LHCb triggerless readout, software trigger and online reconstruction

Alberto Perro on behalf of the LHCb Online Team

Workshop on Fast Real-time Systems and Real-time Machine Learning - Giessen, 8-11 April 2024

Overview

The LHCb Experiment

Going Triggerless

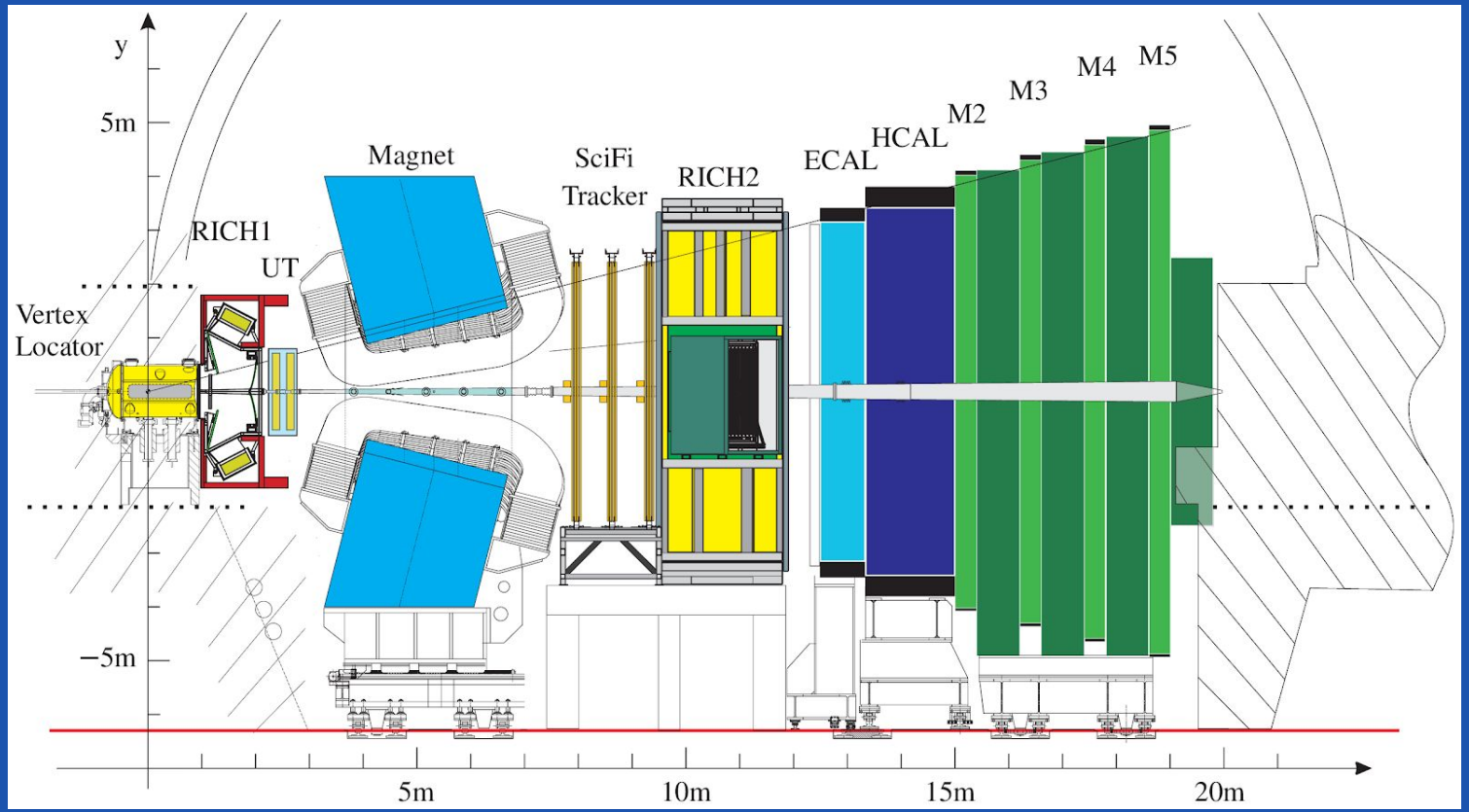
Readout design

Event Building

Software Triggers

Timing and Control

The LHCb Experiment



LHCb Characteristics

Year	2011	2012	2015	2016	2017	2018
L_{int} / fb^{-1}	1.0	2.0	0.30	1.6	1.7	2.1
E_p / TeV	3.5	4	6.5	6.5	6.5	6.5

Some Processes investigated at LHCb and the corresponding estimate of event during the whole Run I (2009-2013) and Run II (2015-2018)

$$BR(B_s^0 \to \phi\gamma) = (3.4 \pm 0.4) \times 10^{-5}$$ ~2.5 M events

$$BR(B_s^0 \to \phi\mu^+\mu^-) = (8.2 \pm 1.2) \times 10^{-7}$$ ~59 k events

$$BR(B_s^0 \to \mu^+\mu^-) = (2.9 \pm 0.4) \times 10^{-9}$$ ~208 events

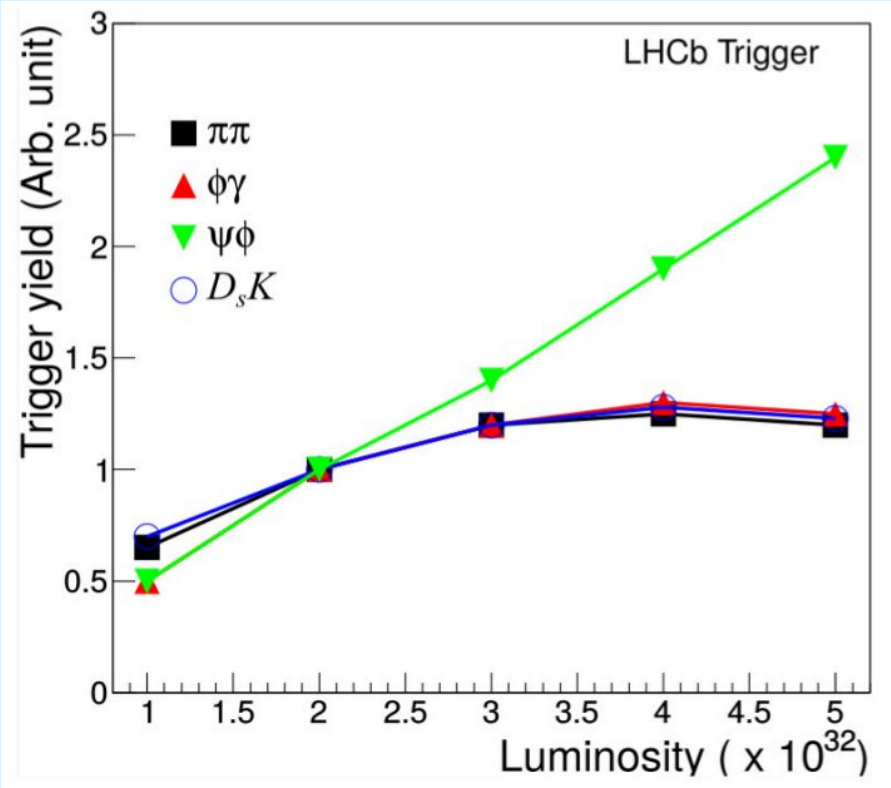
Conventional Trigger Limitations

The first level trigger (L0) is implemented in hardware:

- Selections are made at 40 MHz using Calorimeters or Muon Systems
- Criteria are based on high deposits of transverse energy by charged particles and photons
- Output rate to HLT1 is 1 MHz

Hadronic modes saturate yield in Run II.

An upgrade is needed to run at higher luminosity (5 times the luminosity of Run II).



LHCb Upgrade I: Going Triggerless

- Spectrometer geometry: more space to route cable/fibers
- Zero-suppression on the detectors
- Relatively low radiation levels permit to relax the FPGA/ASIC constraints
- Comparatively small event-size (O~100 kB)
- Efficient and accurate software trigger that can perform online selection with offline-like quality

