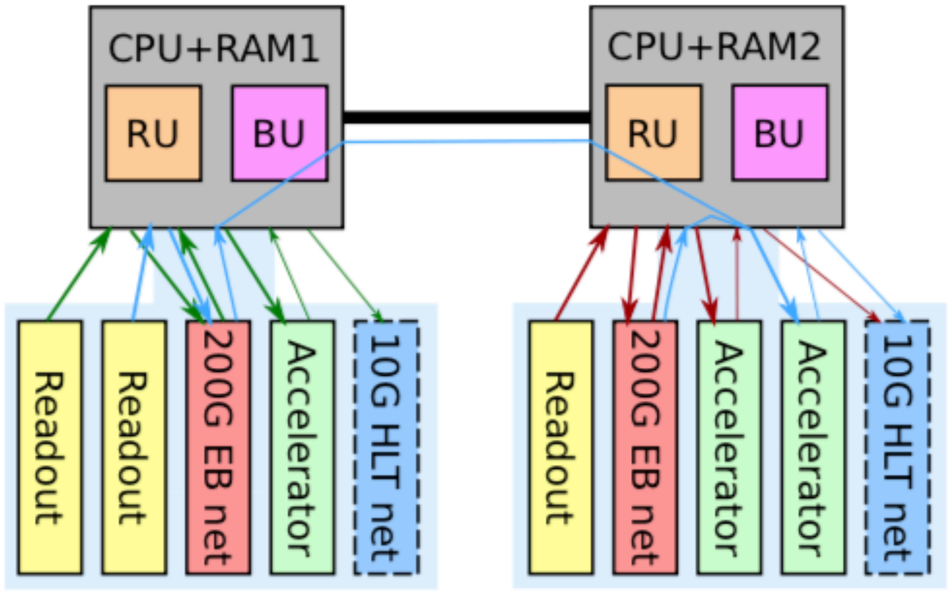
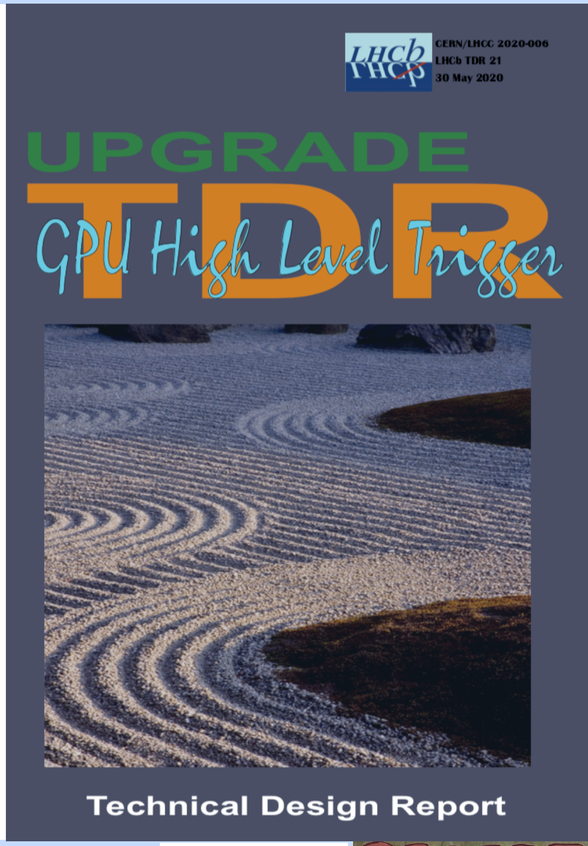
LHCb GPU First Level Trigger status and prospects



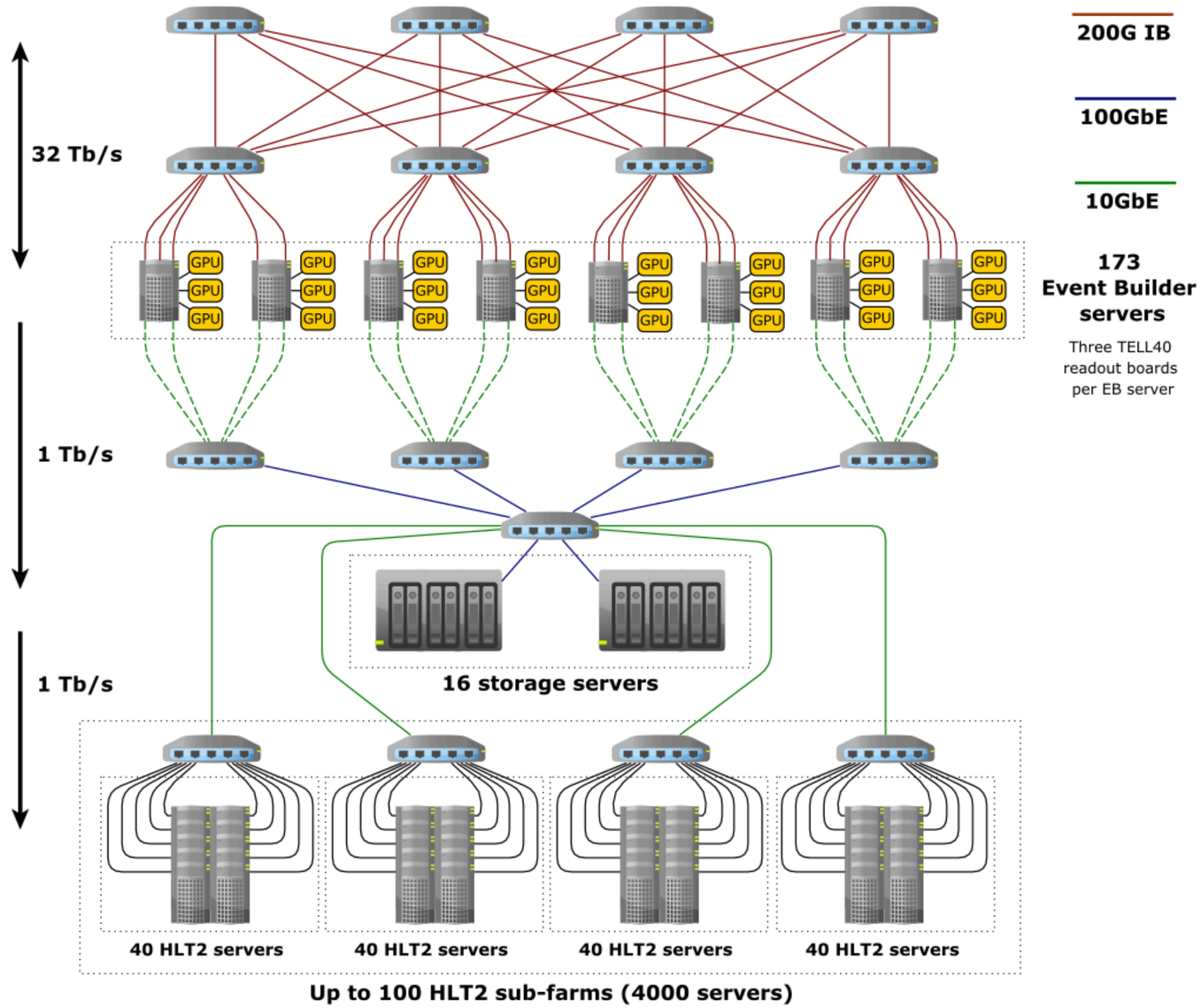


V. V. Gligorov

LPNHE/CNRS

SCF meeting, 22.10.2020

LHCb online dataflow



A flatter network architecture (which would have made us look almost like an HPC centre) was considered and reviewed but in the end was not sufficiently beneficial to justify deviating from the two-layer baseline shown.

