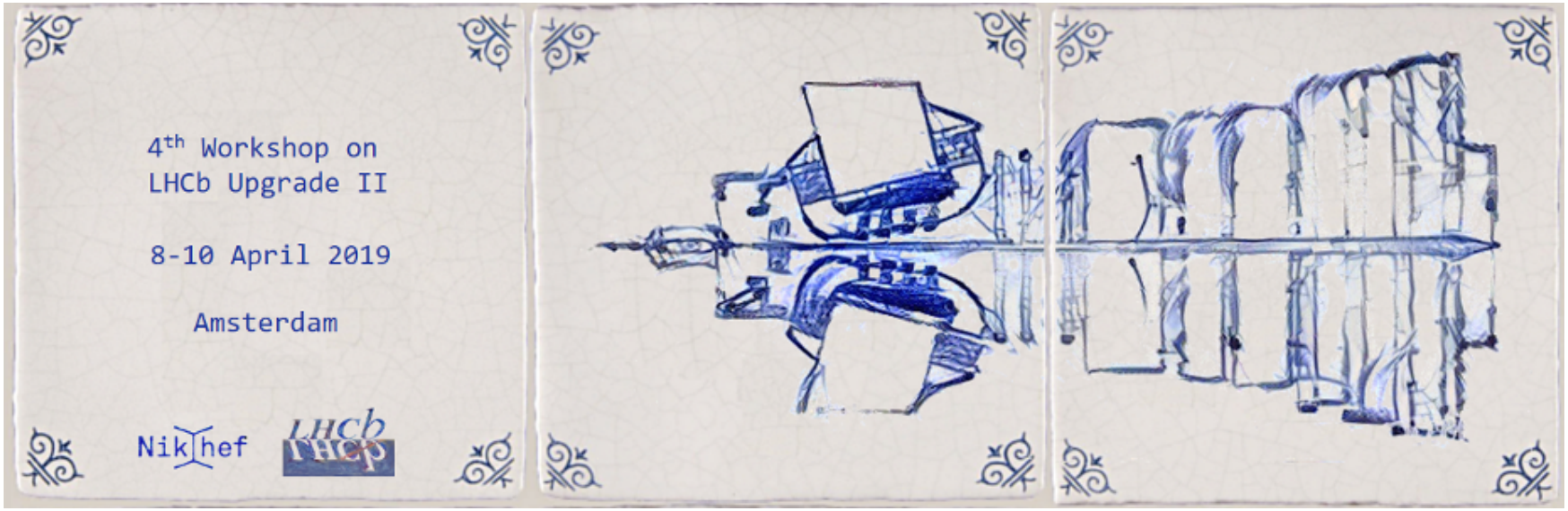
Tracking without CPUs

Giovanni Punzi
Università di Pisa & INFN



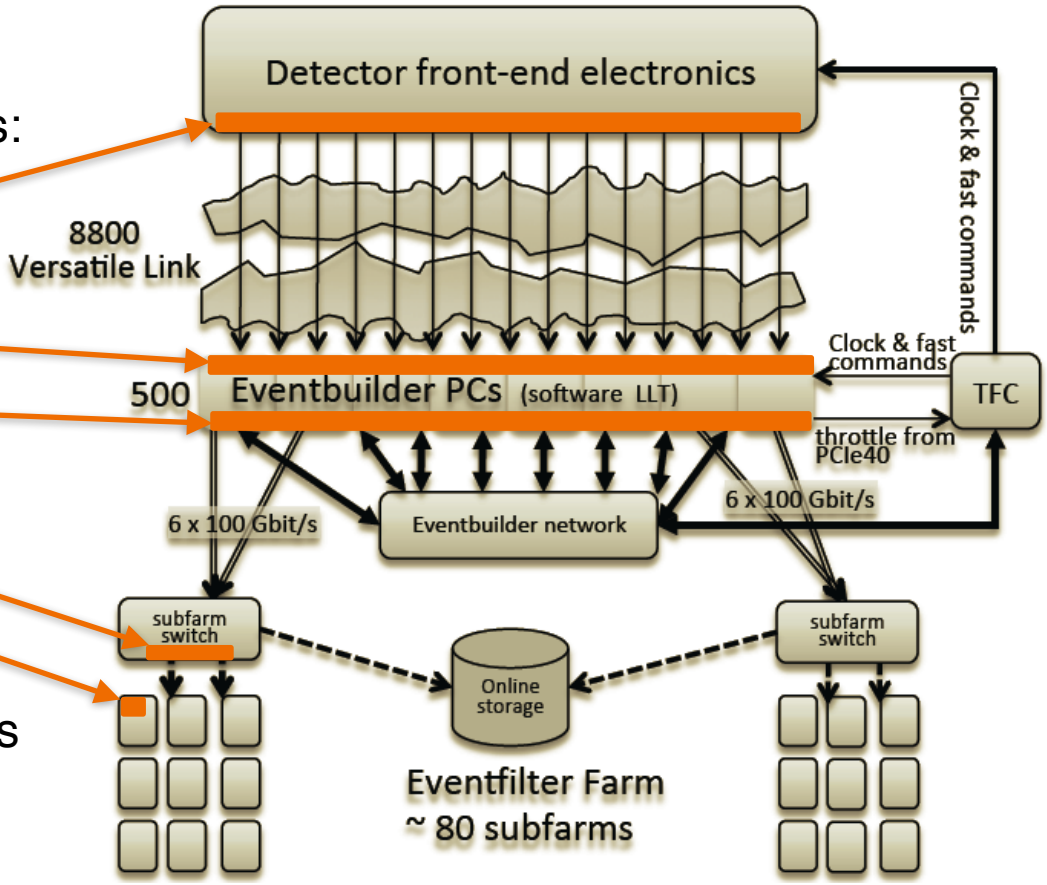
Why without CPUs ?

