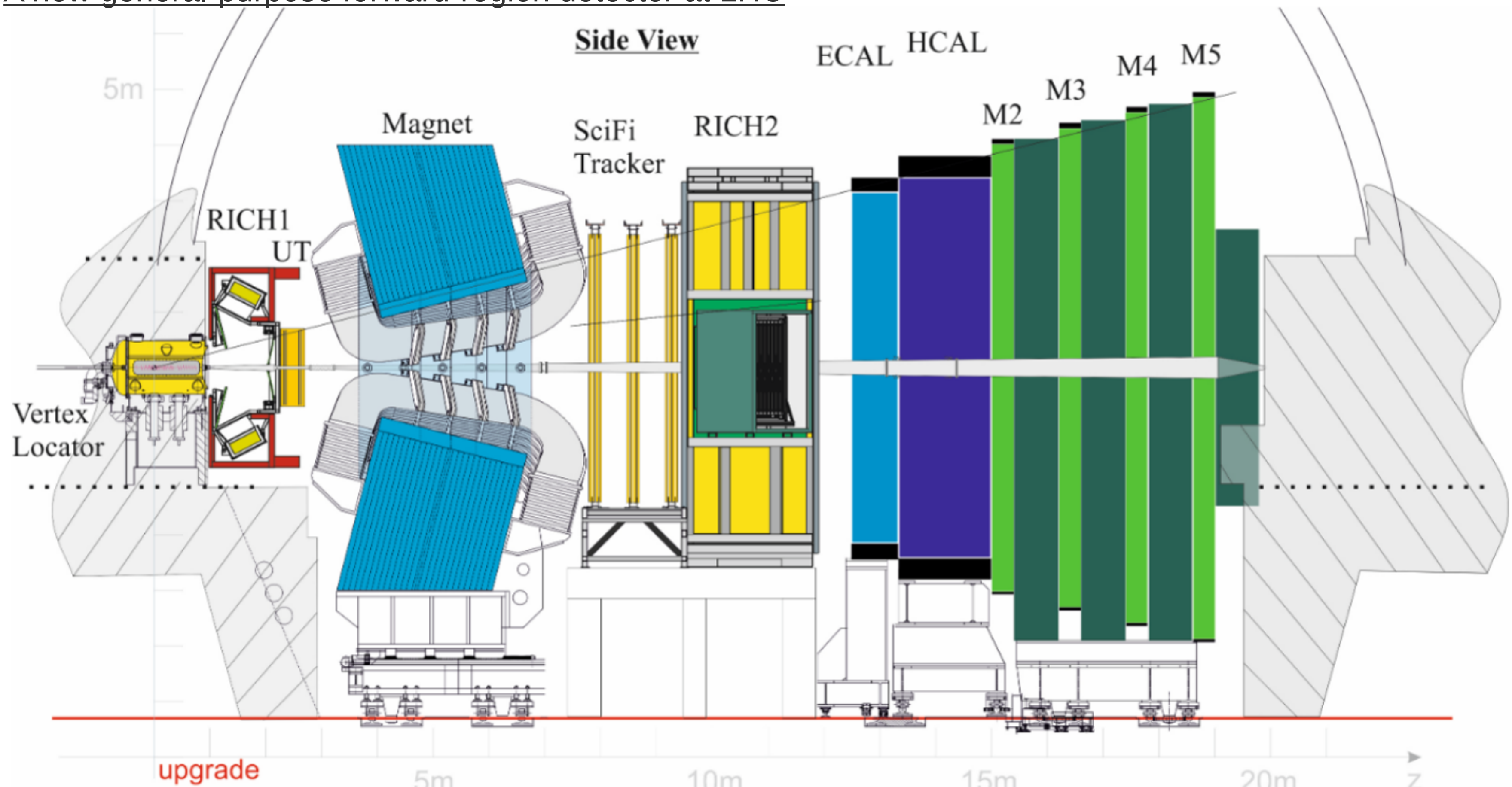
The LHCb trigger in Run 3

Miroslav Saur
(Lanzhou University)

2025/06/26

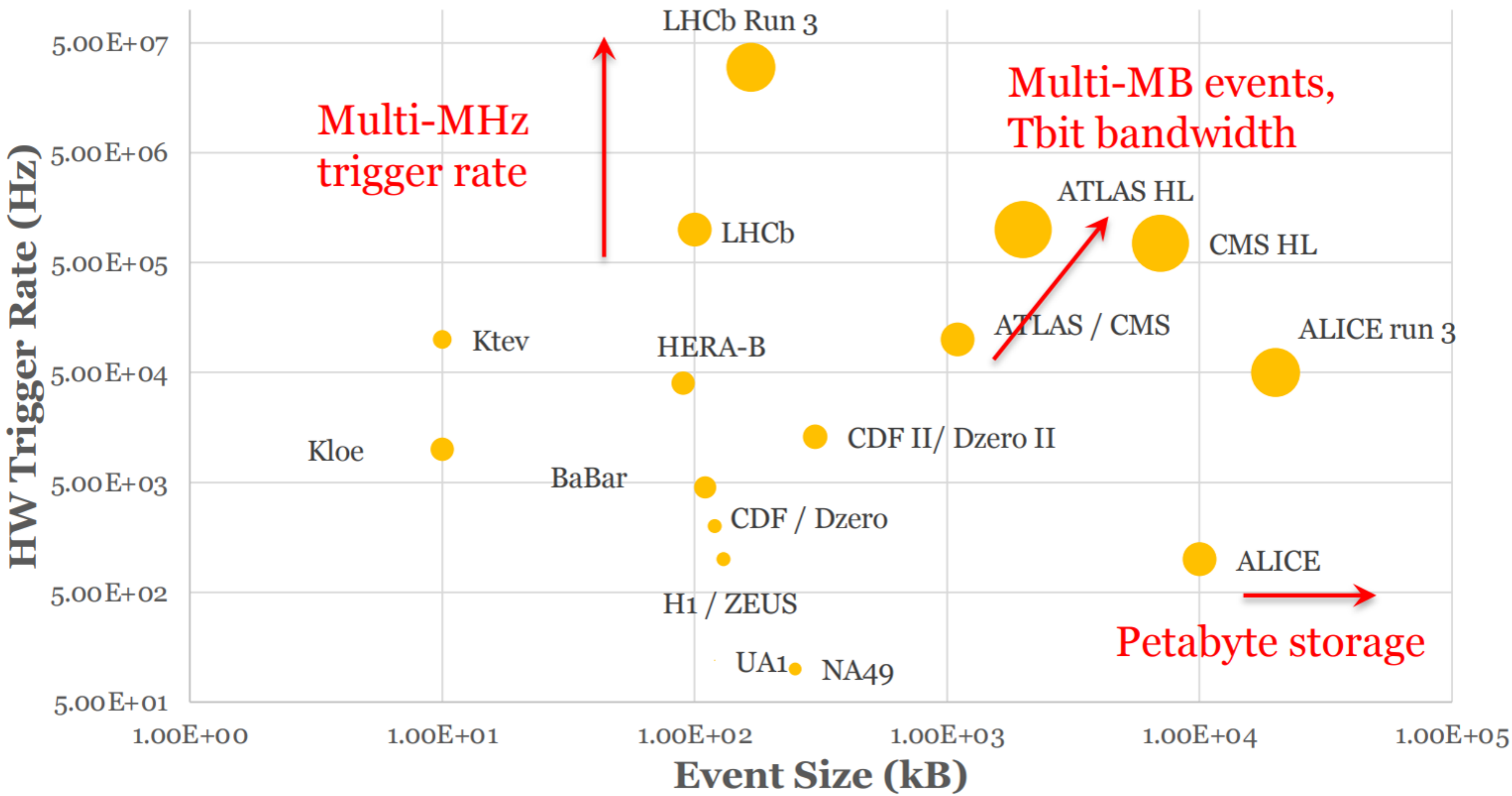
LHCb experiment in Run 3

- LHCb conditions in Run 3: luminosity of 2×10^{33} $cm^{-2}s^{-1}$, $\sqrt{s}$ = 13.6 TeV, visible collisions per bunch $\mu \sim 5$
- New tracker detectors, upgraded electronics, fully software trigger, ...
- A new general-purpose forward-region detector at LHC



Trigger strategies

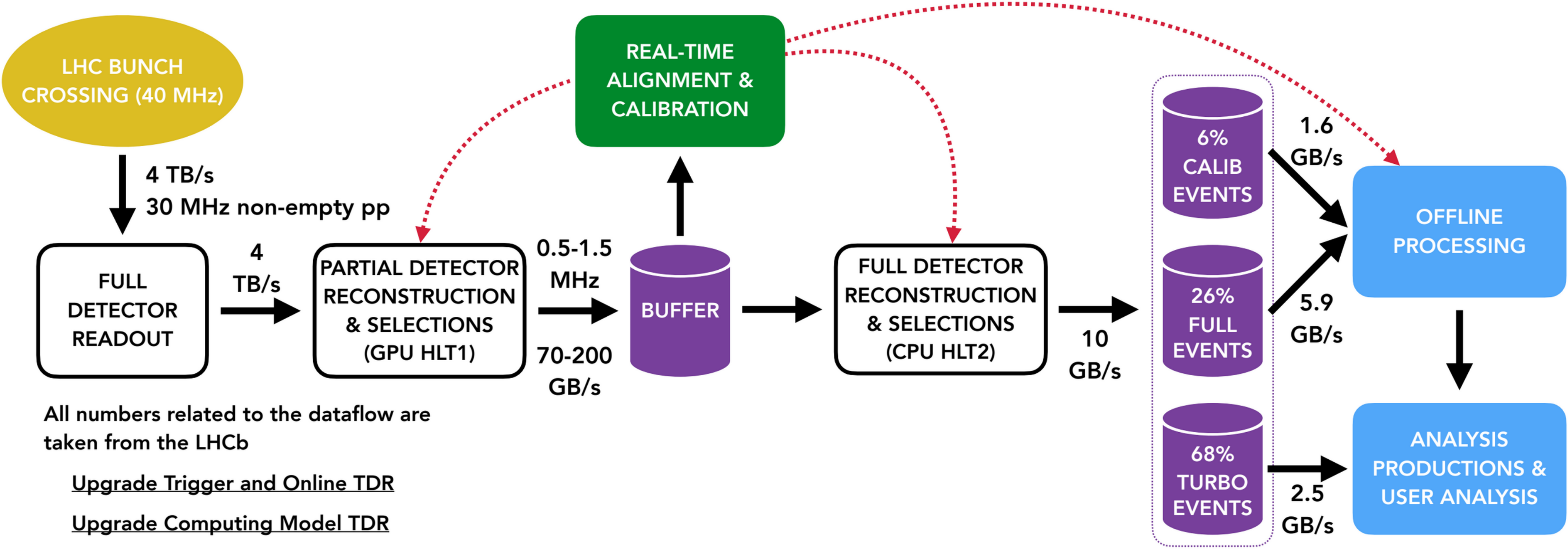
- Almost every pp collision is interesting for LHCb as is contains a heavy quark (*b*, *c*)



A. Cerri LHCP2022

LHCb trigger design

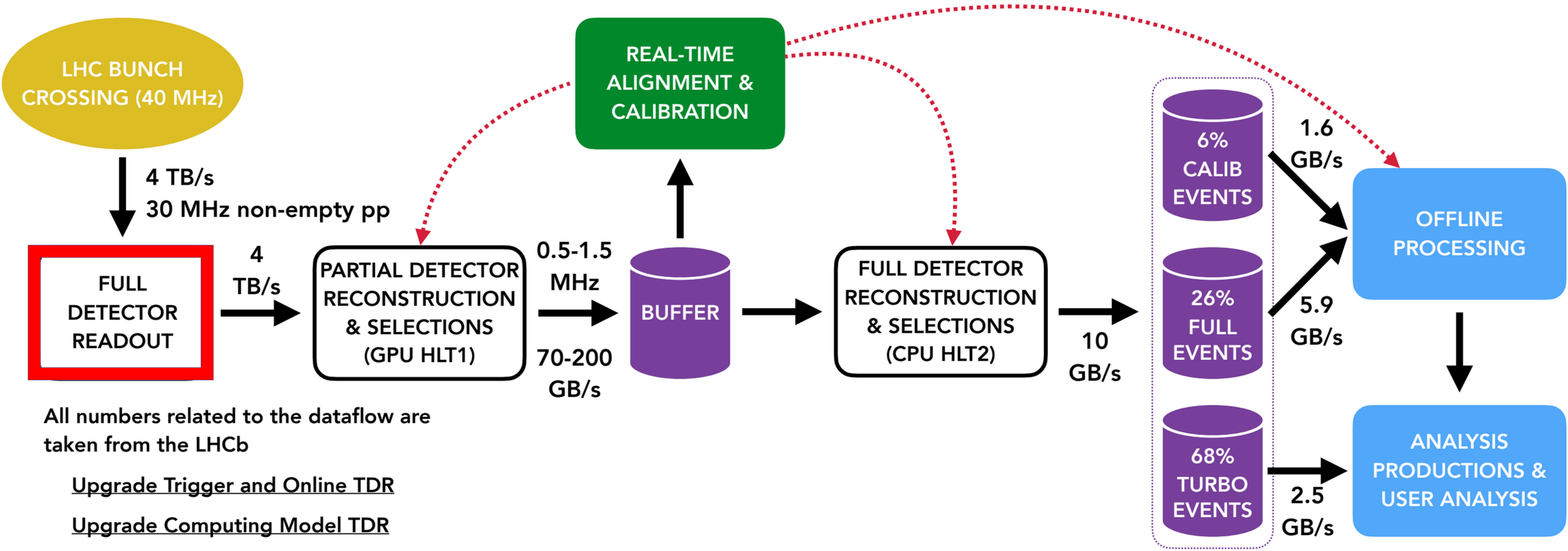
- Full offline-quality reconstruction achieved at the trigger level
- Online and offline processing are using same code base



- Modern computing approaches used in both HLT stages:
  - HLT1: Fully GPU based stage
  - HLT2: modern Multi-threading, task-based scheduling of algorithms, vectorization, ...

LHCb-FIGURE-2020-016

LHCb trigger design: DAQ and Online



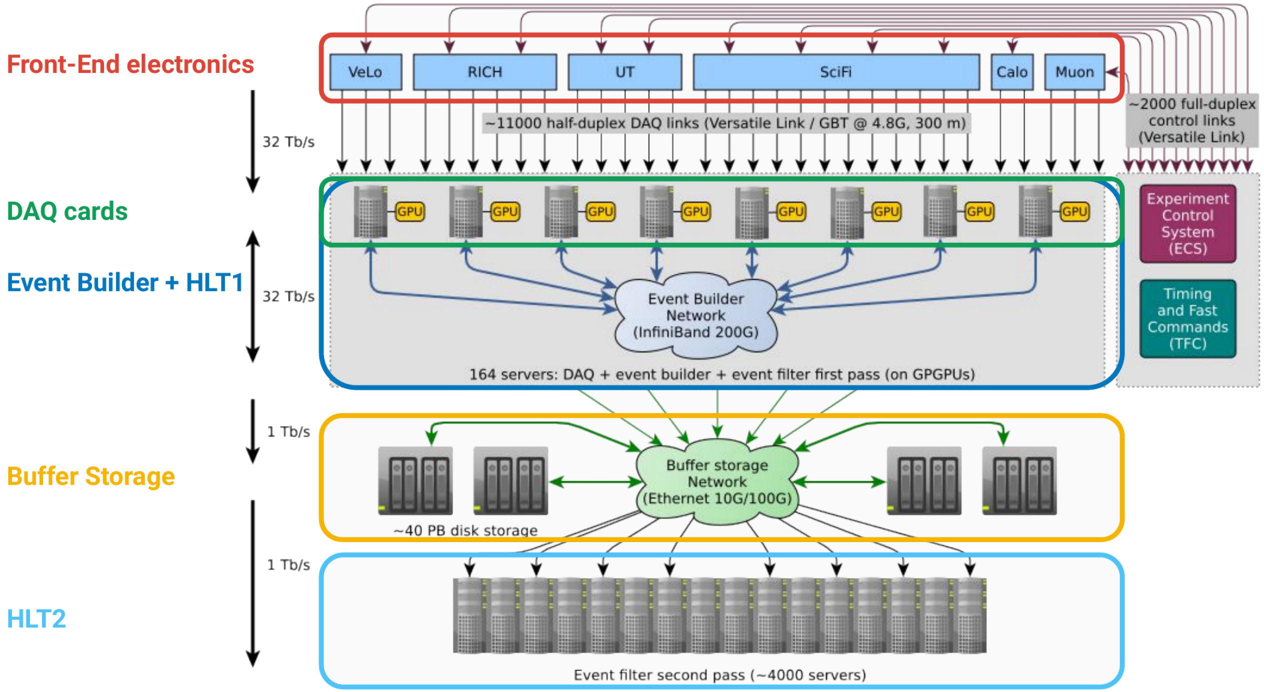
All numbers related to the dataflow are taken from the LHCb