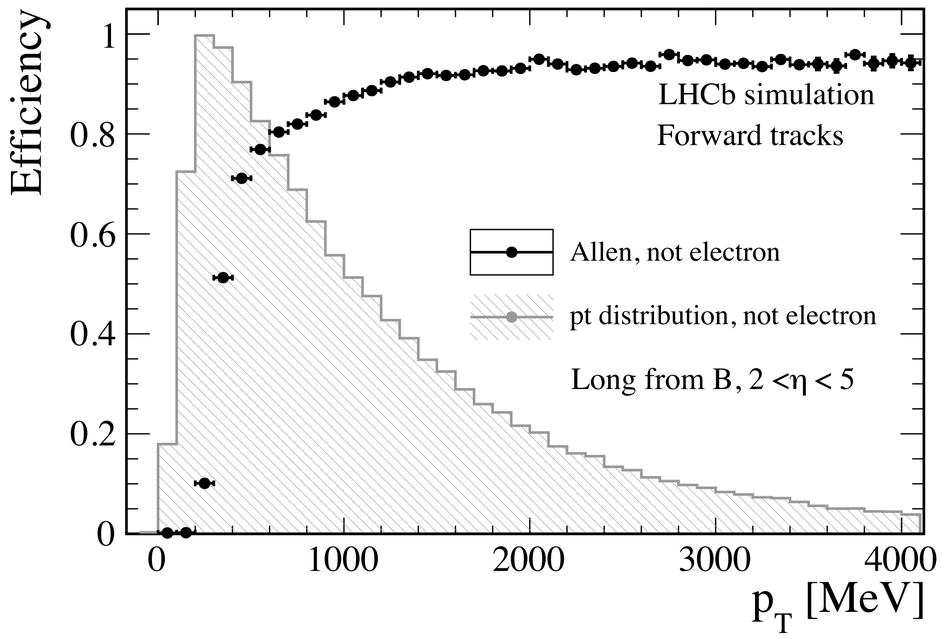
Physics objectives of first-level HLT1 trigger

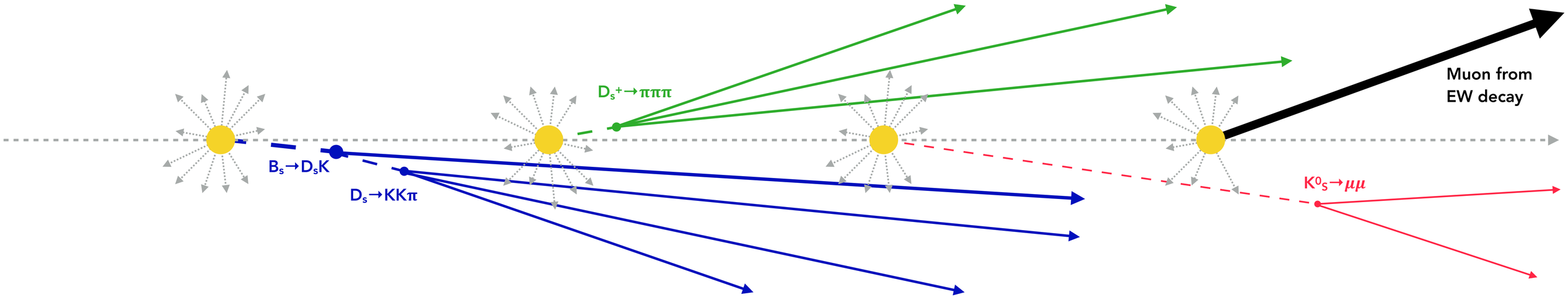
A. Reduce the event rate to 1 MHz which can be written to the online disk buffer and subsequently fully reconstructed by the second stage HLT2 trigger.

B. Remain efficient for soft (e.g. rare Kaon decays), heavy flavour, and EW physics signatures.

Key requirement: maintain a high tracking efficiency with minimal fake rate to as low a P_T threshold as possible

GPU HLT TDR



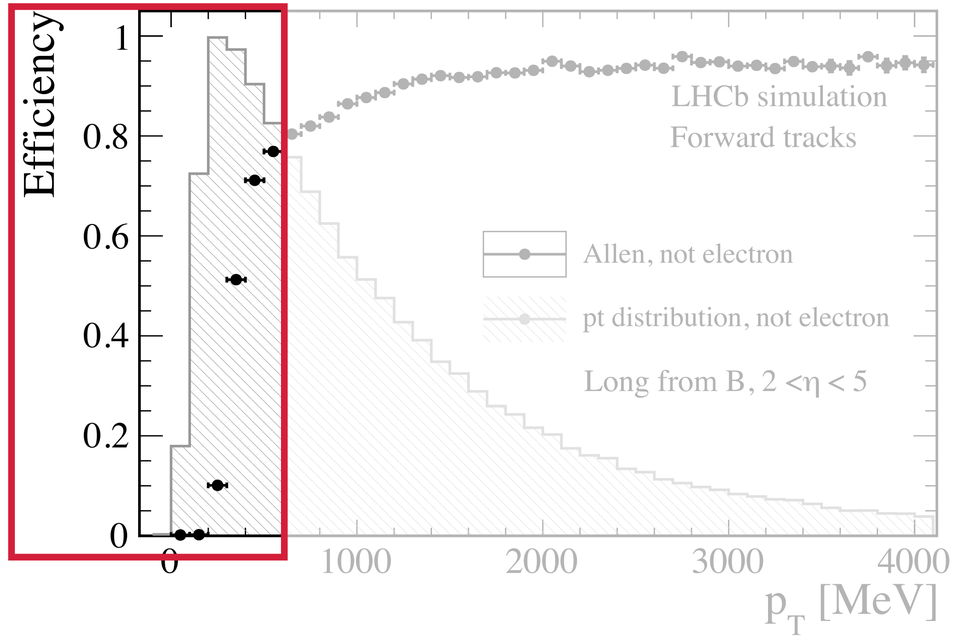


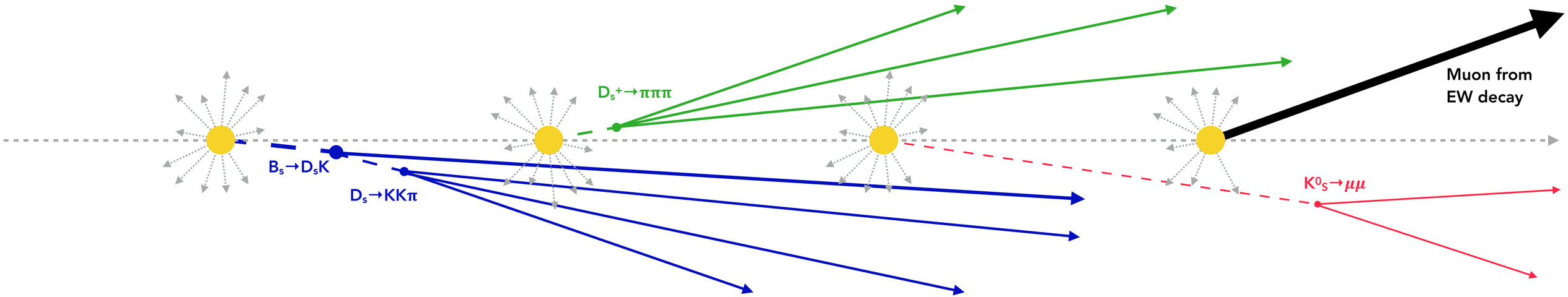
Physics objectives of first-level HLT1 trigger

Reminder that LHCb's triggers are very sensitive to the turn-on region because this is where our physics is.