Online DAQ System Overview

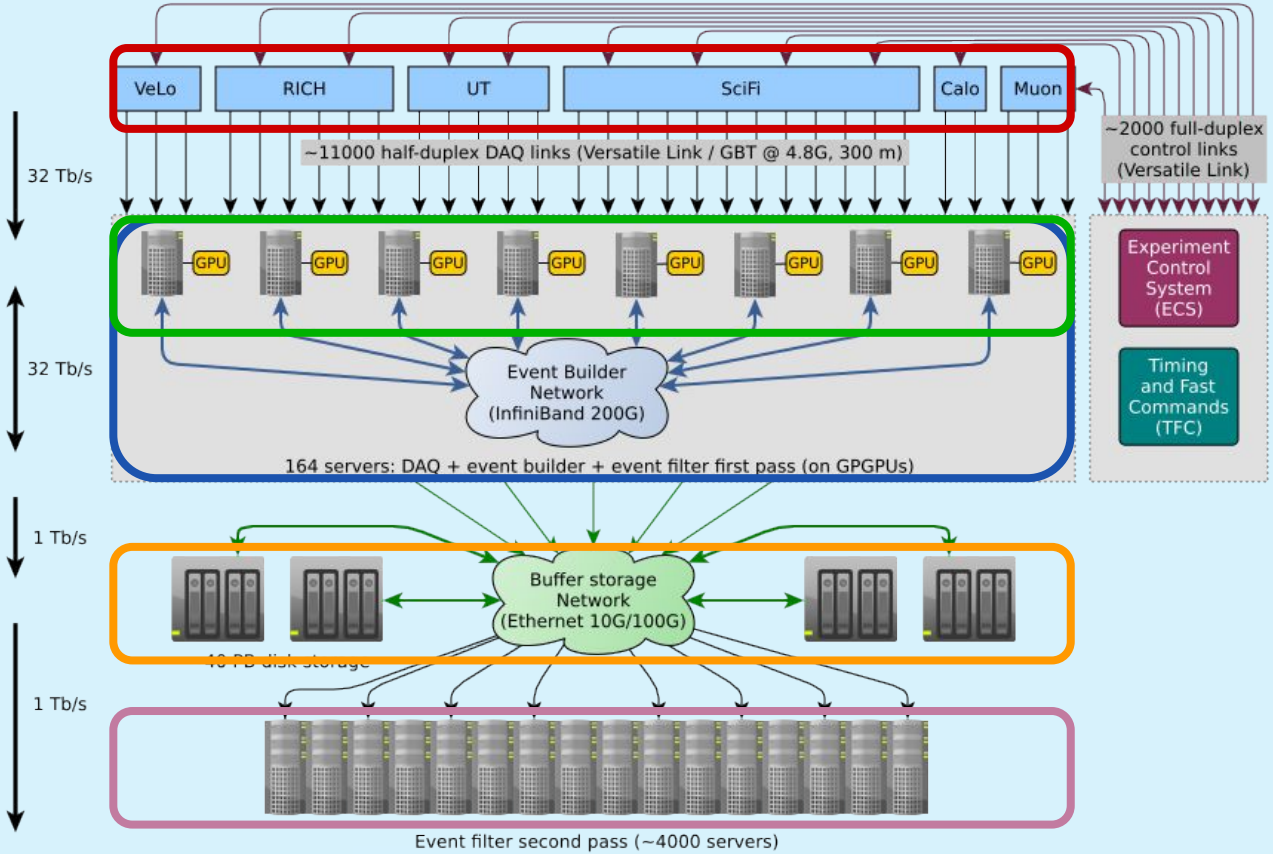
Front-End Electronics

DAQ FPGA Cards

Event Builder and HLT1

Storage Buffer

HLT2

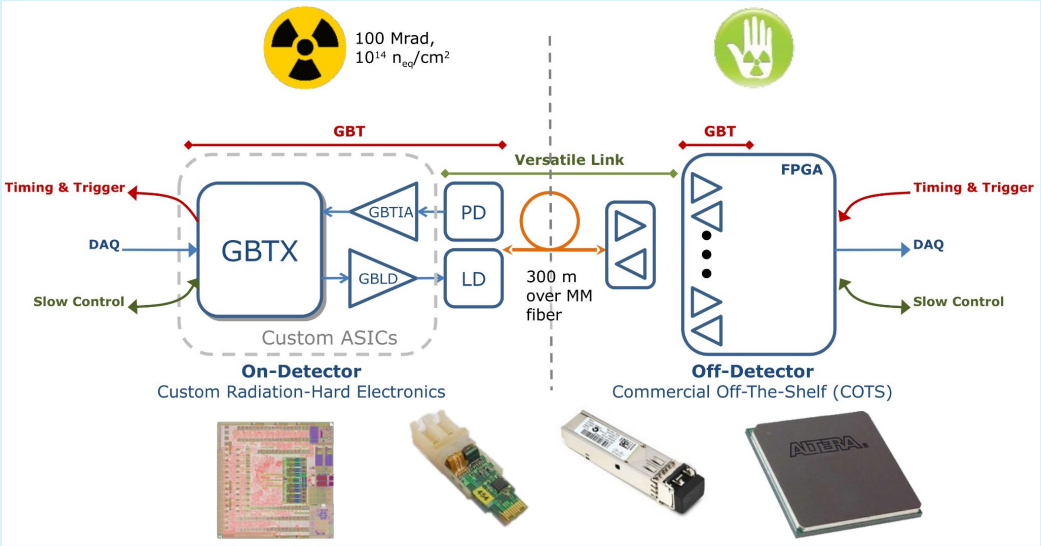


GigaBit Transceiver (GBT) Project

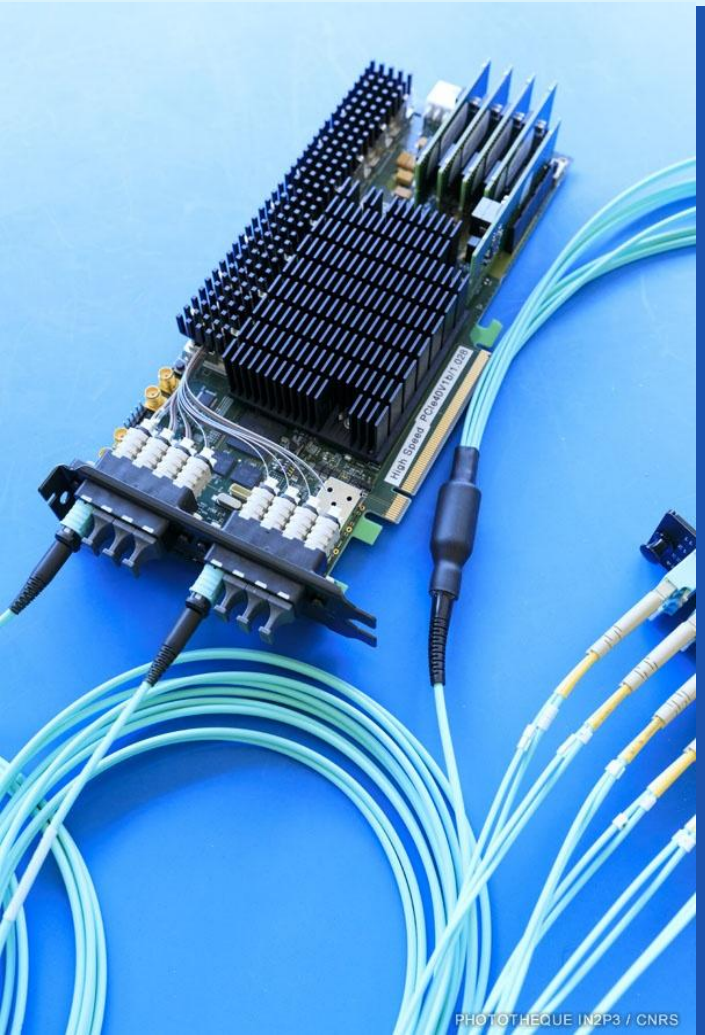
The GBT project provides a radiation hard chipset for handling control and data acquisition on FEE.

It has been designed at CERN and it is widely used across the experiments.

Gateware components (GBT-FPGA) allow to directly interact with the GBTx chip using FPGAs.



10.5170/CERN-2009-006.342



PCIe40 FPGA Cards

- 1.2M Logic Elements
- 48 GBT Radiation Hard Optical Links @ 4.8Gbps
- Two SFP+ modules
- Two PCIe Gen3 x8 interfaces

520 cards are used in LHCb, subdivided in three flavors:

- SODIN: Readout supervisor, source of real-time control commands and entry point for the LHC bunch clock.
- SOL40: It interfaces to the slow and fast control of the FEE and to the fast control of the DAQ cards.
- TELL40: It acquires data from the FEE, processes it, and delivers it to the host through the PCIe interface.

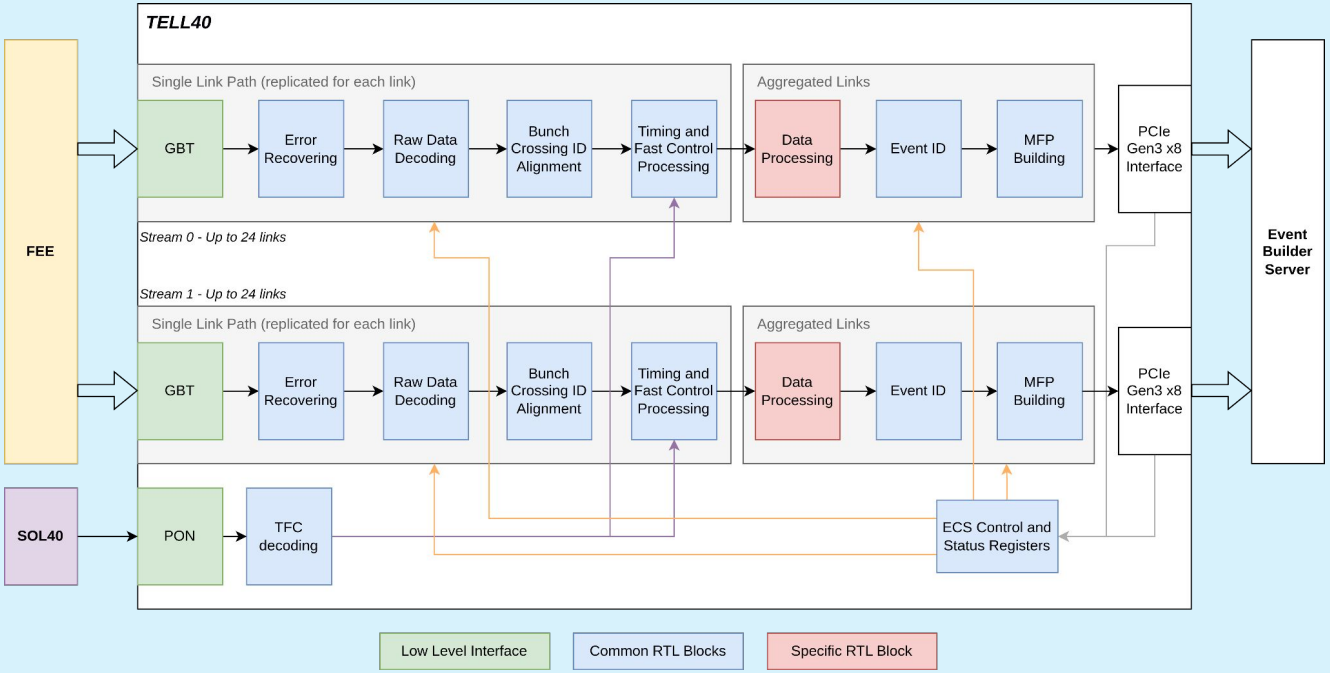
TELL40 Gateware

TELL40 offers a generic gateware platform.