Upgrade Trigger and Online TDR

Upgrade Computing Model TDR

LHCb-FIGURE-2020-016

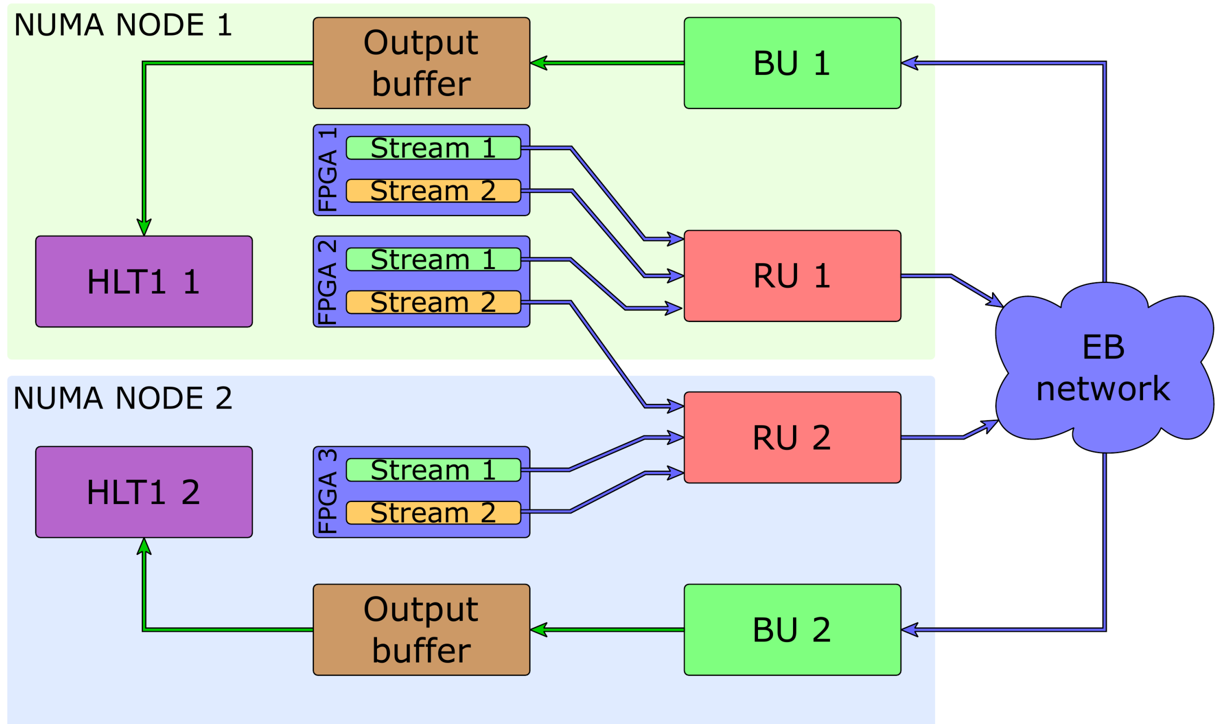
Online architecture: overview



F. Pisani, CHEP2023

- Full detector readout performed by O(500) custom PCIe40 FPGAs
- Event Builder (EB) farm consists of 173 servers with 3 free PCIe slots per server
- HLT1 stage implemented directly within the EB farm as HLT1 approach is inherently parallelizable
  - 2 GPUs (NVIDIA RTX A5000) per server -> 346 GPUs installed in total
- Allows a lighter network post EB

Online architecture: data flow



- Custom-made modular architecture built in C++
- Readout unit (RU): reads the data from DAQ card and send it to the EB network
- Builder unit (BU): reads the data from the EB network and writes the built data into the HLT1 input buffer
- The scheduling synchronization is achieved by using an in-band data barrier
- Buffer-isolated critical section to minimise slowdowns and deadtime

Custom FPGA: PCIe40

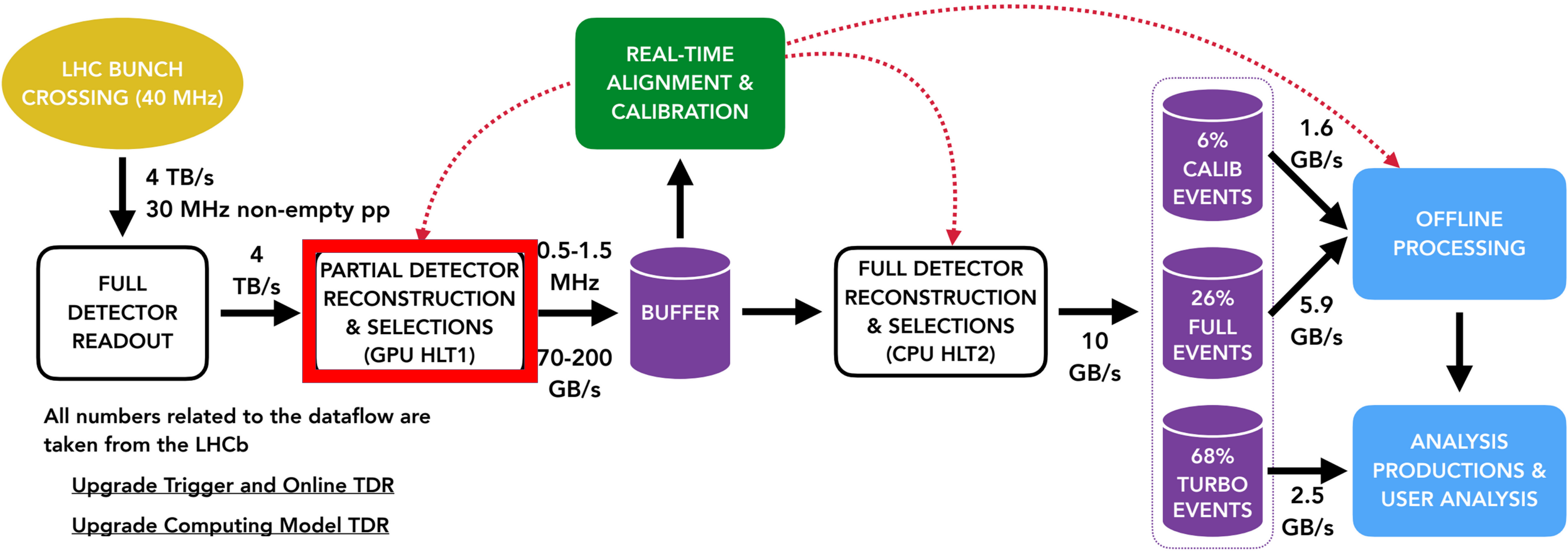
- Custom built card based on Intel Arria10
- 48x10G capable transceiver on 8xMPO for up to 48 full-duplex Versatile links
- 2 dedicated 10G SFP+ for timing distribution
- 2x8 Gen3 PCIe
- One card can serve in different roles based on FW:
  - Readout Supervisor (SODIN)
  - Interface board (SOL40)
  - DAQ card (TELL40)

Data output per sub-detector

- Multiple Event Packet (MEP): HLT1 input format combining several events
  - 1 MEP contains 30 000 events

Sub-detector	fragment size [B]	#tell40 streams	event size [B]	event fraction	MEP size [GB]	MFP size [MB]	RU send size [MB]
Velo	156	104	16250	0.13	0.49	4.69	14.06
UT	100	200	20000	0.16	0.60	3.00	9.00
SCIFI	100	288	28800	0.23	0.86	3.00	9.00
Rich 1	166	132	22000	0.18	0.66	5.00	15.00
Rich 2	166	72	12000	0.10	0.36	5.00	15.00
Calo	156	104	16250	0.13	0.49	4.69	14.06
Muon	156	56	8750	0.07	0.26	4.69	14.06
Total	1000	956	124050	1	3.72	30.06	90.19

LHCb trigger design: HLT1



All numbers related to the dataflow are taken from the LHCb

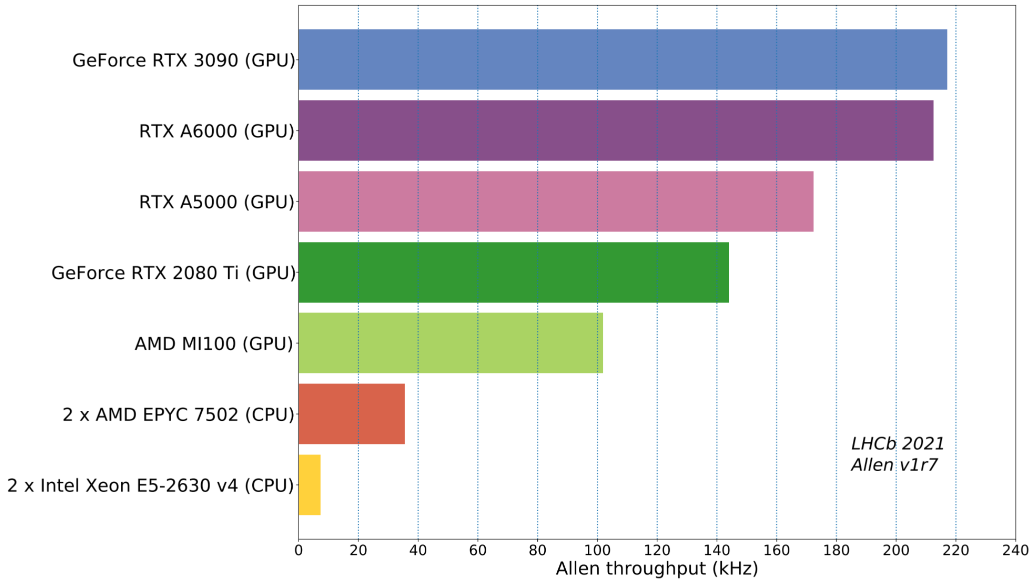
Upgrade Trigger and Online TDR

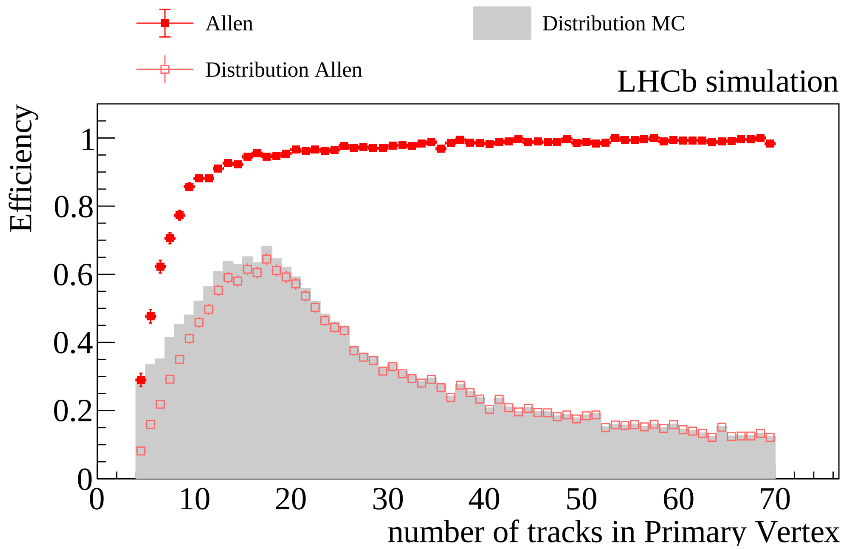
Upgrade Computing Model TDR

LHCb-FIGURE-2020-016

HLT1: overview

- The goal of HLT1 is to process of the LHCb raw data at 30 MHz and reduce rate by a factor of 30 (to 1 MHz)
- HLT1 is implemented in the form of Allen project [Comput. Soft. Big Science 4, 7 (2020)], using RTX A5000
  - Cross-architecture support: x86, CUDA/CUDACLANG (NVIDIA GPUs), HIP (AMD GPUs)
- Partial event reconstruction: vertexing, tracking, muon PID, simplified CALO information
- Rough selection based on O(50) trigger lines covering LHCb physics program
  - High/low pT muons, NN-based one-/two-track selection, detached lines, ...
- Required performance obtainable using O(200) GPUs, 346 RTX A5000 installed

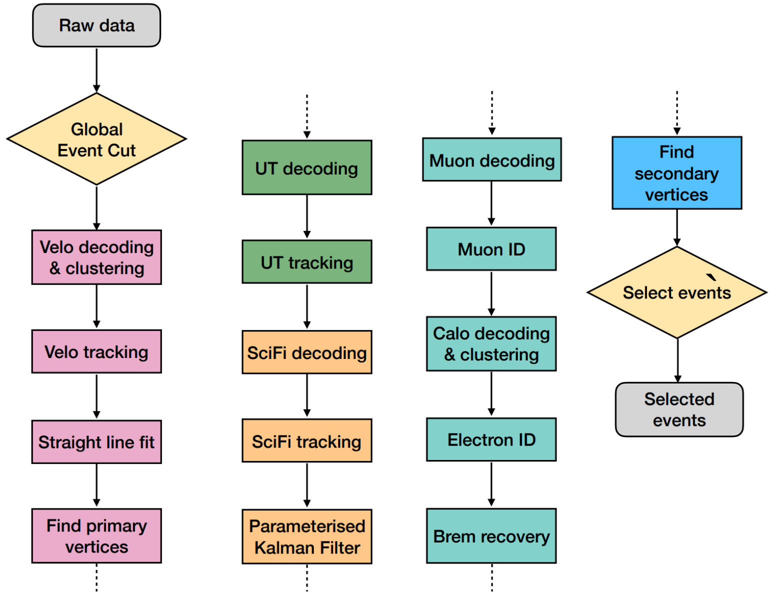


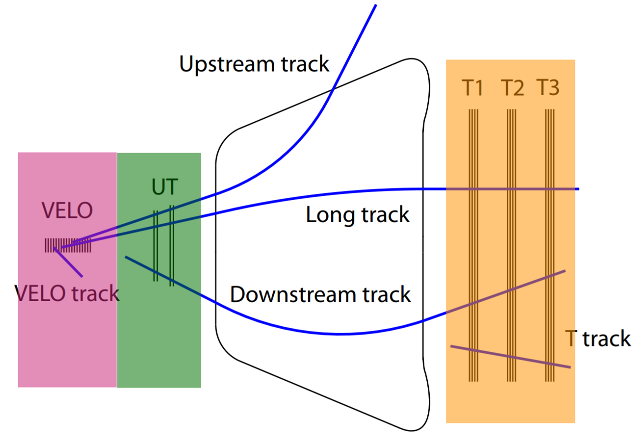


LHCb-FIGURE-2020-014

HLT1: overview

- The goal of HLT1 is to process of the LHCb raw data at 30 MHz and reduce rate by a factor of 30 (to 1 MHz)
- HLT1 is implemented in the form of Allen project [Comput. Soft. Big Science 4, 7 (2020)], using RTX A5000
  - Cross-architecture support: x86, CUDA/CUDACLANG (NVIDIA GPUs), HIP (AMD GPUs)
- Partial event reconstruction: vertexing, tracking, muon PID, simplified CALO information
- Rough selection based on O(50) trigger lines covering LHCb physics program
  - High/low pT muons, NN-based one-/two-track selection, detached lines, ...
- Required performance obtainable using O(200) GPUs, 346 RTX A5000 installed



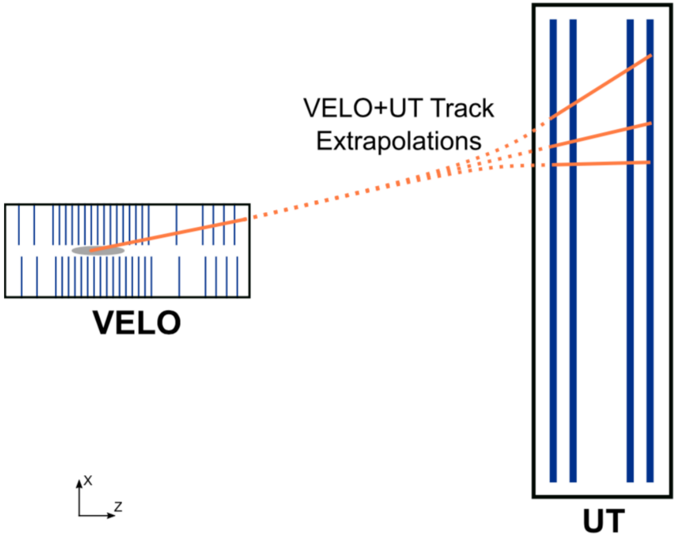


HLT1: Why GPUs?