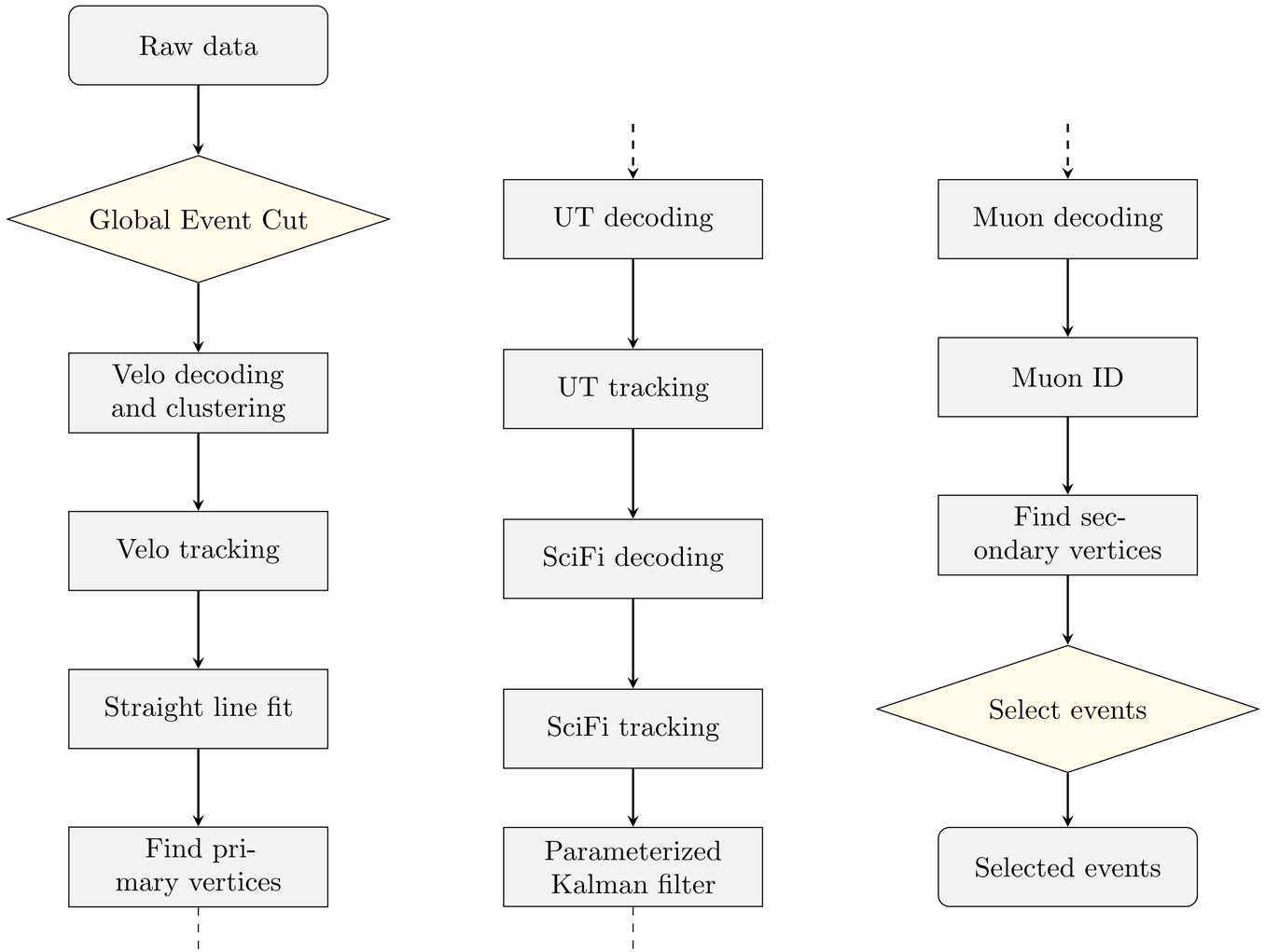
Some loss in this region is irreducible even offline, but the 2014 TDR baseline left substantial room for gains.

Improvements here gain not only extra signal but reduce systematics in modelling the turn-on curve.





LHCb GPU first-level trigger algorithm sequence

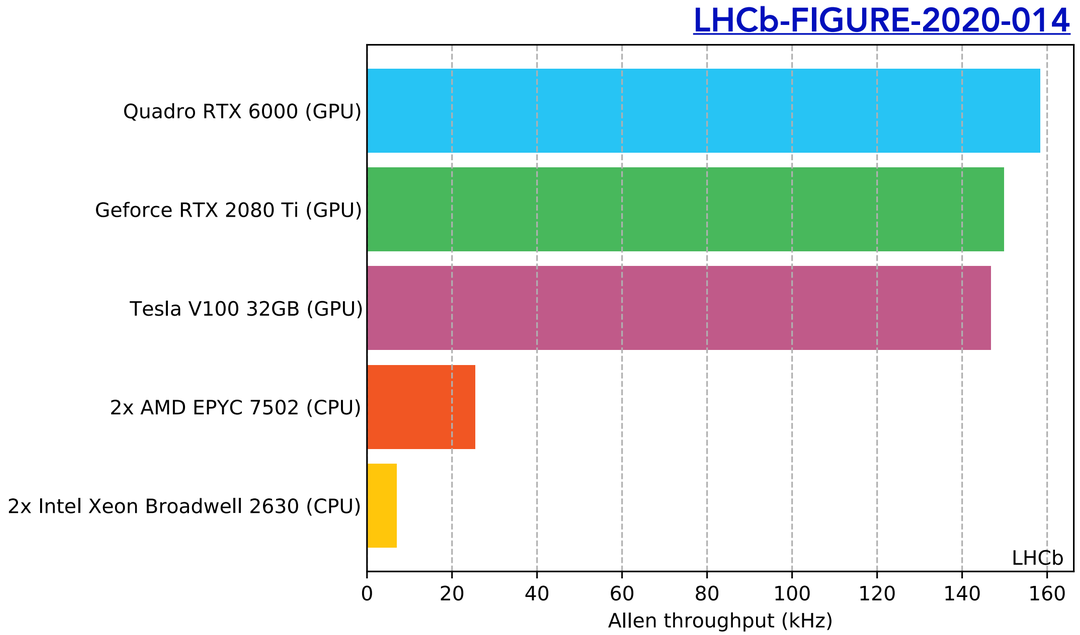


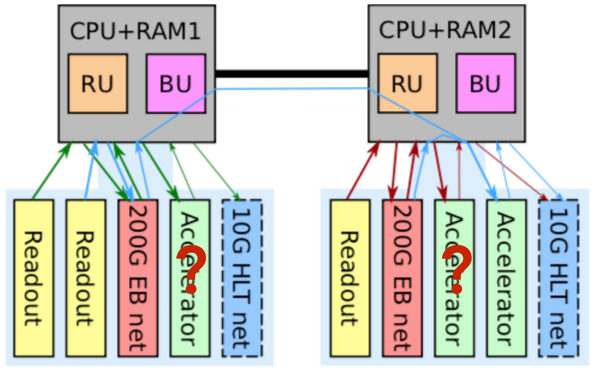
Each entry actually corresponds to multiple algorithms, around 70 in total for the default sequence

The first-level trigger sequence can be compiled for CPU and run in the same job as LHCb's CPU reconstruction.