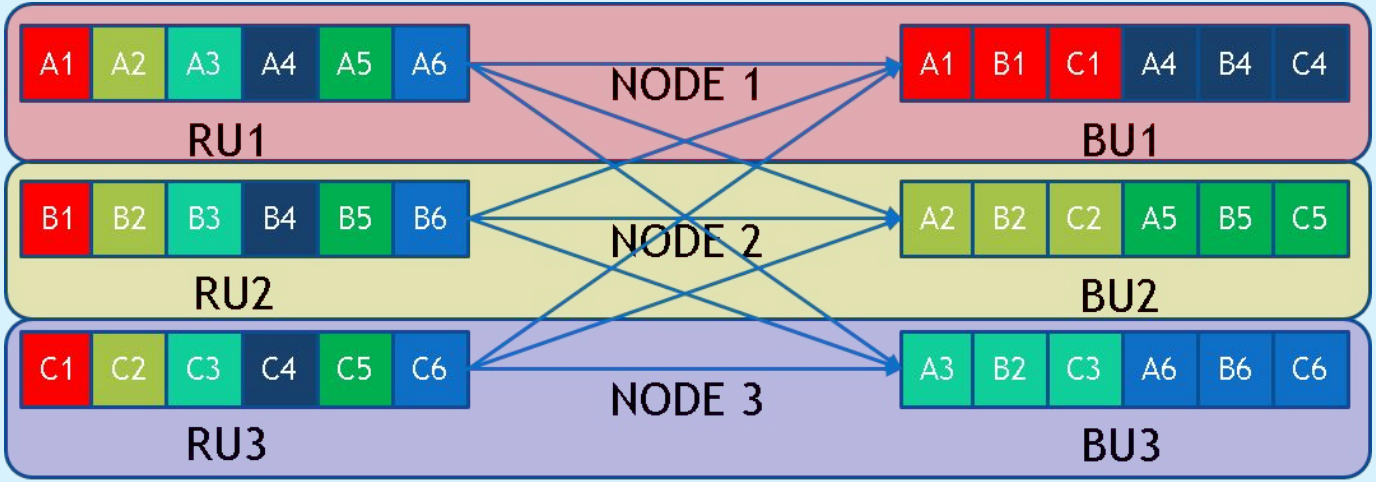
Each sub-detector customises the data processing block.

More than 30 different gateware variants to test and maintain.

Tests, compilations, and deployments done using Continuous Integration (CI).



The Event Building Process



The Event Builder (EB) collects data fragments from all the sub-detectors and builds full events.

Every Readout Unit (RU) read fragments from DAQ board and send them to the Builder Units.

Every Builder Unit (BU) receive fragments and build full events.

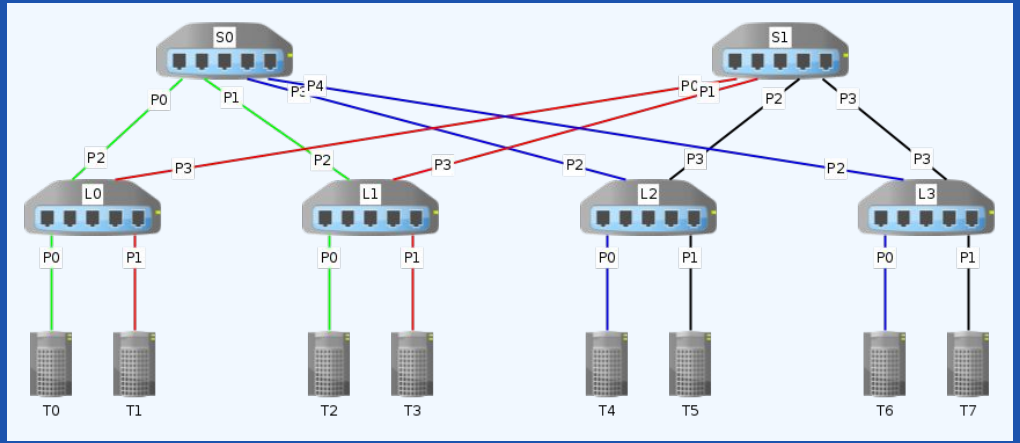
Event Builder Network

Fragments are scattered over all the EB nodes, so a network is necessary to perform the collection.

With a full data rate of 32 Tb/s, ~300 nodes are required (300 * 107Gb/s).

The network is built in a fat-tree topology over InfiniBand HDR (200 Gbps) fabric.

Traffic has an all-to-all pattern and tends to create network congestion.

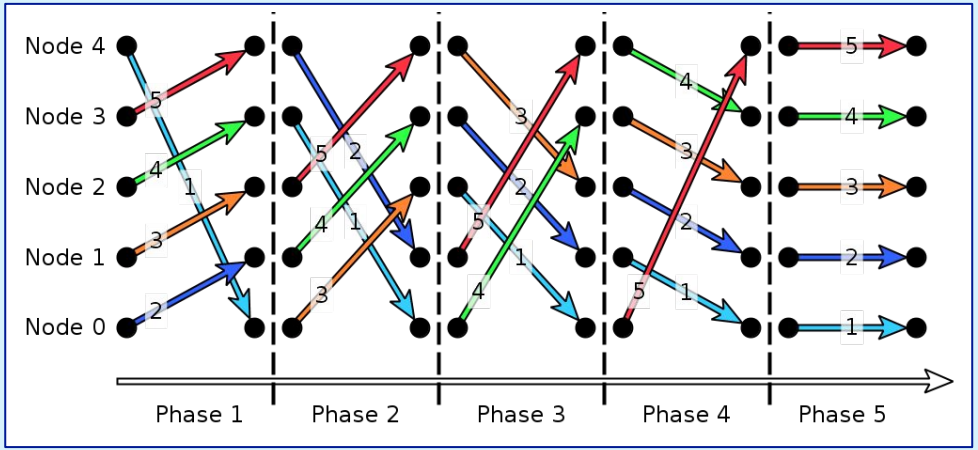


Traffic Scheduling

The processing of N events is divided into N phases:

- In every phase one RU sends data to one BU and every BU receives data from one RU
- During phase n, the i-th RU sends data to BU (i+n) mod N
- All the units switch synchronously from phase n to phase n+1

This way, congestion is avoided by routing messages on selected networks (Fat-tree Networks).



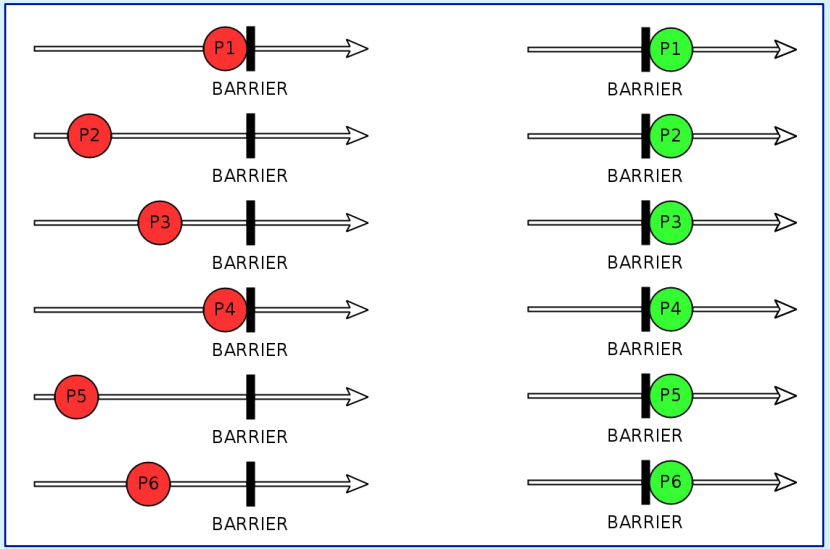
Synchronization

Strong synchronization between all nodes is done at every step of the scheduling.

Synchronization Barrier:

- Processes report they reached the barrier and wait for release
- When all processes have reached the barrier, they are released.

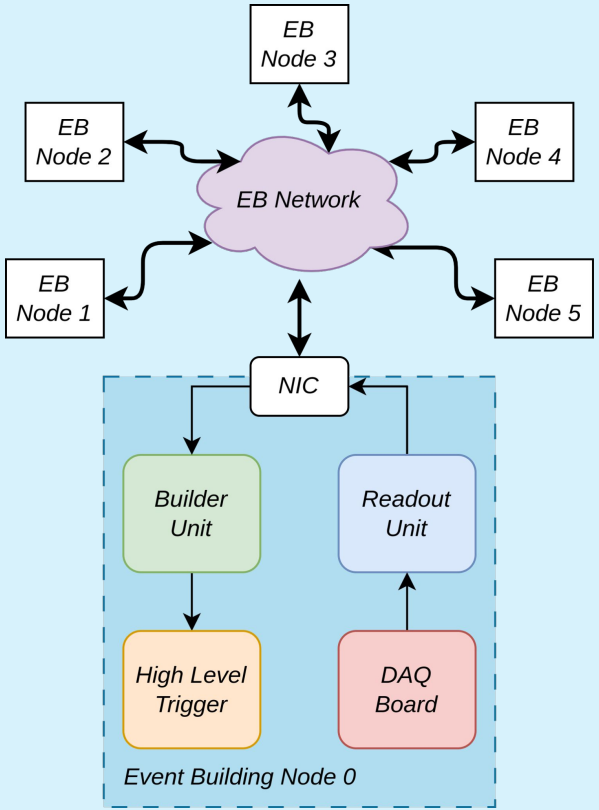
Multiple barrier algorithms are implemented in the communication library.



