FPGA-based real-time cluster finding for the LHCb silicon pixel detector

Giovanni Bassi on behalf of the LHCb RTA project
giovanni.bassi@cern.ch

SCUOLA NORMALE SUPERIORE

INFN

LHCb

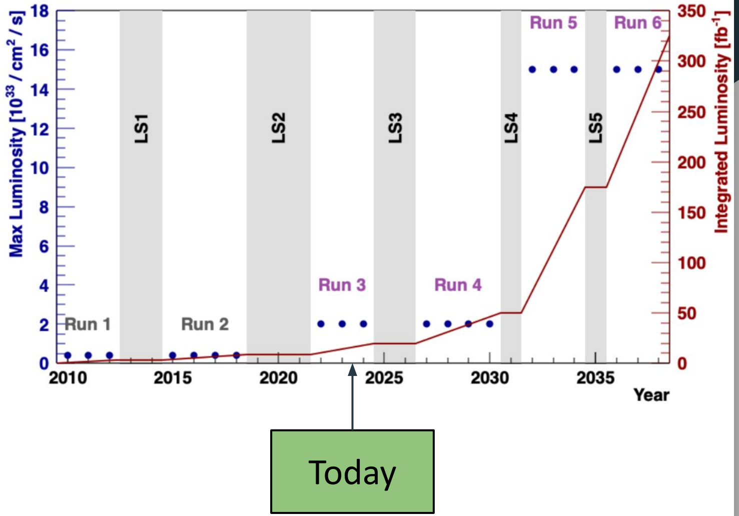
LHCb RTA

26th International Conference on Computing in High-Energy and Nuclear Physics (CHEP2023), 8th - 12th May 2023

CERN-LHCC-2021-012

Background

- LHCb is at the leading edge of high-precision heavy-flavor measurements
- Uncertainties on many key observables are still dominated by statistics ⇒ **bigger data samples, increasing luminosity**
- Need to **process data at low data acquisition levels** to limit data flow, while reducing computing resources needs and power consumption
- Heterogeneous computing is one of the most promising solutions and **FPGA accelerators** are well suited to address, in a high parallel way, heavy repetitive tasks
- Grouping contiguous pixels in single hits (**clustering**) is both a time demanding (2D pixel geometry) and a repetitive task