- LHCb had been pursuing GPU reconstruction algorithms since 2012, and by 2014 most of work on the vertex detector reconstruction algorithm and associated infrastructure done
- However porting single algorithms to GPUs was not going to work: no single algorithm was expensive enough to make an “off loading” model cost-effective, and no model for multi-event processing on a GPU at that time
  - Off loading is instead used at ALICE and CMS for specific tasks
- At the end of 2017 it was decided to try to put the entire HLT1 on GPUs and develop dedicated multi-event scheduling
- Proved to be a successful decision leading to full implementation of HLT1 functionality on GPUs
  - The very first fully GPU-based trigger at HEP

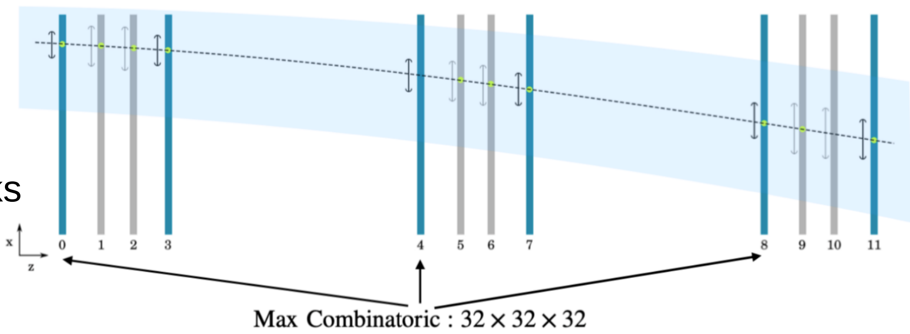
HLT1: Tracking per sub-detector

- Velo tracking [Journal of Computational Science, vol. 54, 2021]
  - 26 silicon pixel modules with $\sigma_{x,y} \sim 5\ \mu m$
  - Local paralleled clustering algorithm (Search by Triplet)
  - Tracks fitted with simple Kalman filter assuming straight line model



- UT tracking [IEEE Access, vol. 7, pp. 91612-91626, 2019]
  - 4 layers of silicon strips
  - Velo tracks extrapolated to UT taking into account B field
  - Parallelized trackless finding inside search window requiring at least 3 hits

- SciFi tracking [Comput Softw Big Sci 4, 7 (2020)]
  - 3 stations with 4 layers of Scintillating Fibres
  - Velo-UT tracks extrapolated using parametrisation
  - Parallelized Forward algorithm to reconstruct long tracks
  - Search windows from Velo-UT momentum estimate
  - From triplets and extend to remaining layers

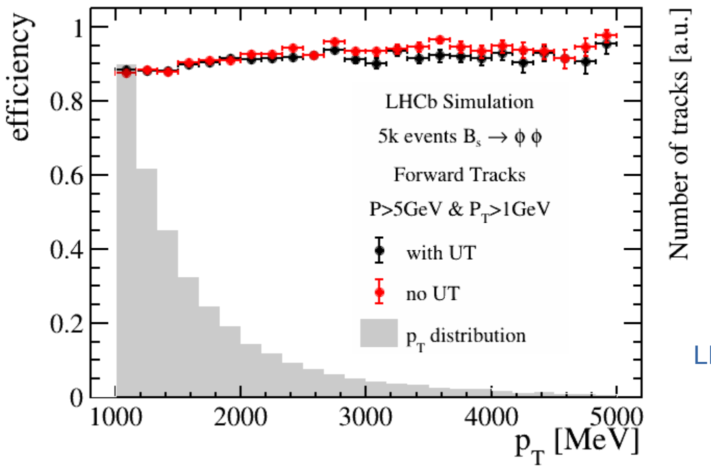


HLT1: Tracking overview

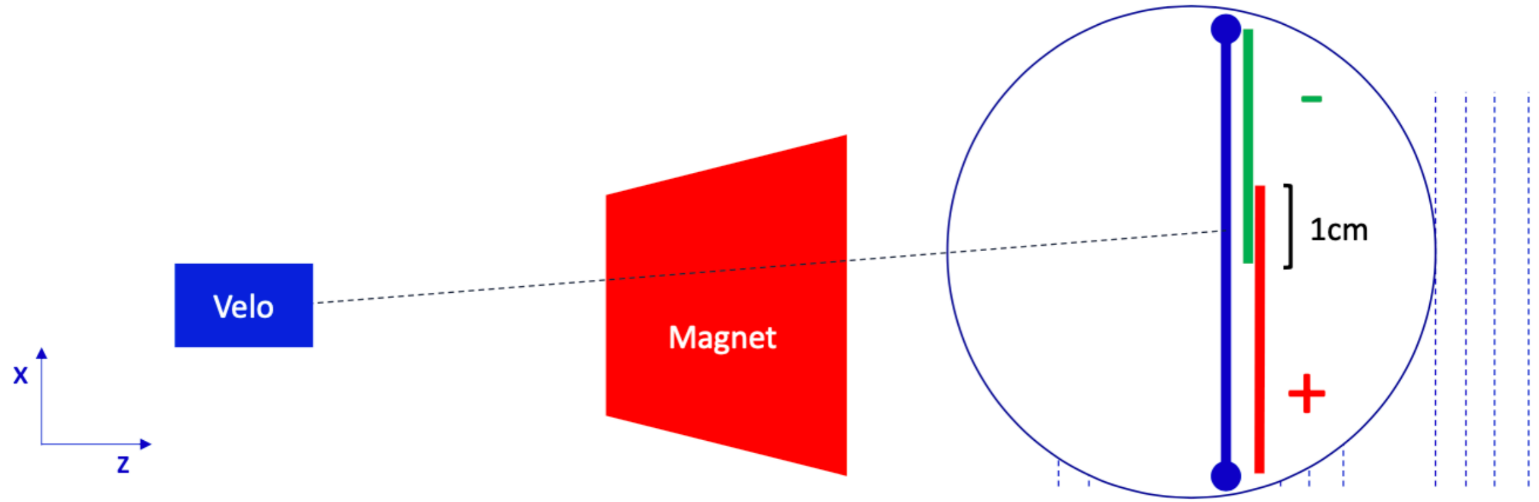
- Due to unavailable UT in 2022 and 2023 two independent tracking approaches were used in HLT1 together

1) Forward tracking without UT

- VELO track is extrapolated as a straight line
- Two search windows in SciFi (based on charge)
- Assumed: p > 5 GeV and p_T > 1 GeV



LHCb-FIGURE-2023-007

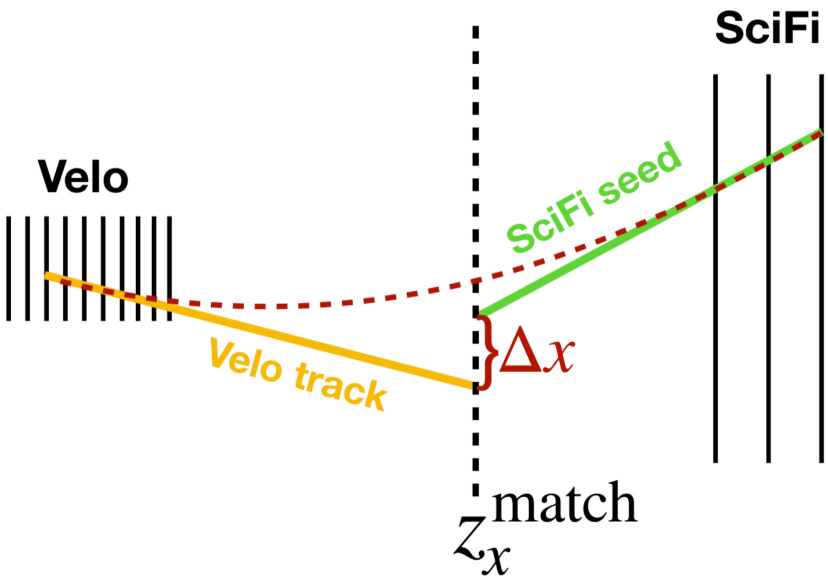


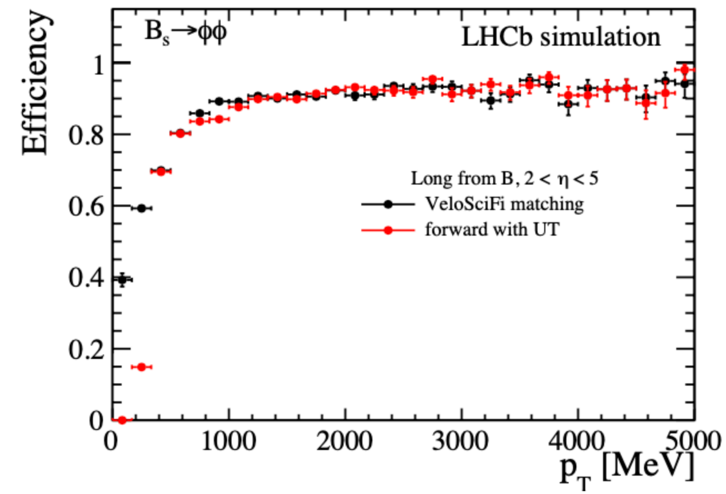
HLT1: Tracking overview

- Due to unavailable UT in 2022 and 2023 two independent tracking approaches were used in HLT1 together

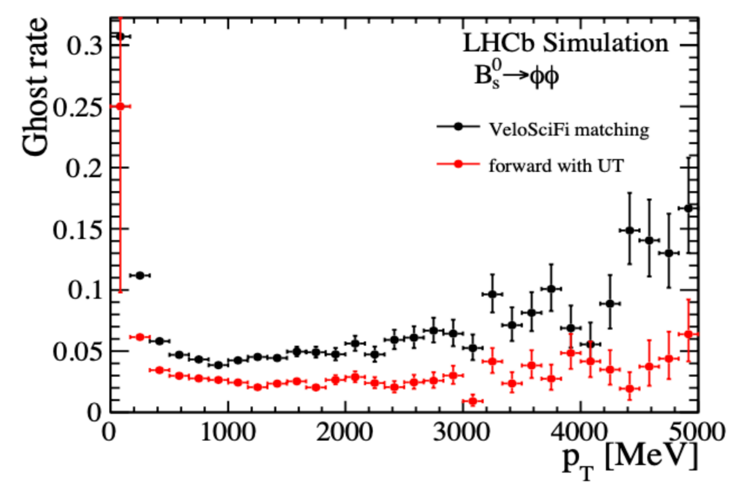
2) Seeding and Matching

- Standalone reconstruction of SciFi tracks
- Matching SciFi tracks to VELO seeds
- Efficient for a low momentum tracks



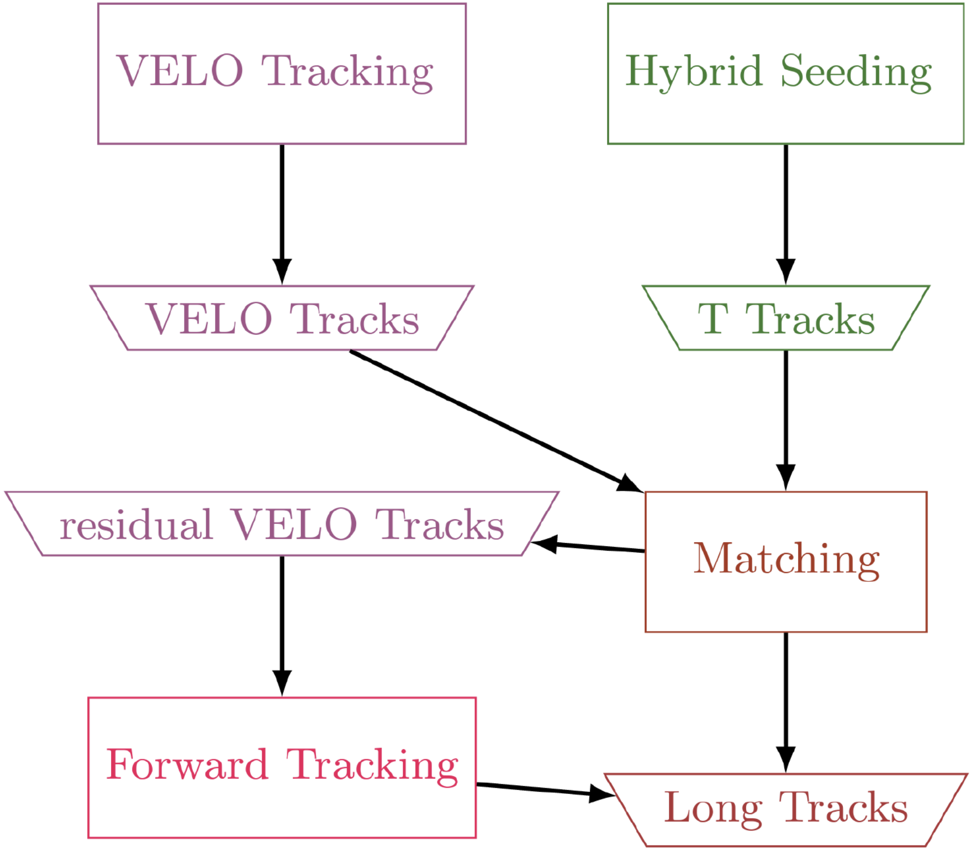


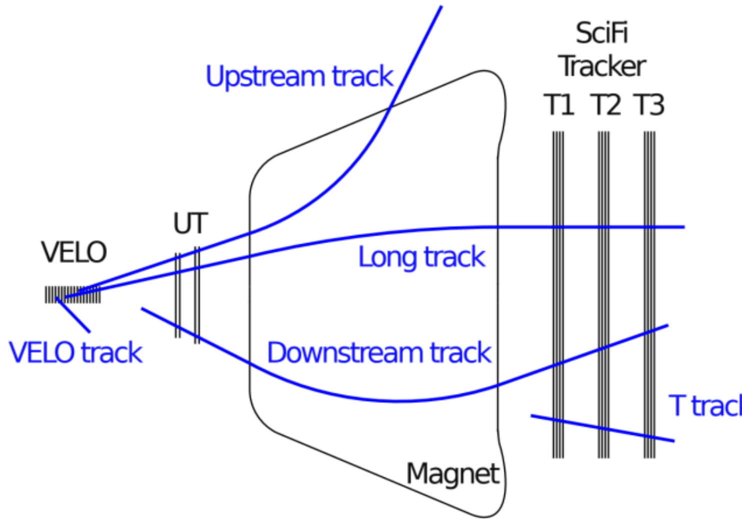
LHCb-FIGURE-2022-010



HLT1: Nominal HLT1 sequence

- Nominal HLT1 tracking sequence combines both forward tracking and Seeding and Matching
- Originally developed for HLT2, lately found efficient also for HLT1

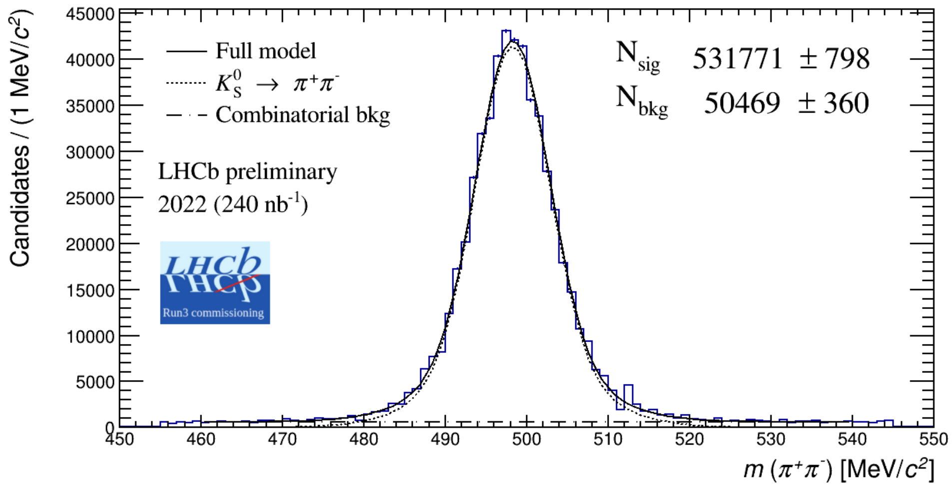


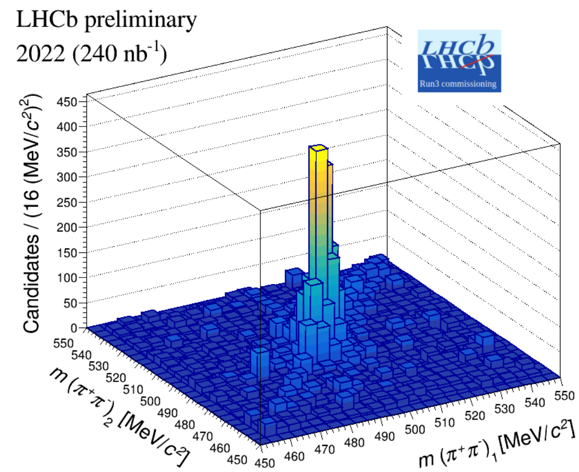


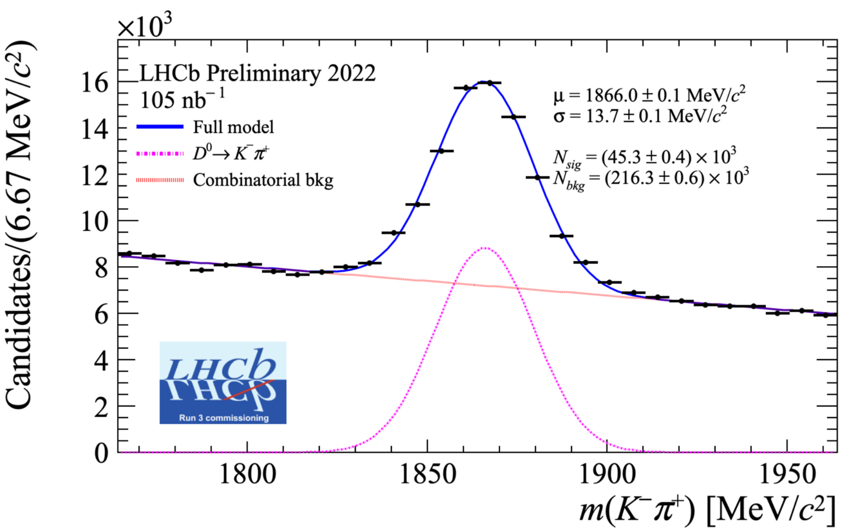
HLT1: 2022 performance

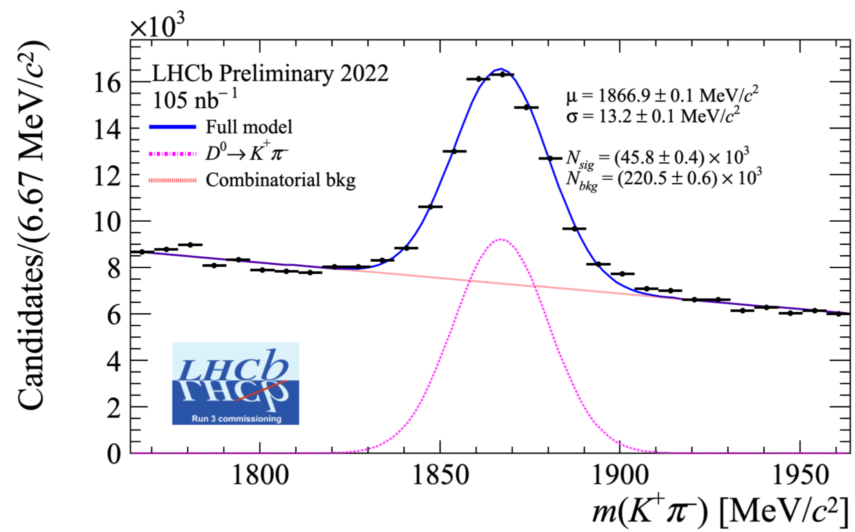
LHCb-FIGURE-2023-005
LHCb-FIGURE-2023-009

