LHCb DAQ & event filter in 2021- 2024 A triggerless readout

Niko Neufeld (CERN)

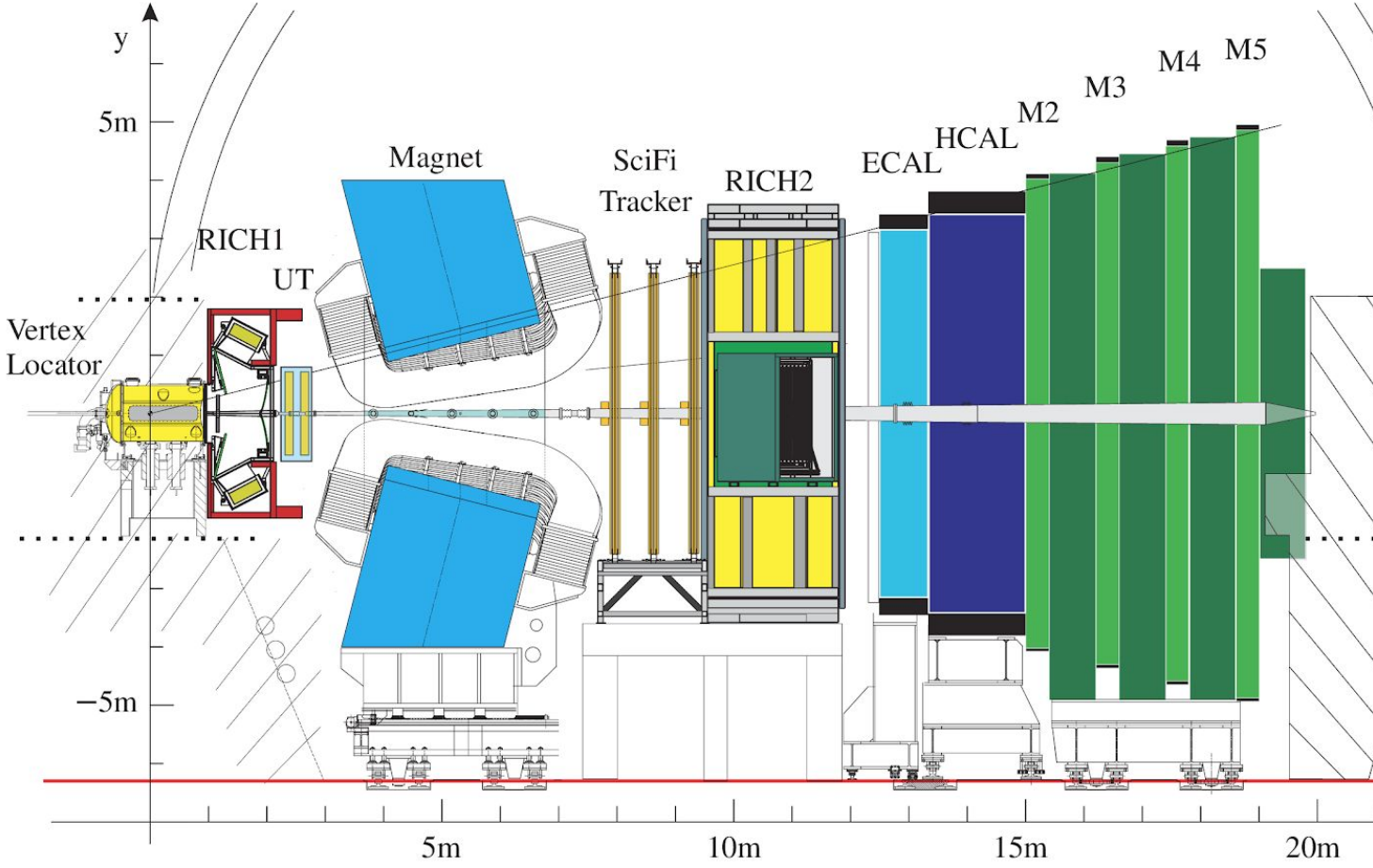
Muon Collider Physics and Detector Workshop
2 Jun - 4 Jun 2021
https://indico.cern.ch/event/1037447/

Acknowledgements

- Drawing on the work of the whole LHCb Online team (and LHCb at large)
- A lot of credit for these slides goes to T. Colombo (CERN/EP)

LHCb in 2021- 2024

- Single-arm forward spectrometer at the LHC
- p-p bunch crossing rate: 30 MHz
- Luminosity: $2 \times 10^{33}\ cm^{-2}s^{-1}$



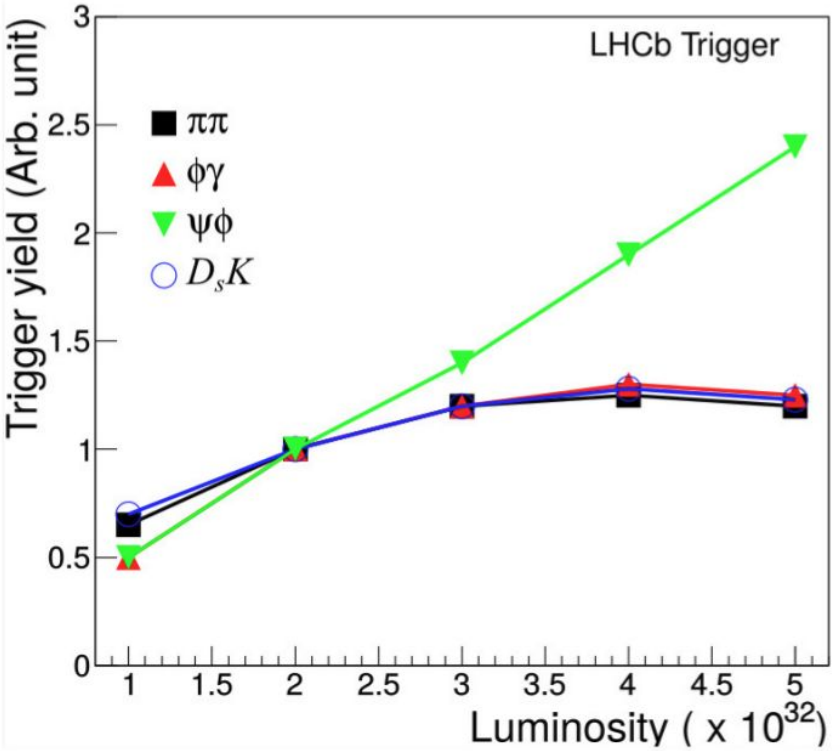
Trigger-less readout: why?