- Why is it worth considering to assist CPUs in tracking with other means?
  - Because Moore's slow is slowing down, and we don't want to do the same.
  - Flavor physics in particular involves huge data: LHCb, with its approach of full readout and processing of each crossing, is at the forefront of a revolution that GPD will have to follow in future.
- Continued progress of high-intensity HEP experiments requires a new approach:
  - Use *specialized*, rather than *general-purpose*, computing architectures to get greater efficiency
  - Pushing processing down to the earliest possible levels of processing ("embedded reconstruction"), to save bandwidth [see GP talks at TTFU and 1st FCC workshop for a detailed discussion of Real-Time analysis, embedded reconstruction, and use of timing, in future experiments].

- For Run-4 (**Upgrade I-b)**, with no luminosity increase, we can foresee some modest-cost, current-tech improvements, targeted to the weakest spots of Run-3
- For Run-5+ (**Upgrade-II)** it is conceivable that we will **want** or **need** to reconstruct most of the events before the HLT -- leaving to the HLT to focus on trigger selection (needing the full flexibility of CPUs)
  - Huge luminosity and new complex detectors, possibly with timing, will constrain DP enormously

Where could we do (part of) track reconstruction?

- Several (non exclusive) possibilities:
  1. On the Front-End
  2. Pre-Build
  3. Post-Build
  4. Within the EFF
  5. Within CPUs.
- At each stage, there may be options to reduce event size OR rate



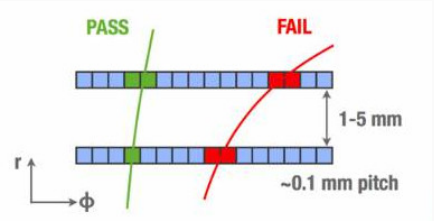
1. On the Front End

- Being on the detector entails several limitations:
  - Space, locality, accessibility, radiation...
  - The main point of full event readout was to break free from limitations...
- Nevertheless, with technology progressing it may be possibile to do some sufficiently intelligent data (pre)processing in the FE that is worthwhile (consider zero-suppression as an example from today).
- Cluster finding, for instance, would certainly be helpful (see later)
- Basically it only makes sense if can reduce data flow and save $$
  - Most likely by data reduction/compression rather than discarding events.
- Well known example in CMS Phase-II, with doublet matching as a way to filter high-pt tracks from loopers, hugely reducing the data flow into the tracking trigger
- Possibility being studied for LHCB as well, although gains must be understood: not obvious in a different geometry
- Current under study at LHCb:
  - Mighty tracker doublet-matching
  - Magnet Side Stations 'stubs'
- Timing info processing also a clear possibility in principle, but still the gain must be proved [see Vava's "fine print" at recent U2-VELO workshop]
- Candidate technologies: ASICs or FPGAs

Current work: tracklets in the Mighty Tracker

Track stubs

- Investigating the possibility of constructing tracklets by correlating doublet of layers
  - Possibility of making doublets in ASIC or FPGA, potentially reducing data size
  - Possibly simplifying pattern recognition by sending the direction of track stubs as well as hit information
  - Not clear that this would give any data reduction or what speed benefit it would give yet
- Concept based on the CMS upgrade

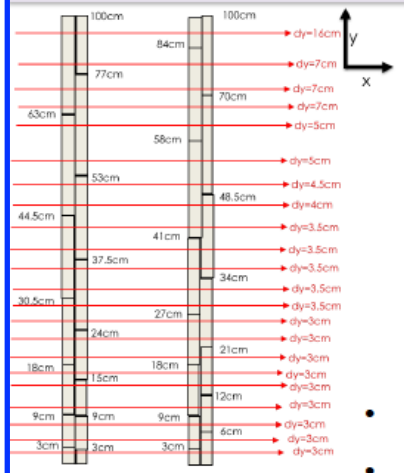


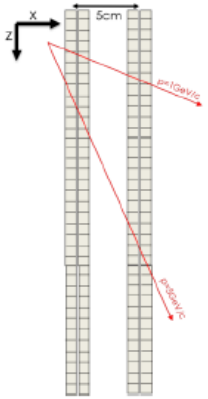
Dónal Murray donal.murray@cern.ch

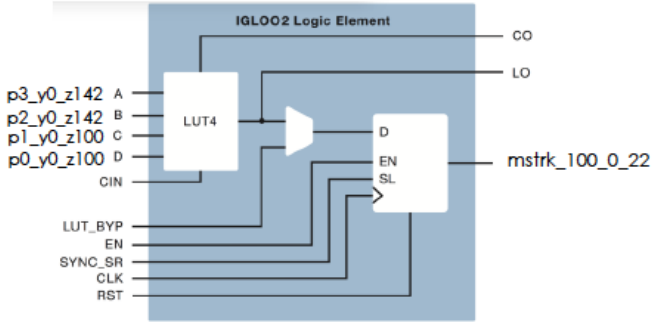
[Murray, General RTA meeting 29/3/19]