→ Good performance in 2022

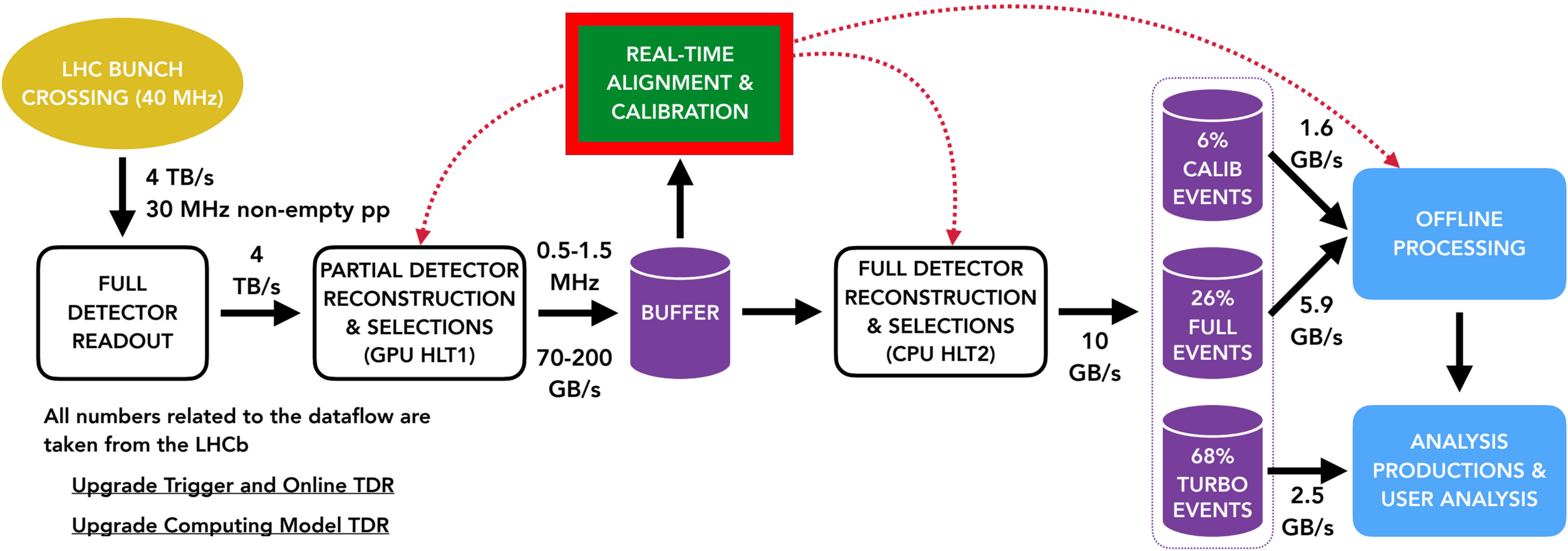








LHCb trigger design: Alignment and calibration



All numbers related to the dataflow are taken from the LHCb

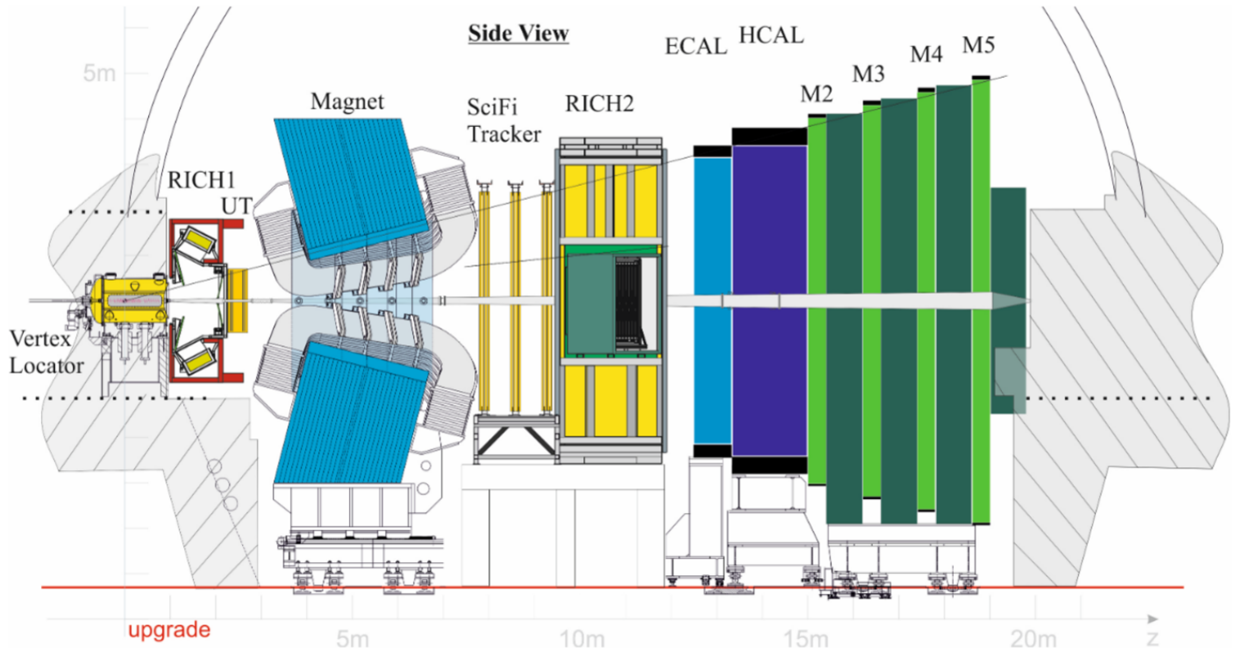
Upgrade Trigger and Online TDR

Upgrade Computing Model TDR

LHCb-FIGURE-2020-016

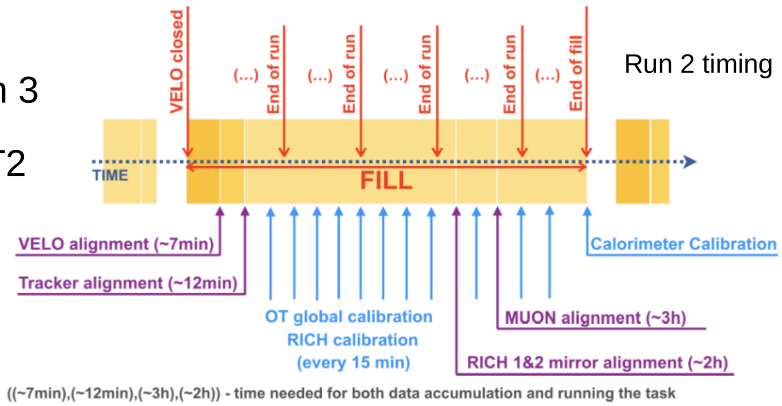
Alignment and calibration

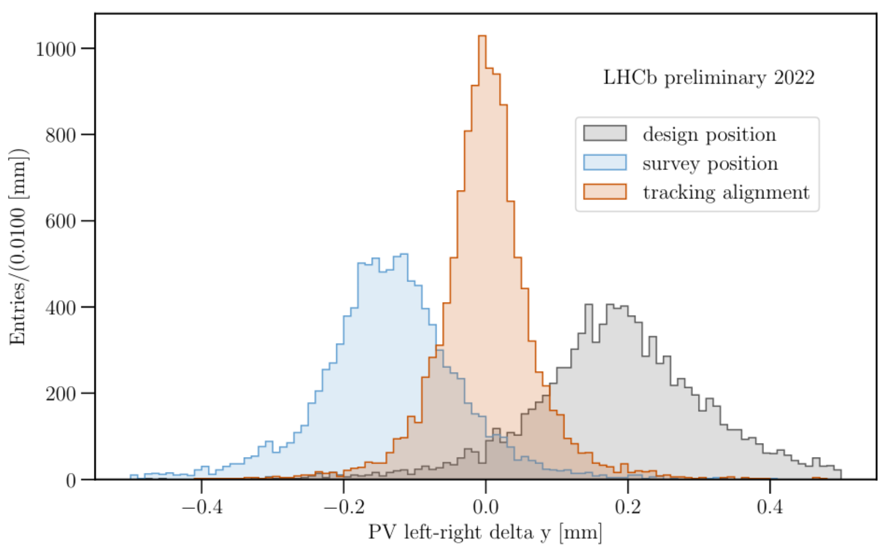
- Fully aligned and calibrated data needed before running HLT2
- Online alignment and calibration pioneered in Run 2, crucial in Run 3
- Buffer with a capacity of O(10 PB) situated between HLT1 and HLT2
- LHCb distinguish two processes:
  - Alignment: VELO, RICH mirrors, UT, SciFi, Muon
  - Calibration: RICH, ECAL, HCAL (also offline part)



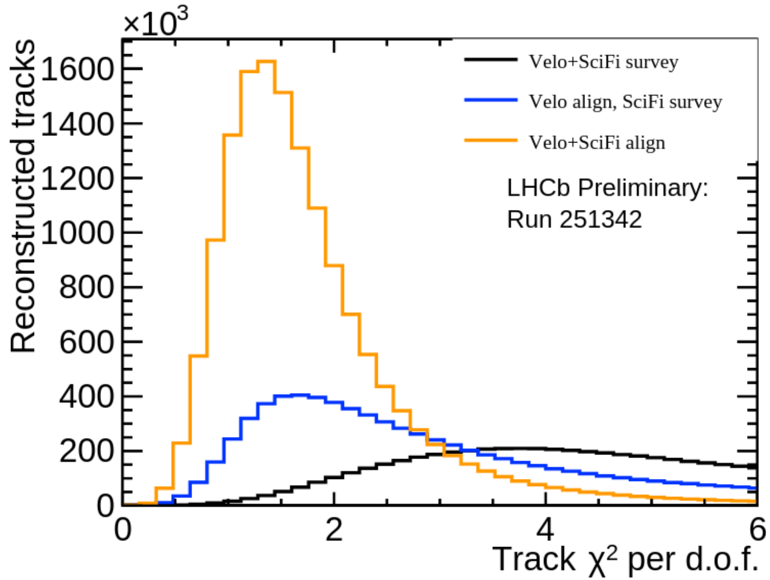
Alignment and calibration

- Fully aligned and calibrated data needed before running HLT2
- Online alignment and calibration pioneered in Run 2, crucial in Run 3
- Buffer with a capacity of O(10 PB) situated between HLT1 and HLT2
- LHCb distinguish two processes:
  - Alignment: VELO, RICH mirrors, UT, SciFi, Muon
  - Calibration: RICH, ECAL, HCAL (also offline part)





LHCb-FIGURE-2022-016

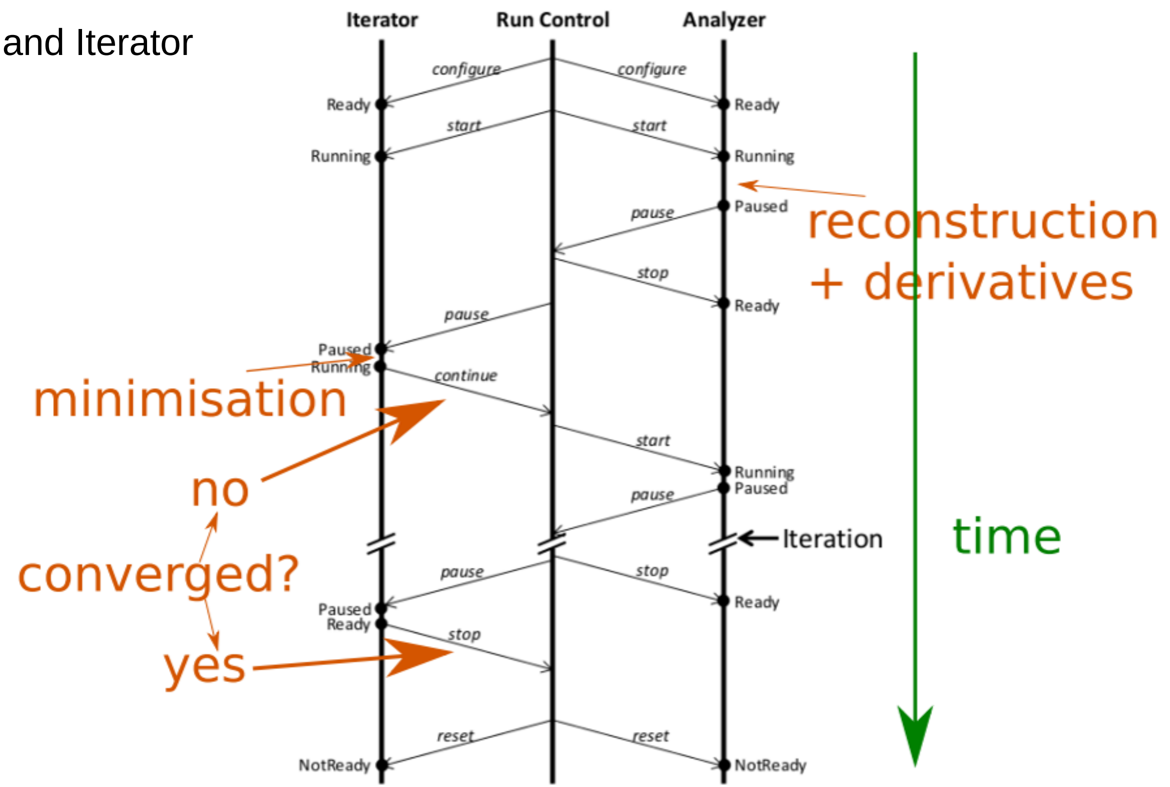


LHCb-FIGURE-2022-018

Alignment and calibration: Alignment

- Alignment and calibration procedure is running multi-threaded on roughly 160 CPU nodes
- Based on a set of HLT1 events selected by a dedicated HLT1 lines
- Implemented as finite-state machine steered by Run Control (fully integrated into ECS)

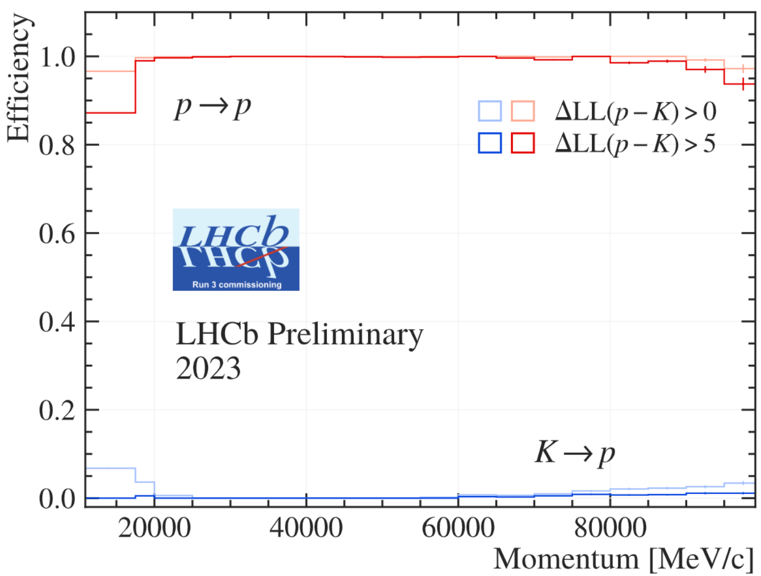
- Alignment procedure is based on Analyzer and Iterator
- Analyzer:
  - Run reconstruction
  - Calculating alignment constants
- Iterator:
  - Collect derivatives/histograms
  - Obtain new constants
  - Convergence check