This is ~4x slower than the native CPU implementation, but as we will see later that is not a problem for us.

Current HLT1 performance





GPU-equipped event builder PC, with traffic of all three readout cards.

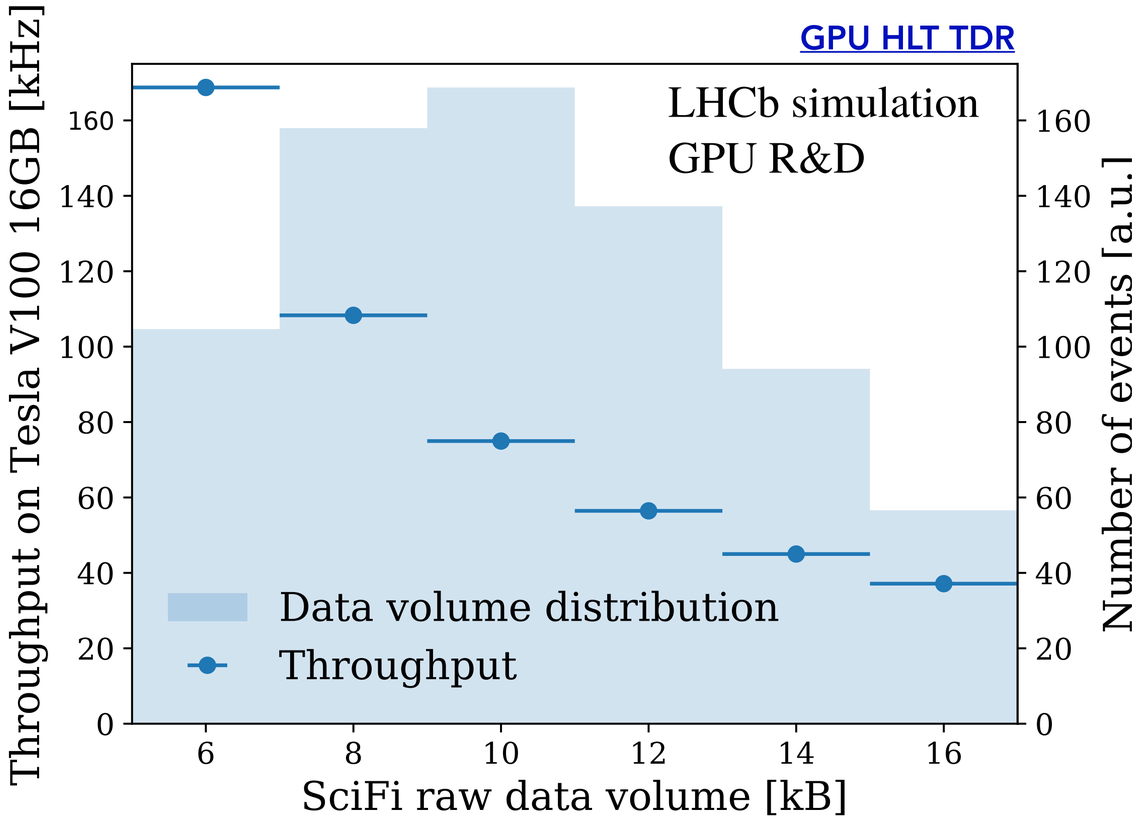
>60% throughput improvement for the same physics since the TDR (this spring!)

Mainly from "technical" improvements in the compilation and GPU configuration, some improvements also to the VELO algorithm logic and the Muon decoding

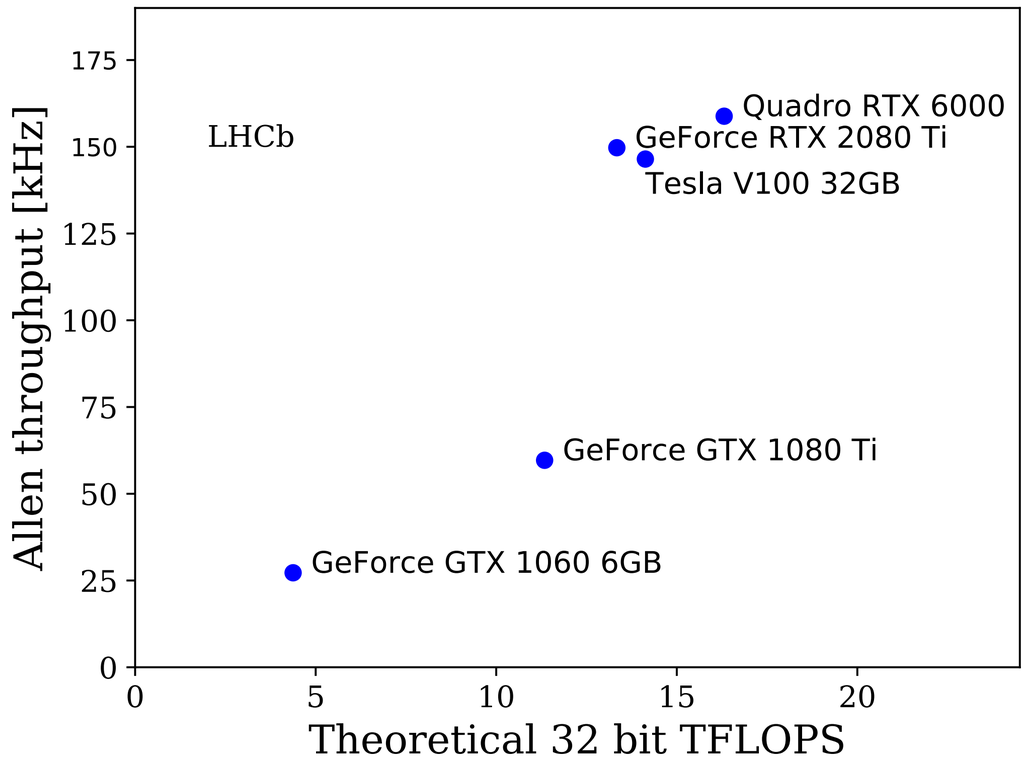
At ~150 kHz per card we are getting close to being able to run HLT1 with one previous generation GPU card per EB node. This is not a practical option because we would saturate the Gen3 PCIe bandwidth of 16 GB/s but it underlines the headroom.