- With traditional calorimeter+muons trigger:
- Increase in luminosity

  ≠

  increase in “interesting” events
- As luminosity grows, thresholds must be increased to keep rate constant
- Trigger inefficiency from higher thresholds is not compensated by higher lumi

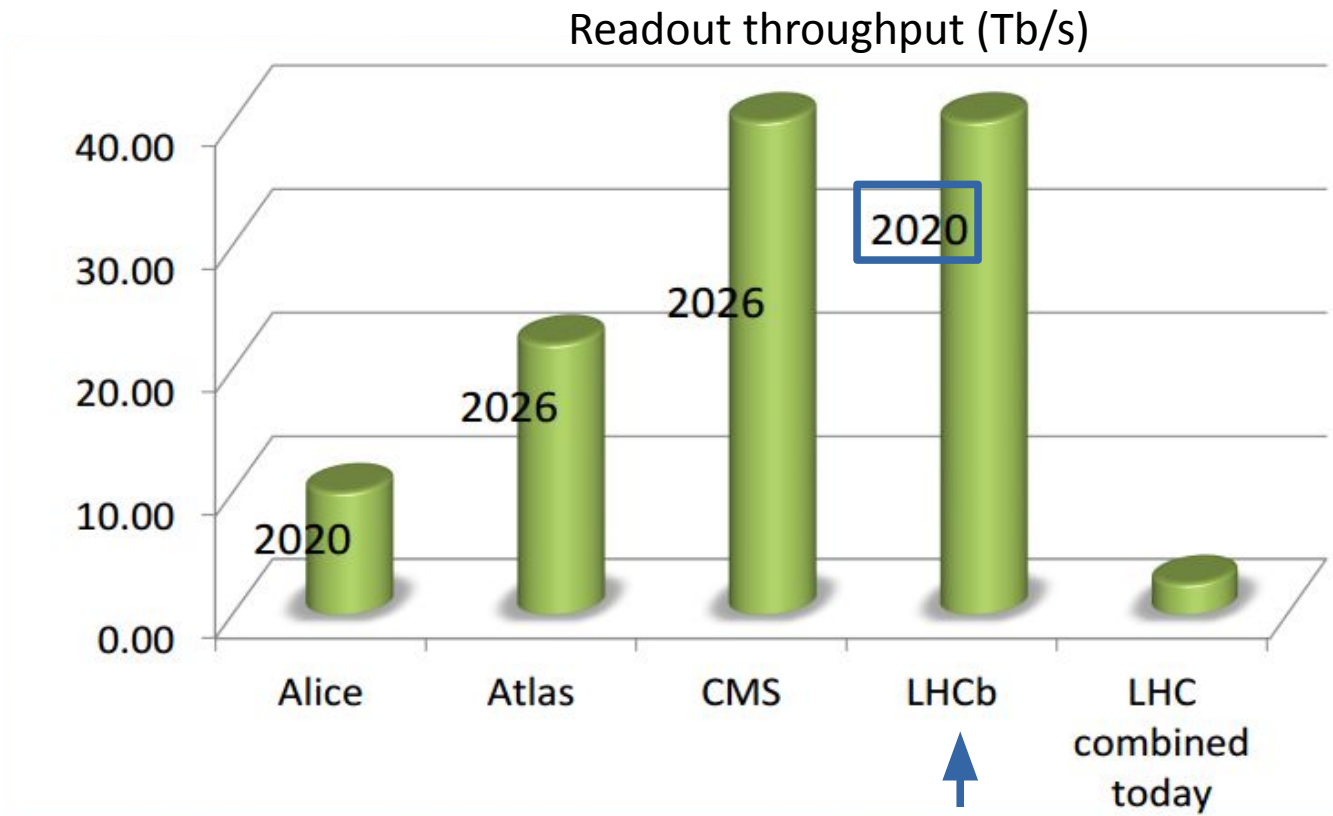
Low level trigger yield vs Luminosity ($cm^{-2}\ s^{-1}$) for a trigger rate of 1 MHz



Trigger-less readout: how?

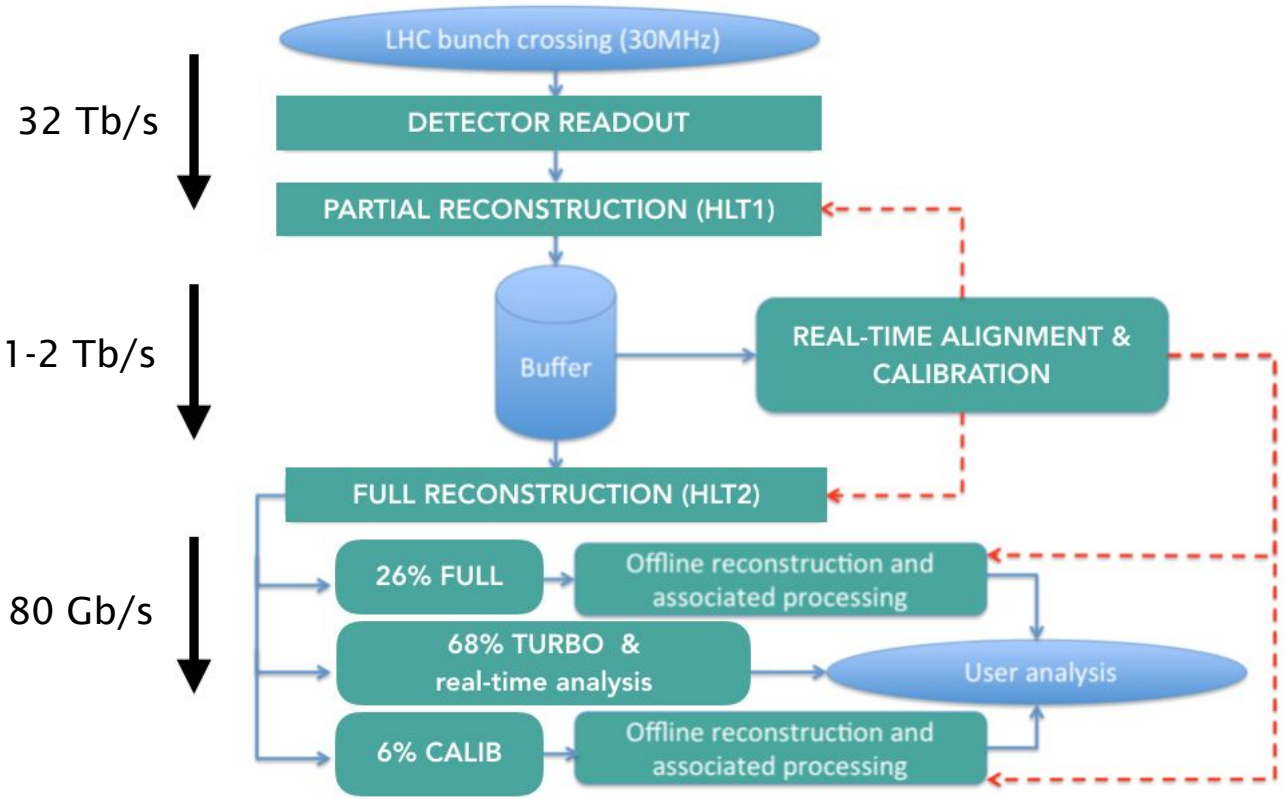
- Spectrometer geometry:
  fibres/cables are not "in the way"
- Relatively low radiation levels allow relaxed radiation-hardness requirements for FPGAs in many detector front-ends
- Zero-suppression on the detectors
- Total event size comparatively small (~100 kB)
- Bonus:
  software trigger can do online selection with offline-like reconstruction

Trigger-less readout: when?