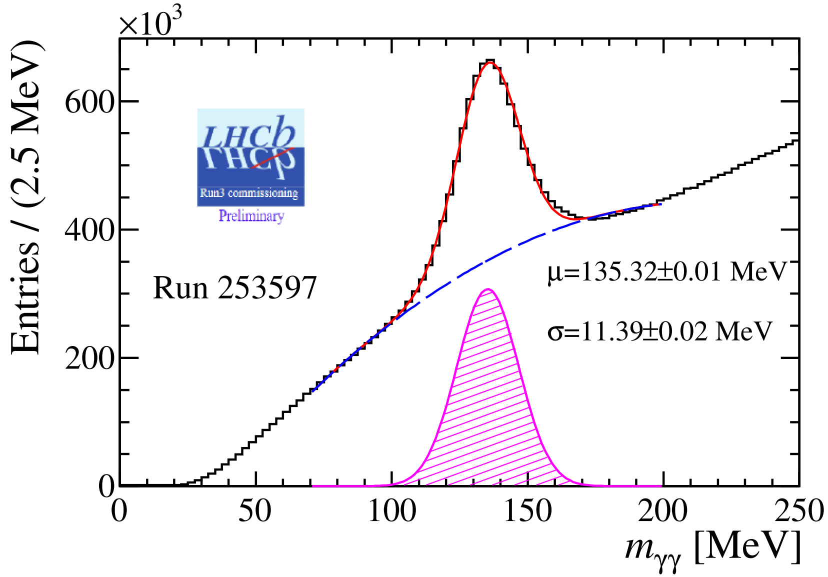
F. Reiss CTD 2022

Alignment and calibration: RICH and ECAL

- Particle identification (PID) is one of the most crucial inputs for HLT2 selection and offline analyses
- RICH detectors are the main PID providers at LHCb used to distinguish between charged hadrons
  - Kaon and pions are obtained from $D^* \to D^0(\to K^-\pi^+)\pi^+$, protons from $\Lambda^0 \to p^+\pi^-$
- With a well calibrated ECAL, neutral pions can be reconstructed extending physics reach of LHCb

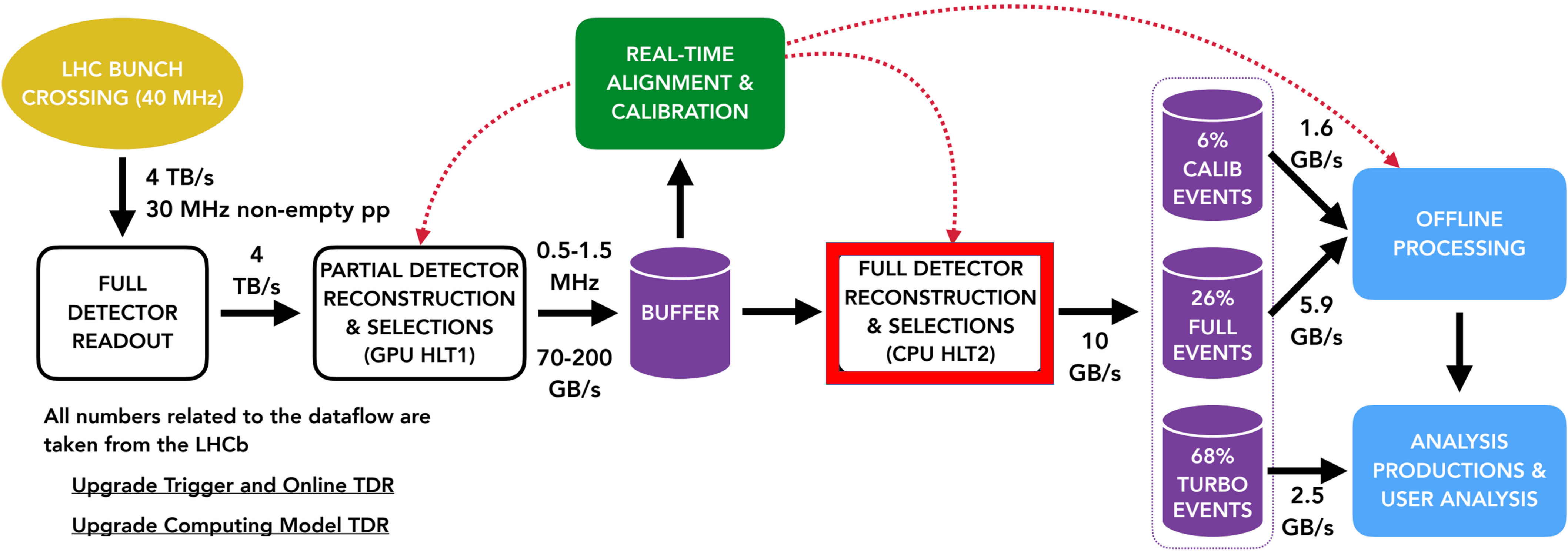


LHCb-FIGURE-2023-023



LHCb-FIGURE-2022-019

LHCb trigger design: HLT2



All numbers related to the dataflow are taken from the LHCb

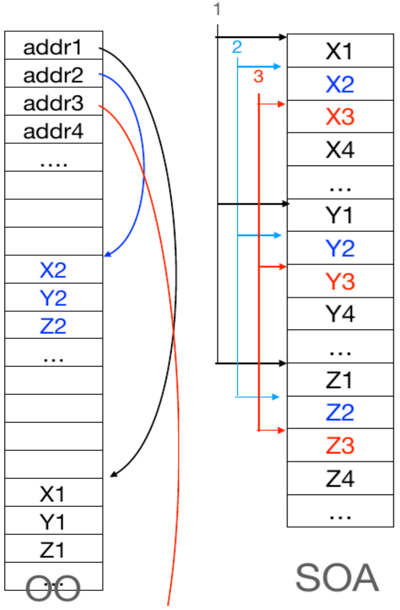
Upgrade Trigger and Online TDR

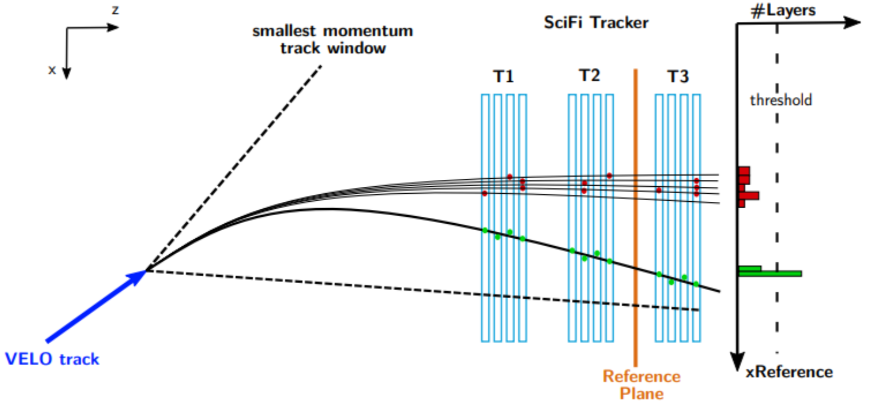
Upgrade Computing Model TDR

LHCb-FIGURE-2020-016

HLT2: Vectorisation and SoA data structure

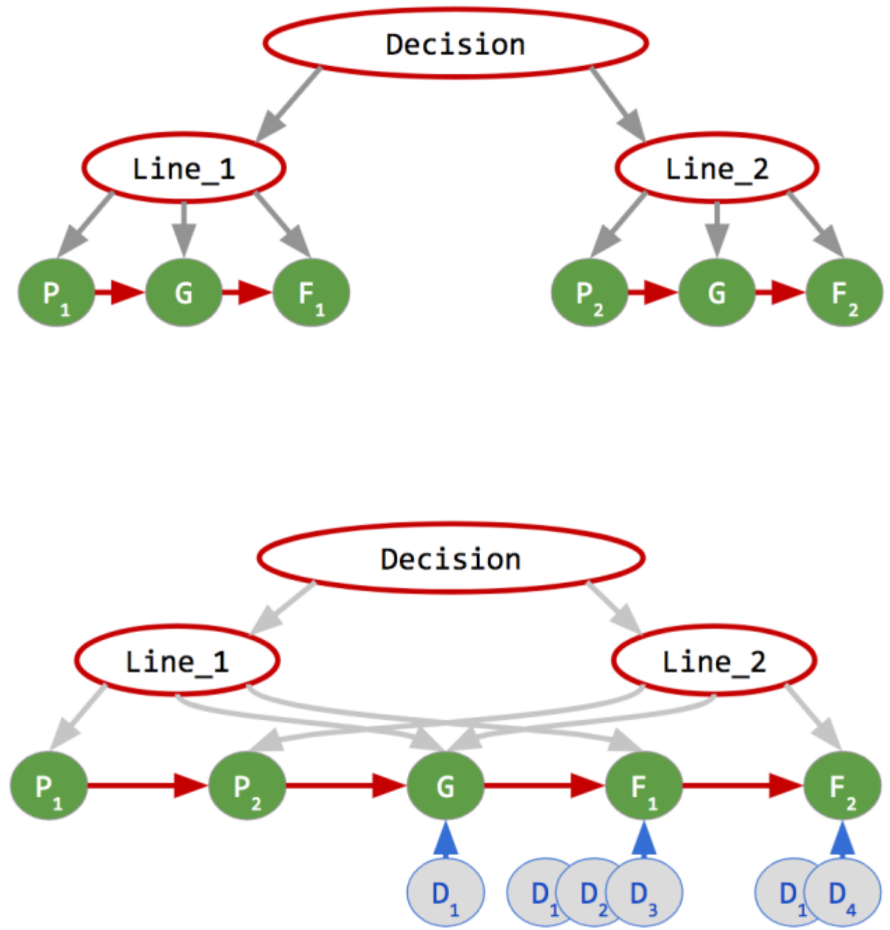
- Underlying data structure directly influences possible routes of the code development
- For a proper vectorisation it is beneficial to switch from OO to SoA representation
  - OO: Object oriented approach
  - SoA: Structure-of-Arrays
- Very different memory behaviour of different models:
  - OO: many small memory allocations, random jumps, copying objects
  - SoA: only reads what will be used, easily vectorisable
- However, SoA-based computing differs significantly in how data are accessed
  - Only a slices of SoA-collection are accessed, no objects (in OO terms)
- SoA relates well to SIMD (single instruction multiple data) approach
- Used in Forward tracking at HLT2:
  - Several thresholds can be scanned in parallel
  - 8 single precision floats/integers in parallel (AVX2)
  - Throughput of Forward Tracking increased up to 60%
- Ongoing studies of SoA structure usage in event model





HLT2: Event scheduler

- Dealing with O(2500) dedicated selection lines
- Multi-threading friendly algorithms needed
- Automatic handling of data and control flow
  - Data flow: Configurable properties with user defined input/output
  - Control flow: what to run and where to stop
- Handle the data flow with specific logical types
  - Order the basic nodes with specific control flow
- Automatically resolve data dependencies by matching input / output
- Static graph with ordered nodes (respecting data constrains)
- Configured during initialization
- Basic node: one algorithm with data dependencies (Basic node)
- Composite node: logic operation (AND, OR, NOT) (Composite node)

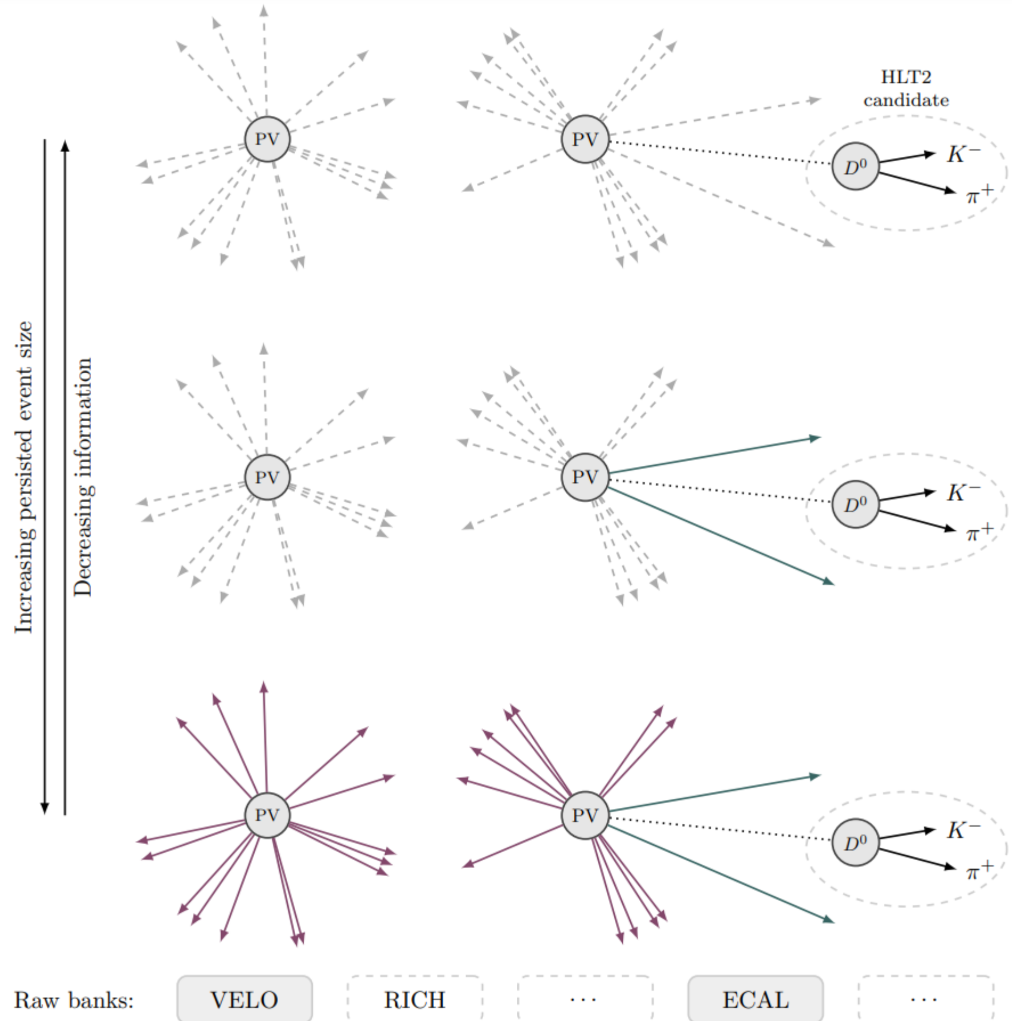


J. Phys.: Conf. Ser. 1525 012052

HLT2: Turbo model

- Bandwidth [GB/s]
  ≈ Accept Rate [kHz] ⨉ Event size [kB]
- Instead of saving of full event, only information needed for a physics analysis can be stored
- Extensively used during the Run II
  - Around 30 % of the trigger rate is Turbo – almost all Charm physics
  - But only about 10 % of the bandwidth!
  - Approximately 2/3 lines keep raw detector information (Turbo SP)
- Significant reduction of data size ⇒ more events at same bandwidth
- Flexible persistence settings
- <u>Baseline approach for Run 3 ≈ 70 % of events</u>

Persistence method	Average event size [kB]
Turbo	O(10)
Turbo++/SP	O(10-100)
Raw event	O(100)



JINST 14 (2019) 04, P04006

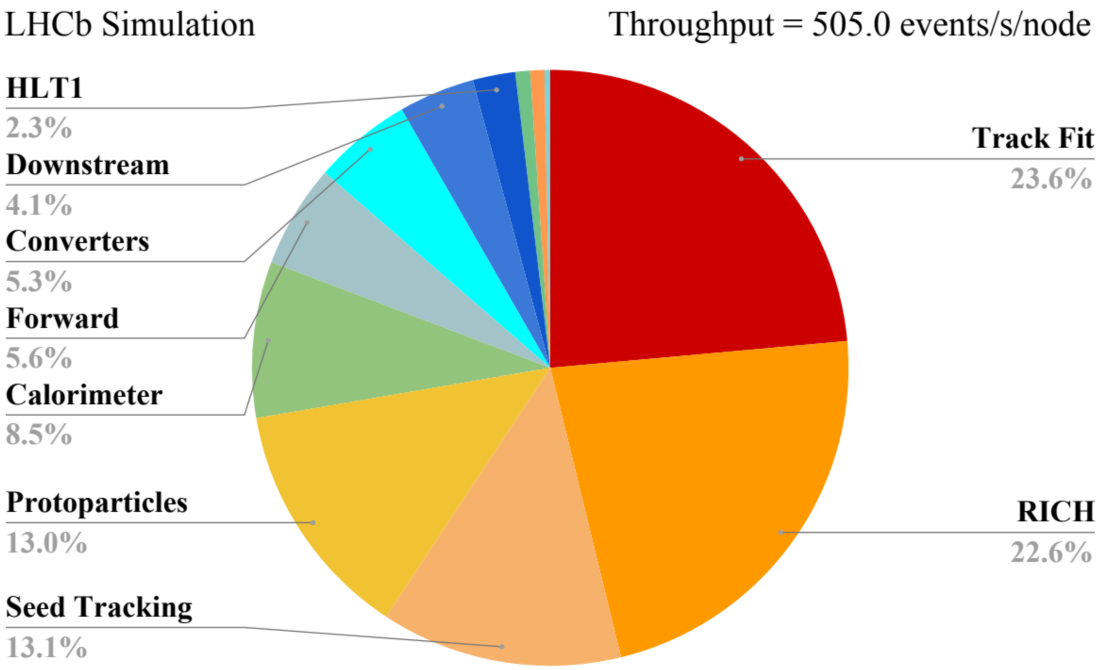
HLT2: Throughput oriented selection

- In Run2 reconstruction was about 70% and selection 30% of time spent
- Selection lines are written using Throughput Oriented (ThOr) functors (function objects)
  - Designed to be agnostic to input / output type and to be flexible on what they operate on
- Functors are composable allowing a simple chaining of basic functors
  - e.g. X @ POSITION @ VERTEX ⇒ Particle.vertex().position().x()
- Simultaneously developed for old and new event model
  - Significant speed up when using SoA model
- Using functor cache instead of just-in-time (JIT) compilation
  - Functors that are defined in python during build ⇒ compiled into a cache
- Compile time to be used natively in the application.