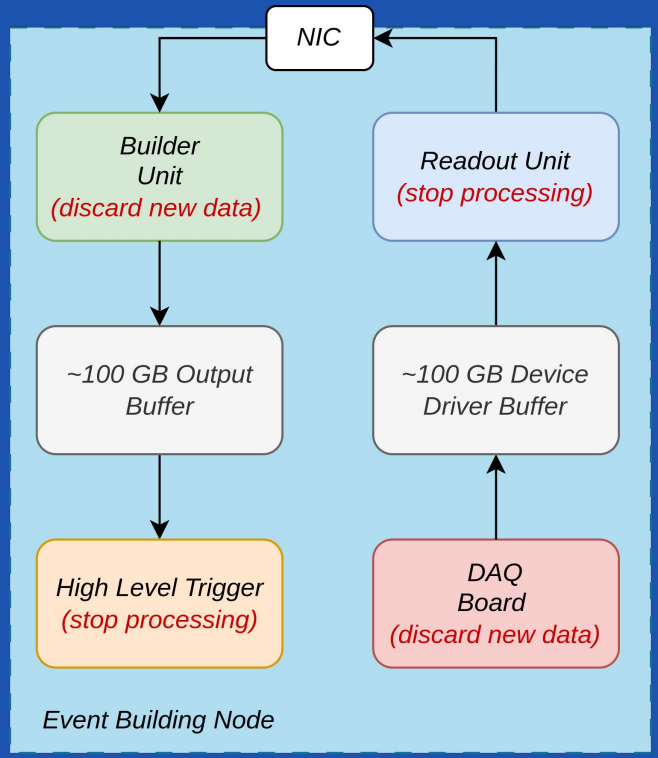
Event Builder Architecture

The key features of the EB architecture are:

- Modular design built in C++
- Large amount of RAM buffers to absorb latency spikes and to shape the traffic
- Use of DMA and Remote DMA to lower memory bandwidth usage
- Dedicated low-level communication library to optimize network performance
- Barrier-based scheduling to guarantee strong synchronisation
- Buffer-isolated critical sections to minimise slowdowns and dead-time



Back-pressure and Fault Tolerance

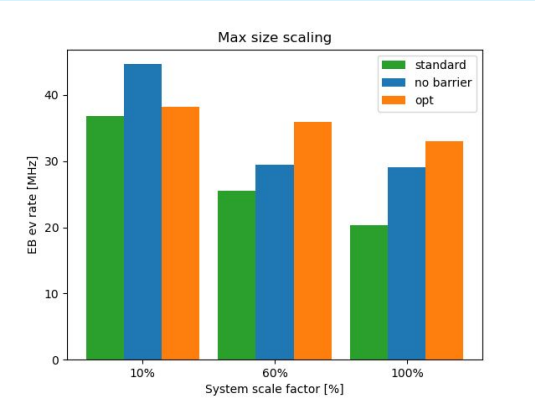


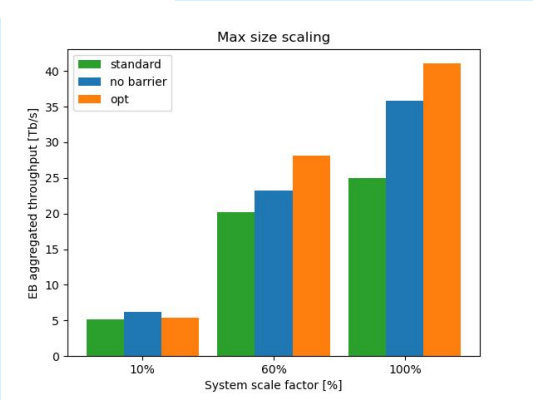
Big RAM buffers at the input and output of the EB are used to reduce latency spikes and back-pressure.

Discard Policies on the Readout Cards and Builder Units also allow to contain back-pressure propagation.

Node-related issues can be solved by moving the EB units to a different host.

Synchronization Performance





With a few nodes, the barrier adds an overhead, reducing the performance.

With the full scale system, the strong synchronization ensures a ~6% higher throughput.

Optimization of the algorithm is crucial to get the maximum performance.

A 40% difference has been measured between the central algorithm and the tournament one.

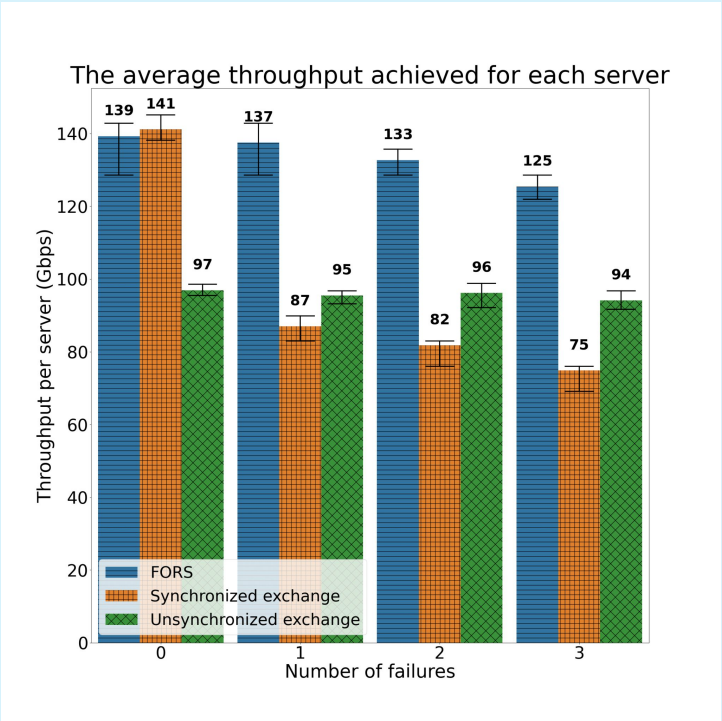
10.1109/TNS.2023.3240514

Network Fault Tolerance

Traffic scheduling requires specific routing to ensure maximum bandwidth.

Link failures can greatly reduce the total throughput by creating congestions.

A fault-adaptive congestion-free routing and scheduling solution has been developed to make efficient use of the remaining bandwidth.



10.1109/LCN58197.2023.10223324

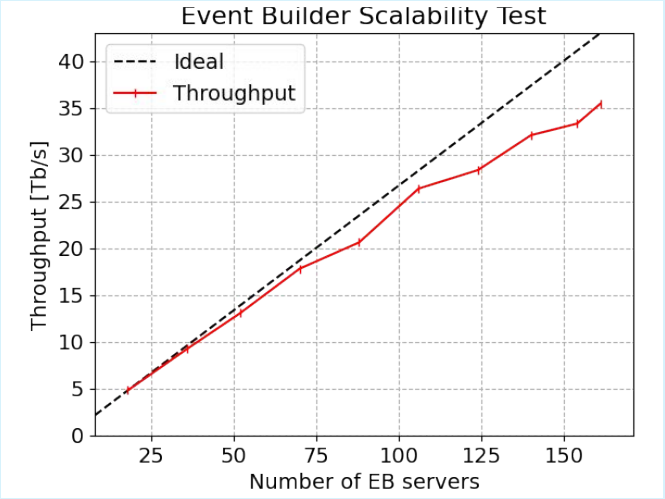
Event Builder: Implementation Results

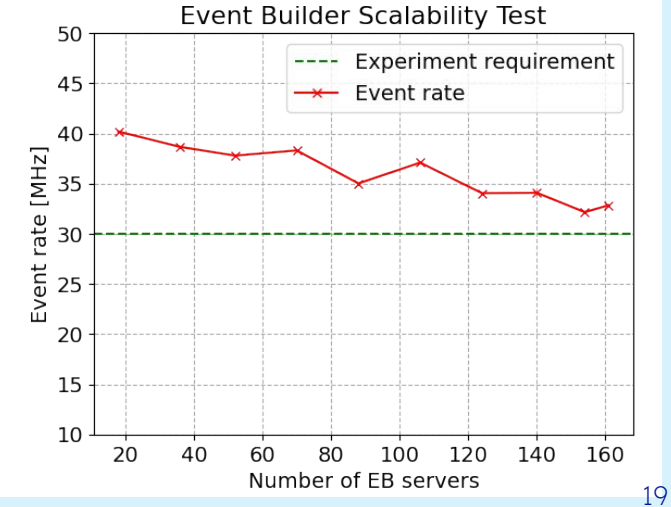
Advantages

- ✓ Removed physics inefficiencies related to the hardware trigger
- ✓ Robust system with respect to network latency spikes
- ✓ Reduced cost thanks to off-the-shelf components and converged architecture

Disadvantages:

- ✗ The trigger provides non physics functionality
- ✗ Tuning required to optimise PCIe communication
- ✗ Converged architectures are less flexible and reliable

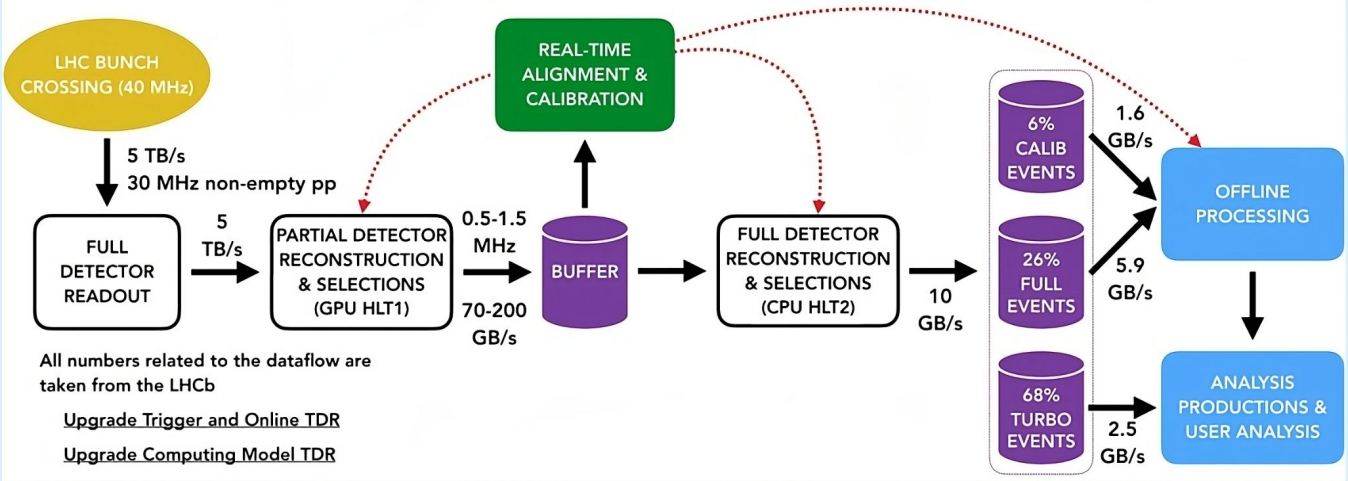




High Level Triggers

Alignment, Calibration, and Full Event Reconstruction in Real-Time

Trigger Layout in Run 3



- 2 stage software trigger: HLT1 and HLT2
- Real-time alignment and calibration
- After HLT2, 10 GB/s of data for offline processing

The First Level Trigger (HLT1)