HLT1 performance robustness

GPU HLT TDR



LHCb-FIGURE-2020-014



Performance scaling with theoretical FLOPS makes us optimistic about next-gen GPU cards coming onto the market. Performance scaling with event size reassures that we won't die because of "weird" events in data.

LHCb GPU first-level trigger in context

Follows in the footsteps of ALICE's pioneering efforts over the past decade(s).

One bullet similarities and differences

1. Both treat the GPU as a complete general purpose processor: raw events in, annotated/reconstructed events out. Minimal CPU-GPU communication.
2. The two systems span the two extremes of event rate: $O(10^2)$ events per GPU per second for ALICE, $O(10^5)$ for LHCb.

LHCb's key innovation with Allen is showing that we can use GPUs as quasi-standalone trigger processors in a domain where we have the equivalent of a few microseconds per event for the reconstruction and selection. Keep overheads to a minimum and use the deep memory buffer of the host CPU nodes to smooth out I/O fluctuations.

Clear today that programmed correctly GPUs can handle complex and even somewhat non-linear data/control flows, as well as complex memory allocation patterns. As with all high-throughput computing the bottlenecks are related to memory management, not TFLOPs for computation. No application is off-limits for GPUs anymore.

So what physics does all this actually get us?

If the GPUs coming on the market now perform as expected could have up to 5x the computing power for HLT1 compared to the 2014 Trigger & Online TDR projections.

The system is limited by the physical space available in the EB nodes, not by money, so we will not buy everything at once. Buying "enough" for 2022 and then upgrading in 2023 for the rest of Run 3 could be very attractive.

An intense effort is underway to figure out how best to use this additional power

1. **Track down to the same P_T thresholds as offline**
2. **Remove the global event cuts which cost 25% of our EW physics signals**
3. **Find tracks originating outside the vertex detector already in HLT1**
4. **Include calorimeter reconstruction in HLT1**
5. **Perform a more complete Kalman fit**

Aim to have public quantitative estimates of how much extra physics this buys (and for which areas of the physics programme) next year to inform purchasing decisions.

Current purchasing plans: let them fight

We need to have a system which can handle 30 MHz in place for the luminosity ramp.

We are now almost at the point where this can be done with one 2018-generation GPU per server node, while we have capacity for up to three GPUs per server node.

This gives us significant flexibility!

Key is remaining vendor-independent: must be able to run with a viable per-GPU throughput on non-NVIDIA GPUs.

Allen already runs on AMD, performance is work in progress for now.

Wait and see what happens with the next-generation GPUs from NVIDIA and AMD (and maybe INTEL) by end of this year before making any decisions. We are in the process of acquiring these under NDA where necessary and starting to test them.

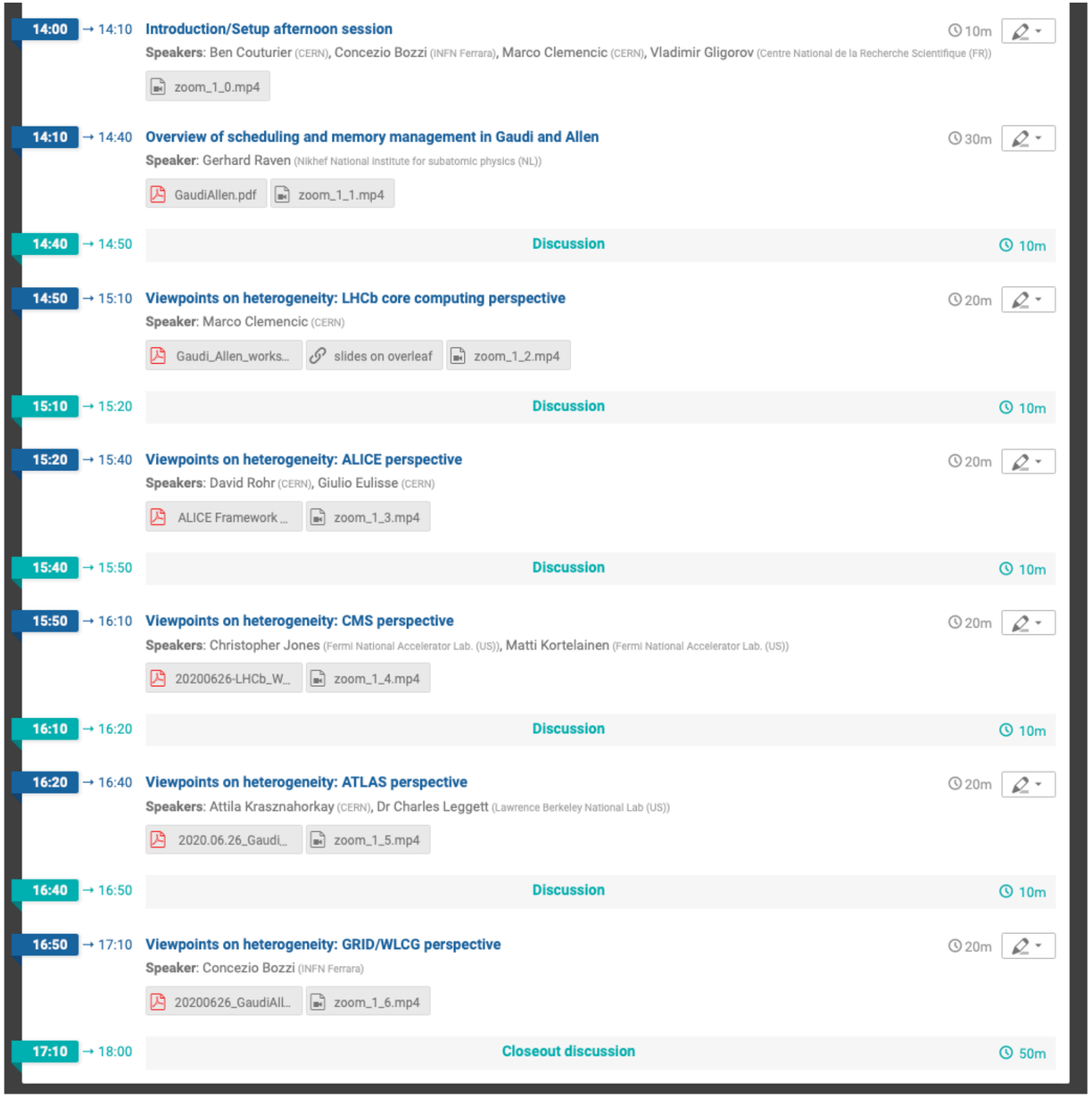
(We already have enough GPUs in hand for commissioning activities.)

Workshop on heterogeneity

Together with ATLAS, CMS, ALICE, and other LHCb software projects organised an intense one day workshop on heterogeneous computing following our June decision.

Very focused objective: now that most of the collaborations are going to be using GPUs, discuss who is using them in what way. In particular understand the relationship between our GPU framework (Allen) and our CPU framework (Gaudi) which we share with ATLAS.

This was very fruitful, and fed into a big ongoing effort inside LHCb to bring Gaudi and Allen closer together where possible.



