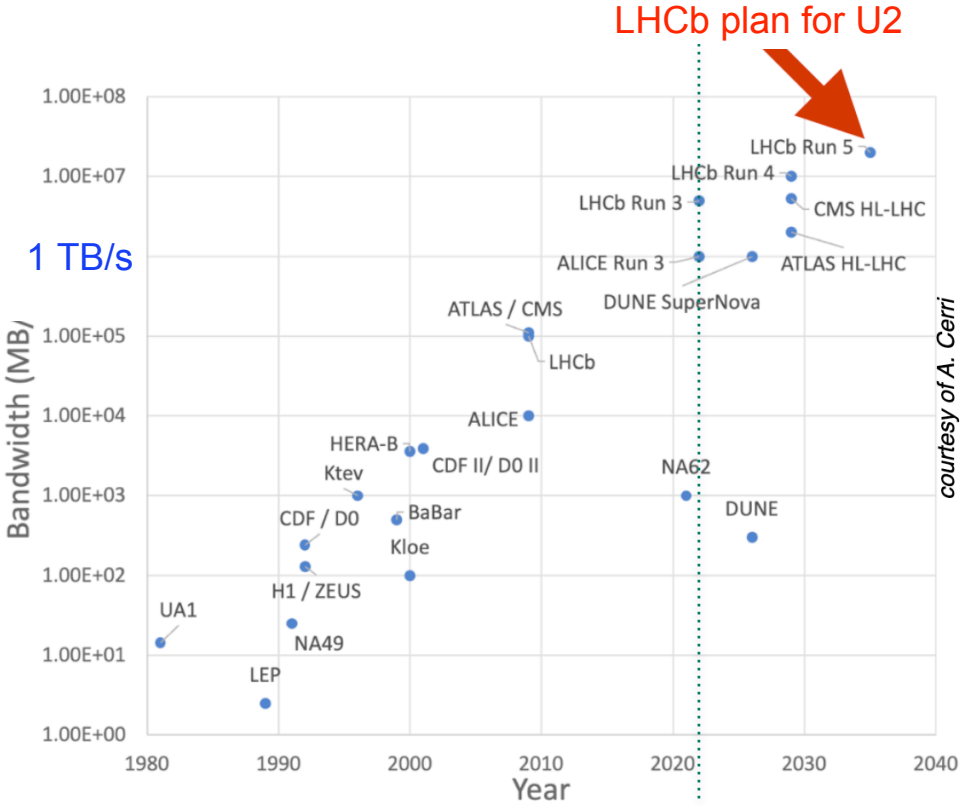
Detector-embedded reconstruction of complex primitives using FPGAs

Giovanni Punzi - Università di Pisa & INFN
on behalf of RETINA group in LHCb Real Time Analysis project

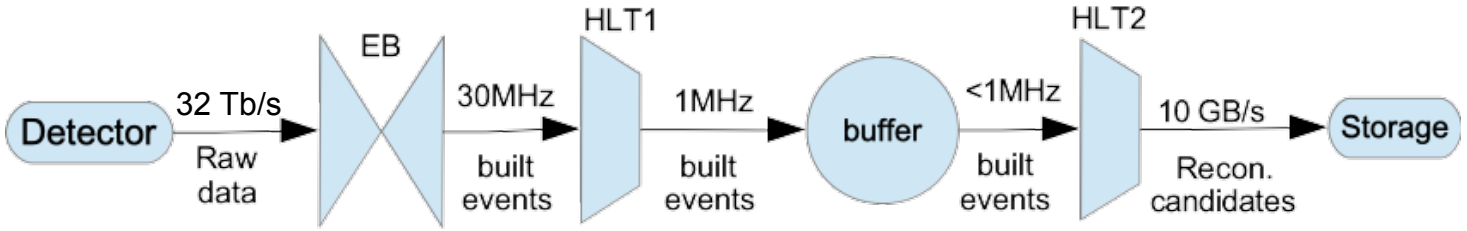
Pisa Meeting on Advanced Detectors - May 28th 2024

The need for speed

- Progress of experiments goes together with increasing data processing rate.
- Flavor physics at low Pt most demanding: LHCb riding on top, in spite of smaller size and lower lumi than other LHC experiments.
- LHCb is effectively "processing-limited"
- I describe an effort to accelerate LHCb reconstruction, using FPGAs closely integrated with detector readout.

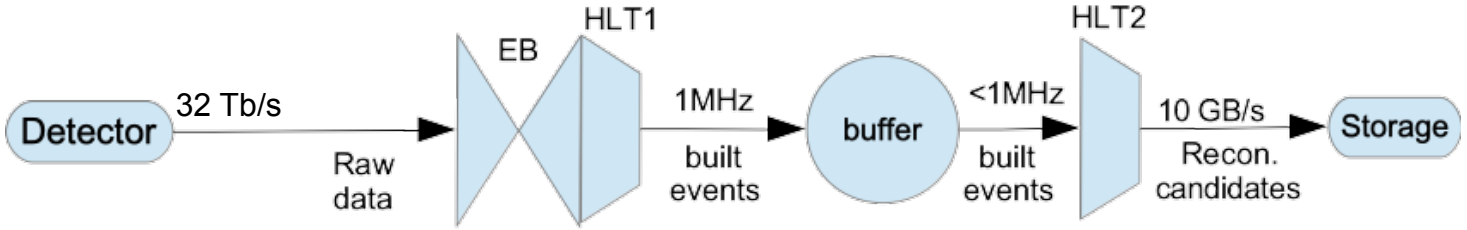


The LHCb Data Processing model



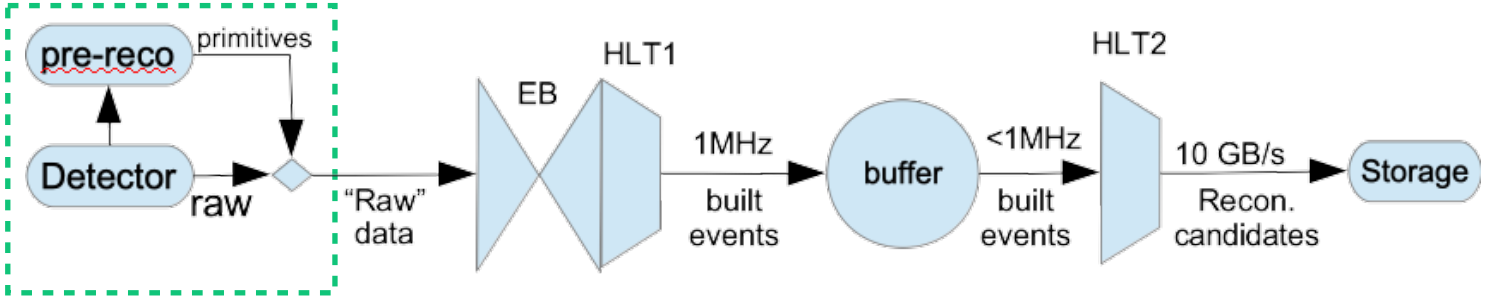
- Physics of Flavor physics at low Pt: no easy Level-0 selection, need to process in detail the whole event.
- Triggerless readout of the whole detector + full event reconstruction before first trigger decision is made
  (often referred to as 'trigger less', or 'full-software trigger')
- Two-level DAQ: HLT1 (full reco, for trigger purpose), HLT2 (physics reconstruction + final selection).
  - Alignment happens between HLT1 and HLT2, to make sure HLT2 reco is final.
    (Large disk buffer in the middle)

The LHCb Data Processing model

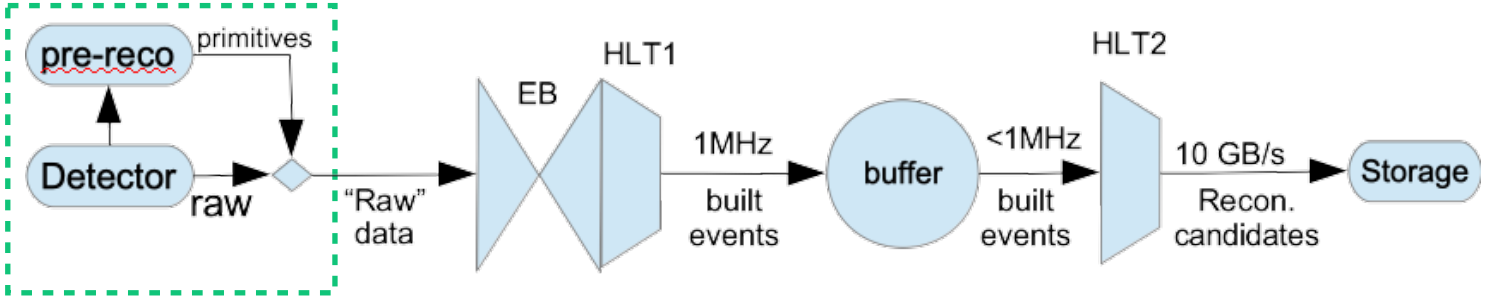


- Physics of Flavor physics at low Pt: no easy Level-0 selection, need to process in detail the whole event.
- Triggerless readout of the whole detector + full event reconstruction before first trigger decision is made
  (often referred to as 'trigger less', or 'full-software trigger')
- Two-level DAQ: HLT1 (full reco, for trigger purpose), HLT2 (physics reconstruction + final selection).
  - Alignment happens between HLT1 and HLT2, to make sure HLT2 reco is final.
    (Large disk buffer in the middle)
  - In Run3, HLT1 moved physically inside the Event Builder to save on data transport, and turned to GPUs for better efficiency, cost.
- What can we still improve, in view of the Upgrade-II of LHCb, with ~7x larger Lumi ?