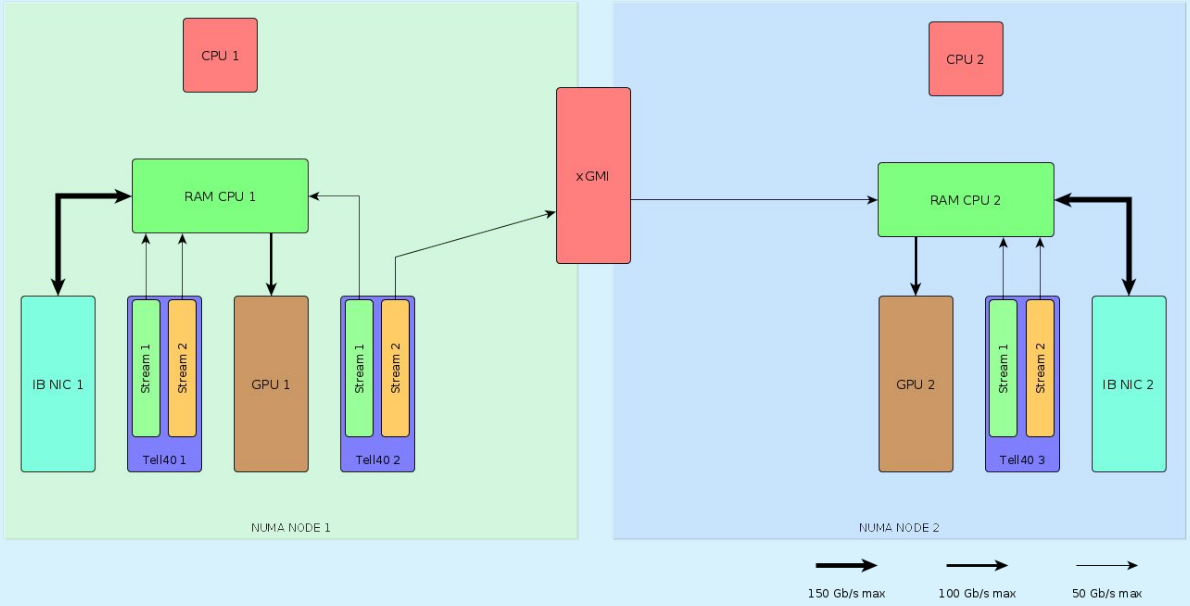
The goal:

- Handle the entirety of the raw data (5 TB/s) at 30 MHz
- Perform partial event reconstruction
- Apply a coarse selection of LHCb physics cases
- Reduce the input by a factor of 30 (~1 MHz output)
- Store selected events for real-time alignment and calibration

How to get there?

GPUs

HLT1 Hardware Architecture



The HLT1 is hosted in the Event Builder farm. (173 servers)

Each server has 3 free PCIe slots to host GPUs, offers sufficient cooling and power, and results in a self-contained system.

This solution results in a cheaper and more scalable architecture than the CPU alternative.

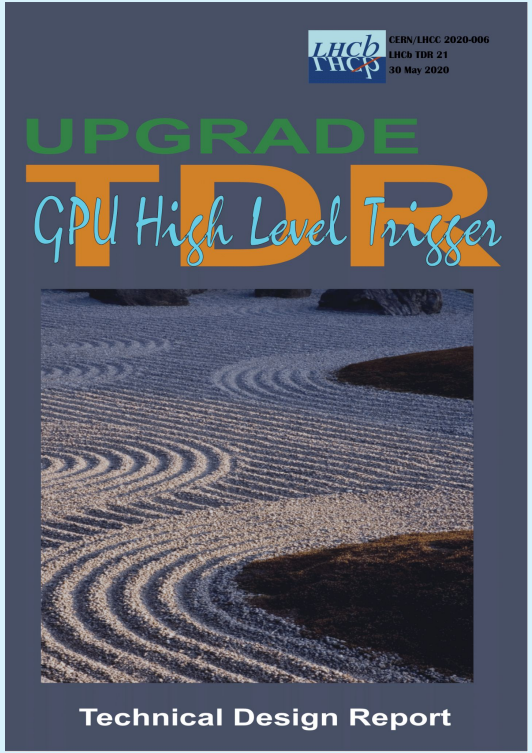
Allen: a GPU HLT1 Trigger Platform

Allen is a open-source software project developed by LHCb.

It can run on CPU and GPU (from both Nvidia and AMD).

The GPU code is written in CUDA and cross-architecture compatibility is configured via macros.

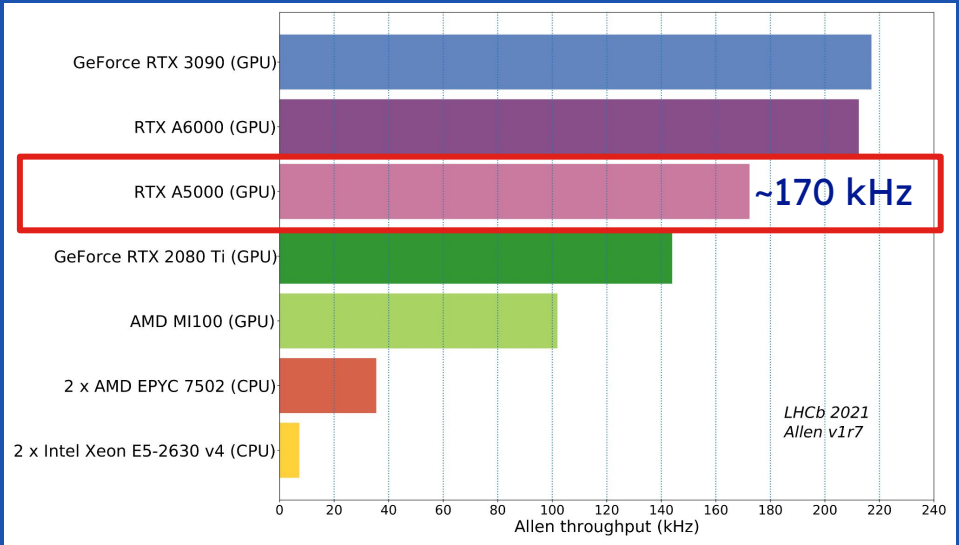
It is integrated in the LHCb stack, DAQ, and control system.



LHCB-TDR-021

10.1007/s41781-020-00039-7

HLT1 Throughput



Throughput is defined as number of events processed in steady-state conditions.

LHCb-FIGURE-2020-014

The requirement of 30 MHz are met with O(200) GPUs, with space in the servers for O(500).

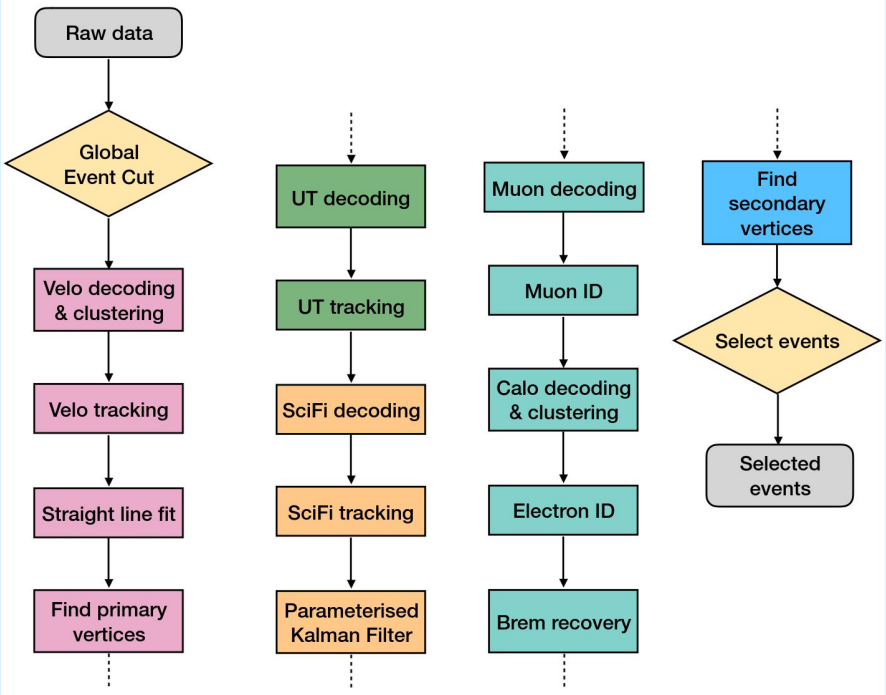
HLT1 scales well with the theoretical TFLOPs available on the GPUs, thanks to its inherently parallelizable tasks.

HLT1 Sequence

Tracking is at the core of the HLT1 reconstruction. It relies on 3 sub-detector systems:

- **VELO** : clustering, tracking, and vertex reconstruction
- **UT**: track reconstruction, momentum estimation, fake rejection
- **SciFi**: track reconstruction, momentum measurements

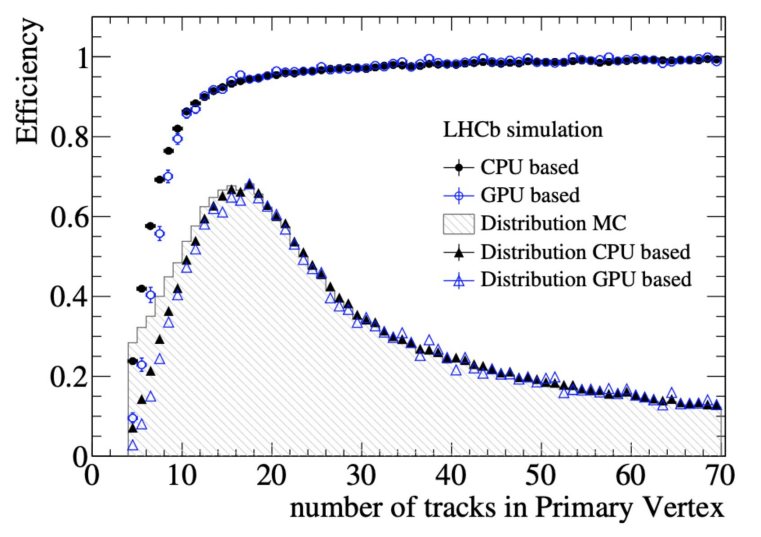
It also uses the muon stations and the calorimeter to do particle identification.