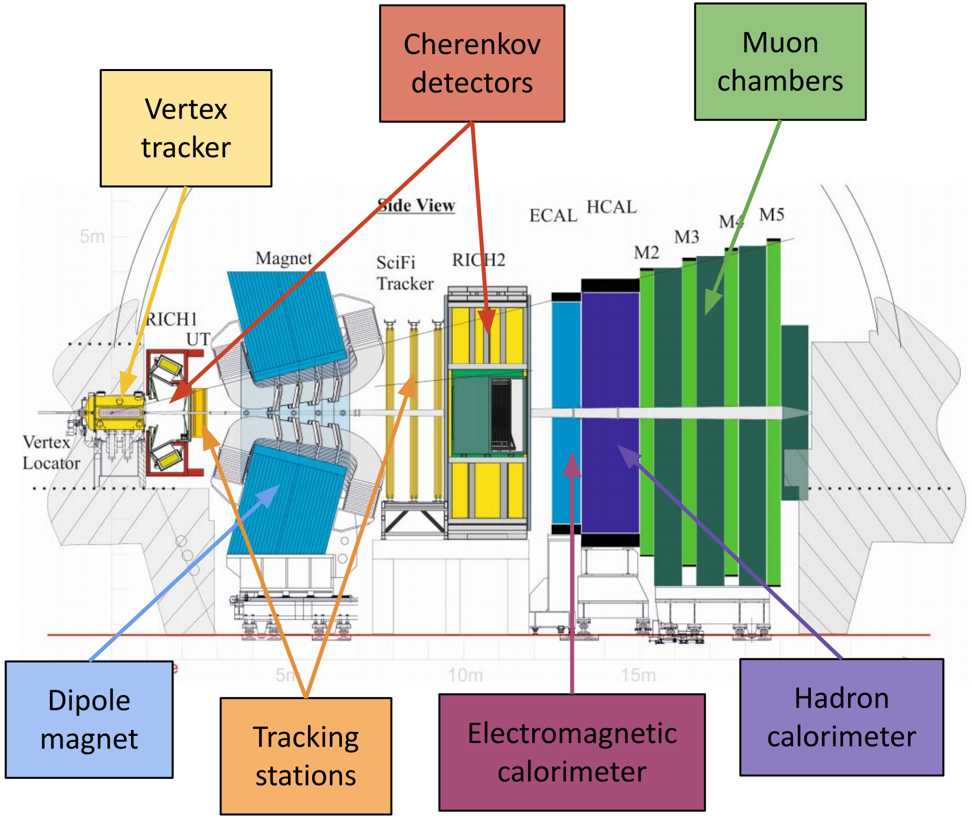


- We developed a FPGA-friendly clustering algorithm, based on the Retina project [CHEP2023, May 9, 14:15, track 2], to tackle 2D clustering during early DAQ stages, while keeping **the same tracking performance** wrt CPU clustering

LHCB-TDR-12

LHCb Upgrade

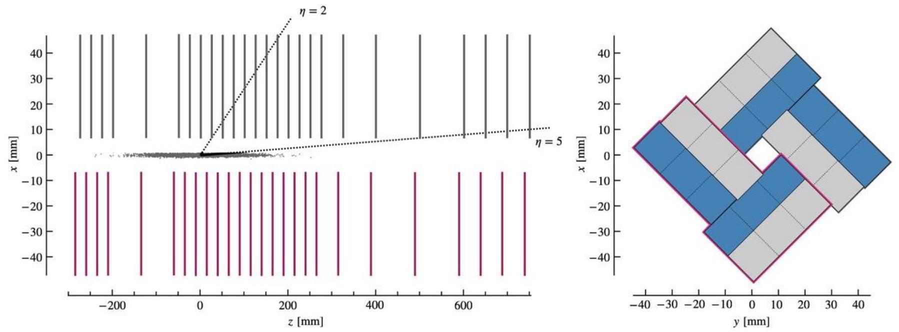
- LHCb is a single arm spectrometer, designed for precision studies of b- and c-hadrons
- LHCb has just completed a major upgrade
- Constraints:
    - $L = 2\times10^{33}$ cm^{-2}s^{-1} (x5 wrt Run 2)
    - 7.6 vertices (1.1 in Run 2)
- Design choices:
    - upgrade of the majority of sub-detectors
    - readout system dealing with a **30 MHz data processing rate** (average LHC bunch crossing rate)
    - **32 Tb/s data flow** from the detector to the High Level Trigger farm (HLT1-GPU + HLT2-CPU)



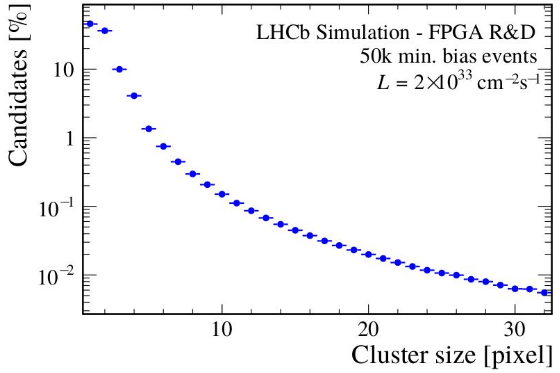
PoS VERTEX 2018

The VELO detector of LHCb

- The clustering algorithm has been tailored for the LHCb Vertex Locator (**VELO**):
    - 26 layers each made of 2 modules
    - Each module consists of 4 sensors
    - 1 DAQ card per module
    - 41 M pixels in total