Current work: stubs in Magnet Side Stations

First thoughts about logic







- 22 y segments per z column
- z0-z1 combinations depends on the distance between planes
- 40 z0-z1 combinations for 5cm distance
- 600 z columns per side
- ~500k combinations per side (18 bits ID in the data package)
- 5 IGLOO2 FPGAs (150K LE each) per side might handle it ?
- Can also stream track packages with 2bit ADC information from each bar associated to the track for further ghost track rejection

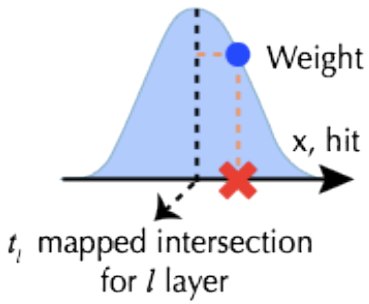
3/29/2019 Magnet Station Overview 11

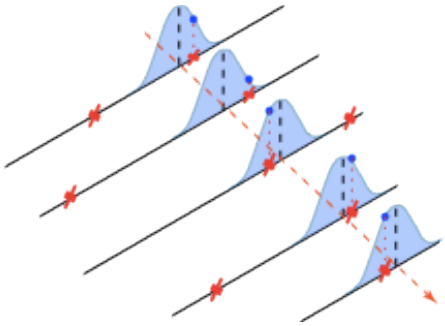
[Da Silva, General RTA meeting 29/3/19]

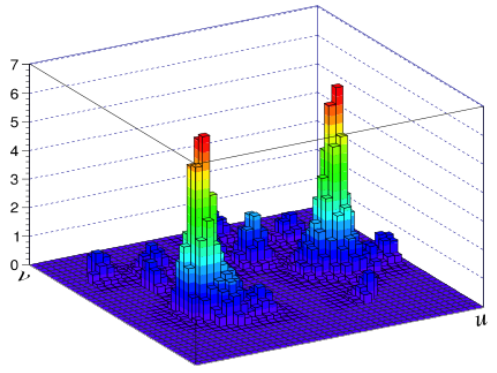
2. Pre-Build level (close to readout units):

- Much easier to establish at least local connections within tracking sections
- Can do data reduction/compression or possibly event selection already.
- Advantages of data reduction at this level is sparing the Event Builder a lot of extra work
- Also some practical advantages:
  - Easier to upgrade/stage/modularize
  - Avoid need for "unpacking" the event to access the portions of interest
  - Can use just the amount of electronics needed for the job and no more
- FPGAs the ideal technology for this: flexible commercial upgradable hardware, with huge bandwidths (TB/chip), low power, low latencies, and Moore's law still running strong.
  - FPGA shown to be the most effective solution for this class of problems in high-end high-speed applications like avionics, military, medical imaging...
  - ASICs still a possibility, but lack of flexibility make it less attractive for this job.
- Possibility to handle several pieces of the tracking (separately or together), and sufficiently mature technologies to be applied starting in Run-3
- A lot of work going on in RTA WP6 [see recent dedicated WP6 meeting]
- CMS-Imperial C. group built a custom FPGA board for this purpose [see McCann's talk at U2-VELO workshop for recent thoughts on LHCB use of the same board]