Evolving towards primitive-based reconstruction

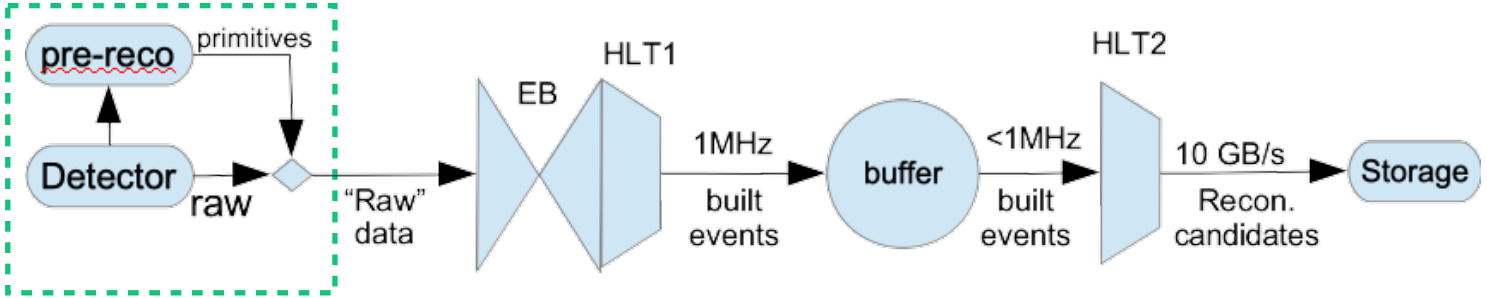


Evolving towards primitive-based reconstruction



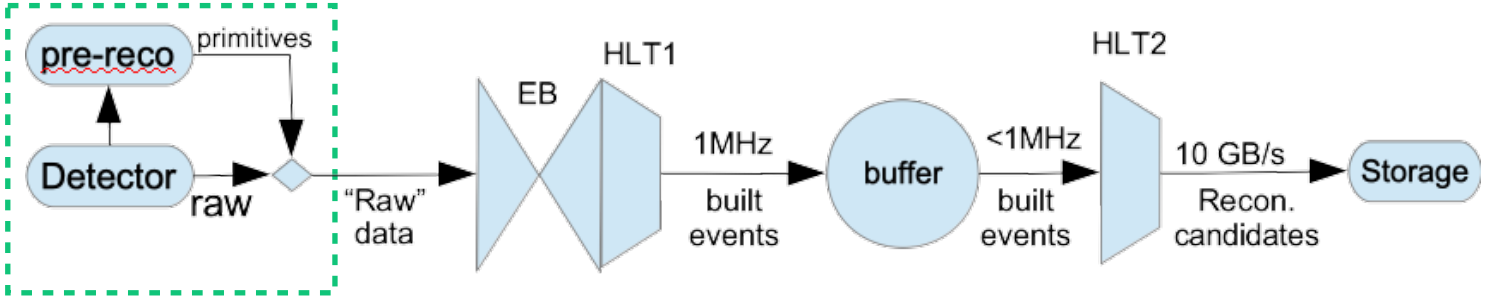
- Push processing before EB: reconstruct intermediate data structures ("primitives") using ~local info.
  - Ex. Track segments, muon stubs...
  - Logically embed in the detector block: make primitives look like "Raw Data" to the DAQ.

Evolving towards primitive-based reconstruction



- Push processing before EB: reconstruct intermediate data structures ("primitives") using ~local info.
  - Ex. Track segments, muon stubs...
  - Logically embed in the detector block: make primitives look like "Raw Data" to the DAQ.
- Advantages:
  - Accelerate HLT reconstruction: easier to combine segments than hits, both in HLT1 e HLT2.
  - Reduce data flow at the source (drop hits not on a track, for instance)

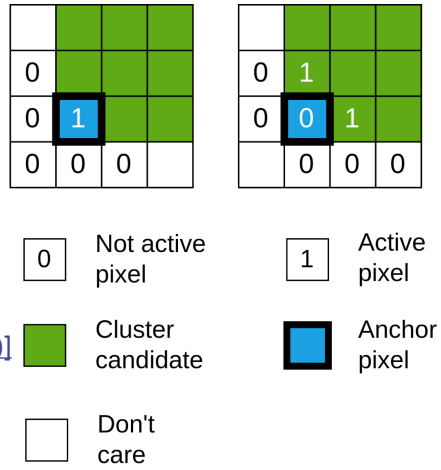
Evolving towards primitive-based reconstruction

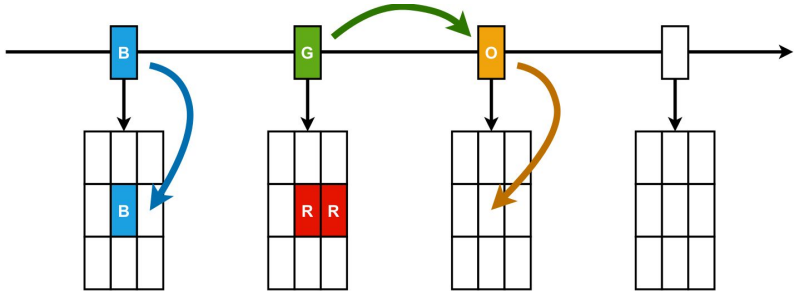


- Push processing before EB: reconstruct intermediate data structures ("primitives") using ~local info.
  - Ex. Track segments, muon stubs...
  - Logically embed in the detector block: make primitives look like "Raw Data" to the DAQ.
- Advantages:
  - Accelerate HLT reconstruction: easier to combine segments than hits, both in HLT1 e HLT2.
  - Reduce data flow at the source (drop hits not on a track, for instance)
- Drawback: it is hard !
  - Can't use time-multiplexing 'a la GPU': (dividing rate by ~300). Need to *actually* process a new event every 25ns.
  - Large b/w, little buffering, constrained latency.
  - CMS' track "vectors/stubs" are a solution using on-detector ASICs [see Macchiolo on Monday]
  - For more complex primitives we adopted (off-detector) FPGAs , programmed at data-flow level.

A 'complex' primitive: hits in the VELO pixel detector

- Hits in the VELO detector of LHCb appear as 2D clusters of pixels [see dedicated VELO talk]
- Firmware deployed in Run3 in FPGA readout boards to make clusters on the fly (Arria 10)
  - Original plan was to do this during HLT1 reconstruction
- Pixels read out as 2*4 arrays (SuperPixels, SP). Clusters found by unpacking them into active matrices, where each pixel actively checks if it belongs to a pattern. Centroid evaluated by LUT.
- Fast solution, but unmanageable to cover the 40M pixels of the VELO
- Solution: dynamically allocate small matrixes where active pixels are found [IEEE TNS 70, 6 (2023)]
  -> allows to process data continuously, yielding a **throughput of 10^{11} hits/s**



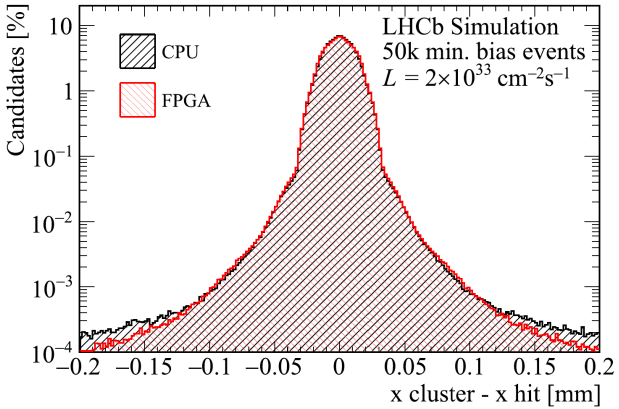


Benefits of embedded Cluster finding

- Quality of real-time cluster reconstruction as good as CPU algorithm
  - Raw pixel information **dropped** and replaced by hit positions during readout (saves 15% of b/w)
- FPGA implementation saves 12% of HLT1 computing power, and uses 1/50th of the electrical power [IEEE TNS 70, 6 (2023)]

-> Now established as the default method at LHCb.

- Side benefits: real-time availability of 10^{11} hits/s **in accessible way** enables further applications (e.g. precision monitoring of beamline)

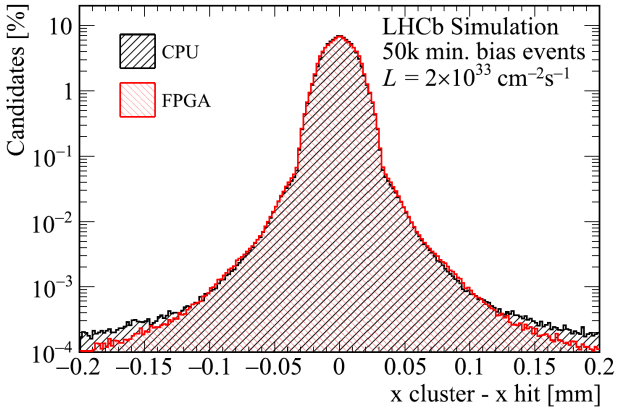


Benefits of embedded Cluster finding

- Quality of real-time cluster reconstruction as good as CPU algorithm
  - Raw pixel information **dropped** and replaced by hit positions during readout (saves 15% of b/w)
- FPGA implementation saves 12% of HLT1 computing power, and uses 1/50th of the electrical power [IEEE TNS 70, 6 (2023)]

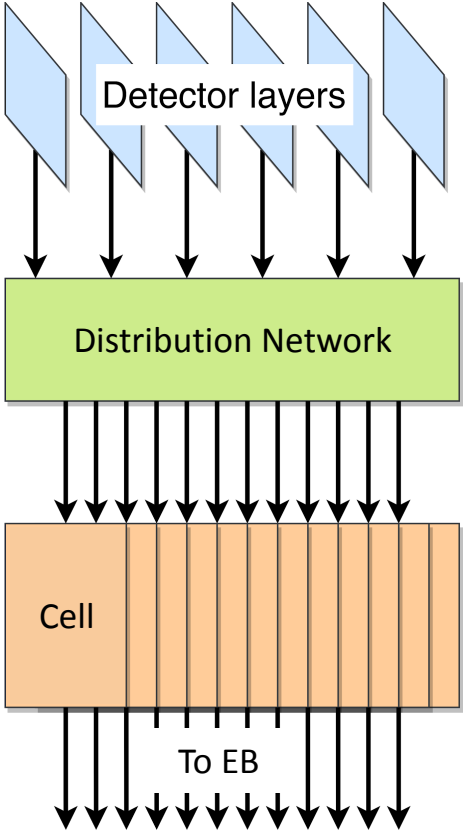
-> Now established as the default method at LHCb.