Compile	JIT compilation (s)	Compilation with cache (s)
Single functor	**11**	**13**
5 different functors	**20**	**13**
Typical HLT2 selection	**70**	**24**
Compilation rerun	**70**	**0**

HLT2: HLT2 throughput

- Full HLT2 throughput
  - Physics selection not included
- Optimised sequence:
  - Removing the redundancy in the reconstruction of Long Tracks
  - Using a partially parametrised Kalman Filter (material scattering)
  - Additional improvements in matching between tracks and ECAL clusters

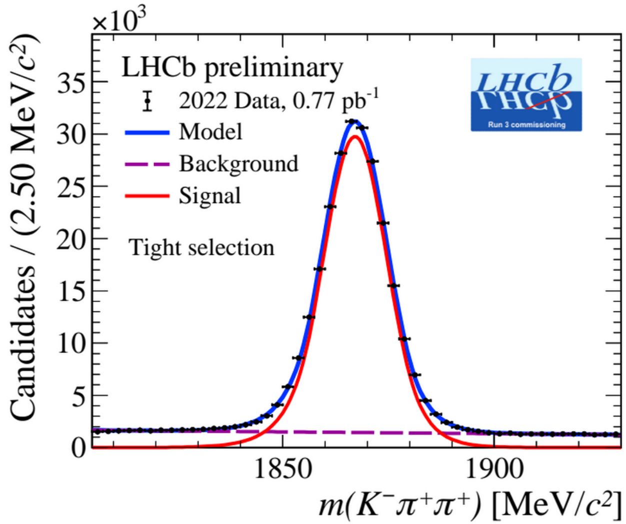


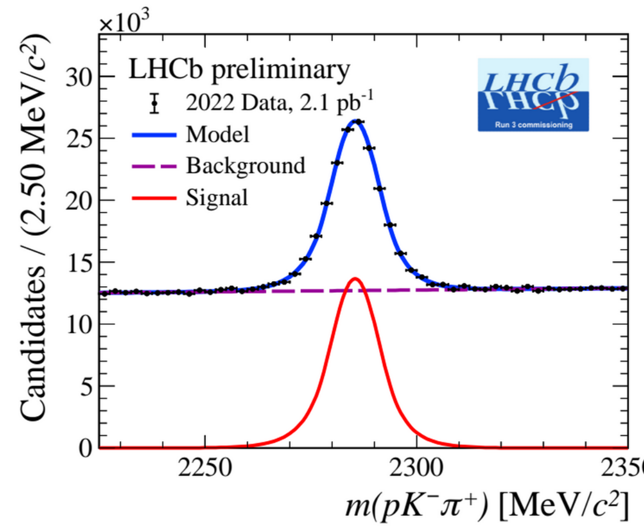
LHCb-FIGURE-2022-005

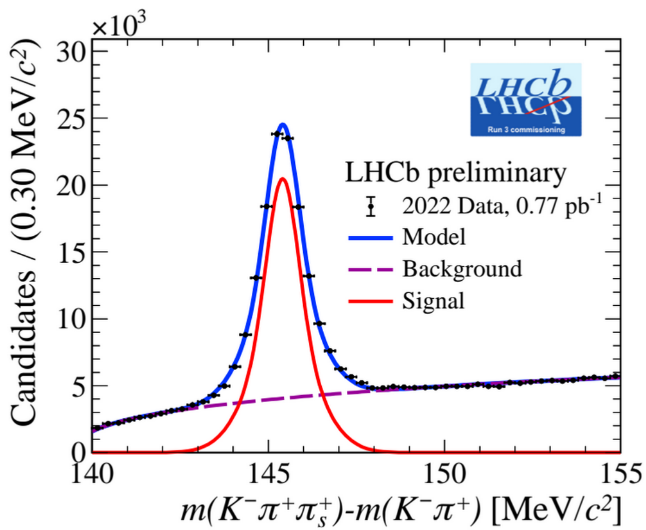
- Achieved the of goal running HLT2 at 500 Hz per node

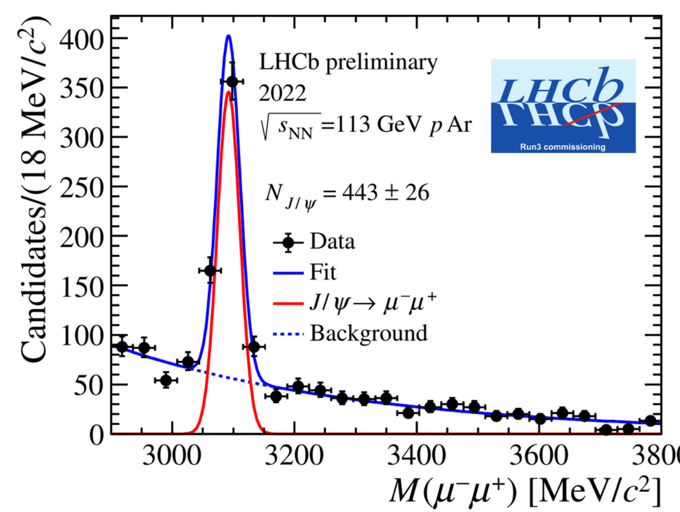
HLT2: First Run 3 results

LHCb-FIGURE-2023-005
LHCb-FIGURE-2023-011

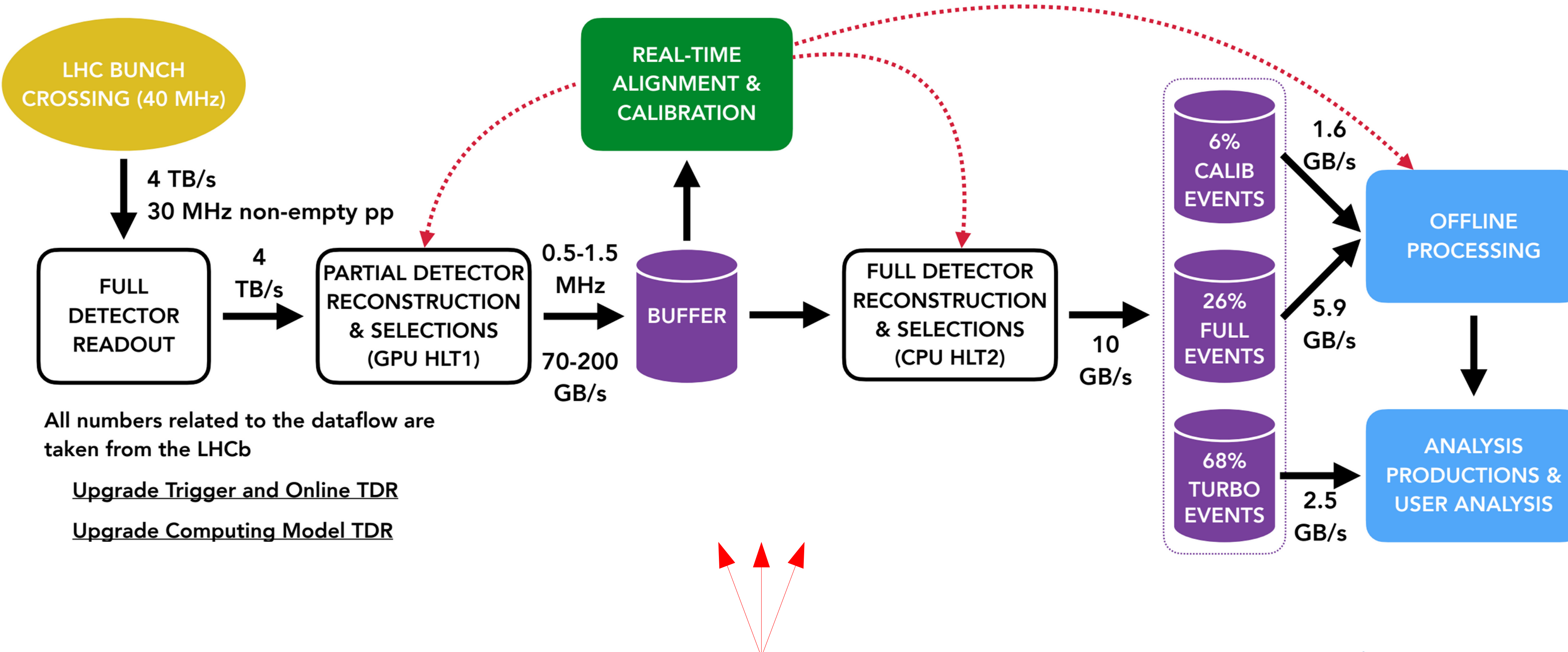








LHCb trigger design - software QA



All numbers related to the dataflow are taken from the LHCb

Upgrade Trigger and Online TDR

Upgrade Computing Model TDR

LHCb-FIGURE-2020-016

Software QA, testing and training

QA: Software quality assurance

- A dedicated working package with the goal is to improve and maintain the quality assurance of code
  - This includes also work on relevant documentation
- The LHCb software stack is a highly modular system based on Gaudi with code being hosted on GitLab
  - Large base of developers, concurrent development of parts of code base.
  - Submission of new code ⇒ Merge Request (MR).
- Changing one part may affect on rest of stack, such an impact may often be hidden
- As any large project, LHCb has an internal review policy for any contribution
  - Any MR must be reviewed before merging
- System depends on consisting of maintainers (experts) and shifters (junior members)
  - This assures that each line of code is fully reviewed by a relevant experts and senior LHCb software expert (maintainer)
  - Shifter helps with checking requirements of each MR and evaluating tests
  - Each contribution should be written as accessible to shifters who then can learn more about the LHCb code base – ideal place to learn both about LHCb software and computing

QA: Testing infrastructure at LHCb

- Testing infrastructure has two main parts:

- The LHCb nightly build system
  - Compile & Test & Compare: Built? Ran? Finalized? (code error or not)
  - O(300) cores, jobs managed by Jenkins
  - Can be run directly from GitLab using web-hooks for any MR

- The LHCb Performance Regression (LHCbPR)
  - Utilities same infrastructure as nightlies
  - Focus on physics variables (momentum, tracking efficiency, vertex…)
  - Configured by python scripts
  - The LHCbPR front-end (browser-based)
  - Quickly check and comparison of test results (histograms)

QA: Software training

- Extremely important for any code development is not to only gather experts but also to pass the knowledge
  - Basic introduction to software used at LHCb: StarterKit [lhcb.github.io/starterkit/]
  - Many experiments are missing more advanced tutorials covering core online and offline software

- LHCb organized 28 dedicated upgrade software hackathons during the last 6 years

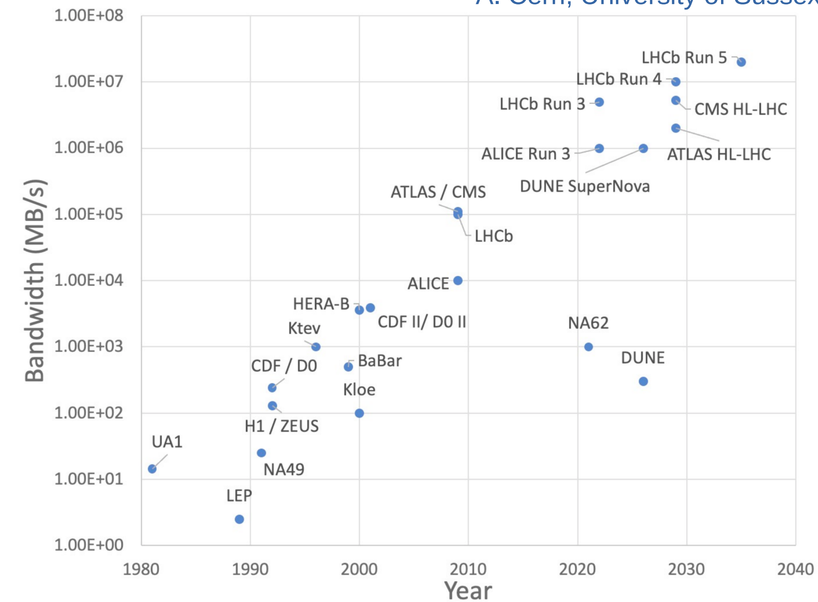
- Development of the new framework and training of a new contributors to all relevant aspects
  - Modern computing methods in general, heterogeneous (GPU) programming, FPGAs, …
- These skill are necessary for any modern HEP experiment, but (often) not taught at universities or even recognized in hiring / promotion
- Community-wide effort needed to train and keep those who decided focus also on computing aspects

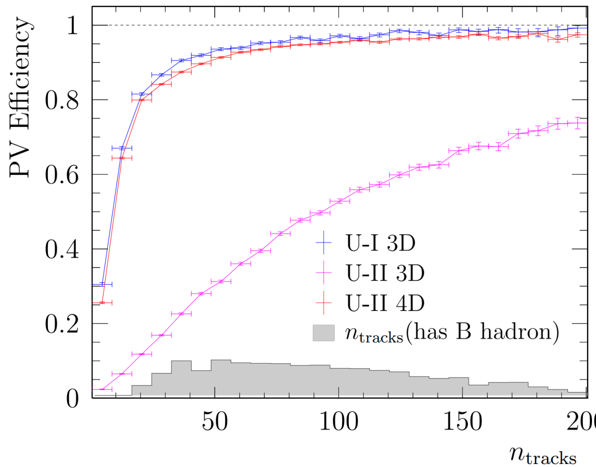
LHCb Upgrade 2
+
View on building a new trigger system

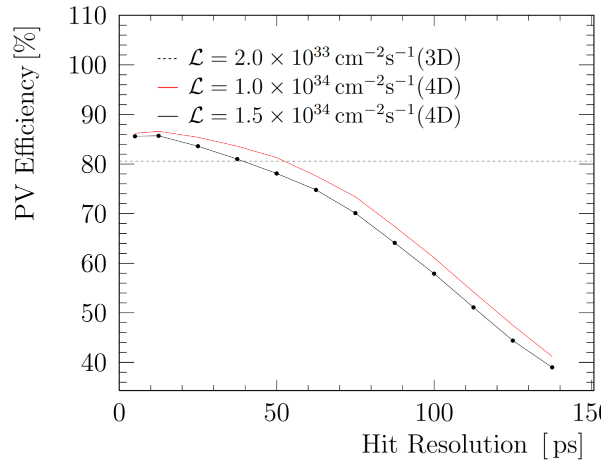
LHCb trigger in Run 5

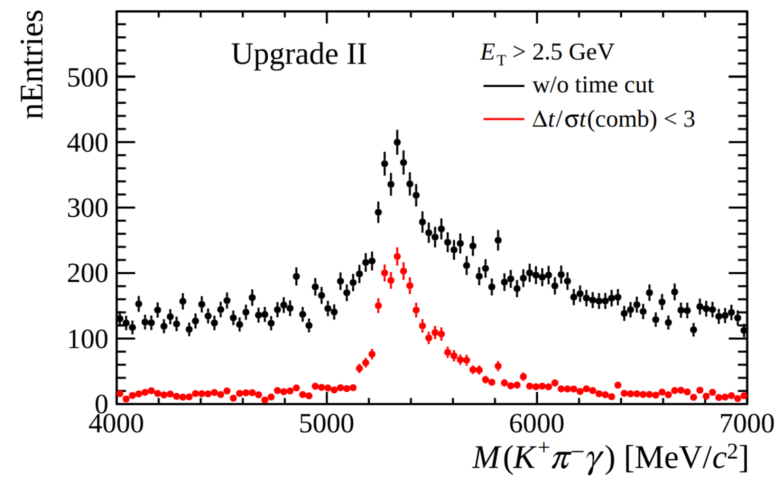
- LHCb is planning the Upgrade 2 for Run 5 and 6
  - FTDR approved this March: LHCb-TDR-023
- Luminosity: $1.5\text{x}10^{34}$ $cm^{-2}s^{-1}$; pileup: 40
- Significant challenge for DAQ and trigger
  - Currently estimated bandwidth around 25 TB/s
  - The highest expected value at HEP
- Adding timing to the tracking would mitigate effect of pileup
  - Ongoing hardware and software development