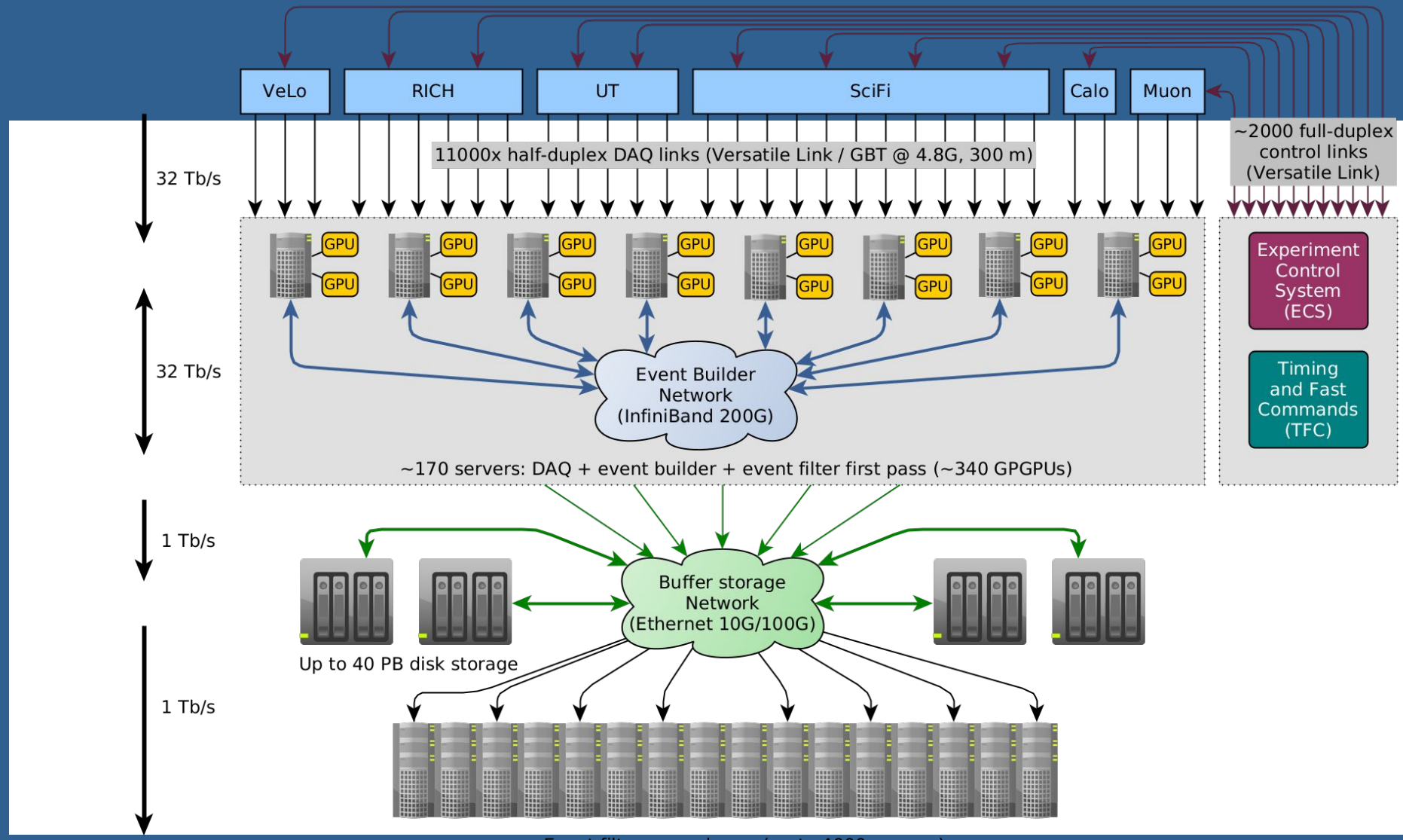


Data-processing and event selection

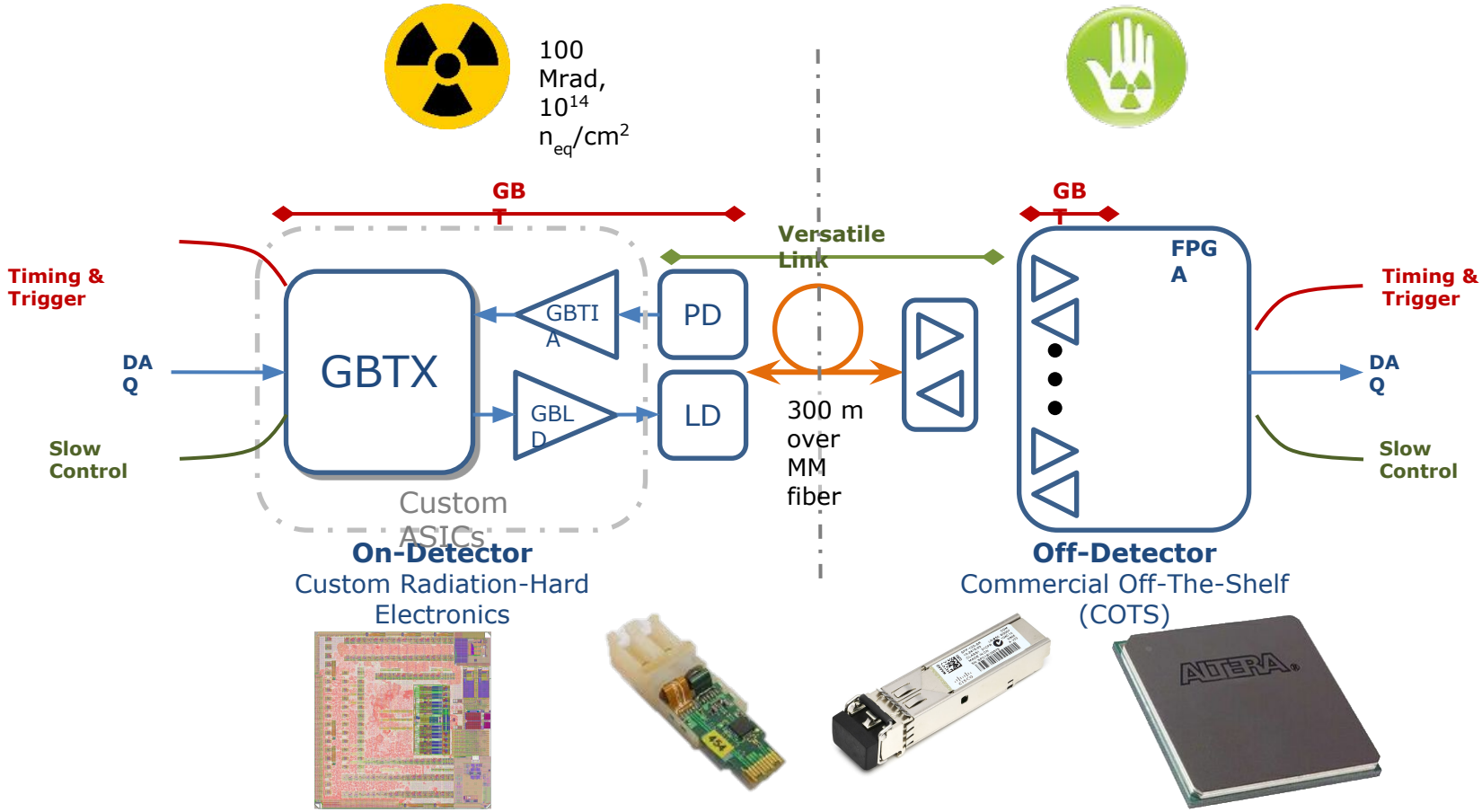
- Two stages of software filtering:
  1) "HLT1" on GPGPUs
  2) "HLT2" on CPUs
- Large storage buffer to decouple the two
- Calibration and alignment are performed "semi-live", while the data are buffered