LHCb-TALK-2022-175

HLT1 Performance

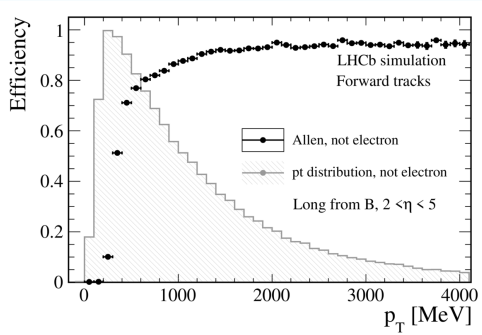
Vertex Reconstruction:
Efficiency > 90% for vertices with >10 tracks



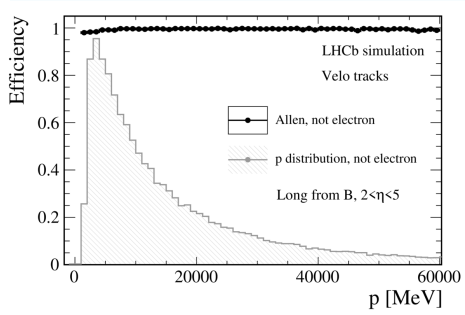
10.1007/s41781-021-00070-2

Tracking Performance:

- **Efficiency > 99%** for VELO, **95%** for high-p forward tracks
- Good momentum resolution and fake rejection



LHCb-FIGURE-2020-014

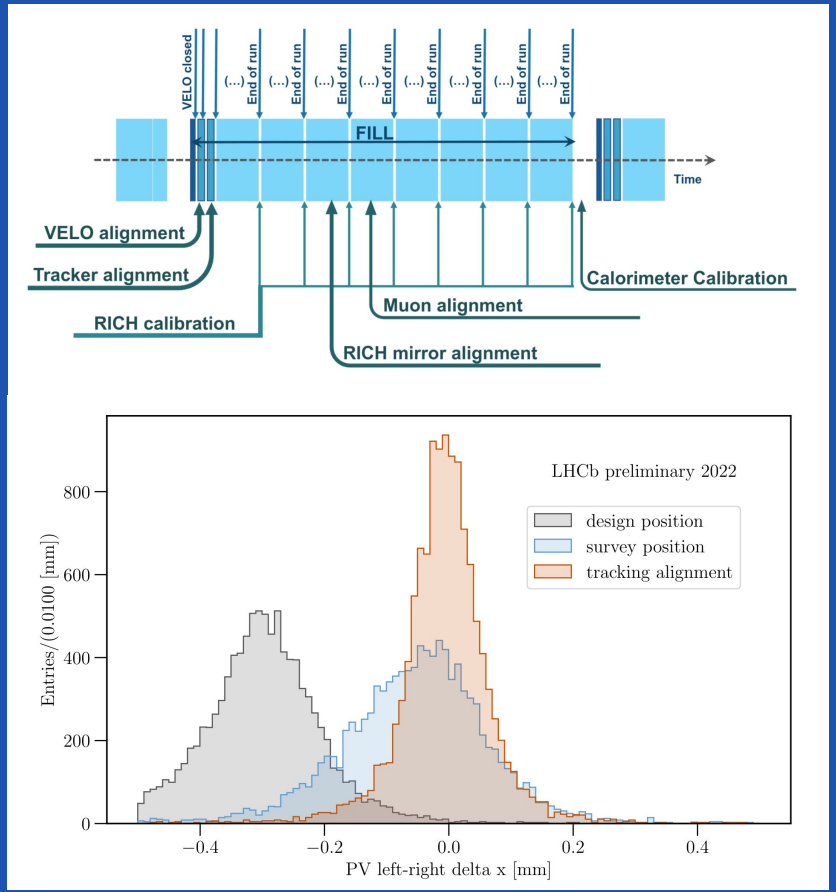


Real-time alignment and calibration

Dedicated trigger lines in HLT1 are used to select events for the alignment procedure.

Data must be fully aligned and calibrated before running the second High Level Trigger (HLT2).

This is managed by storing the events in a buffer of O(30) PB, giving enough time for the alignment and calibration procedure to complete.



LHCB-FIGURE-2022-016

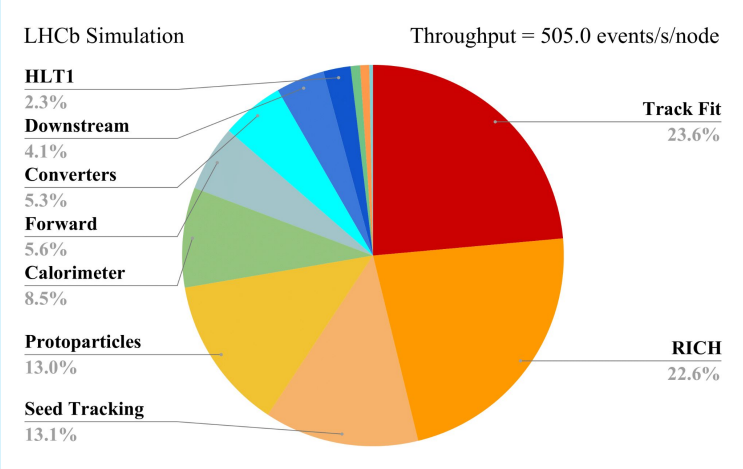
The Second Level Trigger (HLT2)

HLT2 fully reconstruct offline-quality tracks and neutral objects, including particle identification information.

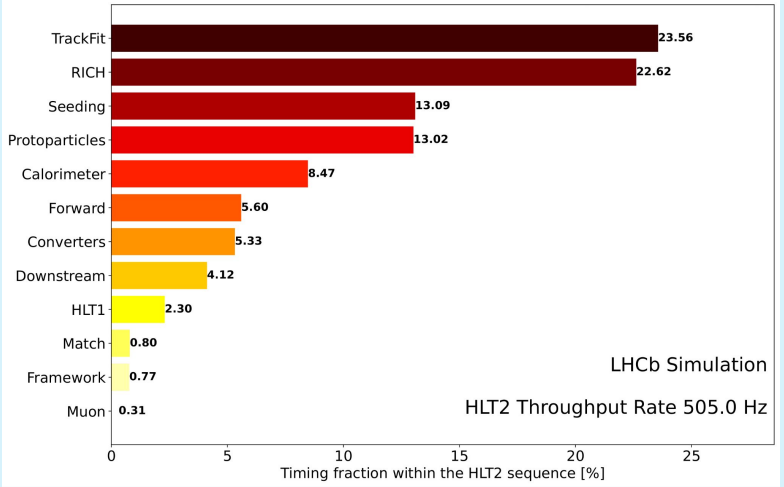
Fixed output throughput limited at 10 GB/s requires an output event rate of ~100 kHz.

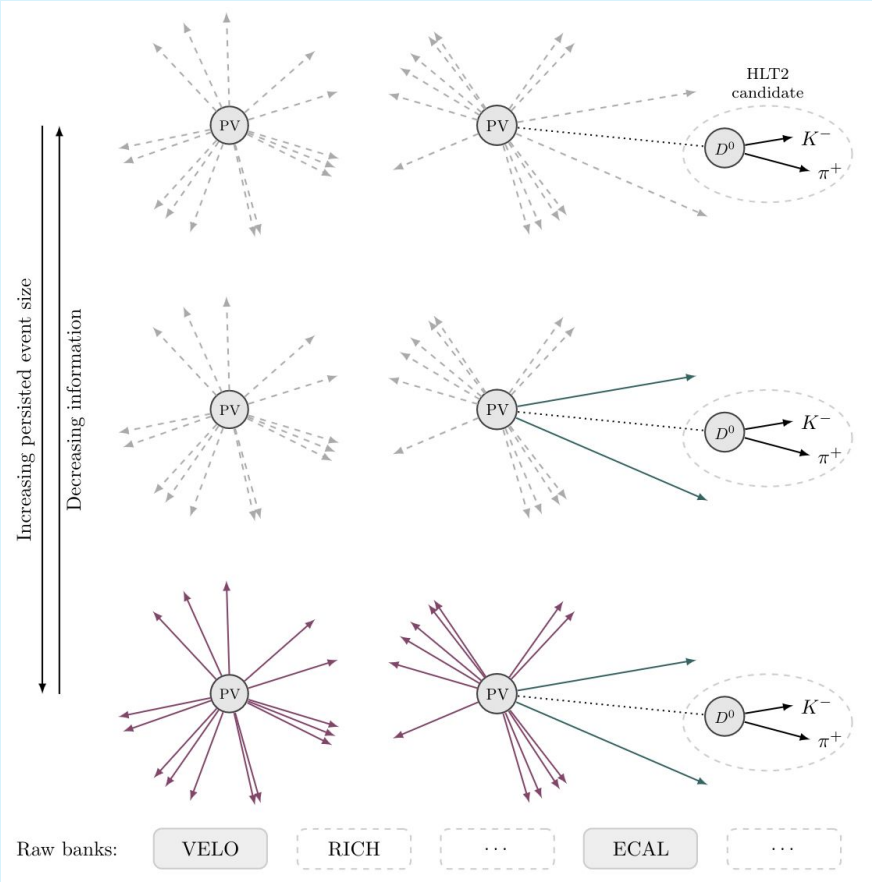
HLT2 runs asynchronously to the LHC, distributing the workload over times where no stable beam is present.

A total of O(1000) trigger lines are used to decide over the persistence of a given event.



LHCB-FIGURE-2022-005





HLT2 Event Persistence

Due to bandwidth limitations, different persistence levels are available:

- **Turbo**: Save raw and reconstructed data only from signal candidate that passed a HLT2 line selection
- **Selective Persistence**: Save selected raw and reconstructed data
- **Full stream**: Save the raw and reconstructed data of the full event

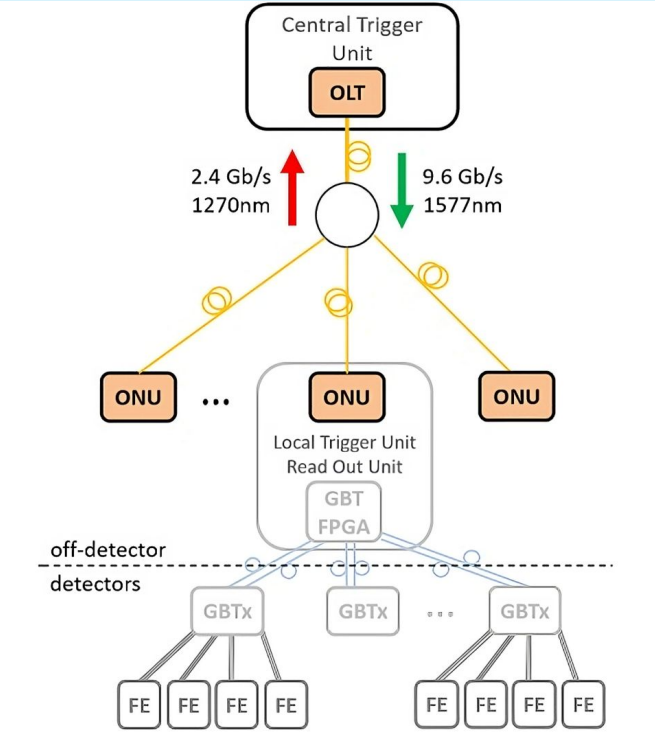
Going from Full Stream to Turbo is a ~10x decrease in size.