A. Cerri, University of Sussex









LHCb-TDR-023

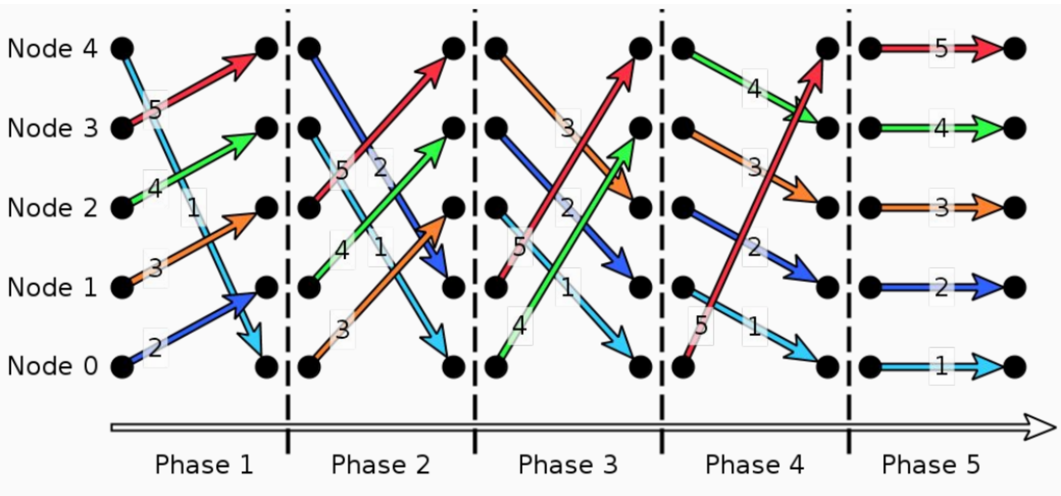
Future: assumptions for LHCb trigger in Run 5

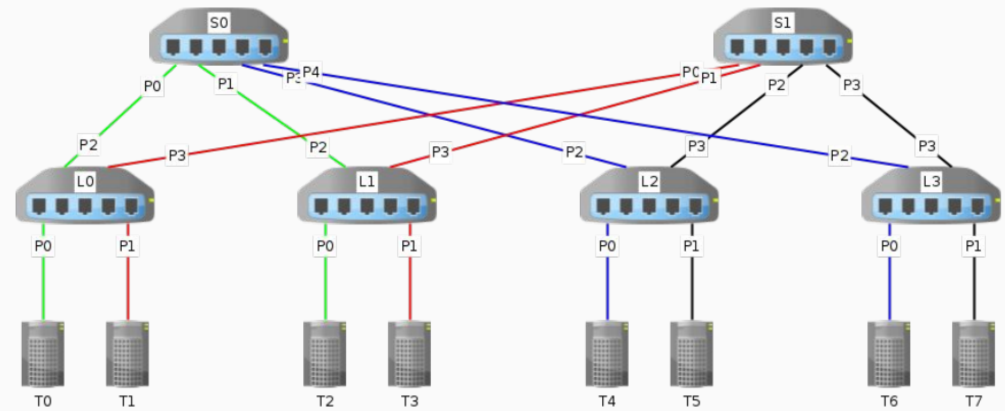
- Investigating even a broader interplay between various architectures
  - RETINA project: Ongoing study to use FPGAs for downstream tracking [JINST 17 C04011]
  - Investigate new architectures as TPUs, IPUs, … ?
- Implementing HLT2 using GPUs
  - Seems to be necessary to keep up with the ever rising input rate and broad physics goals
  - How to keep writing selection accessible to any member of the collaboration?
- Output bandwidth and offline storage has to be taken into account
  - Hard drive writing speed scaling is rather bad
  - Offline storage (data + MC) becoming a problem for any LHC experiment
- ML/AI algorithms are evolving fast -> mostly classifiers, anomaly detection, experiment control system, ...