System overview

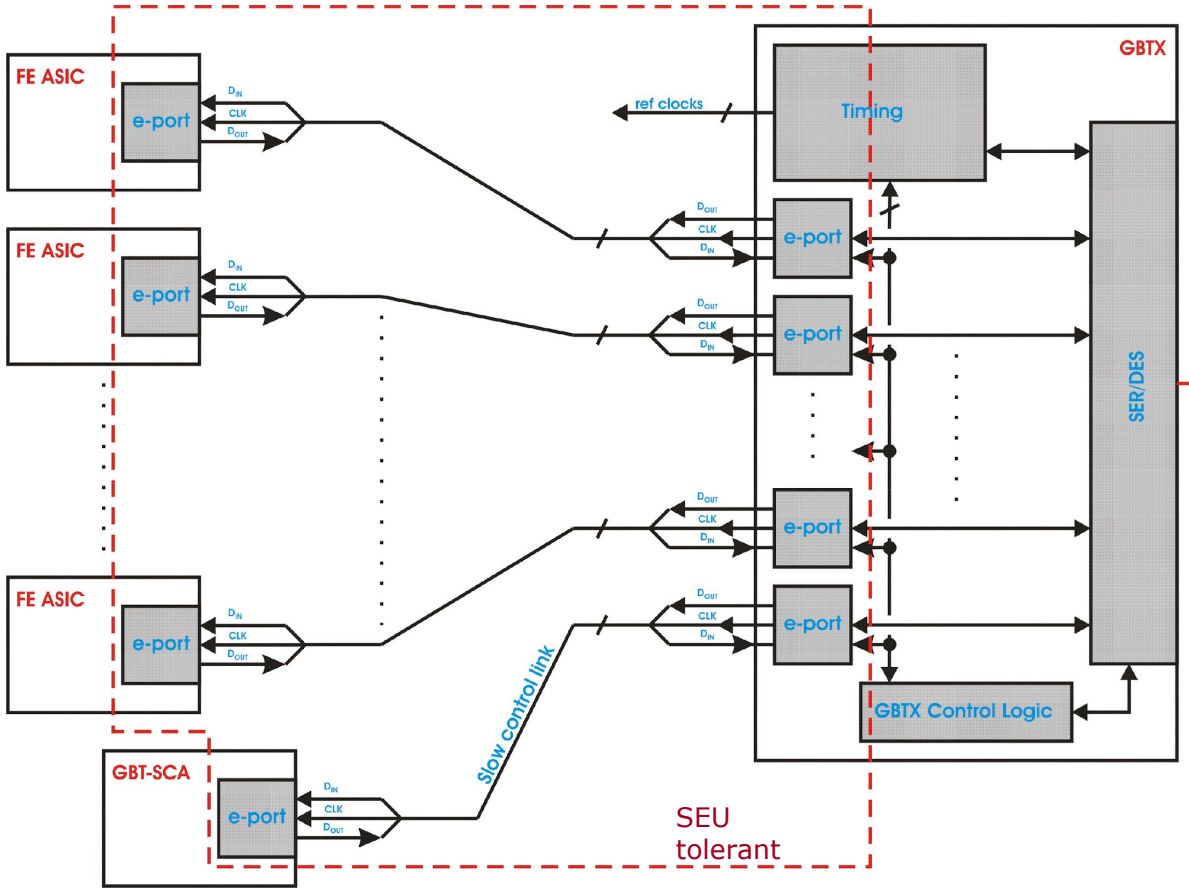


Front-end: GBT over Versatile Link



Credit: P. Moreira S. Baron (CERN)

Front-end: GBTx multiplexing



- GBT/Frontend interface: Electrical links (e-link)
  - Serial, bidirectional
- Up to 40 links per ASIC
- Programmable data rate:
- 40×80, 20×160, or 10×320 Mb/s

Credit: P. Moreira (CERN)

Back-end: PCIe40

A single custom-made FPGA board for DAQ and Control

- Based on Intel Arria10
- 48x10G-capable transceivers on 8xMPO for up to 48 full-duplex Versatile Links
- 2 dedicated 10G SFP+ for timing distribution
- 16x PCIe 3.0

One board, many firmware personalities

1 Readout Supervisor (SODIN)

- Reception and distribution of global 40 MHz timing
- Generation and distribution of synchronous and asynchronous commands
- Event type (physics, calibration, empty) generation

One board, many firmware personalities

42 Interface Boards (SOL40)

- Distribution of the global timing to the front-ends
- Interface bridge between the control system and the front-ends