- Side benefits: real-time availability of 10^{11} hits/s **in accessible way** enables further applications (e.g. precision monitoring of beamline)



'Local' application: all required data accessible in a single FPGA
Next we discuss a more complex solution involving multiple FPGAs

'Retina' Architecture



Input from detector over multiple lines

Distribution network:

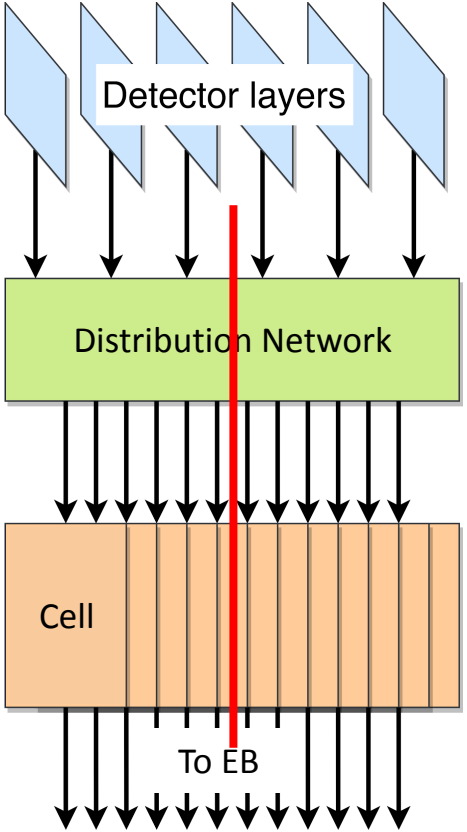
- Custom Switch: routes hits only to appropriate cells.

Cell:

- Evaluates presence of primitive in specific region

Primitives are sent to the Event Builder.

'Retina' Architecture



Input from detector over multiple lines

Distribution network:

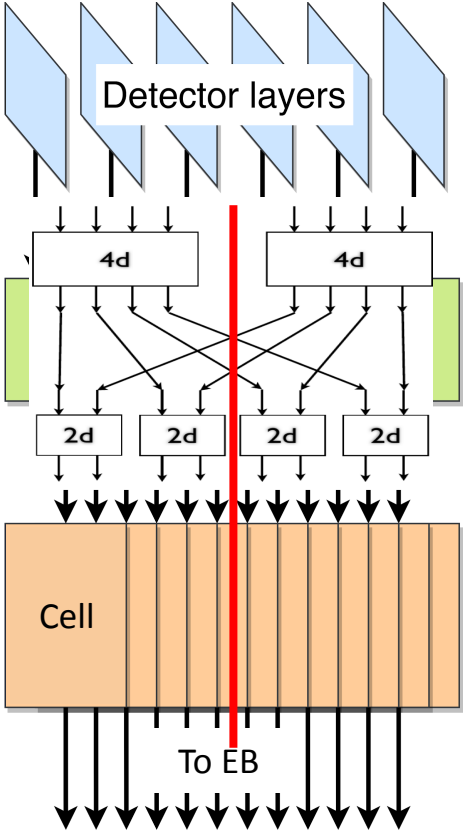
- Custom Switch: routes hits only to appropriate cells.

Cell:

- Evaluates presence of primitive in specific region

Primitives are sent to the Event Builder.

Modular design



Input from detector over multiple lines

Distribution network:

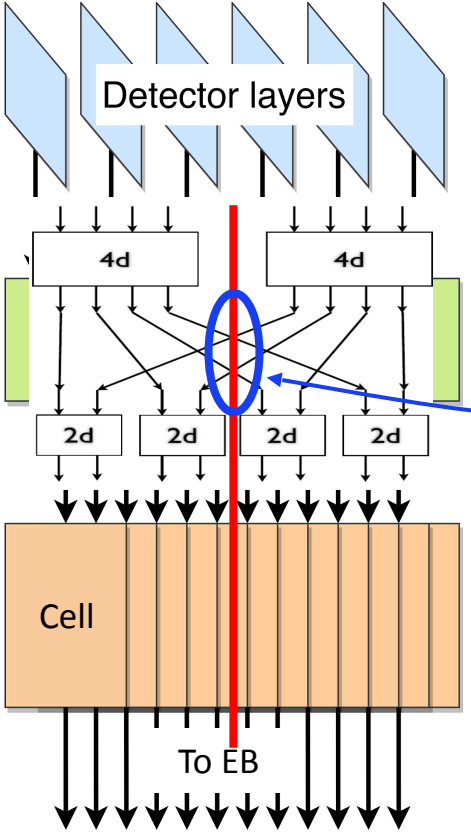
- Custom Switch: routes hits only to appropriate cells.

Cell:

- Evaluates presence of primitive in specific region

Primitives are sent to the Event Builder.

Modular design



Input from detector over multiple lines

Distribution network:

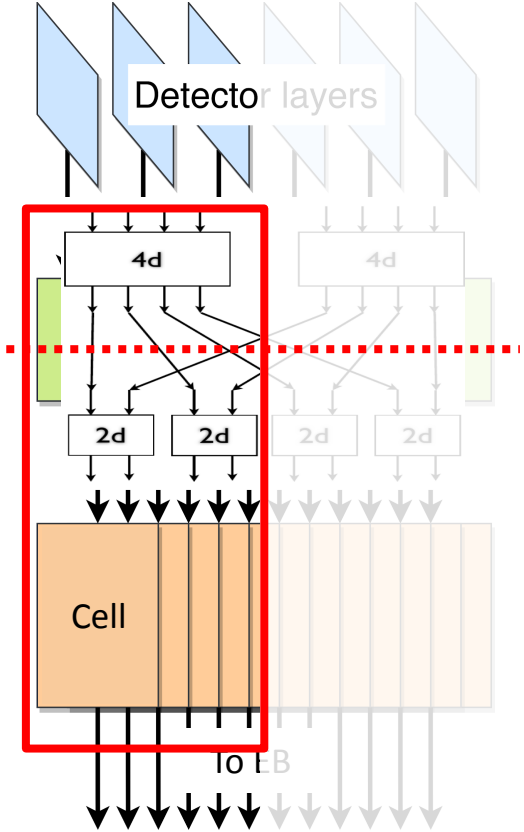
- Custom Switch: routes hits only to appropriate cells.
- Optical communication: exchanges hits between boards.

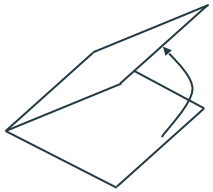
Cell:

- Evaluates presence of primitive in specific region

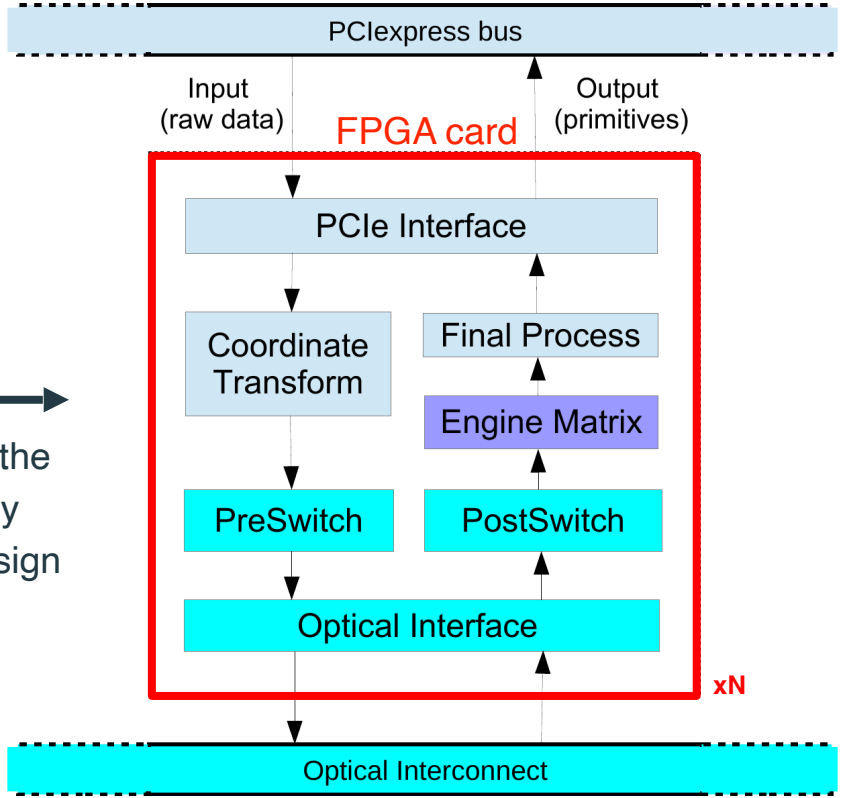
Primitives are sent to the Event Builder.