Thank you for the attention

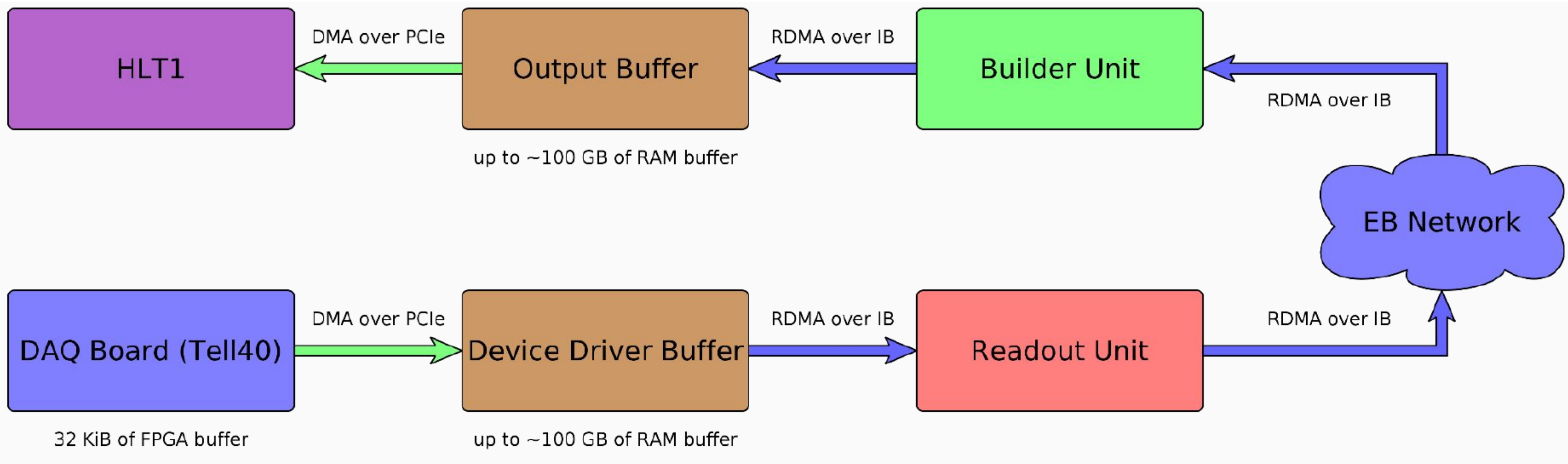
Spare slides

Online: traffic scheduling



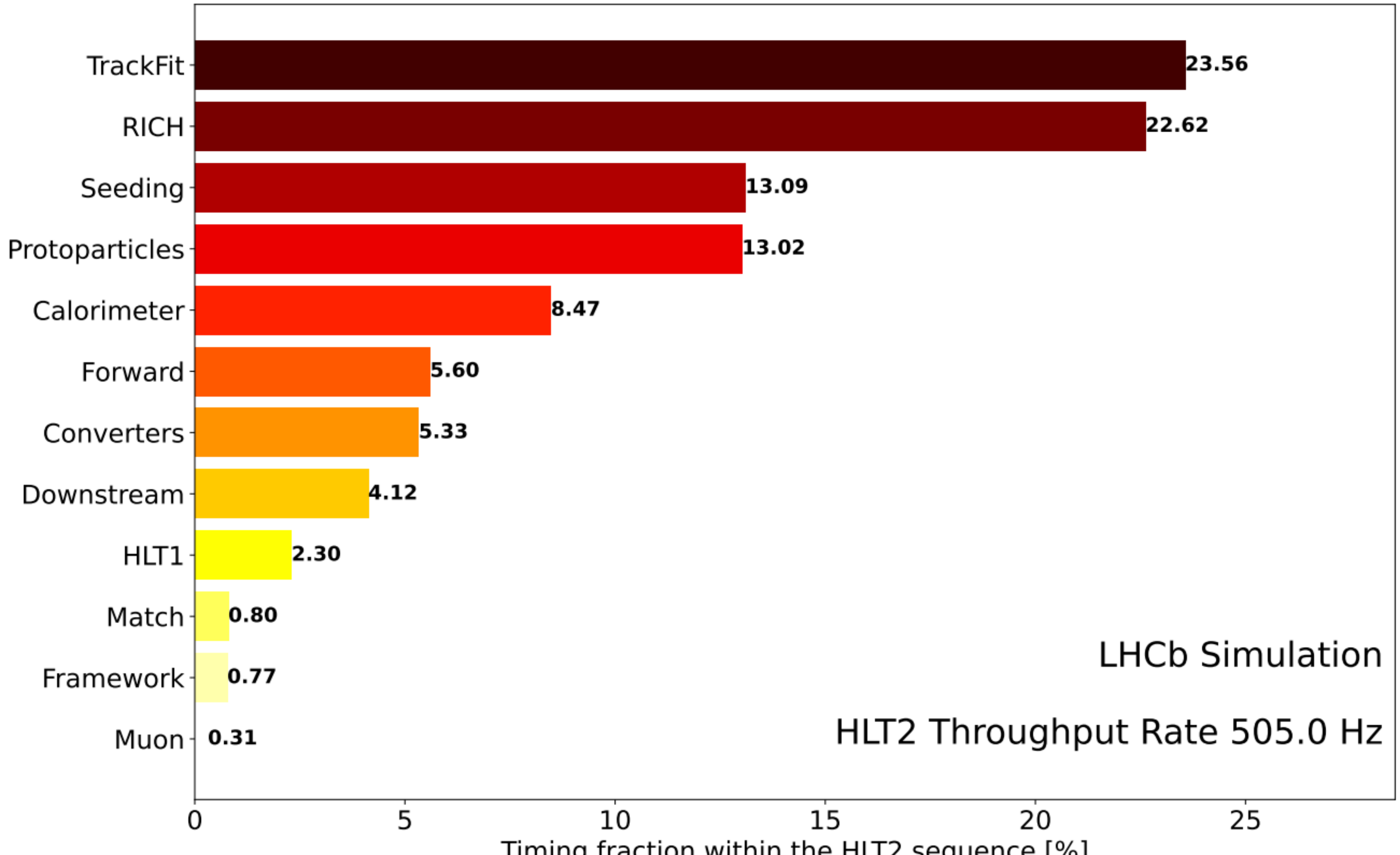


Online: latency and server flow

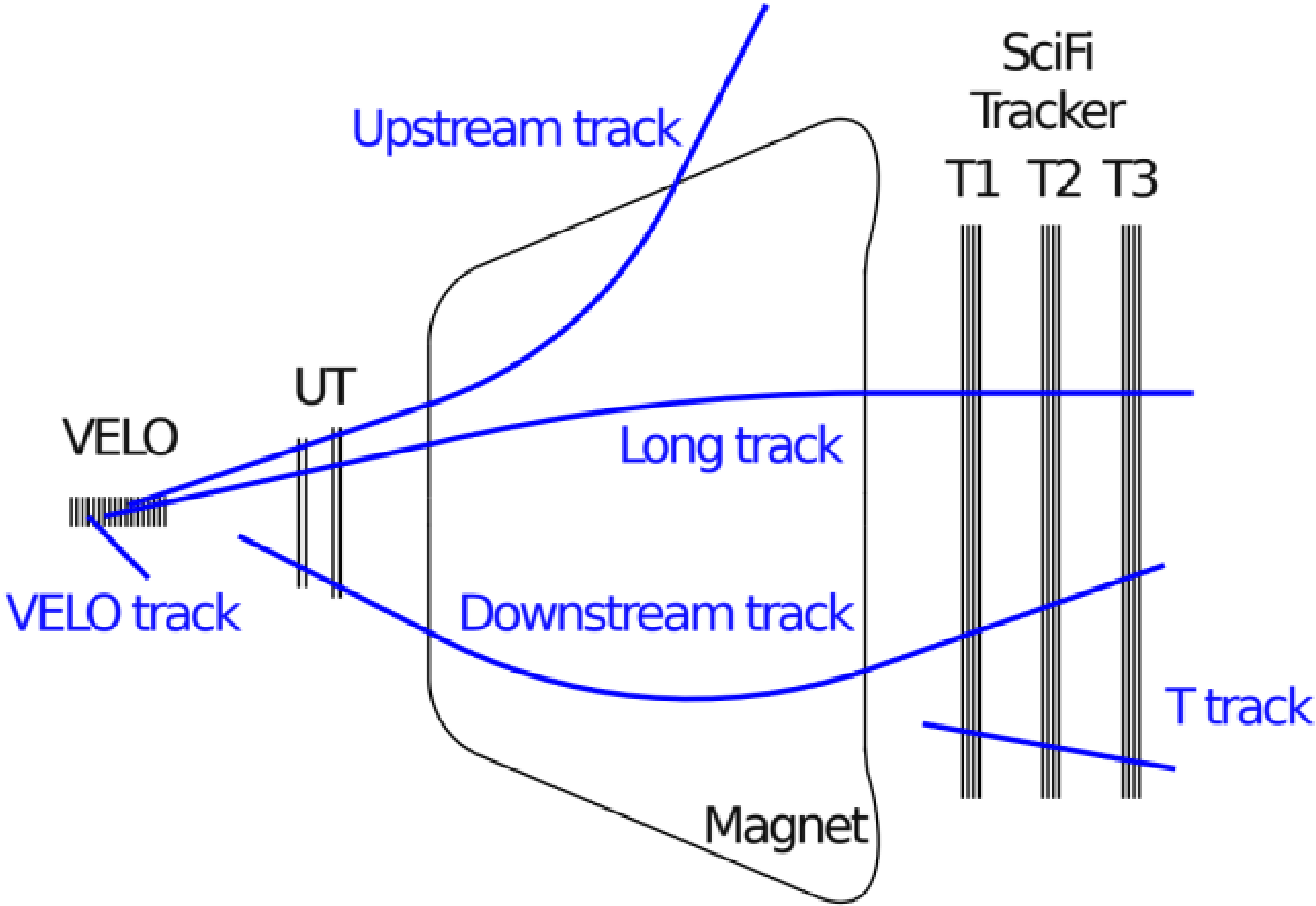


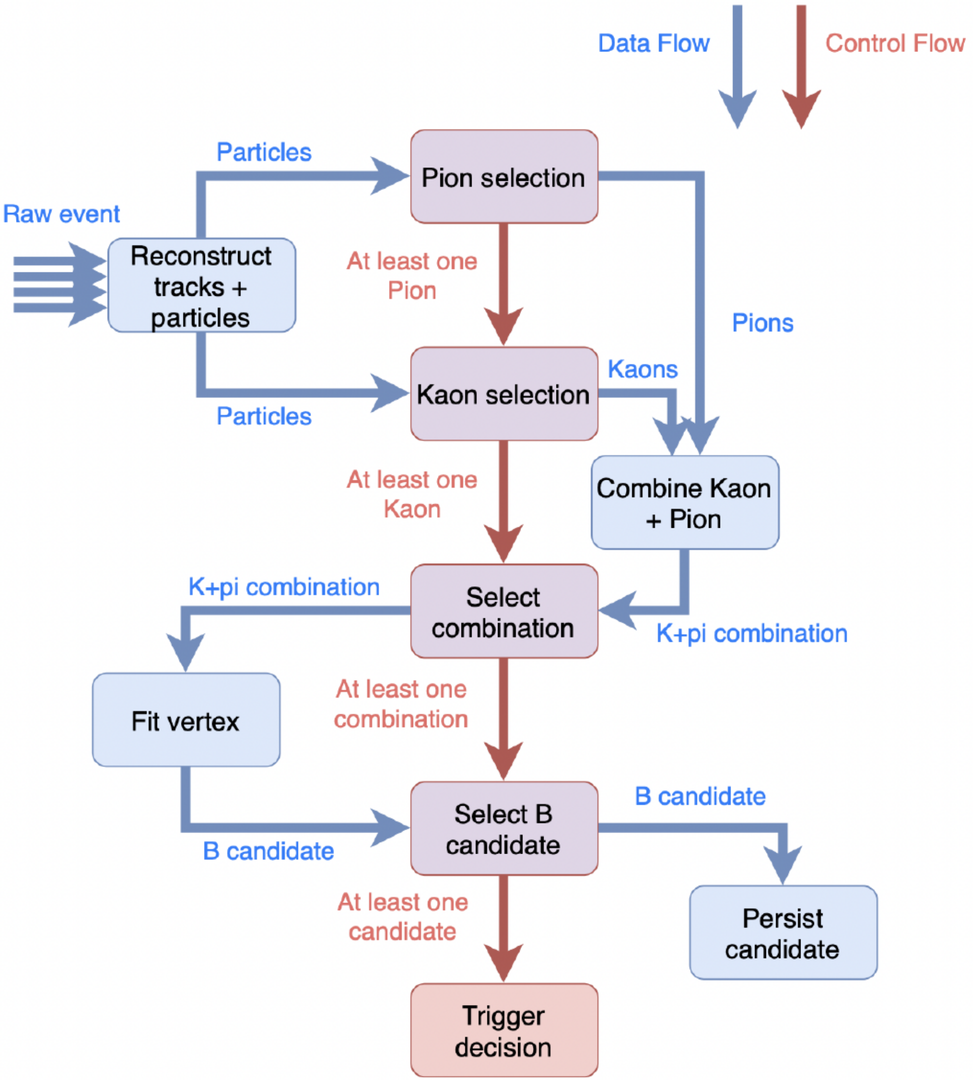
Data output per sub-detector

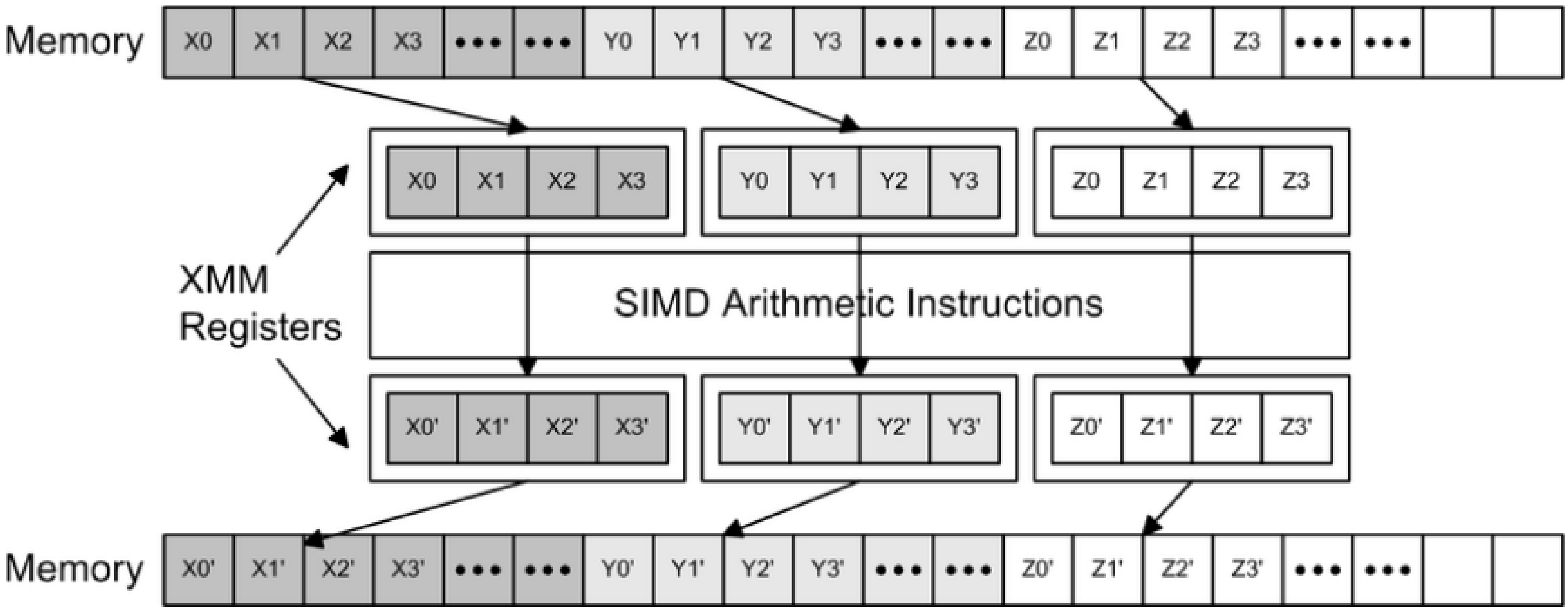
Sub-detector	fragment size [B]	#tell40 streams	event size [B]	event fraction	MEP size [GB]	MFP size [MB]	RU send size [MB]
Velo	156	104	16250	0.13	0.49	4.69	14.06
UT	100	200	20000	0.16	0.60	3.00	9.00
SCIFI	100	288	28800	0.23	0.86	3.00	9.00
Rich 1	166	132	22000	0.18	0.66	5.00	15.00
Rich 2	166	72	12000	0.10	0.36	5.00	15.00
Calo	156	104	16250	0.13	0.49	4.69	14.06
Muon	156	56	8750	0.07	0.26	4.69	14.06
Total	1000	956	124050	1	3.72	30.06	90.19



TrackFit
RICH
Seeding
Protoparticles
Calorimeter
Forward
Converters
Downstream
HLT1
Match
Framework
Muon
23.56
22.62
13.09
13.02
8.47
5.60
5.33
4.12
2.30
0.80
0.77
0.31
LHCb Simulation
HLT2 Throughput Rate 505.0 Hz
0
5
10
15
20
25
Timing fraction within the HLT2 sequence [%]

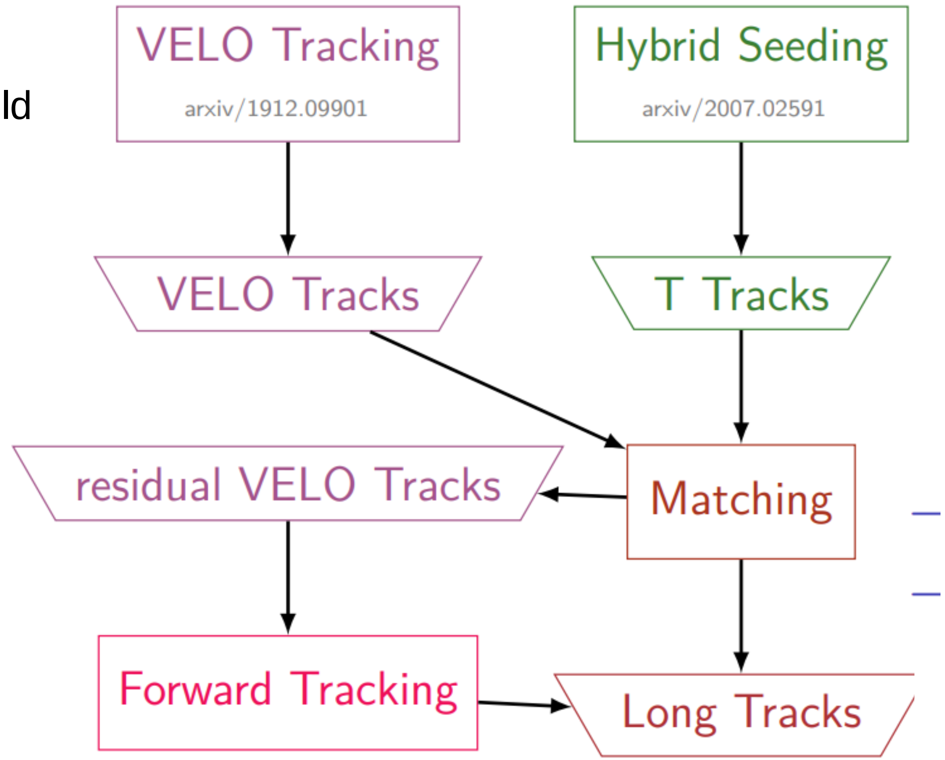

Upstream track
SciFi
Tracker
T1 T2 T3
UT
VELO
Long track
VELO track
Downstream track
T track
Magnet


Data Flow
Control Flow
Raw event
Reconstruct tracks + particles
Particles
Pion selection
At least one Pion
Pions
Particles
Kaon selection
Kaons
At least one Kaon
Combine Kaon + Pion
K+pi combination
Select combination
K+pi combination
Fit vertex
At least one combination
B candidate
Select B candidate
B candidate
At least one candidate
Persist candidate
Trigger decision


Memory
X0
X1
X2
X3
Y0
Y1
Y2
Y3
Z0
Z1
Z2
Z3
XMM
Registers
SIMD Arithmetic Instructions
X0'
X1'
X2'
X3'
Y0'
Y1'
Y2'
Y3'
Z0'
Z1'
Z2'
Z3'
Memory

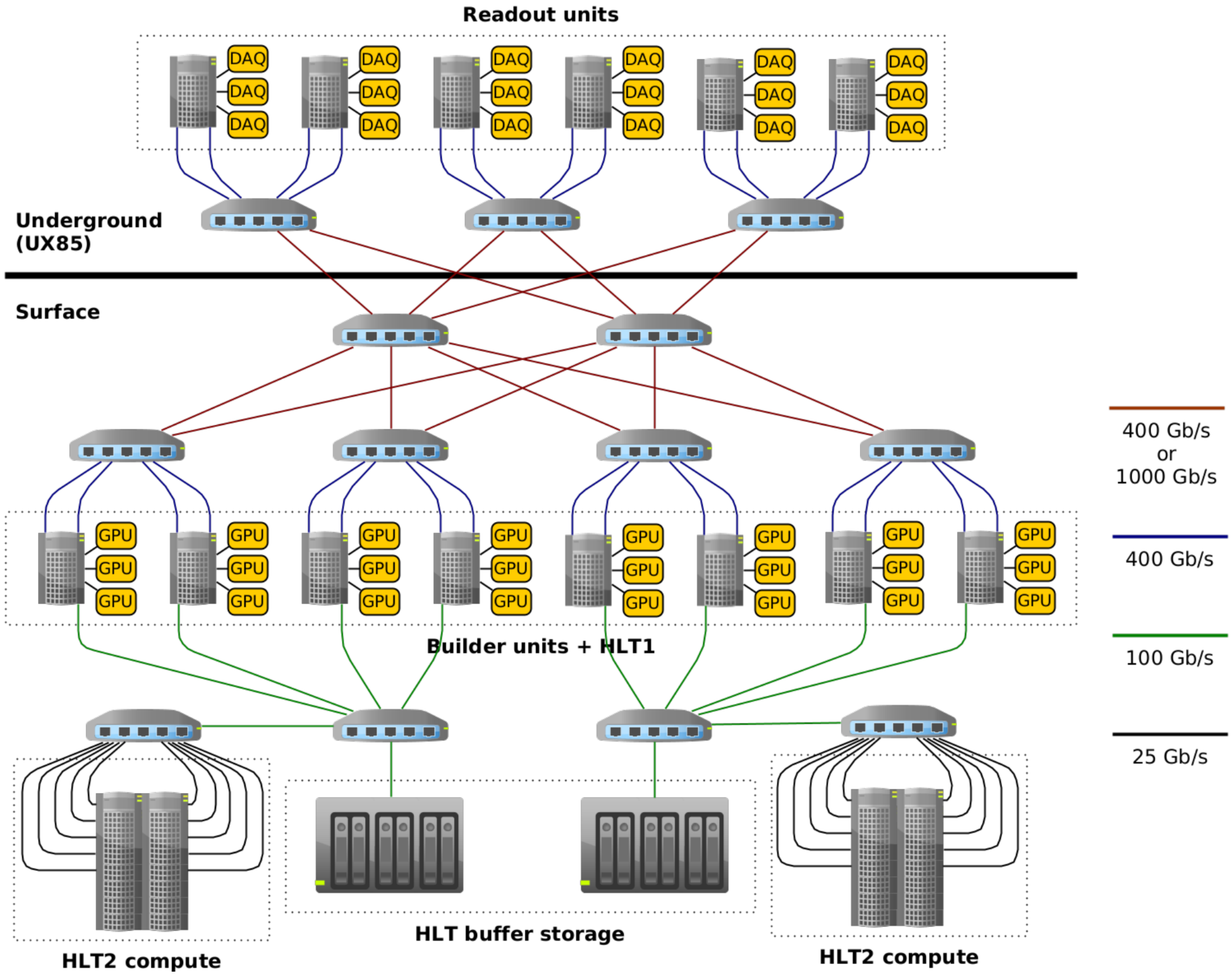
Forward tracking

1) Define hit search window
2) Treat magnet as optical lens to simplify track and hit projection
3) Hough-like transform: project all hits in window to reference
4) Plane and count number of SciFi layers in histogram
5) Scan histogram, collect hits from bins above threshold
   - found set of SciFi hits extending VELO track
6) Clean-up hit set and fit using 3rd order polynomial
7) Estimate q/p from fit result



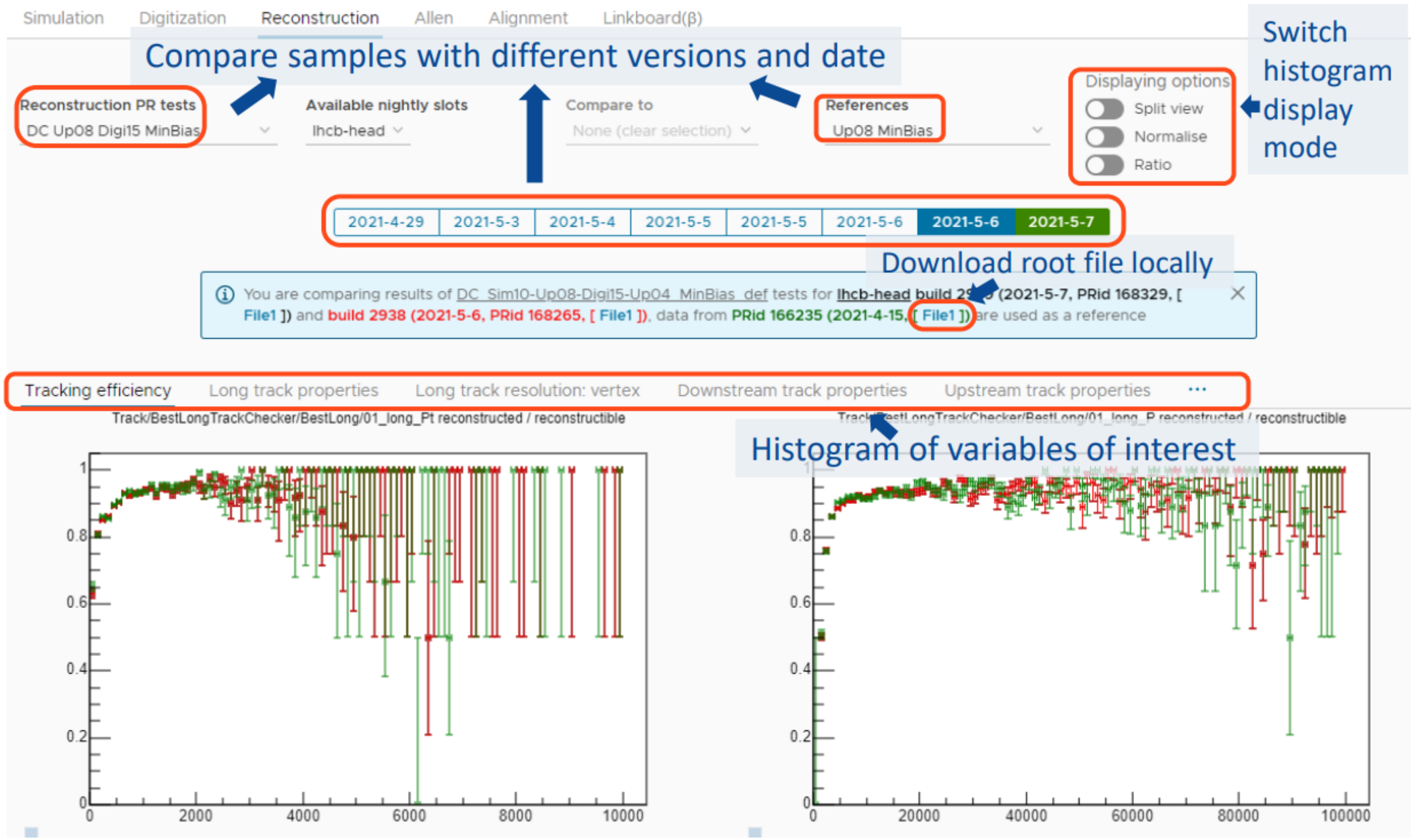
LHCb Upgrade 2

- Online scheme



Reconstruction dashboard

- Additionally a high-level physics properties can be obtained from nightlies and compared between each other and / or references
- Results are then visualised and accessible via LHCbPR front-end
- Used extensively to compare results between various HLT2 settings and evaluate their impact



Summary

- HLT1 represents the first complete high-throughput GPU trigger at HEP
  - 350 installed GPUs allows to explore additional ideas during Run 3
- HLT2 is based on modern computing methods such as SoA, vectorisation, multi-threading
  - Turbo model is a baseline for Run 3 allowing to record higher statistics at same bandwidth
  - Declared goal of running HLT2 at 500 Hz per node was successfully achieved
- Extended QA and testing system is well established part of the development cycle
- Upgrade II of LHCb is expected to results into the highest bandwidth at any HEP experiment
- Various studies and initiatives already started aiming to tackle such an interesting challenge

Personal points on building a new trigger system

Disclaimer:
- Purely personal opinion
- CepC trigger throughput taken from various talks this week
- All mistakes are purely my own

- Trigger strategy should be considered from the beginning of planning a new experiment
- Important interplay between a hardware (detector) and trigger groups
  - Proper sharing of information to keep overview about a properties of developed system
  - Important to know what type of information can be accessible in the online processing
- CepC is aiming for a bandwidth O(55) Tbps in case of trigerless mode
  - Significant amount of data to be processed and moved within the online system
  - Is there a clear strategy for offline processing? Integration of online and offline code is highly beneficial
  - Can it be that different running modes would benefit from a different trigger strategies?
- Event model and general data structure has a profound effect on the general software architecture
  - One approach for everyone may not be optimal, at the same time cost for supporting many different structure has to be evaluated
- Software become an integral part of any HEP experiment on same level as the hardware
- CepC is an ambitious project with several high-level physics goals, but only a thorough preparations will make it possible