10.1088/1748-0221/14/04/P04006

Timing and Control

Fast and slow control, monitoring, and clock distribution

Timing and Fast Control (TFC)



10.22323/1.313.0124

The TFC is responsible for:

- Distributing LHC bunch clock with fixed phase
- Centrally generating real-time commands and distributing them with fixed latency
- Monitoring all the readout modules in real-time

The TFC is based on the *Timing, Trigger and Control for Passive Optical Networks* (TTC-PON) project.

Multiple Passive Optical Networks (PON) interconnects all the component of the DAQ system, distributing clock and commands with fixed latency.

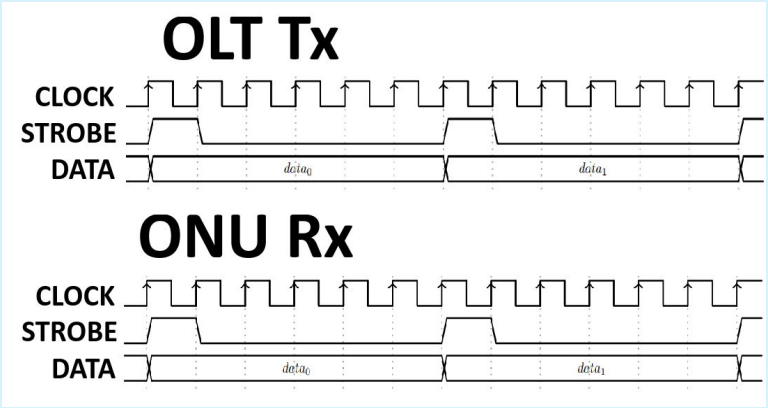
The endpoint devices are PCIe40 FPGA cards, implementing the supervisor, control, and DAQ roles.

Clock Recovery

TTC-PON manages clock and data recovery using FPGA transceivers' functionality.

The OLT begin to transmit a new word containing a fixed header when a strobe pulse is provided by the user.

The ONU identifies the header position in the serial stream and generates a strobe pulse phase-aligned with the header.

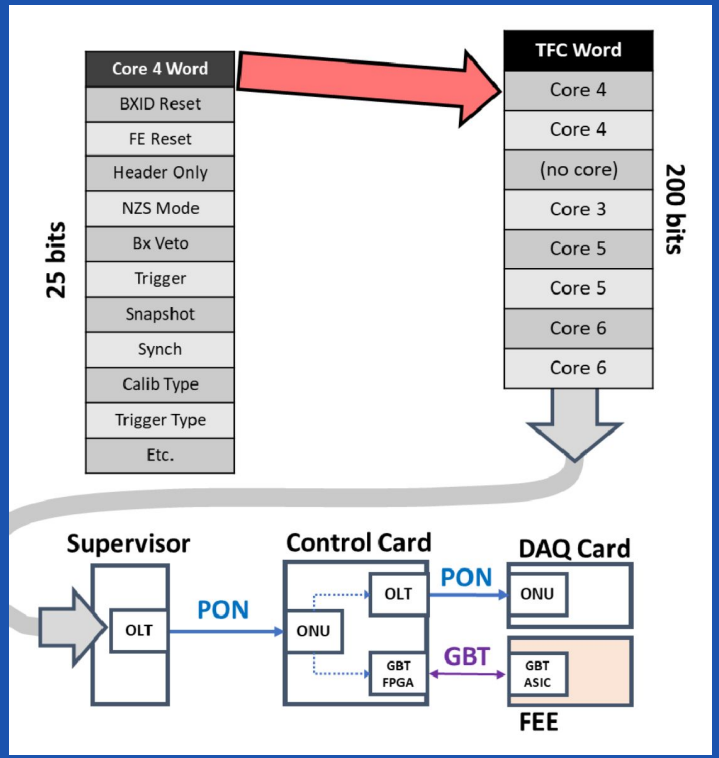


10.1109/TNS.2023.3273086

The waveform might look the same, but:

- In the OLT, strobe drives the data
- In the ONU, strobe is generated from data

The Readout Supervisor (SODIN)



10.1109/TNS.2023.3273086

Central component of the TFC system implemented in custom gateware.

Implements 10 independent cores that control different partitions of the detector simultaneously.

Each core can generate commands towards the DAQ and the FEE, controlling a set of subsystems.

25 bits are available to each partition to forward commands to the control cards.

Control cards relay information to FEE or DAQ sequentially.

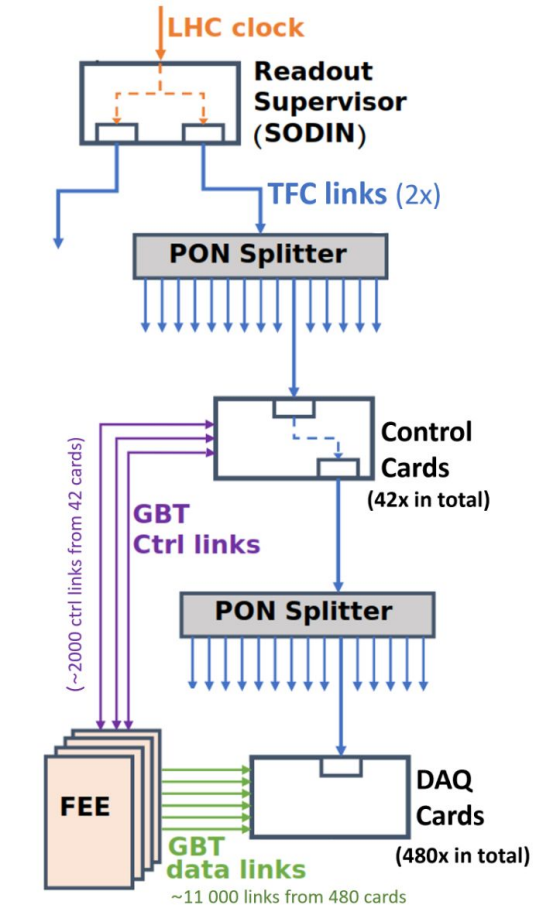
TFC Distribution

The TFC network uses the PCIe40 bidirectional 10G SFP+ modules.

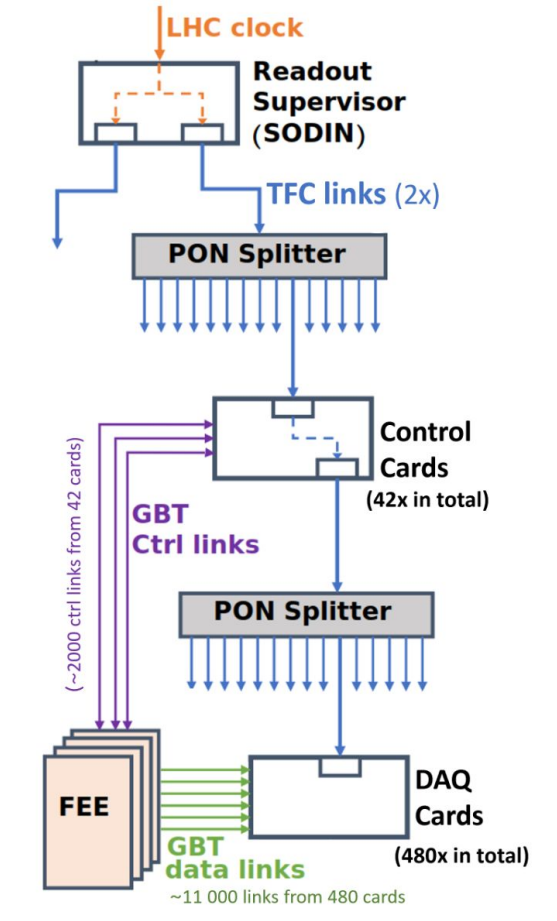
The supervisor feeds two PON splitters 1:32, which are connected to one of the links on the control cards.

The other link is connected to a PON splitter, which distributes to the DAQ cards.

Front End Electronics is directly connected to the control cards via the available GBT links.



10.1109/TNS.2023.3273086



Real-Time Monitoring

The TFC endpoints can also send data upstream through the PON interface.

Data is transmitted using a Time Division Multiplexing (TDM) scheme. Each ONU occupies a pre-assigned time slot when transmitting back.

In the worst-case scenario (reading from *all* endpoints in the system), the readout period would be 8us (320 clocks).

An application of this upstream link is throttle management. A throttle signal is transmitted when buffers in the DAQ cards are full. The supervisor then reduces the trigger rate accordingly.

10.1109/TNS.2023.3273086

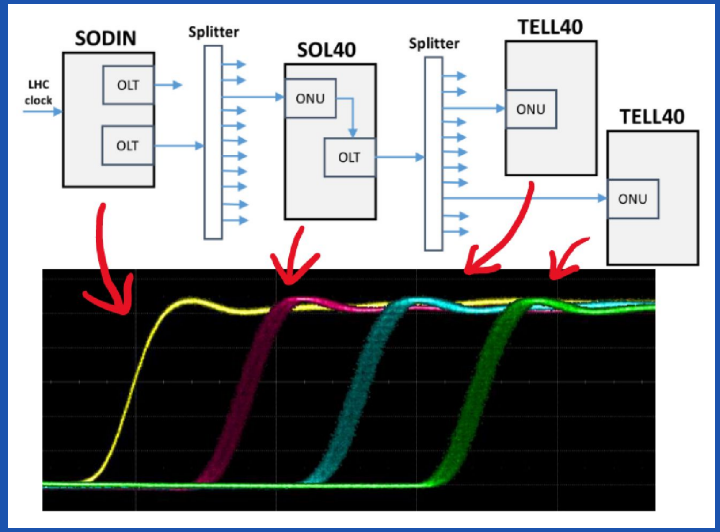
Clock Jitter and Skew Measurements

The requirements of the TFC are:

- **Clock Skew:** Phase difference between the LHC bunch clock and each recovered clock must be < 500 ps
- **Clock Jitter:** The total RMS jitter of each clock must be < 100 ps
- **Bit Error Rate:** The links' BER must be < 1.0×10^{-15}

Measurements reported:

- Maximum clock skew of 254 ps (σ = 76 ps)
- Random Jitter of 13.5 ps (considering 76 ps as max)
- BER < 3.1×10^{-17} (95% C.L.)