Current work: the RETINA project



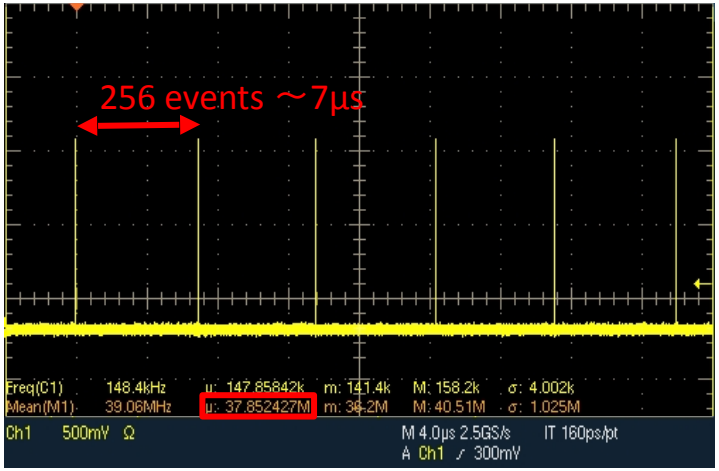




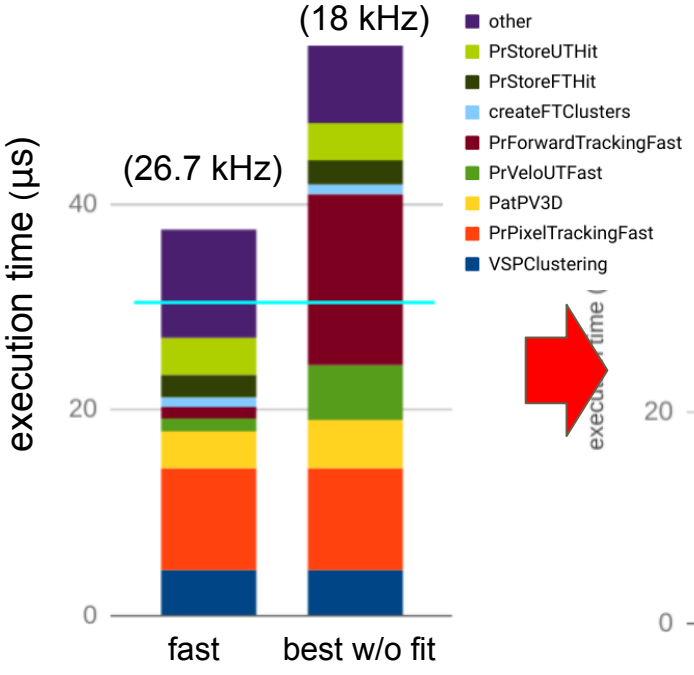
- RETINA is an *architecture* for very fast reconstruction, addressing real-time reconstruction by ***extreme parallelism*** and ***high connectivity,*** imitating some features of natural neural systems.
- Its main features are:
  - Ultra-low latency, allowing reconstruction to happen transparently "on the fly" while data is being readout, with no buffering overhead. Ideal for Pre-Build processing.
  - Produces results similar to a Hough-transform
  - Implemented by low-level software programming to exploit of every available FPGA gate, leading to ~maximum theoretically achievable computational efficiency
  - Bitwise simulation of every piece is available
- LHCb-PUB-2014-026 already described in detail a monolithic FPGA-based system based on this new custom architecture to perform VELO+UT tracking at 40 MHz.
- More recent developments based on array of commercial FPGA cards connected by optical network
- Various applications today at various stages of development (modular, could in principle handle HLT1)

Status: VELO clustering+tracking already mature in Run3

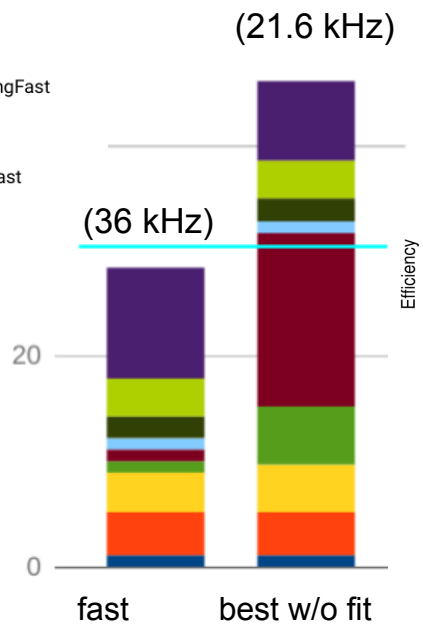
- Clustering already proved in hardware at 38 MHz, modest hardware needs -> being integrated in VELO PCIe40 [see ACAT talk]
- Pattern recognition of VELO tracks, close to offline efficiency, still being optimized a bit
- Both save significant fraction of data rate into the EB.
- Fully emulated in official software
- Real experience in data taking will be interesting, but seems a safe bet for Run4, and naturally extendable to Run5
  (maybe clustering -> FE if shown to be advantageous ?)