Physical implementation

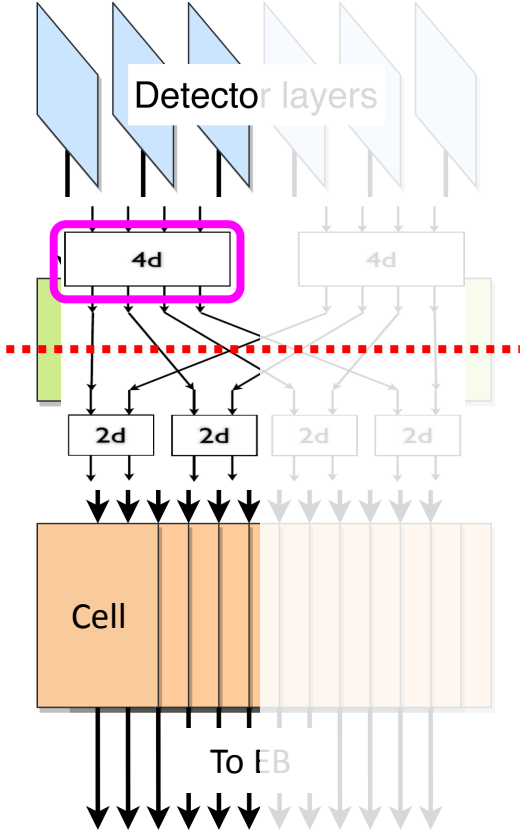


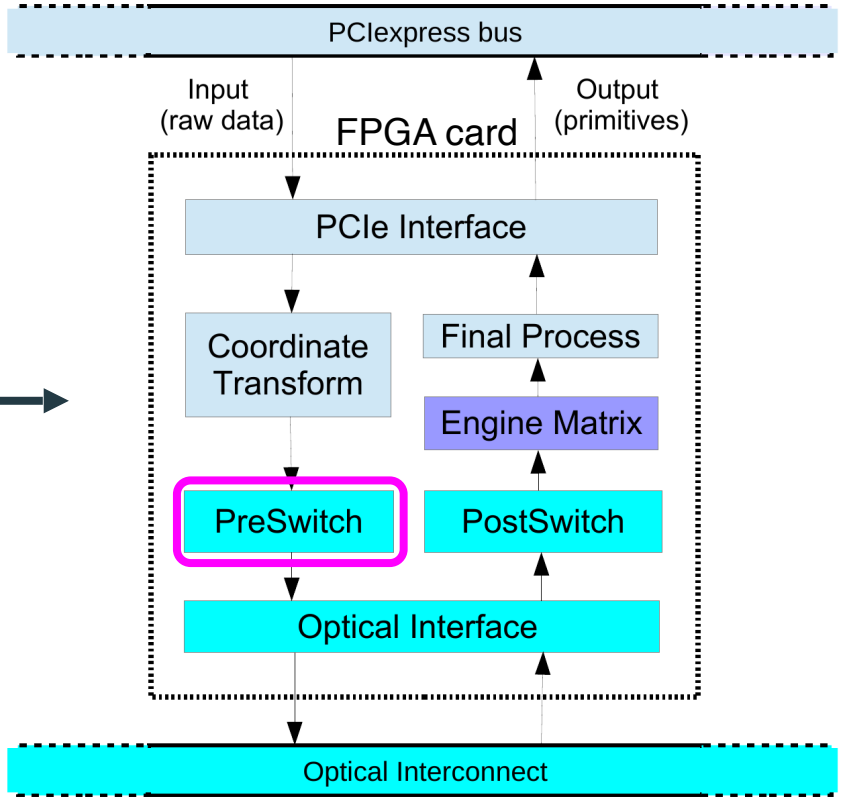


Embedded in the same FPGA by folding the design

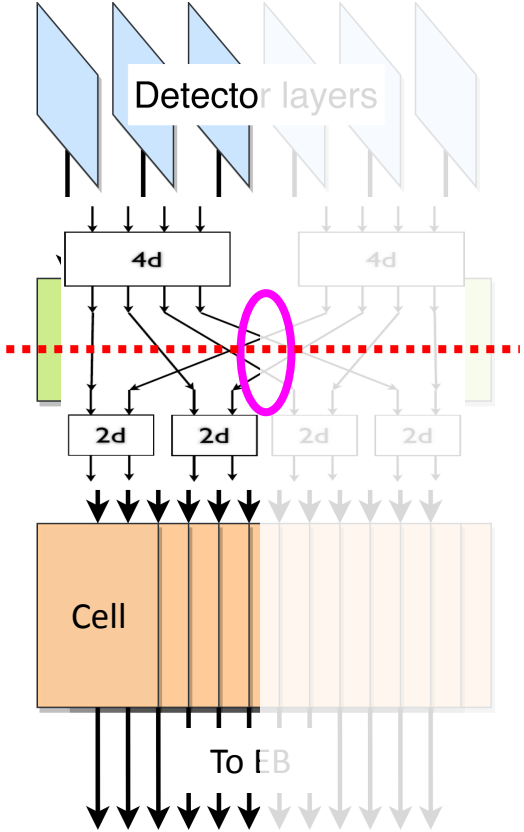


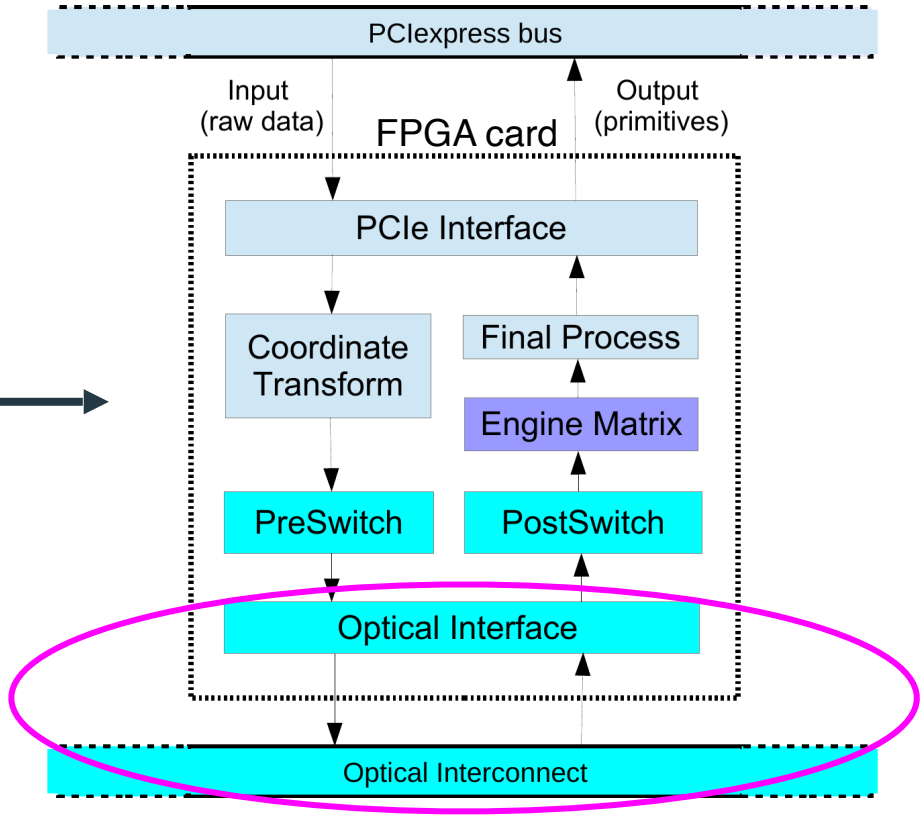
Physical implementation



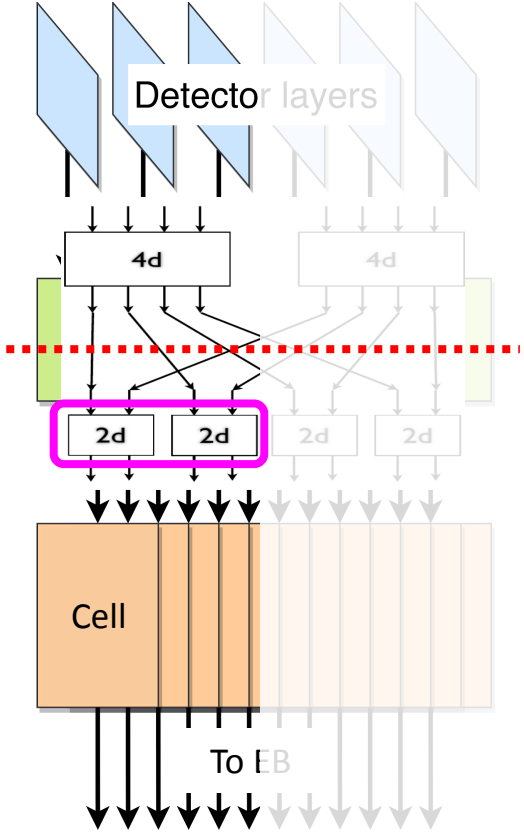


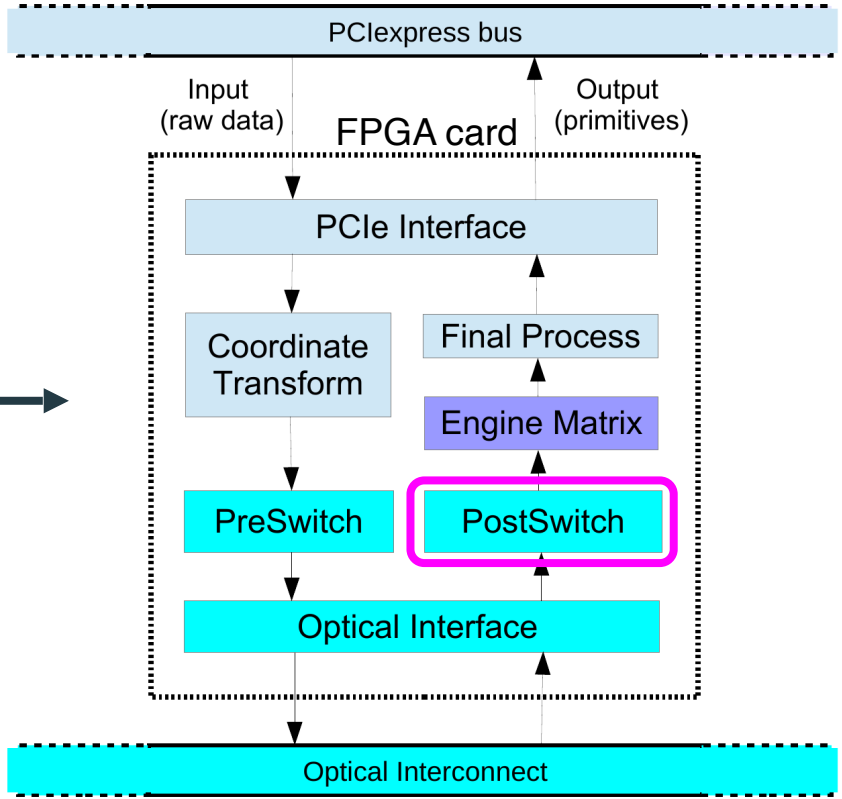
Physical implementation



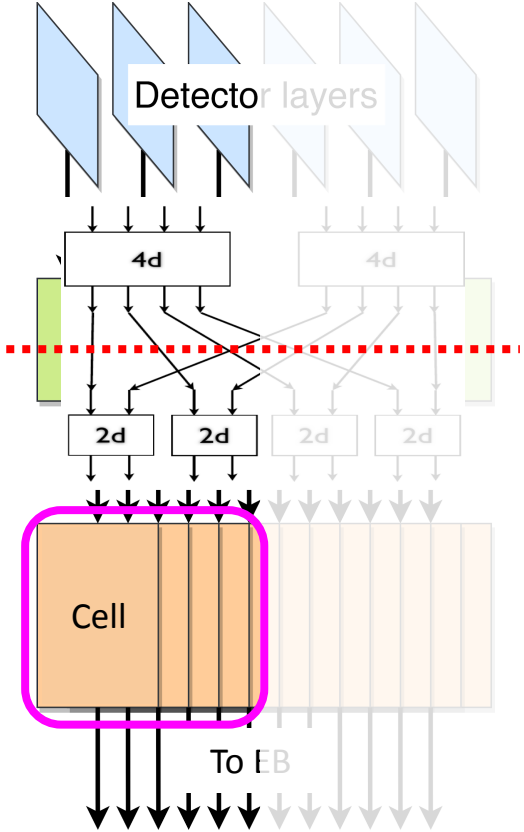


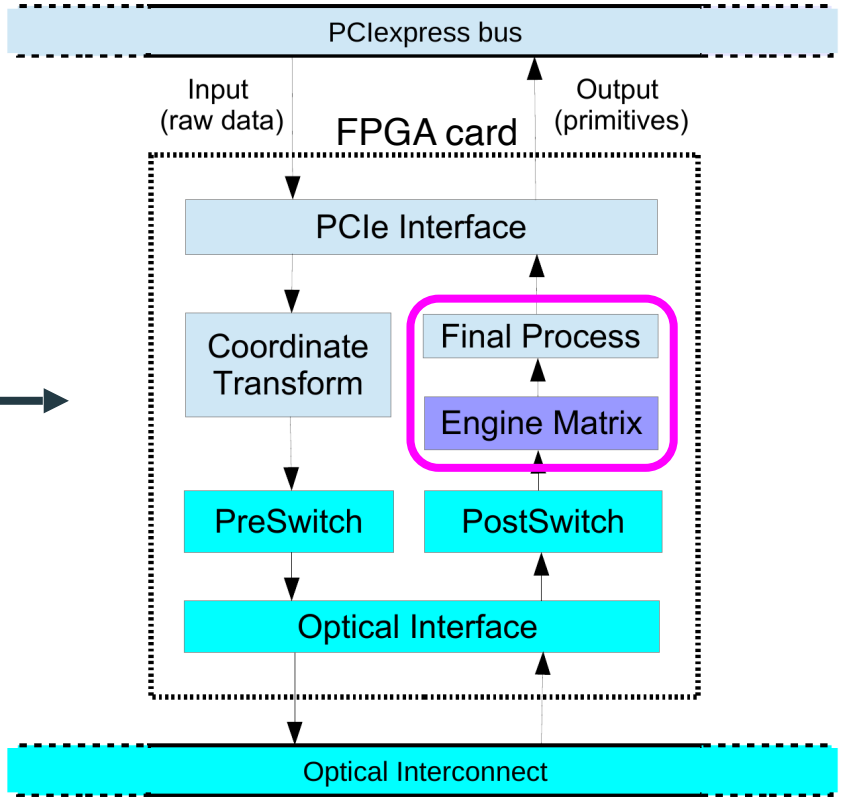
Physical implementation



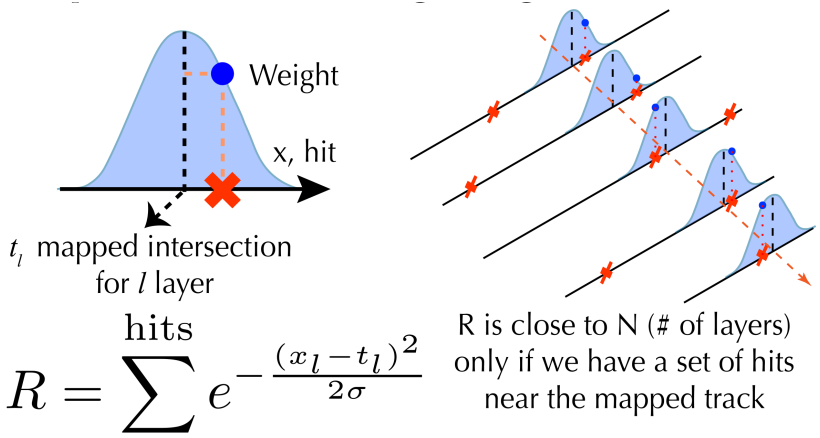


Physical implementation



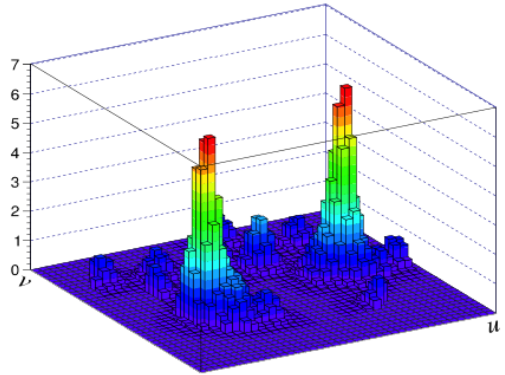


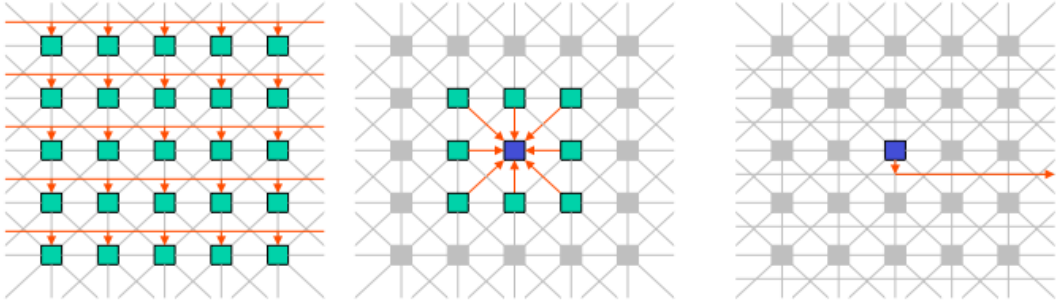
The “artificial retina” architecture: what happens inside a cell



- Each cell computes its response (*R*) as the weighted sum of inputs
  - For tracking, hits closer to the reference track get larger weight (Gaussian in the example)
- Digital analogue of "receptive fields" in vision processing in the natural brain
  - Hence the historical name 'retina architecture'
  - More specific than a generic 'neural net'
  - Calculation must happen is zero-time for the system to work

The “artificial retina” architecture: what happens in the cell matrix





INPUT
all cells in parallel

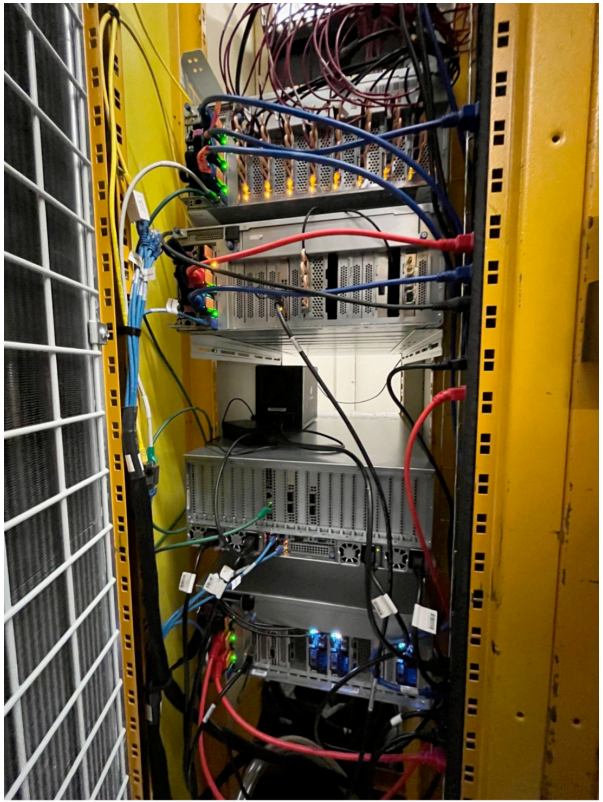
CLUSTER FIND
all cells in parallel

OUTPUT
sequential