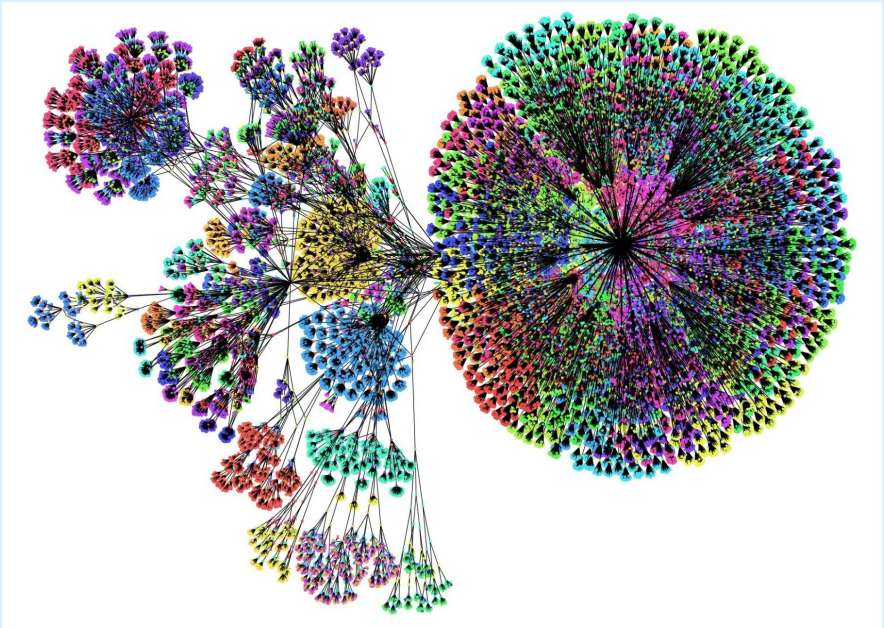
10.1109/TNS.2023.3273086

Experiment Control System

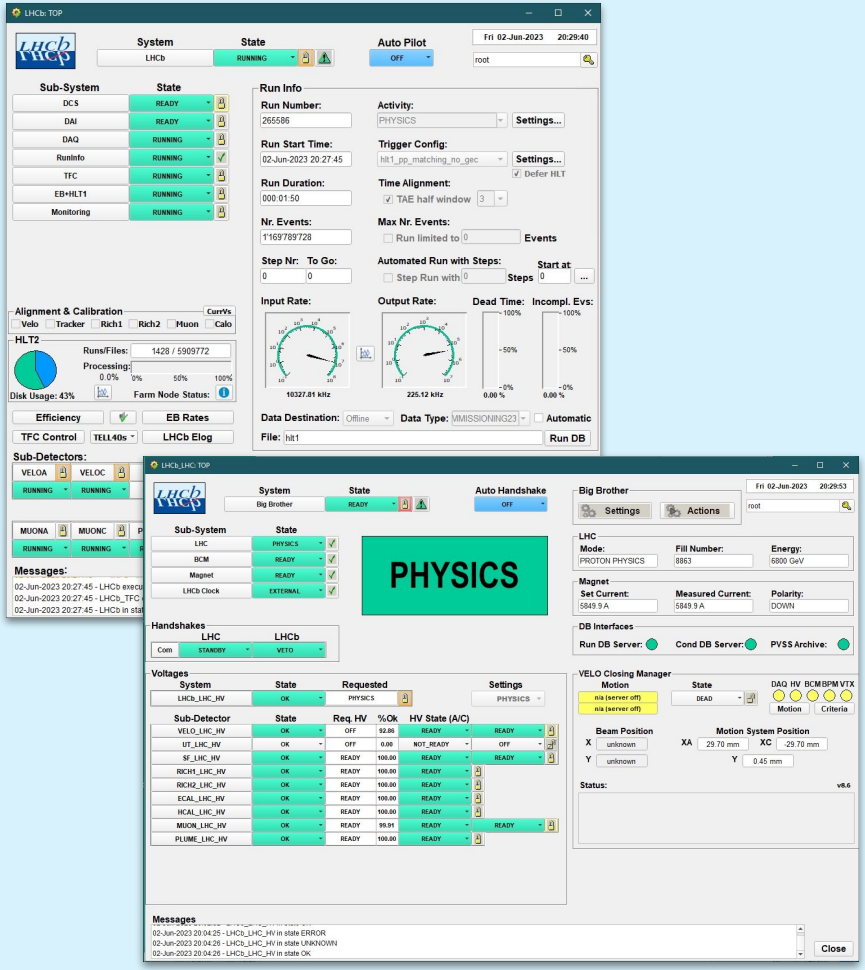
The LHCb ECS is based on the JCOP project, a common CERN framework for control systems.

The ECS is structured as a distributed hierarchical system running on Virtual Machines.

Control Units are designed as Finite State Machines that can take local decisions. Device units provide the interface to hardware and software endpoints.



Courtesy of CMS DCS team



Operations Tools

The main tools available are:

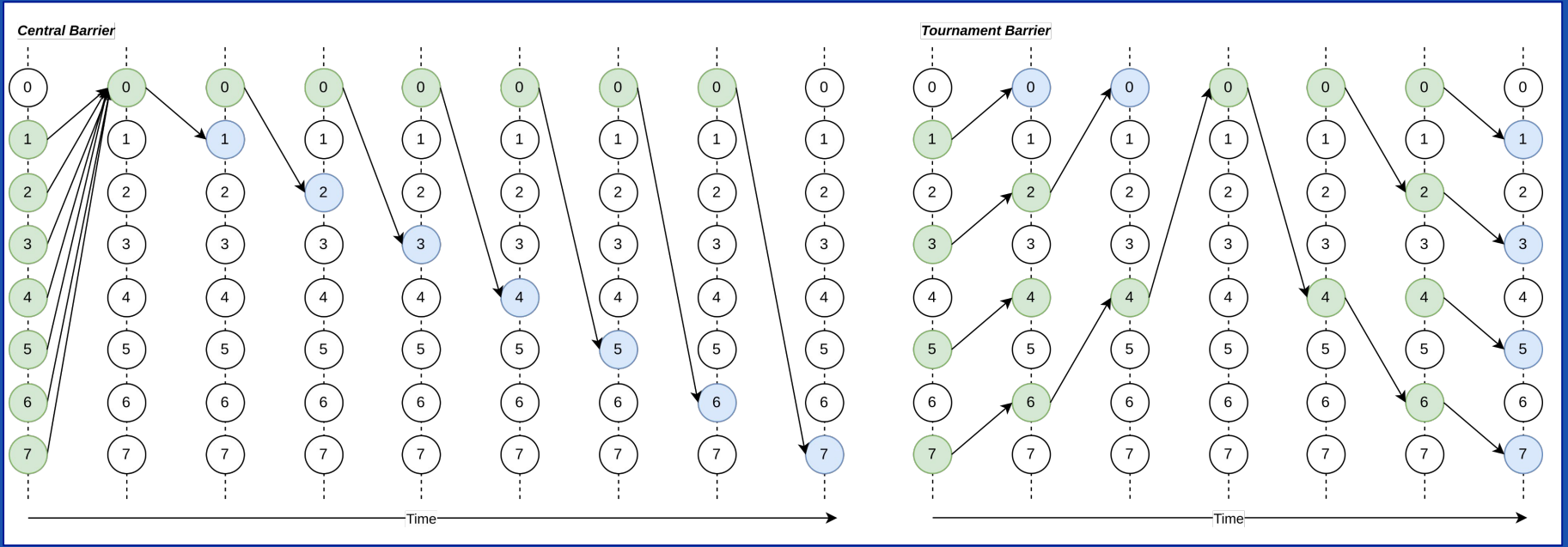
- **RunControl**: handling of DAQ and dataflow, configuration of the system, and controls run state
- **AutoPilot**: responsible of keeping the system running, automatically starts and recover from any state
- **BigBrother**: handling of dependencies between LHCb and the LHC. Controls the VELO closure, voltage subsystems, and RunControl

Thank you for your attention

… and thanks to the LHCb Collaboration for this Experiment

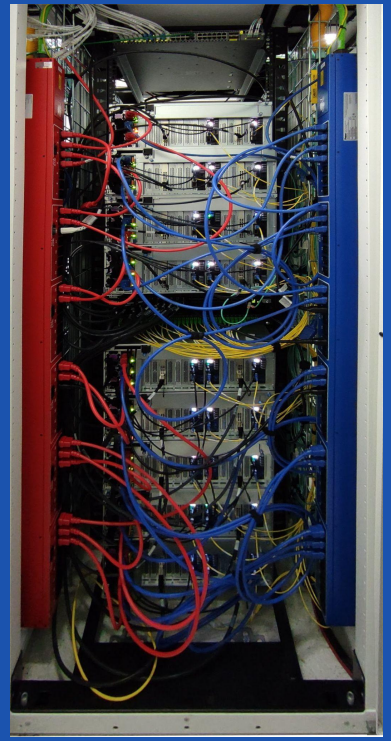
BACKUP

Synchronization Algorithms



Event Builder Farm

- 163 EB servers
- 24 racks: 18 EB, 2 control, 4 storage
- 28 40-port IB HDR switches: 18 leaf and 10 spine



HLT1 GPU Scheduling

Events are physically independent and can be reconstructed in parallel.

Parallelism is achieved in three ways:

- Run several sequences in parallel
- Batch multiple events in each sequence
- Exploit data-parallel patterns in each reconstruction algorithm

Batching events introduces a scheduling problem.

A multi-event scheduler has been created in order to optimise the execution of the algorithms.

The scheduling heuristics is based on two main principles:

- Spreading algorithms with similar profiles (e.g. data-dependent or compute-dependent)
- Exploit previous sequence knowledge to reduce branching

Event Size Model

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

* Actual Size depends on Luminosity