Official sequence, CPU only

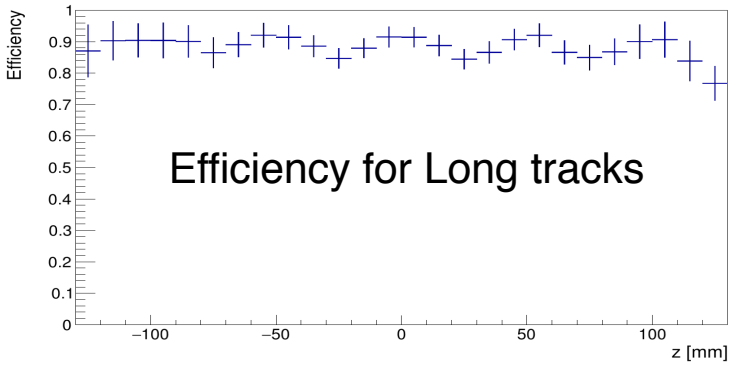


Clustering+PattRec on FPGA



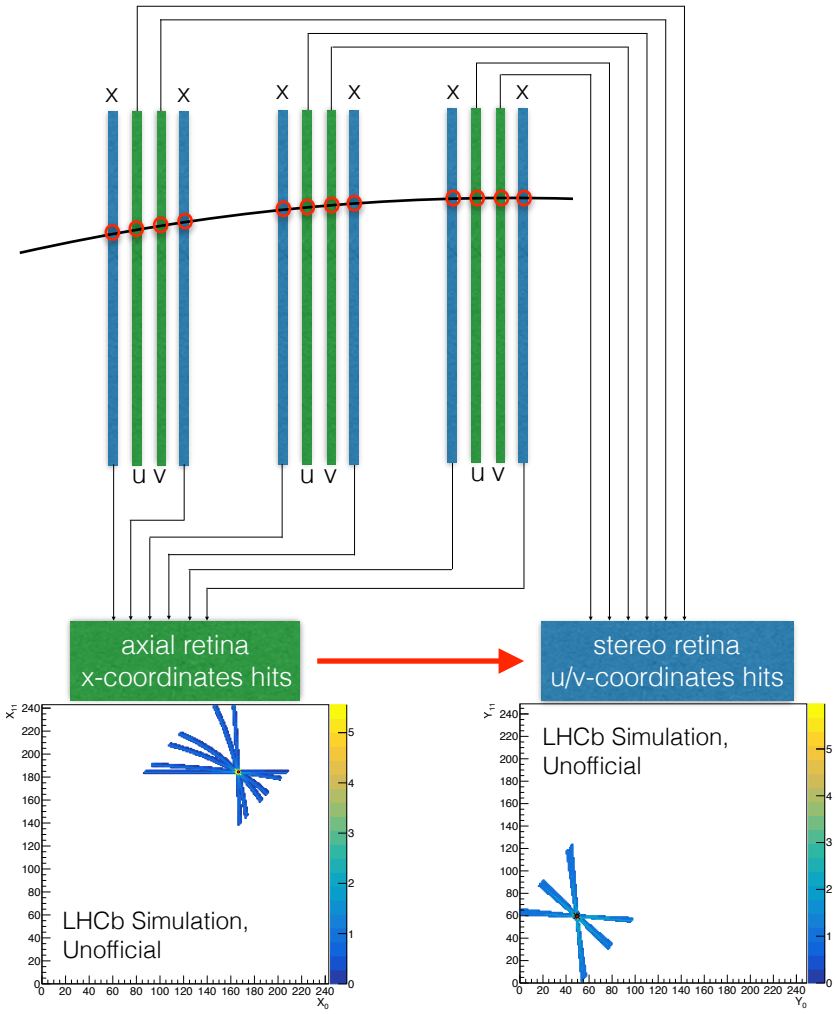
Comparison with "extreme scenario" of official Run-3 sequence (no backwards)

26.7 MHz → 36 MHz



Ongoing: Reconstruction of T-tracks in SciFi

- Much harder computation problem than the VELO - proportionally larger saving of CPU (currently not in the sequence, certainly not for HLT1)
- RETINA reconstruction factorized in two stages.
  - **Pattern recognition:** find the x-z track projection using only axial layers.
    - tracks approximated as straight lines (2-dim Retina).
    - for each local maximum found, a linear χ^2 fit to a parabola is executed (on DSP blocks of FPGAs)
  - **Stereo association**: x-z projection of axial track candidate used as "seed" to extract y-coordinates from stereo layers and associate y-z projection. Same firmware of axial Pattern Reco stage, with lower granularity.
- **Already proved feasible** with extra electronics corresponding to ~150 FPGA cards [see ACAT talk] Easily extendable to Mighty Tracker (even cheaper)



Most natural application: downstream tracks up-front of the HLT1

Run-4 proposal: The Downstream Tracker (DWT)

[see 2nd workshop in Elba]

Having T-tracks available up-front offers many physics benefits:

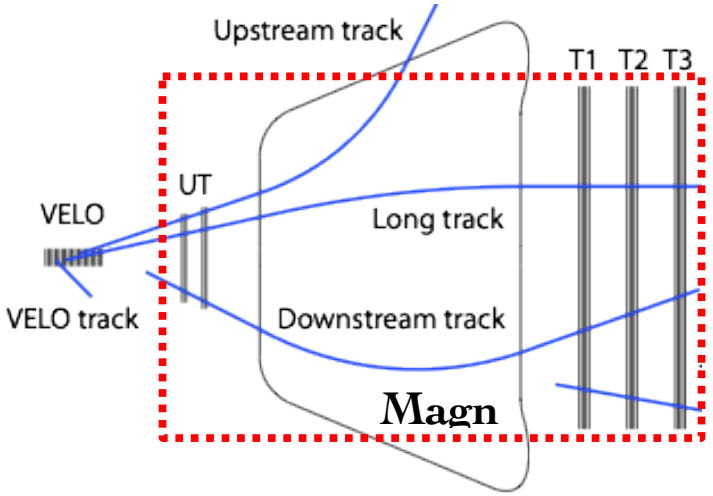
HLT1 trigger on long-lived into downstream tracks:

- **Neutral kaons** (Ks and KL) from charmless decays of B and Bs to CP eigenstates and hadronic charm decays → Increase yields > 2x-3x
- **Lambdas** in decays of charm and beauty baryons → increase yields more than for Ks
- **Very rare** Ks decays (e.g. Ks -> $\mu\mu$)
- **"Lifetime-unbiased"** D^0 -> Ks π π
- **Exotic Long lived particles**: dark sector bosons, ALPs / Higgs, majorana neutrinos → increased sensitivity at long lifetimes

	M (MeV)	τ (ps)
B_s	5300	1.5
K_S	500	90
K_L	500	50000
Λ	1100	260

General reconstruction:

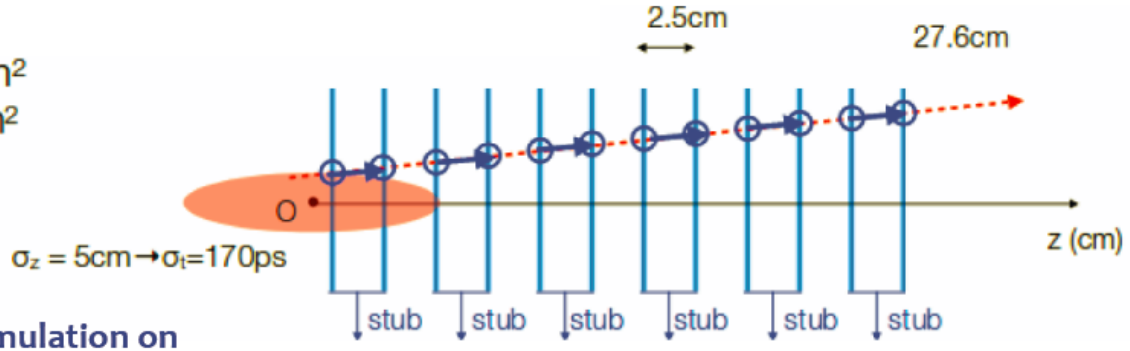
- **Up-front muons** with Muon-T-track matching
- **Long tracks** from VELO-T-track matching
- **V0 decays inside magnet** with T-tracks only [see D. Marangotto at T&A of 29/3]