- 3 steps happen simultaneously (pipelined) while input is coming, in order not to stop the flow:
    1. All cells are filled in parallel
    2. Clusters are found by local negotiations between neighboring cells
    3. Output of cluster centers are queued to output
- A final pipeline stage may be added to perform application-dependent processing

Hardware demonstrator

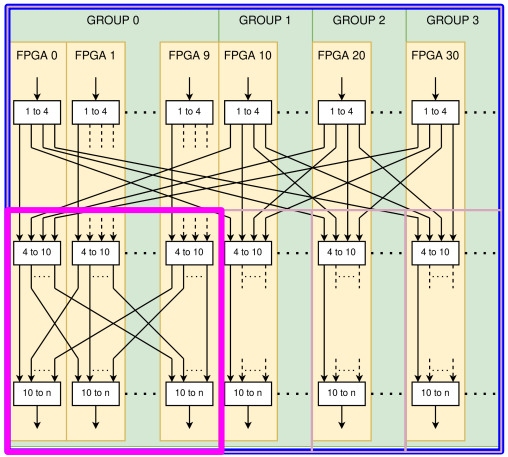
- A complete Retina demonstrator was installed and tested at the LHCb TestBed facility. Culmination of a decade-long effort.
- Reconstruct a VELO quadrant using 8 PCIe-hosted FPGA cards (Stratix-10, 2.8 MLE). (Takes VELO clusters as input)
- Test on LHCb MC @Run3 luminosity (2×10^{33} $cm^{-2}s^{-1}$).
  - Bit-by-bit comparison with software emulator gives perfect matching - running uninterruptedly for weeks.
  - Achieved **20 MHz event rate** (LHCb rate ~27 MHz)
    - Easily on target with optimization and current FPGAs
  - No buffering, sub-μs latency
  - Low power consumption 550 W

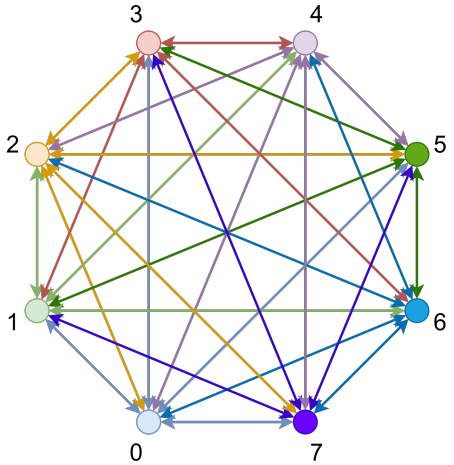


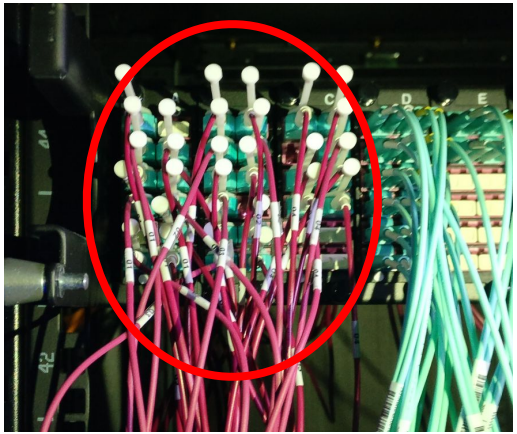
Detail of the switching network

- Topology: 8-nodes full-mesh network.
- 28 full-duplex optical links at 25.8 Gbps, total bandwidth 1.4 Tb/s.
- Open source protocol Intel SuperLite II v4
- Traffic managed by LUTs - dedicated optimization code for load-balancing