GPUs as part of a heterogeneous computing architecture

As often it is wise to separate

1. What our framework allows us to do
2. What it is actually intelligent (efficient, cost-effective, etc.) to ask the framework to do

Broadening (1) has few downsides (costs developer time, possibly increased maintenance).

Broadening (1) has the enormous benefit that we can react quickly if market-driven changes to the processing technologies (which we do not control) alter (2)

Concrete example: how do we optimally define & configure an algorithm sequence, namely its data and control flows, in a parallel environment?

The answer is practically the same whether you are doing it for multithreaded CPU or GPU!

That doesn't mean you'd want to configure the same algorithms or sequences in both cases. It just means that you don't have to guess today what will be the right choices in the future.

Follow this logic and move towards general use of cross-architecture algorithms.

Offline use of GPUs in LHCb for reconstruction

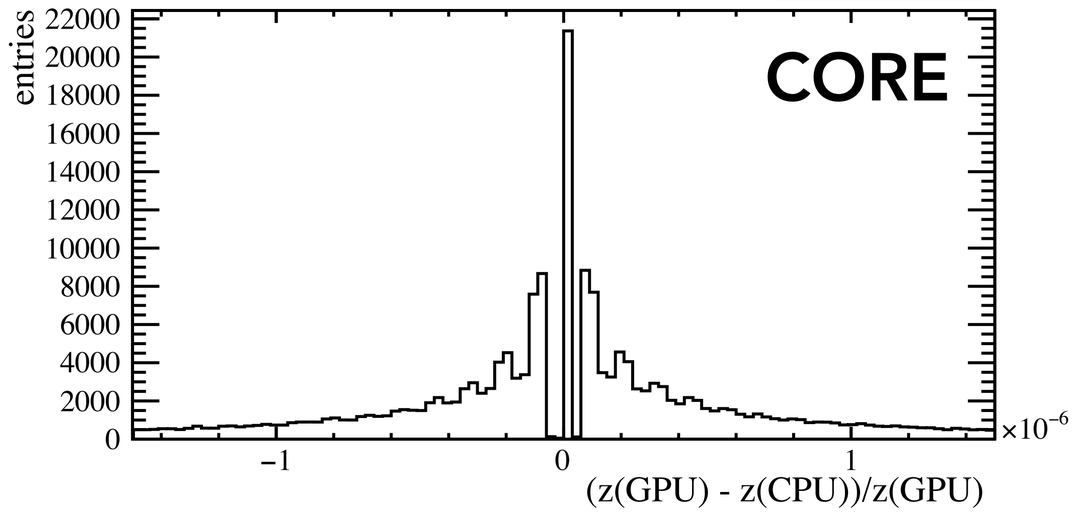
HLT1 is a tiny part of LHCb's offline processing cost. Will run the GPU code compiled for CPU in simulation. Gives the same answer as when running on a GPU to better than permille level.

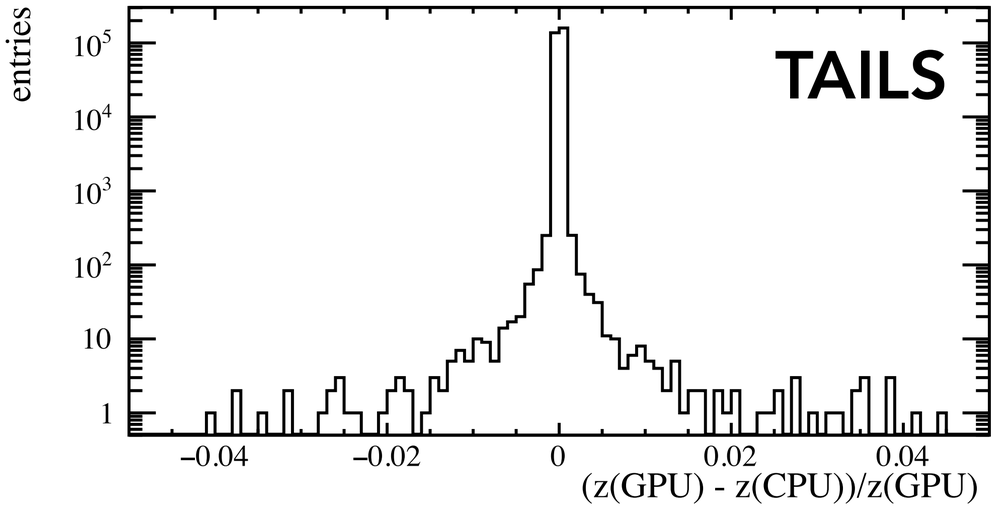
Ongoing rewrite of CPU reconstruction to make it thread-safe and vectorizable will help. Once done it is a much smaller step to make algorithms able to run on both CPU and GPU, with architecture-specific speed optimizations where necessary.

Progress crucially depends on keeping the core developers together while integrating the best younger colleagues as they come through the system. If we can keep the team in place by Run 4 we should be able to have a single cross-architecture parallel reconstruction and selection codebase and have maximum flexibility over our architecture choices.

A reminder that dedicated FPGA efforts also exist on LHCb — we want to see to what extent they can be brought into this cross-architecture development environment as well.

GPU HLT TDR





Offline use of GPUs in LHCb for analysis

GPU HLT TDR

All applications require work on DIRAC to enable transparent user job submission to GPU resources.

Until GPUs become widely available on GRID, use the EB farm. Commissioning experience needed but should be available in TS and YETS at the very least.

Potential applications:

1. ML algorithm training
2. “Full fat” Feldman Cousins for limits/coverage
3. High-statistics fits particularly for amplitude analyses and fits based on TensorFlow
4. … will surely discover many more once the resources are widely available, as usual.