Run5 project: 4D VELO reco [INFN-TIMESPOT]

- **RETINA-like processing of VELO stubs+timing**

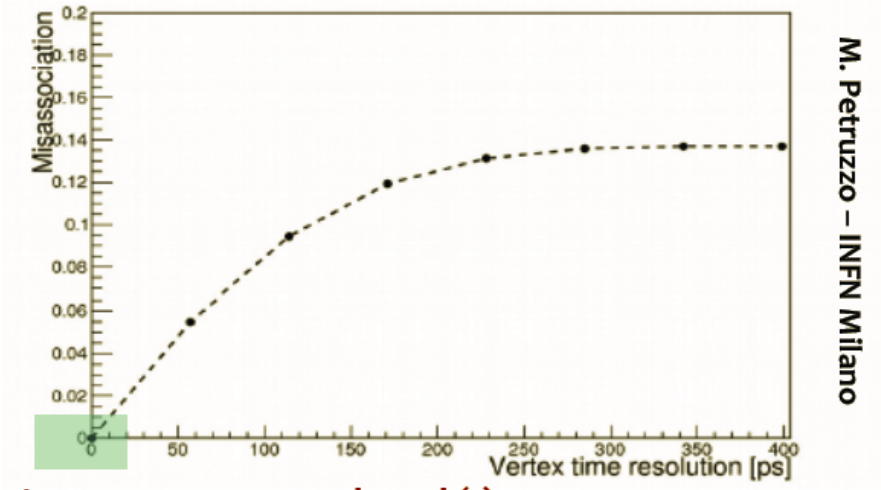
Sensor area = 6x6cm²
pixel size = 55x55μm²
thickness = 200 μm
time res σ_t=30 ps



Stub algorithm tested by simulation on a LHCb-like vertex detector:

- 12 planes of silicon vertex detector
- Pilup = 40
- 1200 tracks/event
- Interaction region of gaussian shape (σ_z = 5 cm, σ_t = 167 ps)

Mis-association vs vertex time resolution →



M. Petruzzo – INFN Milano

The 4D fast tracking algorithm has also been in FPGA on a custom board (1):

Two Xilinx Virtex Ultrascale FPGAs
High-speed optical transceivers → up to 1 Tbps input data rate per FPGA
One Xilinx Zynq FPGA

(1) M. Petruzzo et al., A novel 4D finding system using precise space and time information of the hit , TWEPP 2018

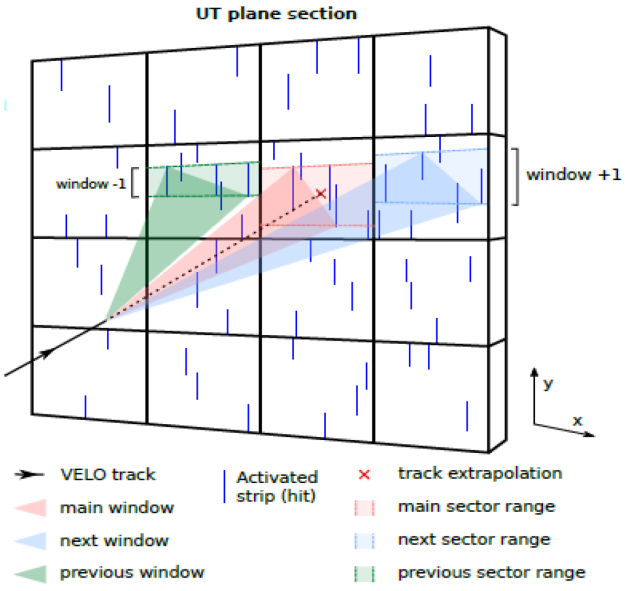
3. Post-Build reconstruction

- Could be physically located in the EB (if *dedicated* architecture), before network distribution to EFF - this would save on data transport of the output
- No latency issues, can operate in a CPU-like environment
- Most natural specialized type of computing to use at this level is the GPU
  - Built to be an effective accelerator for graphics, SIMD structure promises to provide an efficiency boost also in tracking jobs
  - Shown to deliver good performance for scientific workloads in recent years.
  - Need many events in each load to properly match resources to computation, and preferably strong data reduction to reduce memory footprint and output BW
  - Can use high-level languages, C++ - like, without sacrificing efficiency
    - Possible cross-fertilization with CPU algorithms

- Lots of GPU studies in HEP. In LHCb the Allen Project proposes full HLT1 in GPUs
- See WP6 dedicated meeting