- Implemented via optical patch panel, allows for easy reconfiguration

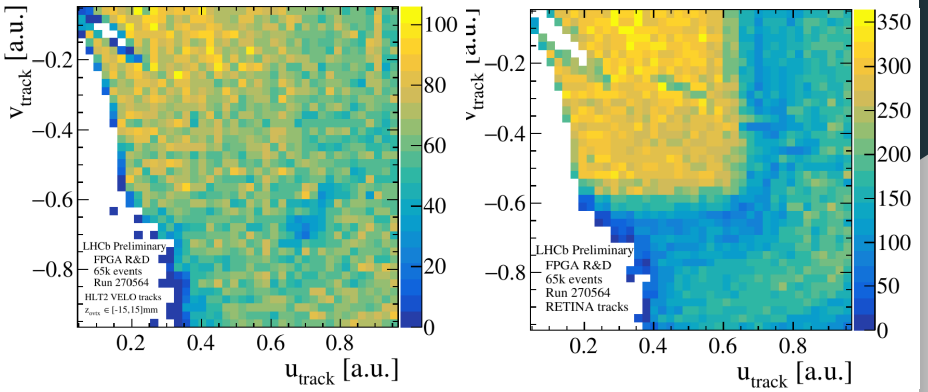






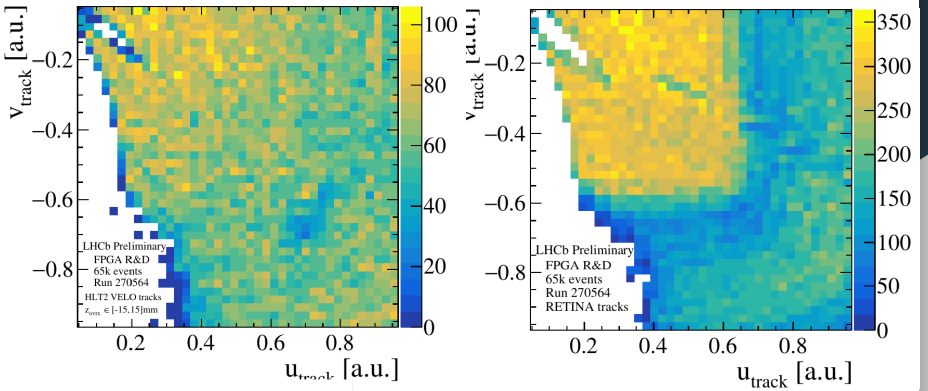
Results on live LHCb data

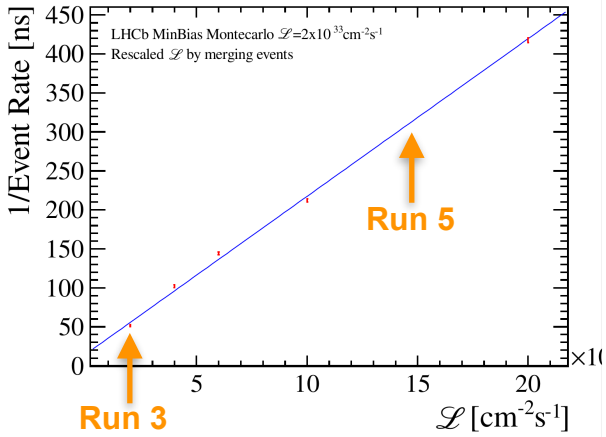
- Currently Running parasitically on real LHCb data during Run 3 physics data taking (at reduced rate)
- Online LHCb alignment constant applied on the fly.
- Tracks distribution from demonstrator (right) very similar to HLT2 output (left).

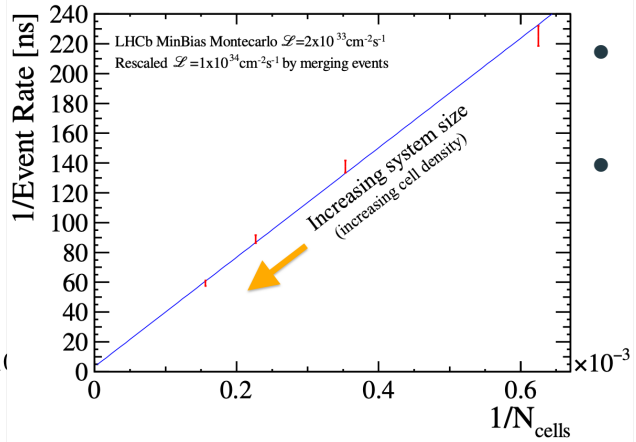


Results on live LHCb data

- Currently Running parasitically on real LHCb data during Run 3 physics data taking (at reduced rate)
- Online LHCb alignment constant applied on the fly.
- Tracks distribution from demonstrator (right) very similar to HLT2 output (left).



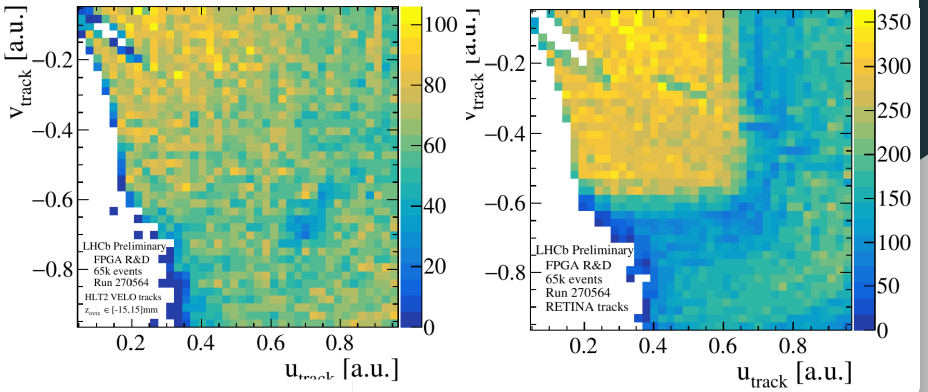


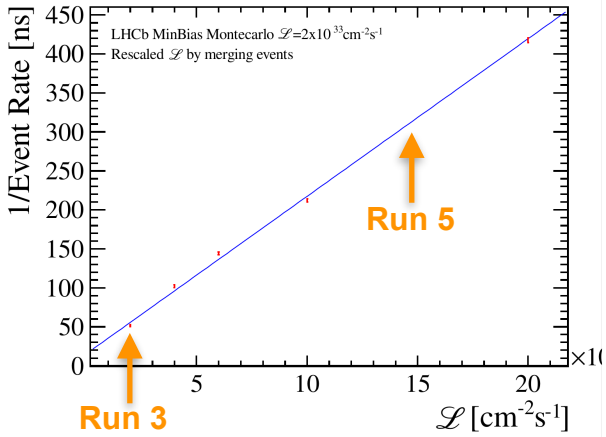


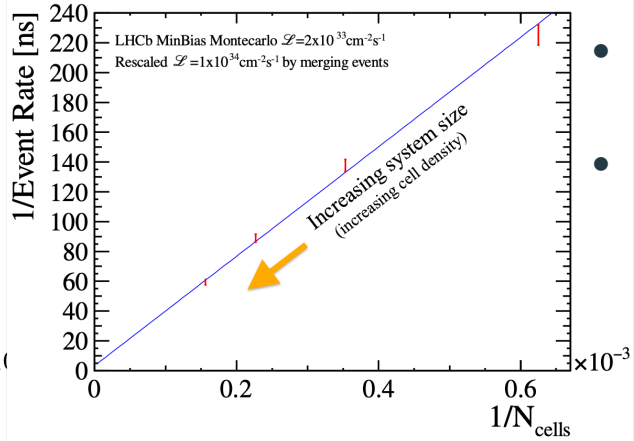
- Emulate higher luminosities by event overlapping
- Performance LINEAR with occupancy and size, up to very high lumi

Results on live LHCb data

- Currently Running parasitically on real LHCb data during Run 3 physics data taking (at reduced rate)
- Online LHCb alignment constant applied on the fly.
- Tracks distribution from demonstrator (right) very similar to HLT2 output (left).



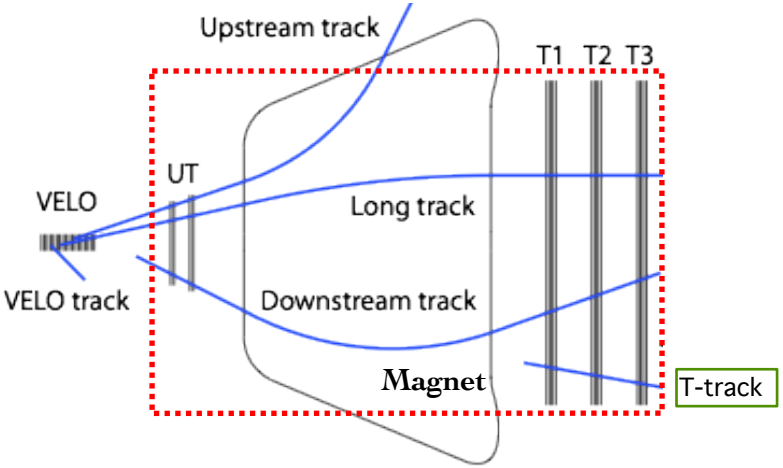


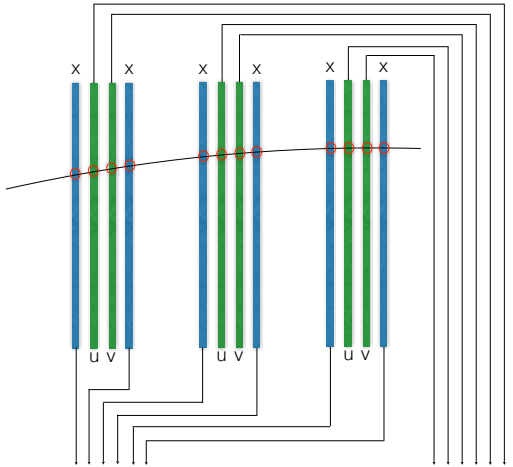


- Emulate higher luminosities by event overlapping
- Performance LINEAR with occupancy and size, up to very high lumi

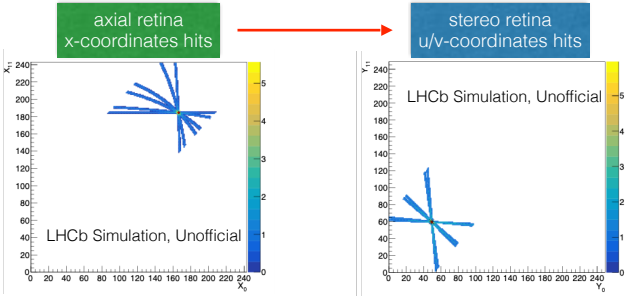
Promising for High-Lumi
⟹ Build a real application