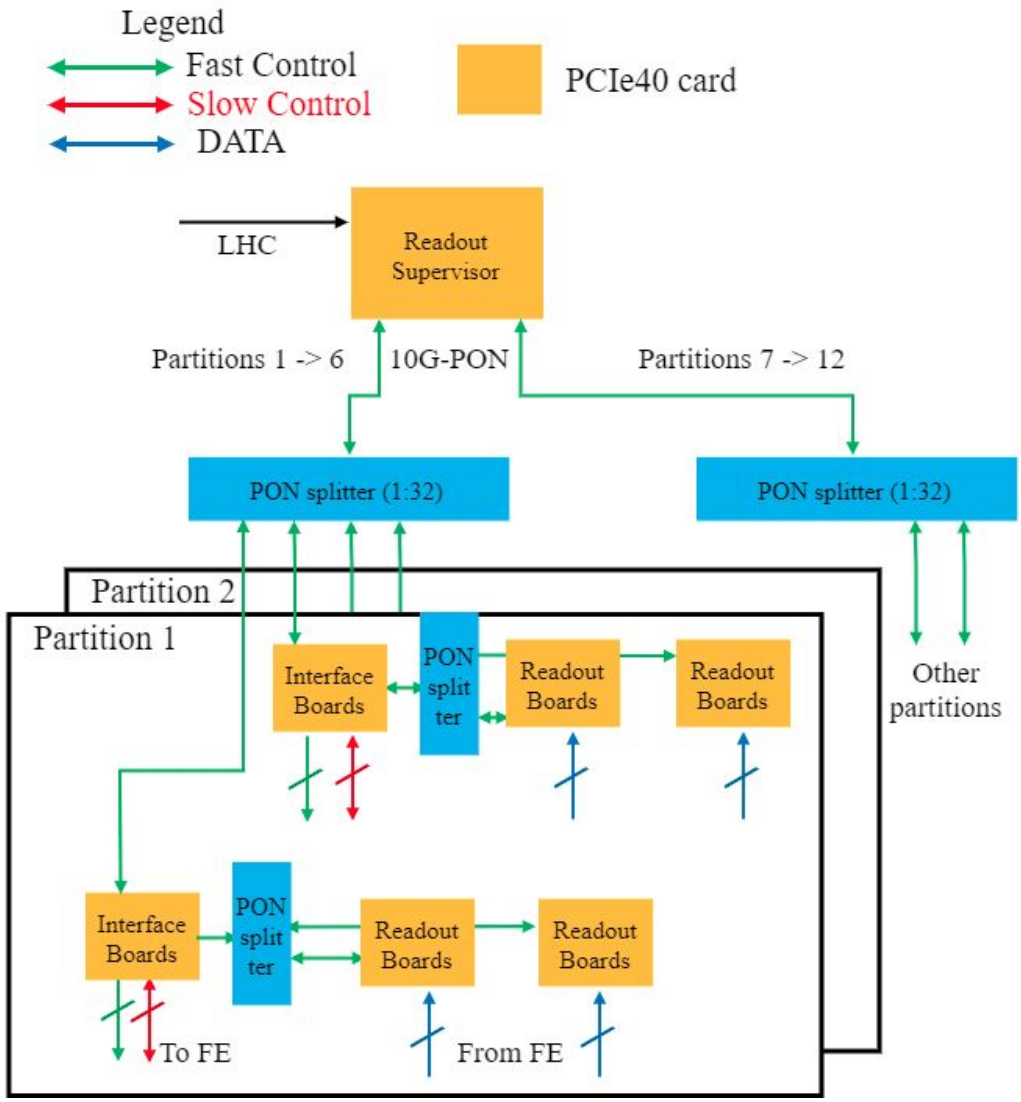
One board, many firmware personalities

478 Readout Boards (TELL40)

- Data Acquisition
- First pre-processing of the data
- E.g.:
  - Re-ordering and separation on event boundaries of streaming data
  - Hit clustering

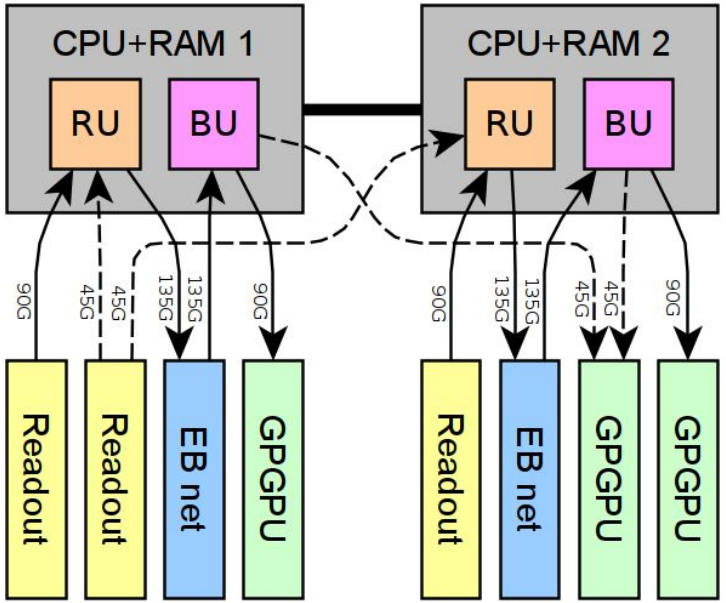
Timing and Fast Commands

- Synchronously driving the Front-End electronics over GBT
- 10G-PON for efficient Back-End signal distribution and fixed phase clock recovery
- Partitioning for debugging and commissioning



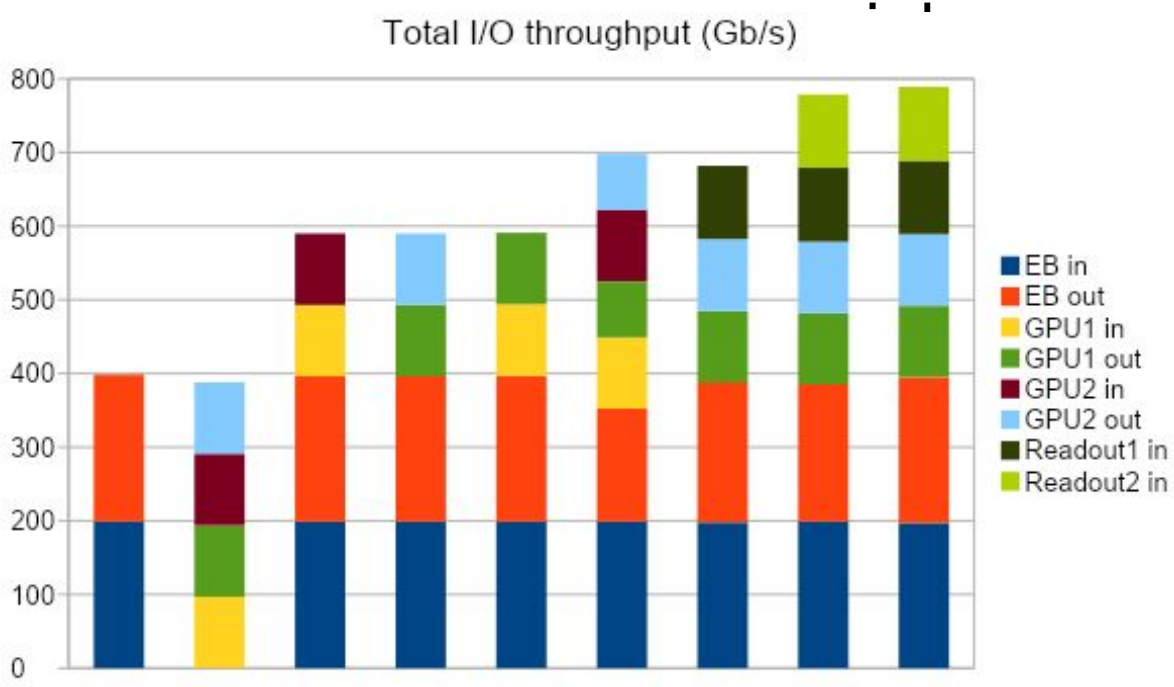
Event builder server

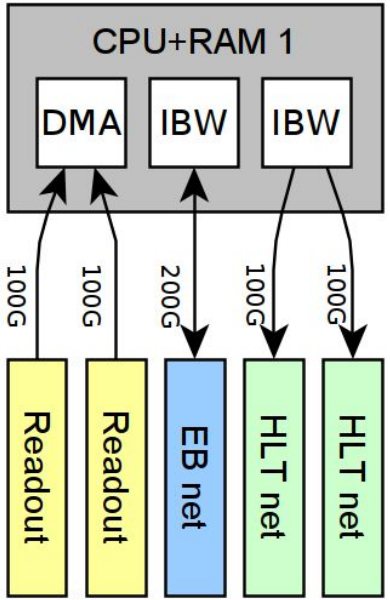
- 2 AMD EPYC 7002-series CPUs
  - PCIe 4.0
  - 8+8 DDR4 channels
- 3 readout boards
- 2 InfiniBand 200G NICs
- Up to 3 GPUs
- 512 GiB RAM (buffer to decouple EB and readout)