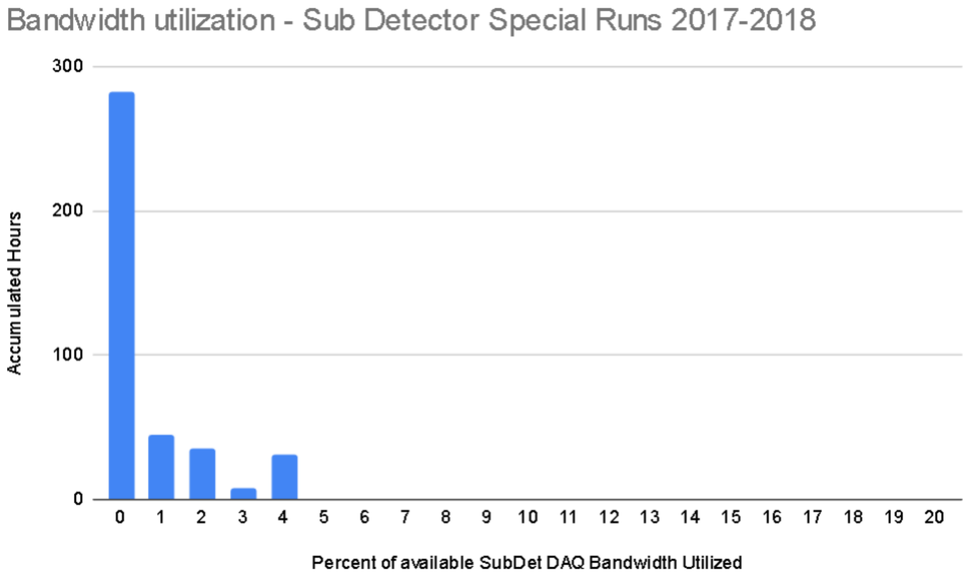


Figure 46: DAQ utilisation during special detector runs

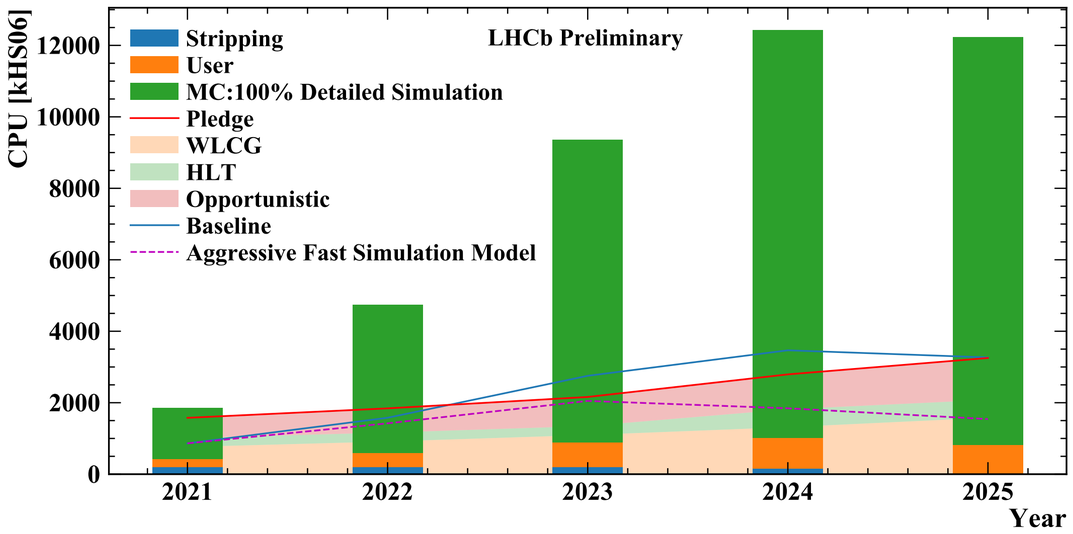
Developments of GPU processing for simulation in LHCb

This problem is not unique to LHCb, and there are now many community initiatives on this topic.

Some examples:

1. Opticks/Optix (GPU ray tracing for optical photon transportation)
2. AdePT initiative for CALO sim on GPUs
3. Celeritas GPU acceleration for Geant developed by FNAL/ANL/Oak Ridge

LHCb-FIGURE-2019-018

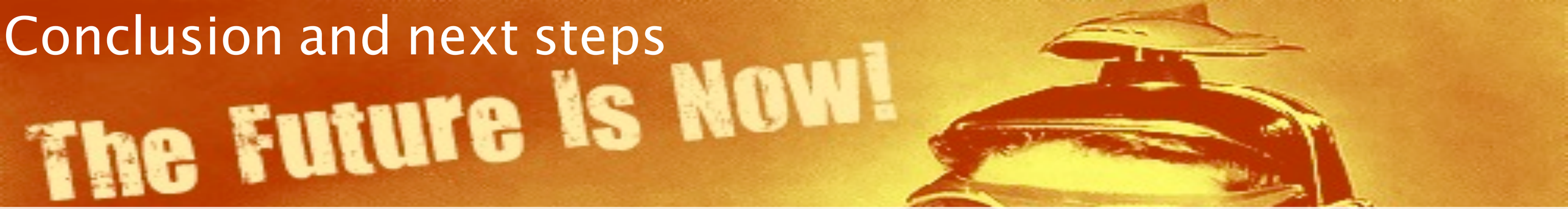


Our new Gaussino simulation framework is designed for multithreaded applications and aims to support GPUs and other accelerators. A lot of developer time is going into making sure that if the community initiatives pay off, we can take advantage.

Simulation is the key driver of LHCb offline processing requirements, significantly bigger than user jobs

Conclusion and next steps

Conclusion and next steps



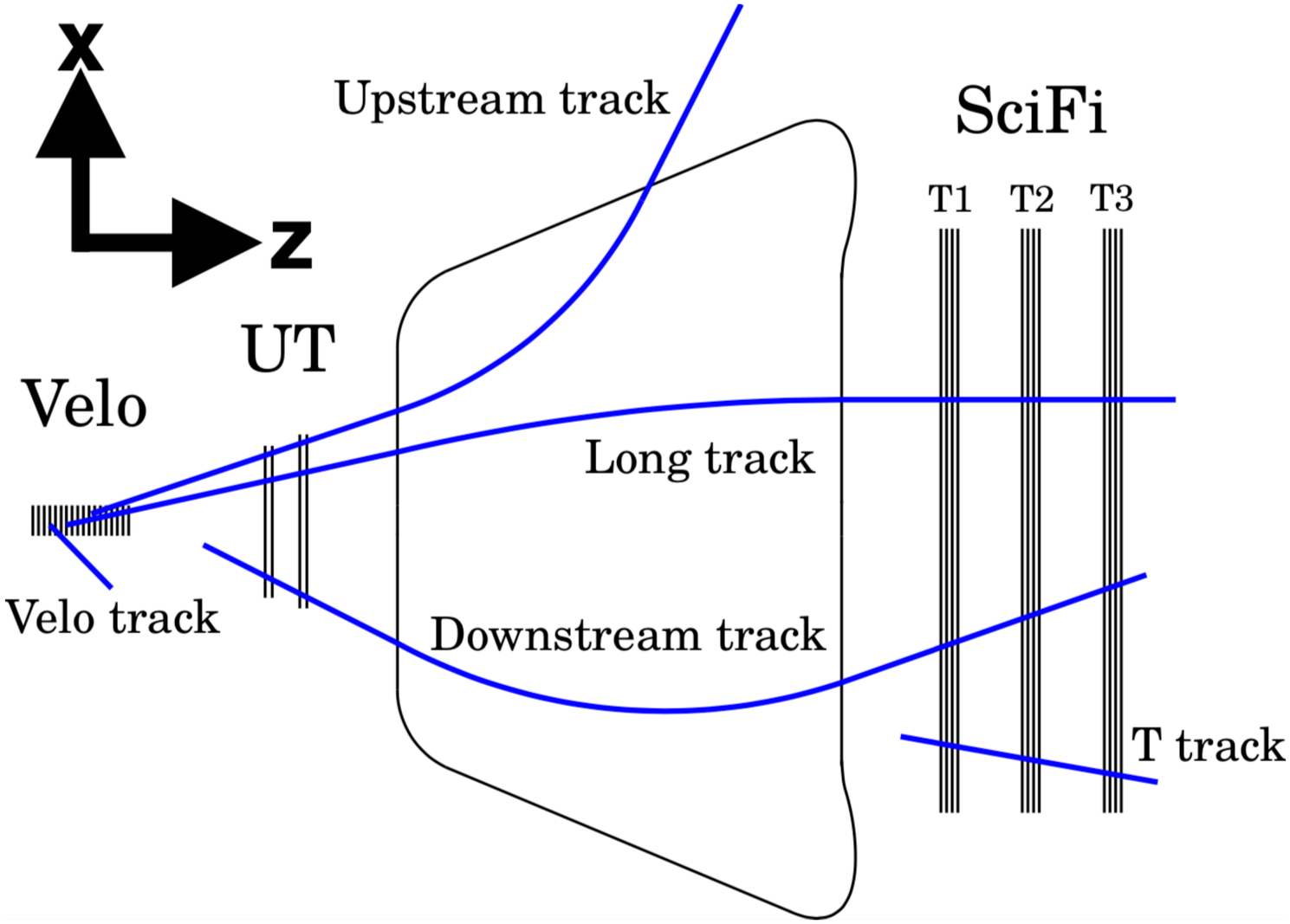
LHCb first-level GPU trigger is well on track for 2022 datataking

GPUs are increasingly a viable alternative to CPUs as general purpose quasi-standalone processors. Focus on writing architecture-independent or portable parallelized code, evaluate right CPU/GPU mixture on a cost benefit basis for any given application.