DWT project: reconstruction of SciFi track primitives

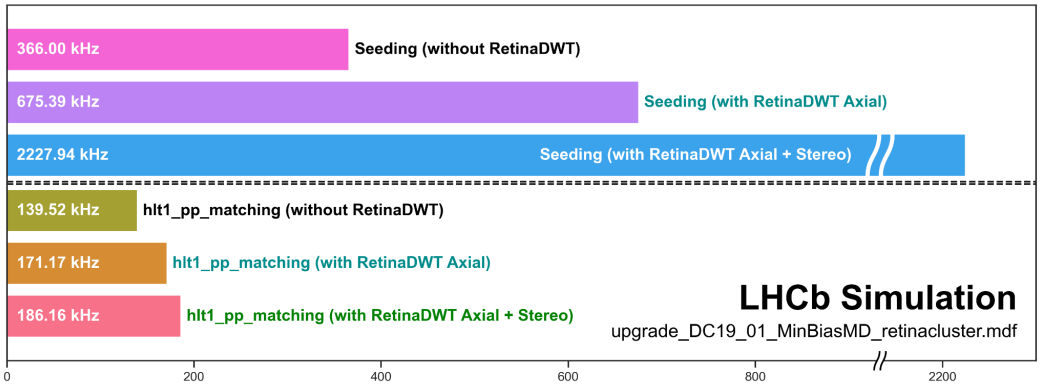




- Track segments in the SciFi detector play important role in LHCb
  - Currently used as 'seeds' for HLT1 tracking sequence
  - Heavy to compute (before GPUs only possible at HLT2)
- Implementation as 2-step retina device (axial layers, then stereo)
- Requires ~100 FPGA boards (new LHCb readout boards)



Throughput tests on the actual HLT1 system

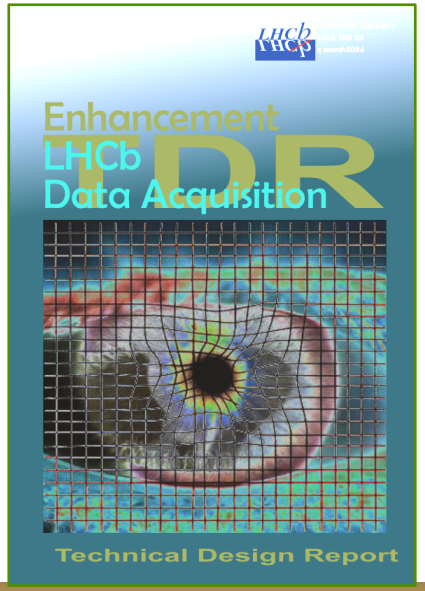


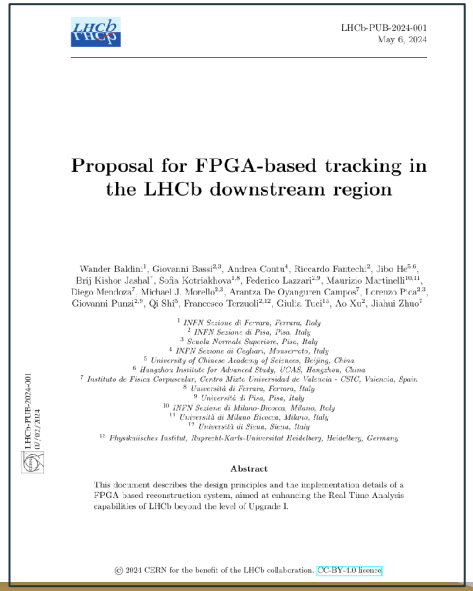
- Effect on Full HLT1 sequence, long tracks matching VELO tracks and T-tracks:
- Execution time:
  - Total: **7.2 µs**
  - Seeding: **1.5 µs**
- Replacing seeding with *primitives* decoding and refitting.
  - Total: **5.4 µs**
  - *Primitives* decoding and refitting: **0.06 µs**
- Overall HLT1 throughput **increased by 33%.** Makes room for **further HLT1 functionality**.

Plan to implement for next LHCb run (Run 4)

For more information

- [DAQ enhancement TDR] submitted by LHCb to the LHCC (not yet public)
  - Contains Run 4 proposal for Both the DWT and the new FPGA board of LHCb
- Detailed technical description already available as a [LHCb public note]




LHCb-PUB-2024-001
May 6, 2024

Proposal for FPGA-based tracking in the LHCb downstream region

Wander Baldini[1], Giovanni Bassi[2,3], Andrea Contu[4], Riccardo Fantechi[2], Jibo He[5,6], Brij Kishor Jashal[7], Sofia Kotriakhova[1,8], Federico Lazzari[2,9], Maurizio Martinelli[10,11], Diego Mendoza[7], Michael J. Morello[2,3], Arantza De Oyanguren Campos[7], Lorenzo Pica[2,3], Giovanni Punzi[2,9], Qi Shi[5], Francesco Terzuoli[2,12], Giulia Tuci[13], Ao Xu[5], Jiahui Zhuo[7]

[1] *INFN Sezione di Ferrara, Ferrara, Italy*
[2] *INFN Sezione di Pisa, Pisa, Italy*
[3] *Scuola Normale Superiore, Pisa, Italy*
[4] *INFN Sezione di Cagliari, Monserrato, Italy*
[5] *University of Chinese Academy of Sciences, Beijing, China*
[6] *Hangzhou Institute for Advanced Study, UCAS, Hangzhou, China*
[7] *Institute de Fisica Corpuscular, Centro Mixto Universidad de Valencia - CSIC, Valencia, Spain*
[8] *Università di Ferrara, Ferrara, Italy*
[9] *Università di Pisa, Pisa, Italy*
[10] *INFN Sezione di Milano-Bicocca, Milano, Italy*
[11] *Università di Milano-Bicocca, Milano, Italy*
[12] *Università di Siena, Siena, Italy*
[13] *Physikalisches Institut, Ruprecht-Karls-Universität Heidelberg, Heidelberg, Germany*

Abstract

This document describes the design principles and the implementation details of a FPGA based reconstruction system, aimed at enhancing the Real Time Analysis capabilities of LHCb beyond the level of Upgrade I.

List of Authors

Wander Baldini[1], Giovanni Bassi[2,3], Andrea Contu[4], Riccardo Fantechi[2], Jibo He[5,6], Brij Kishor Jashal[7], Sofia Kotriakhova[1,8], Federico Lazzari[2,9], Maurizio Martinelli[10,11], Diego Mendoza[7], Michael J. Morello[2,3], Arantza De Oyanguren Campos[7], Lorenzo Pica[2,3], Giovanni Punzi[2,9], Qi Shi[5], Francesco Terzuoli[2,12], Giulia Tuci[13], Ao Xu[2], Jiahui Zhuo[7]

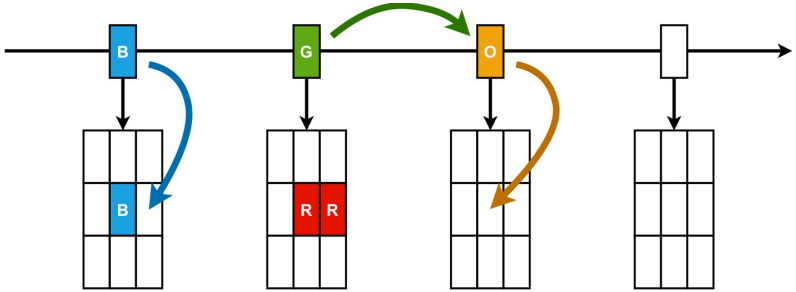
[1] *INFN Sezione di Ferrara, Ferrara, Italy*
[2] *INFN Sezione di Pisa, Pisa, Italy*
[3] *Scuola Normale Superiore, Pisa, Italy*
[4] *INFN Sezione di Cagliari, Monserrato, Italy*
[5] *University of Chinese Academy of Sciences, Beijing, China*
[6] *Hangzhou Institute for Advanced Study, UCAS, Hangzhou, China*
[7] *Instituto de Fisica Corpuscular, Centro Mixto Universidad de Valencia - CSIC, Valencia, Spain*
[8] *Università di Ferrara, Ferrara, Italy*
[9] *Università di Pisa, Pisa, Italy*
[10] *INFN Sezione di Milano-Bicocca, Milano, Italy*
[11] *Università di Milano Bicocca, Milano, Italy*
[12] *Università di Siena, Siena, Italy*
[13] *Physikalisches Institut, Ruprecht-Karls-Universitat Heidelberg, Heidelberg, Germany*

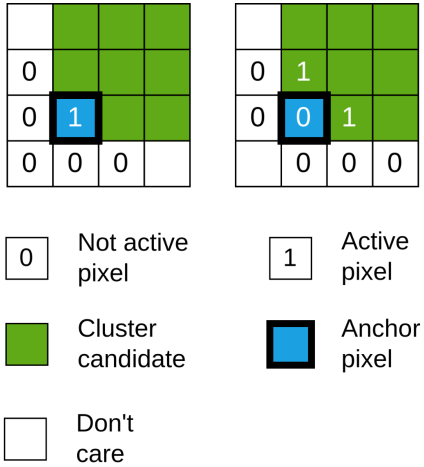
Backup

A 'complex' primitive: hits in the VELO pixel detector

- Hits in the VELO detector of LHCb appear as clusters of pixels [see dedicated VELO talk]
- Firmware deployed in Run3 in FPGA readout boards to reconstruct clusters on the fly (Arria 10)
  - Original plan was to do this during HLT1 reconstruction
- Pixels read out as 2*4 arrays (SuperPixels, SP). Clusters found by unpacking them into active matrices, where each pixel actively checks if it belongs to a pattern. Centroid evaluated by LUT.
- Fast solution, but unmanageable to cover the 40M pixels of the VELO
- Solution: dynamically allocate small matrixes where active pixels are found.
  Input data travel along a chain of empty matrices:
  - When a SP hits an empty matrix, it allocates it to its position
  - It a SP hits a matrix it belongs to, it fills the matrix at the right position
  - Cluster finding happens in parallel in all matrices

-> allows to process data continuously, yielding a **throughput of 10^{11} hits/s** [IEEE TNS 70, 6 (2023)]



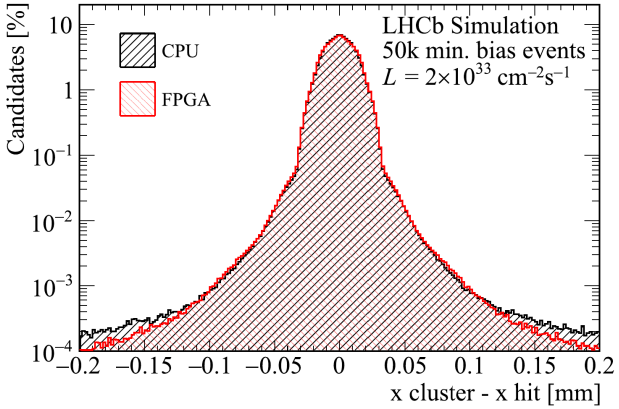


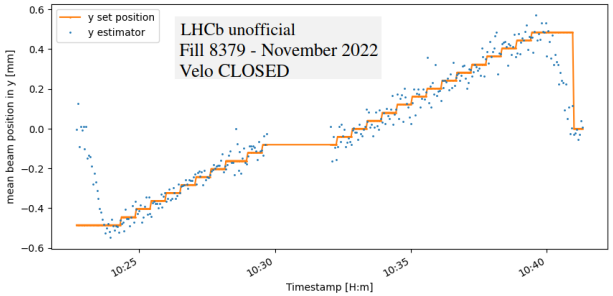
Benefits of embedded Cluster finding

- Quality of real-time cluster reconstruction as good as CPU algorithm
  - Raw pixel information **dropped** and replaced by hit positions during readout (saves 15% of b/w)
- FPGA implementation saves 12% of HLT1 computing power, and uses 1/50th of the electrical power [IEEE TNS 70, 6 (2023)]

-> Now established as the default method at LHCb.