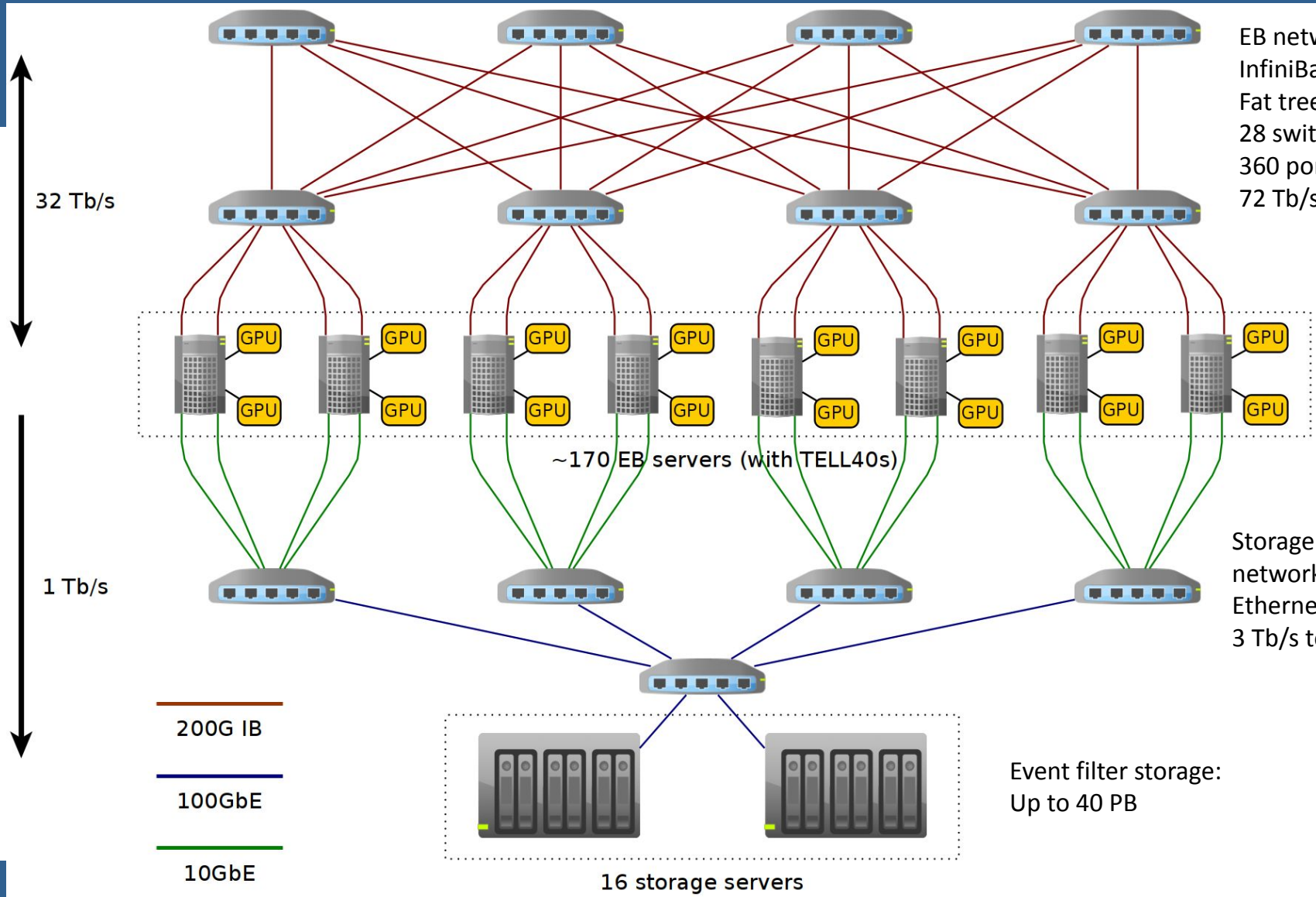
Challenges for EB servers

Memory subsystem pushed to the limits! RDMA is





Event builder networks



EB network:
InfiniBand 200G
Fat tree
28 switches
360 ports
72 Tb/s total

Storage / filter network:
Ethernet 10/100G
3 Tb/s total

Event filter storage:
Up to 40 PB

Challenges for the EB network

- Needs to collect data from 478 readout boards into a single "location"
- And hand it over to GPGPUs + CPUs for further processing
- Want high link-load (keeping costs low)
- Want to use some kind of remote DMA to reduce server-load
- Traffic is inherently congestion-inducing
- → Our solution: careful application-level traffic scheduling
- → Specialized routing algorithm for our network topology (fat tree)

Summary

- LHCb can do and afford a full read-out at bunch-crossing rate
- Single stage synchronous readout built around GBT and a single flexible FPGA board
- Detector control uses the same FPGA boards as the timing distribution system
- AMD Rome (PCIe Gen4) based servers make compact, very-high-I/O event-builder, connected with 200 Gb/s InfiniBand
- Event-selection is entirely in software to maximize physics yield, increase the amount of data collected, flexibility and minimize cost
- The system is very well scalable, by up to 3 a factor without any substantial changes

Discussion

Triggerless read-out: Why?

- No trigger is 100% efficient, hardware triggers always have to compromise
- A trigger-less front-end is much *less complex* and *more robust*
  - No buffering, no selection logic,
- Selection in “software” / using compute infrastructure from the data-centre world has a lot of advantages:
  - Scalability
  - Cost: costs in electronics are driven (down) by scale
  - **Cost-efficiency**: 1 USD spent on an ASIC trigger board will be “active” only during the operation of the trigger, 1 USD spent on a CPU or GPU can be “active” round the clock
  - Flexibility: new hard and software can be integrated without impacting the detector or operations
  - Operations: compute for filtering does not need to be “on-prem”, *not operated by experiment, not even “owned”*

Triggerless read-out: Why not?

- More data from the front-end
  - 0-suppression / compression / clever encoding required
  - more links → more power, more cost, more material
- Cultural shift:
  - from electronics to software engineers
  - from hardware projects to software commitments: What does your funding agency say?