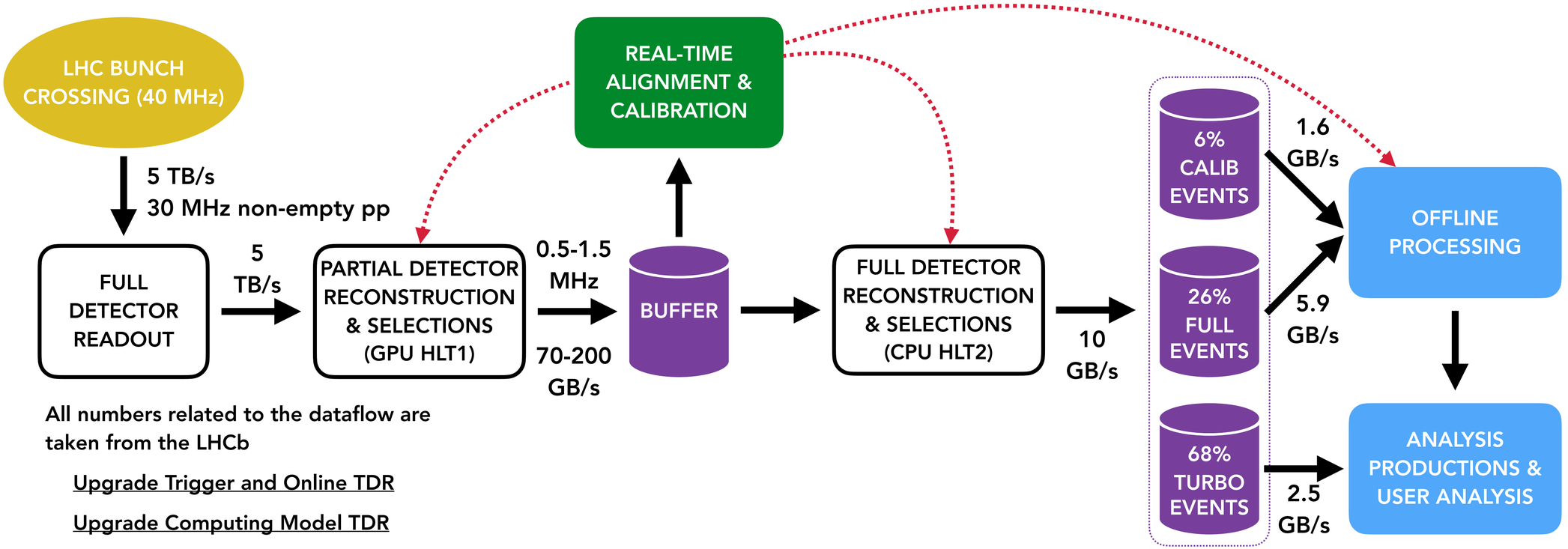
Main bottleneck is not hardware but the limited number of people and the difficulty to retain the most skilled software developers. Having to knowledge transfer every 3 years is very inefficient and disincentivizes the needed long-term planning and development. Cross-collaboration efforts (OpenLab, HSF, IRIS-HEP, etc) and shared libraries are crucial!

Backup

Track types


x
z
Upstream track
SciFi
T1
T2
T3
UT
Velo
Long track
Velo track
Downstream track
T track

Dataflow



All numbers related to the dataflow are taken from the LHCb

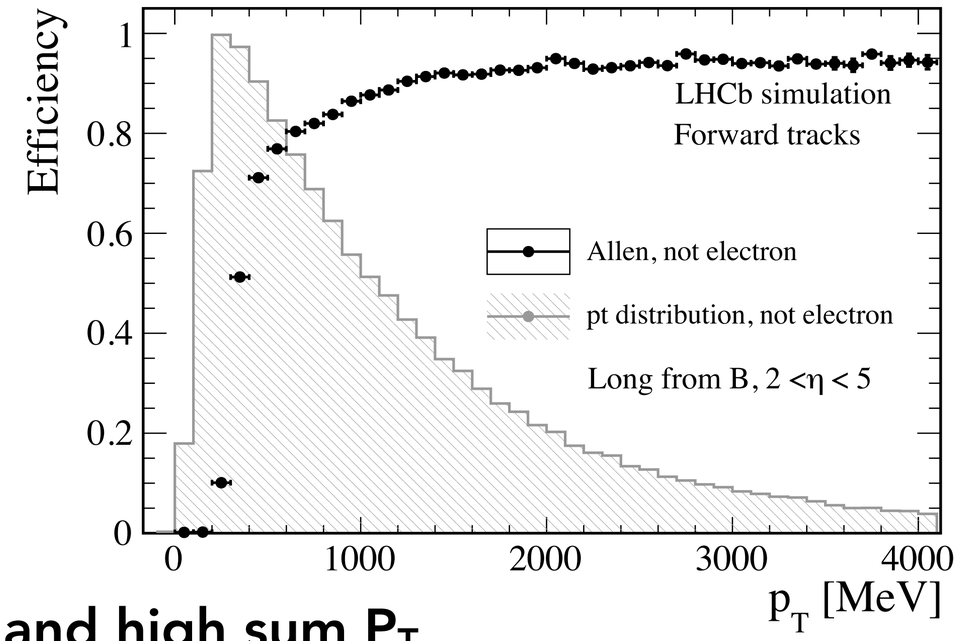
Upgrade Trigger and Online TDR

Upgrade Computing Model TDR

Physics objectives of first-level HLT1 trigger

A. Reduce the event rate to 1 MHz which can be written to the online disk buffer.

B. Remain efficient for physics ranging from rare kaon decays to EW physics and everything in between.



Core physics signatures of HLT1

1. Single displaced charged particle with high P_T
2. Displaced vertex with two (or more) charged particles and high sum P_T
3. Displaced muon and dimuon (displaced or prompt)
4. Very high P_T leptons independently of their displacement for EW physics
5. Exclusive charged-particle decays of softer (strange or charmed) hadrons.

Key to all these: high tracking efficiency with minimal fake rate down to as low P_T as possible

Several different bottlenecks in the system limit the HLT1 output rate to 1 MHz : I/O to the online disk buffer, disk buffer size for holding data until interfill periods, and the computing speed of the second-level trigger.