- Side benefits: real-time availability of 10^{11} hits/s **in accessible way** enables further applications
- Example: measurement of beam position vs time exploiting cylindrical symmetry of hit distribution
  - Large rate require no track reconstruction ('trackless')
  - O(μm) precision, continuous monitoring



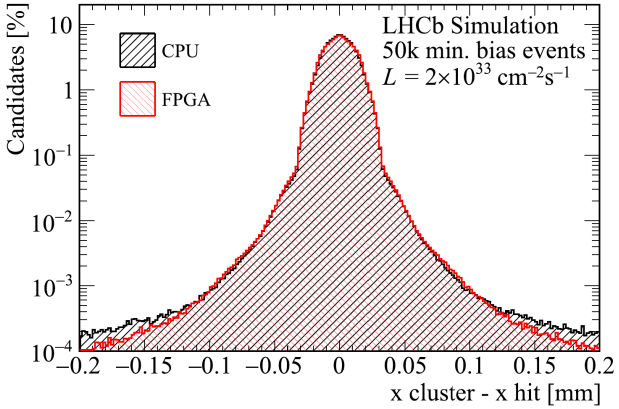


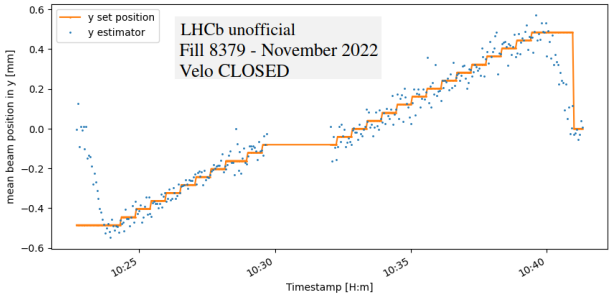
Benefits of embedded Cluster finding

- Quality of real-time cluster reconstruction as good as CPU algorithm
  - Raw pixel information **dropped** and replaced by hit positions during readout (saves 15% of b/w)
- FPGA implementation saves 12% of HLT1 computing power, and uses 1/50th of the electrical power [IEEE TNS 70, 6 (2023)]

-> Now established as the default method at LHCb.

- Side benefits: real-time availability of 10^{11} hits/s **in accessible way** enables further applications
- Example: measurement of beam position vs time exploiting cylindrical symmetry of hit distribution
  - Large rate require no track reconstruction ('trackless')
  - O(μm) precision, continuous monitoring

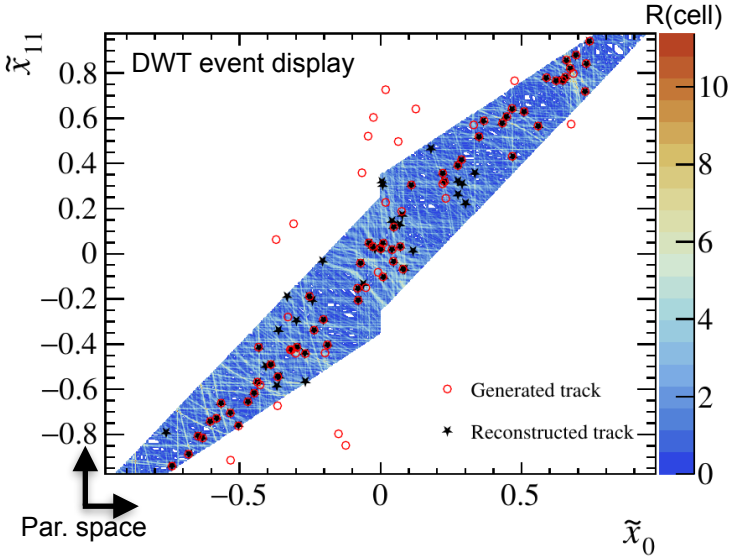




'Local' application: all required data accessible in a single FPGA
Next we discuss a more complex solution involving multiple FPGAs

Emulation study of DWT performance

- Studies performed with realistic device Emulator, running on official LHCb MC productions.
- Tracking quality of primitives is at a level close to HLT1 - will be refined to tracks in HLT1 processing
  - Efficiencies ~90% , Ghost rates ~15%



Track type	MinBias	$D^0 \to K^0_S\pi^+\pi^-$	$B^0_s \to \phi\phi$
Long, $p > 3\,\mathrm{GeV}/c$	85 (86)	83 (84)	84 (85)
Long, $p > 5\,\mathrm{GeV}/c$	90 (91)	89 (90)	89 (89)
Long from B not $e^\pm$, $p > 3\,\mathrm{GeV}/c$	-	-	88 (87)
Long from B not $e^\pm$, $p > 5\,\mathrm{GeV}/c$	-	-	90 (90)
Down, $p > 3\,\mathrm{GeV}/c$	84 (85)	83 (84)	83 (84)
Down, $p > 5\,\mathrm{GeV}/c$	89 (91)	88 (89)	88 (89)
Down from strange not $e^\pm$, $p > 3\,\mathrm{GeV}/c$	-	83 (83)	-
Down from strange not $e^\pm$, $p > 5\,\mathrm{GeV}/c$	-	88 (88)	-
Down from strange not long not $e^\pm$, $p > 3\,\mathrm{GeV}/c$	-	83 (83)	-
Down from strange not long not $e^\pm$, $p > 5\,\mathrm{GeV}/c$	-	88 (89)	-
ghost rate	16 (10)	17 (12)	17 (13)
ghost rate / (1 - ghost rate)	0.2 (0.1)	0.2 (0.1)	0.2 (0.1)

DWT tracking performance

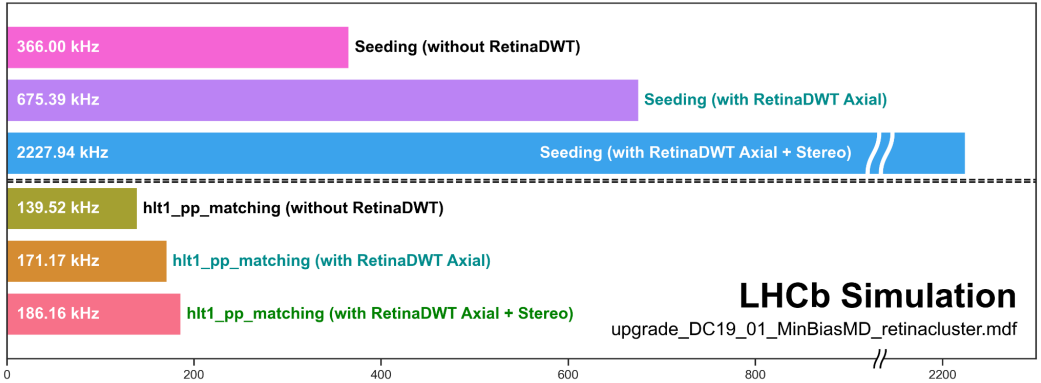
- Fiducial requirements: $p_T > 200$ MeV/c; $2 < \eta < 5$.

Event-averaged values in brackets

Track type	MinBias	$D^0 \to K^0_S \pi^+ \pi^-$	$B^0_s \to \phi\phi$
Long, $p > 3\,\mathrm{GeV}/c$	85 (86)	83 (84)	84 (85)
Long, $p > 5\,\mathrm{GeV}/c$	90 (91)	89 (90)	89 (89)
Long from B not $e^\pm$, $p > 3\,\mathrm{GeV}/c$	-	-	88 (87)
Long from B not $e^\pm$, $p > 5\,\mathrm{GeV}/c$	-	-	90 (90)
Down, $p > 3\,\mathrm{GeV}/c$	84 (85)	83 (84)	83 (84)
Down, $p > 5\,\mathrm{GeV}/c$	89 (91)	88 (89)	88 (89)
Down from strange not $e^\pm$, $p > 3\,\mathrm{GeV}/c$	-	83 (83)	-
Down from strange not $e^\pm$, $p > 5\,\mathrm{GeV}/c$	-	88 (88)	-
Down from strange not long not $e^\pm$, $p > 3\,\mathrm{GeV}/c$	-	83 (83)	-
Down from strange not long not $e^\pm$, $p > 5\,\mathrm{GeV}/c$	-	88 (89)	-
ghost rate	16 (10)	17 (12)	17 (13)
ghost rate / (1 - ghost rate)	0.2 (0.1)	0.2 (0.1)	0.2 (0.1)

- Performance similar to current HLT1 already at the primitive level.

Throughput tests on the actual HLT1 system



- **T-track seeding** (computational heavy) **x6 speedup with primitive-based** reconstruction
- Effect on Full HLT1 sequence, long tracks matching VELO tracks and T-tracks:
- Execution time:
  - Total: **7.2 µs**
  - Seeding: **1.5 µs**
- Replacing seeding with *primitives* decoding and refitting.
  - Total: **5.4 µs**
  - *Primitives* decoding and refitting: **0.06 µs**
- Overall HLT1 throughput **increased by 33%.** Makes room for **further HLT1 functionalities**.

Plan to implement for next LHCb run (Run 4)