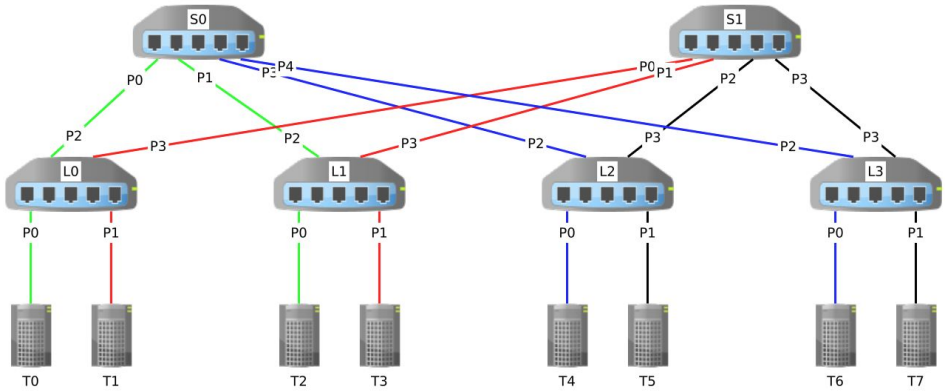
Food for thought

- Power and cost for front-end links must be weighed against power and cost for intelligence and trigger logic on the the front-end
- Only removing the trigger altogether brings simplification benefits
- Even in a trigger-less system it is very useful (indispensable?) to have a synchronous command (and monitoring) channel
- Trigger-less systems are more easily read out asynchronously (but time-stamps will be needed) or in a streaming fashion which can be an advantage for certain slower detector technologies (e.g. the readout of the ALICE TPC for Run3 and beyond)

Details on Multi-Tbit/s event-building

Event building, a.k.a. MPI_Alltoall

- Traffic pattern is *all-to-all gather*: For each event, one “builder” server receives fragments from all servers
- Schedule: linear shift
  - With N servers, the transfer of N events is divided into N phases
  - In every phase each server exchanges data with only one server
- If the start of a phase is synchronized, and the network is non-blocking
  → no link conflicts!



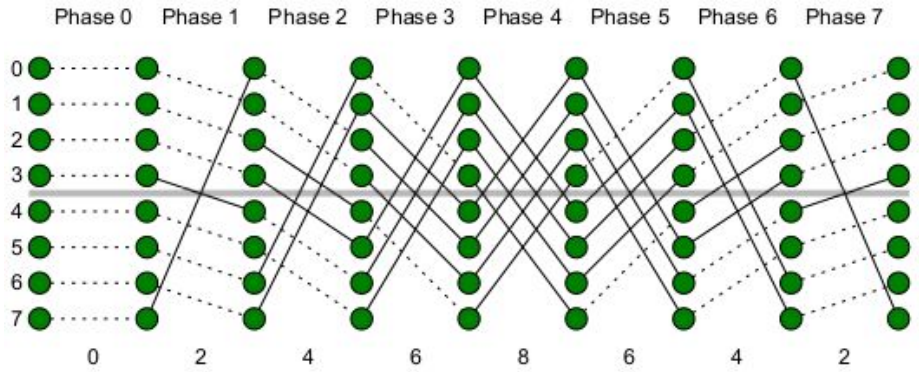
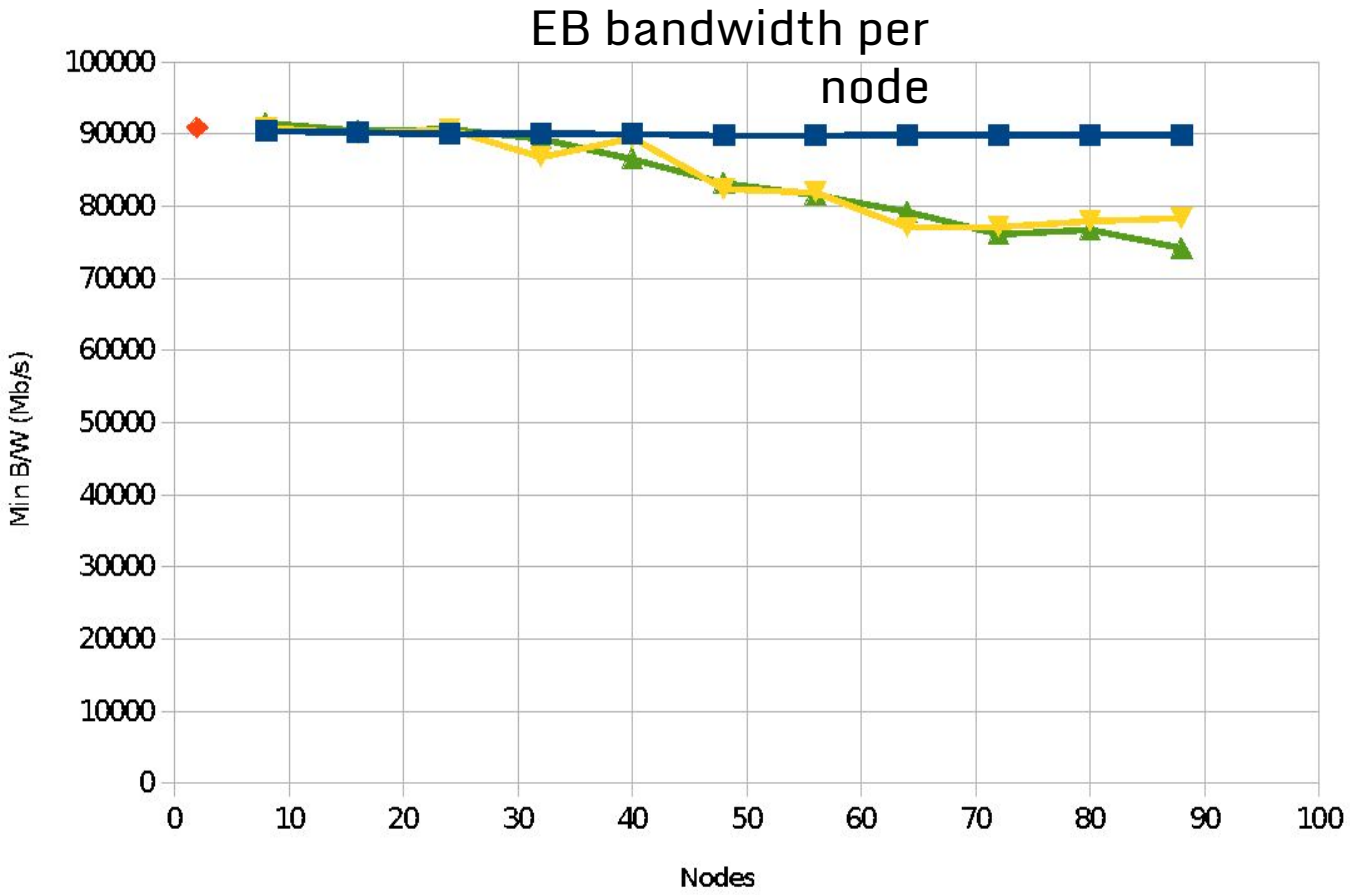


Image credit: B. Prisacari et al.