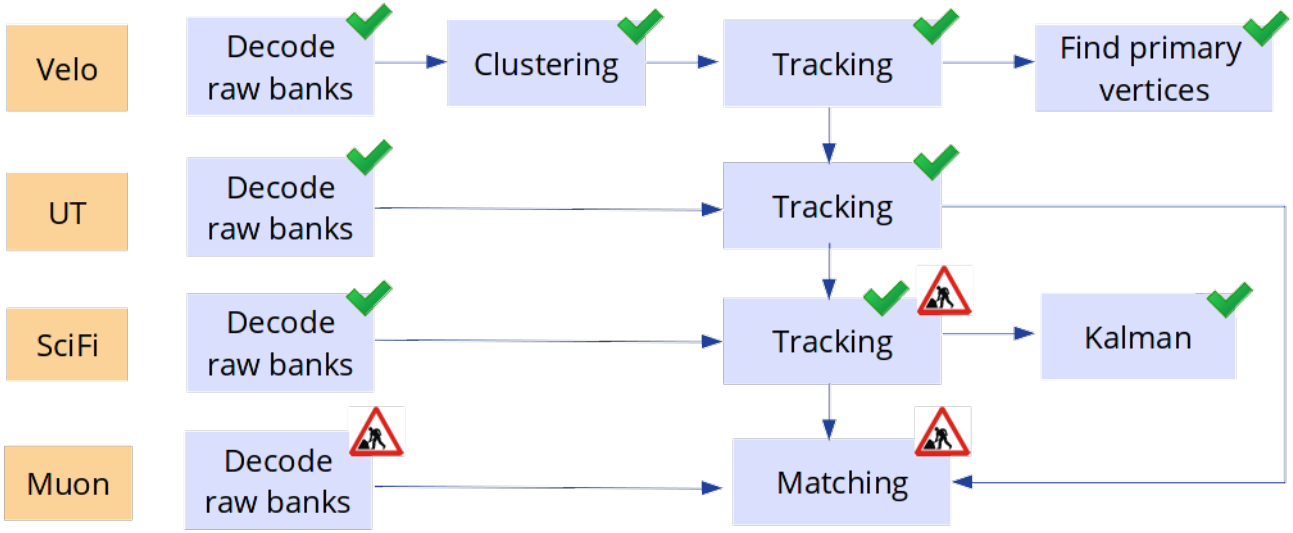
Current work: the ALLEN project

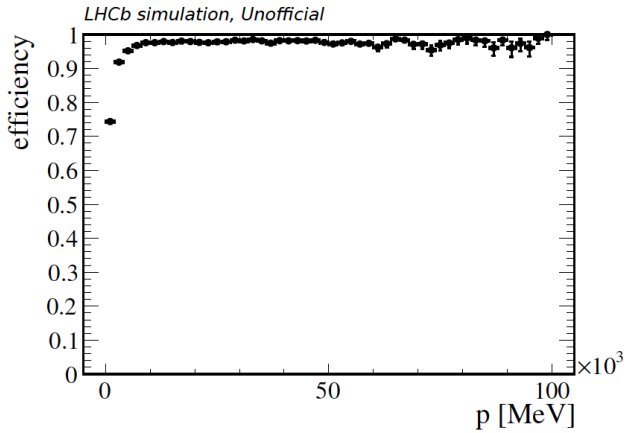
- A project to implement HLT1 selection on a farm of GPU's, using CUDA [https://gitlab.cern.ch/lhcb-parallelization/Allen]
- GPU implementation does not simply require running a special compiler on the same old code
- The 5 Challenges [see D.Campora ACAT talk]:
    1. Modularity. Parallelize both event and within-event
    2. Custom memory manager to avoid time overheads
    3. Spatio-temporal Locality: data access patterns need to restrict to a portion of memory
    4. Multiplicity reduction (cache memory limitations)
    5. Balance workload across cores. Reduce branches and inter-process communication



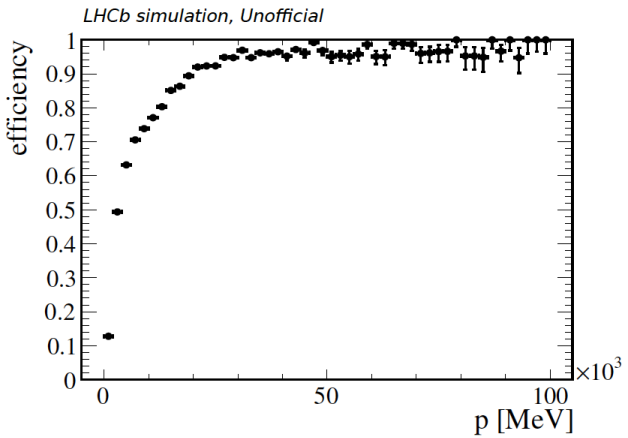
- Not only for GPUs: Intel's ISPC compiler allows porting the same code to SIMD-capable CPUs via a SPMD programming model.

Current status of the Allen project





Velo



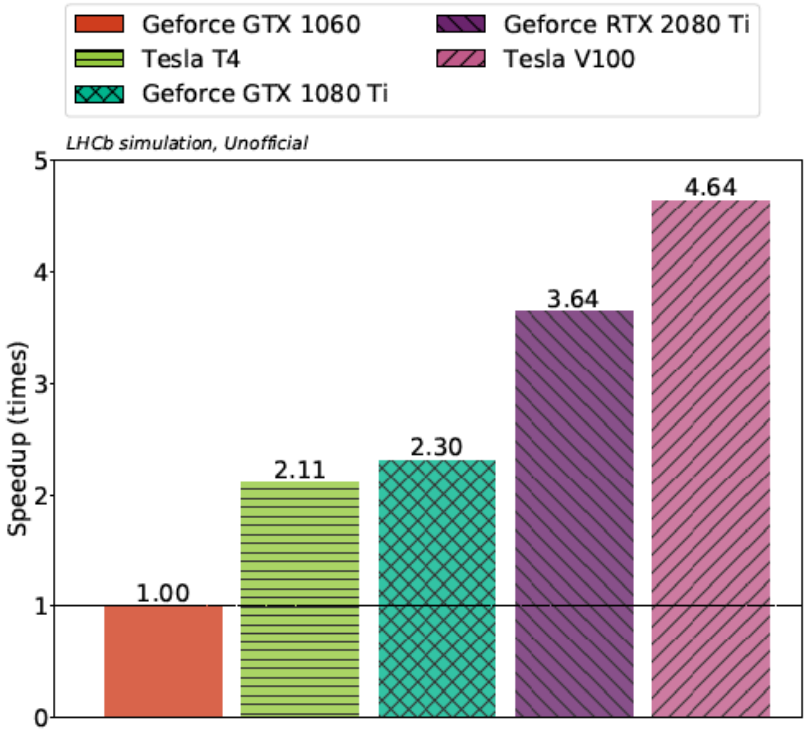
Velo+UT

Allen timing performance tests

- Sequence with Velo, PV, UT and decoding of SciFi
- Has been proved to have reasonable reconstruction performance
- The time performance in figure is relative to the *Geforce GTX 1060*
- Whether this is a convenient route will depend on evolution of GPU costs

- Geforce GTX 1060: 24 kHz
- Geforce GTX 1080 Ti: 56 kHz
- Geforce RTX 2080 Ti: 88 kHz
- Tesla T4: 51 kHz
- Tesla V100: 112 kHz

Feature	Geforce GTX 1060	Geforce GTX 1080 Ti	Geforce RTX 2080 Ti	Tesla T4	Tesla V100
# cores	1280 (CUDA)	3584 (CUDA)	4352 (CUDA)	2560 (CUDA)	5120 (CUDA)
Max freq.	1.81 GHz	1.67 GHz	1.545 GHz	1.59 GHz	1.37 GHz
Cache (L2)	1.5 MiB	2.75 MiB	6 MiB	6 MiB	6 MiB
DRAM	5.94 GiB GDDR5	10.92 GiB GDDR5	10.92 GiB GDDR5	16 GiB GDDR6	32 GiB HBM2
CUDA capability	6.1	6.1	7.5	7.5	7.0
TDP	120W	250W	250W	70W	250W



4./5. Reco in the EFF -- close to, or within CPUs

- Little specific work in LHCb (to my knowledge)
- Same considerations of Post-Build apply to EFF, with a couple differences:
  - No BW saving, only save if raw computation efficiency better than CPU
  - Might still be more convenient that EB placement for practical reasons (e.g. number of units needed), depending on implementation details
- "Within CPUs" means to have special commercial units where CPU is more or less tightly coupled to a dedicated accelerator unit
  - Most notable example is Intel's CPU-FPGA hybrid
  - Advantage of a high bandwidth and shared memory between CPU and its FPGA
  - Here can again perform specific acceleration targeted at specific parts of the tracking algorithm that are particularly suitable for it
  - Must have one per each CPU, imply special coding and rather rigid structure - but might turn out to be efficient
- **A lot depends on implementation details and commercially available line of products that are still unclear, so it is hard to say more than it is a field to watch**

Possible scenarios for Run-4

- Several technologies for non-CPU tracking are already quite mature, and should definitely be usable tools in Run-4:
  - Some limited FE preprocessing, like doublets in the MT, SM
  - Clustering in back-end for all detectors should be well established
  - A Downstream Tracker to reduce tracking load and enhance LLP
  - GPU-accelerated HLT1 selection to reduce CPU and BW load
- In principle all of them could happen. However, constant luminosity and limited budget mean that the Run-3 system may not really need to change, and their actual adoption will be driven by cost savings, new detectors, and low-cost improvements to physics.
- Some early version of these could already be in Run-3, and will be crucial experience to establish feasibility. For the same reason, it will be important path finders for further future.

Possible scenarios for Run5+

- It's hard to make predictions about the future...
- What is clear is that the large step in luminosity, unprecedented for flavor experiments, and likely new advanced detectors, will put us in a very new situation, in which real time data processing takes a very central role in the success of the experiment.
- My 2c: while commercial CPU power will still grow, the large luminosity and possibly timing data will make the use of specialized tracking devices *unavoidable*.
- Exact scenario will depend on the lessons learned from Run3+Run4.
  A strong motivation to gain experience as soon as possible, if one was needed !
- What we can see in the crystal ball:
  - Some low-level preprocessing, possibly with data reduction (not events)
    - Doublets, rejection of out-of-time hits....
  - *Most* of the track reconstruction done pre-build
  - Higher-level reconstruction (vertexing, candidate reconstruction ?) done on GPU
  - CPUs doing a lot of actual physics analysis, leaving very little to do off-line:
    DTF , "pre-packaged analysis" -> online histograms, other non-O(N) statistics...

- **An interesting time - a time for LHCb to be pioneer in HEP technology and Physics**