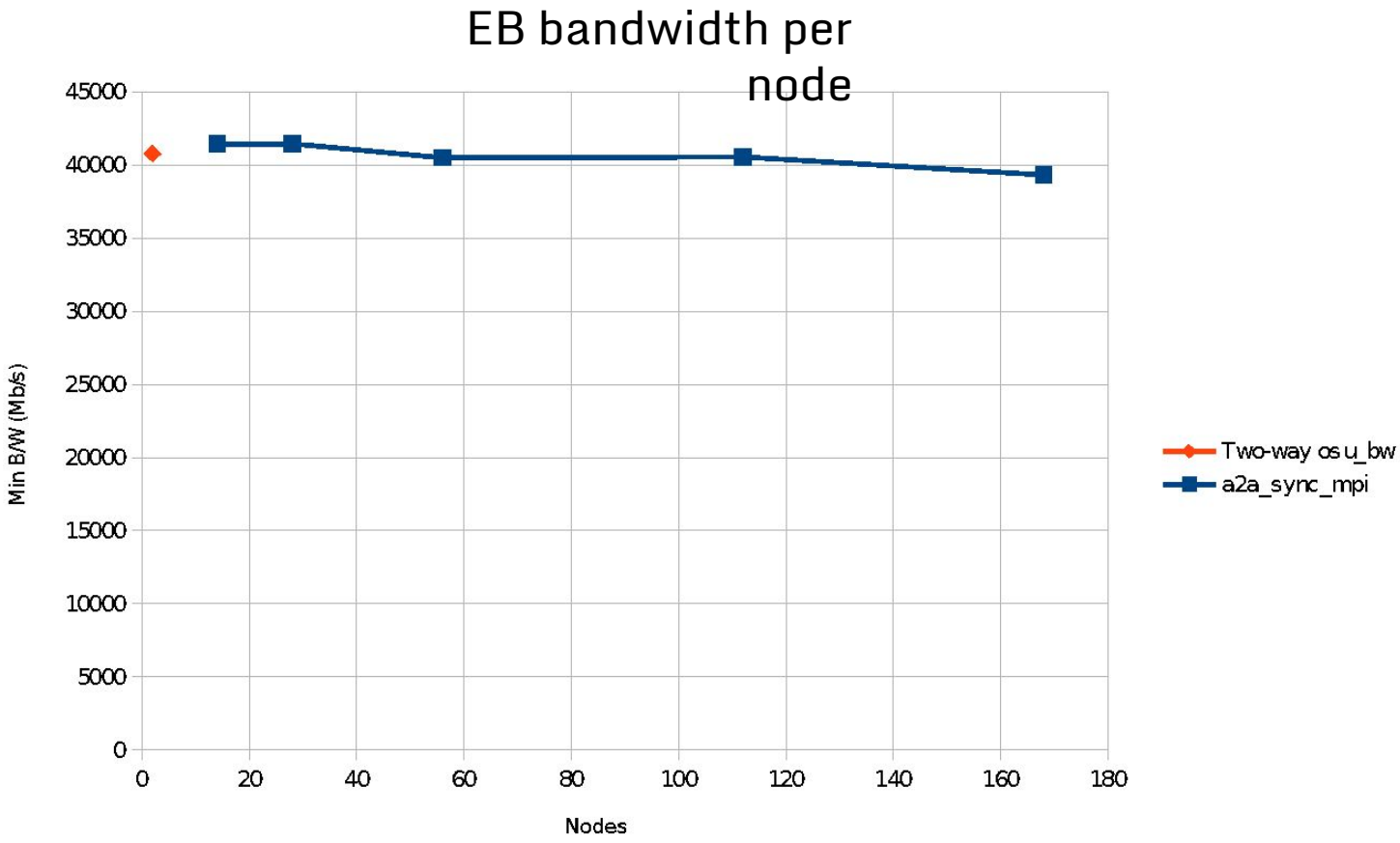
Scalability on InfiniBand



Tested at the Goethe-HLR HPC cluster (InfiniBand 100G)

With the right traffic shaping, almost perfect scalability!

Scalability on InfiniBand



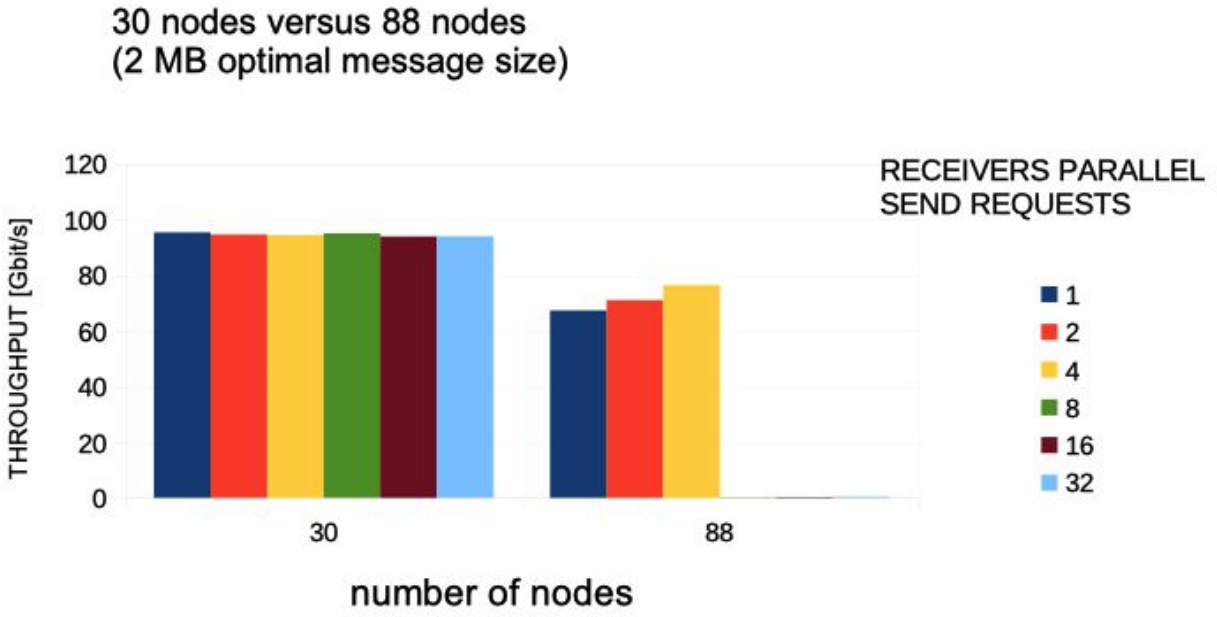
Tested on the CMS DAQ (InfiniBand 56G)

Very good scalability with almost 200 nodes

Why InfiniBand?

- PCIe Gen4 allows using 200 Gbit/s connections:
  Lower cost, better scalability, but so far only effectively exist for IB!
- Remote DMA is crucial for EB server performance:
  - RDMA implementations do not like packet drops:
    either deep buffers or good flow control are needed.
  - Deep buffers @ 100G = expensive/non-existent RAM tech.
  - Many flow-control bugs found on available reference platforms.
- **Could never get access to a really big Ethernet test system:**
  Network congestion issues only appear at scale.
  For InfiniBand we have used super-computer sites.
- Lowest risk solution – within our budget – is the InfiniBand solution

Scalability on Ethernet with deep buffers



- Deep buffers alone don't save us
- Hardware flow control from many Ethernet vendors is flakey