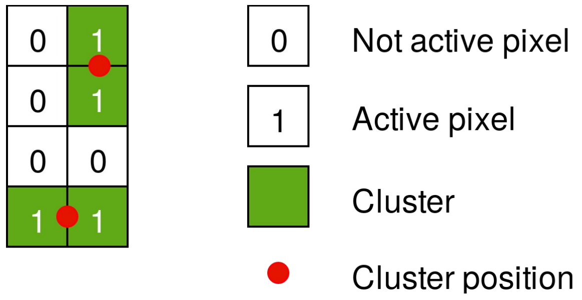


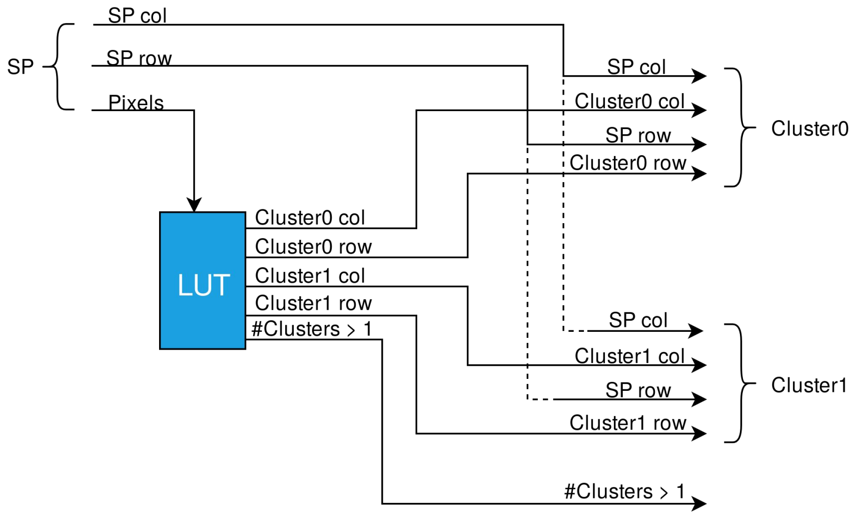
- Pixels are read in groups of 2x4 pixels (**SuperPixels**)
- VELO clusters are typically made of few pixels (1-4)
- The first step of the cluster reconstruction is to flag isolated SuperPixels (isolated = none of the 8 neighbors SPs have any active pixel)



Algorithm overview

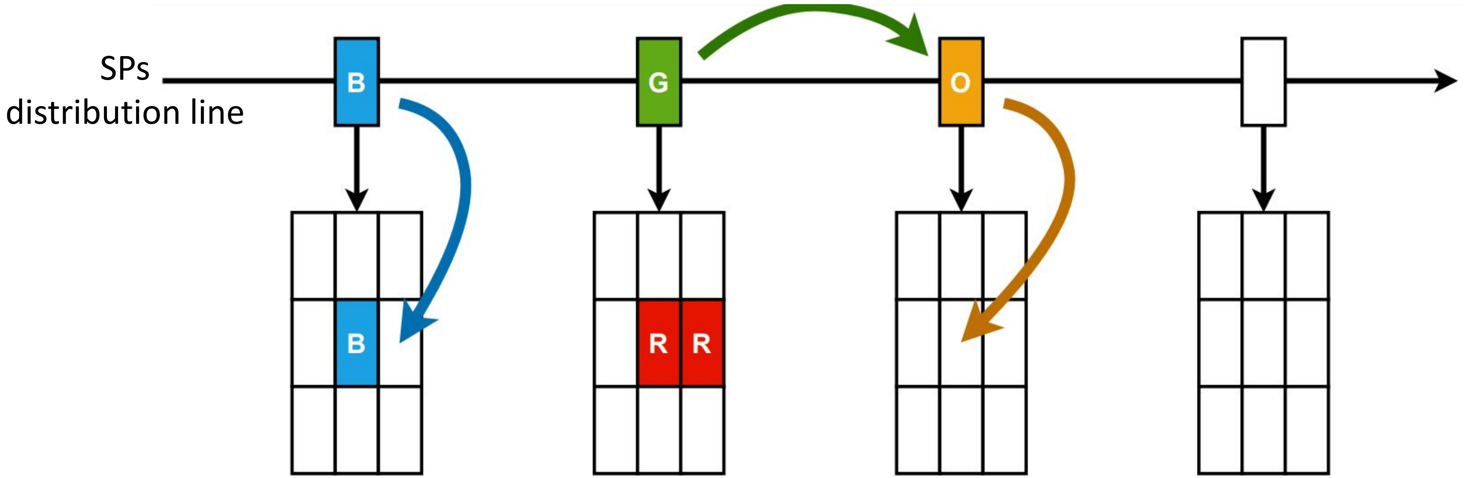
- Isolated SPs are resolved with a Look Up Table (LUT) allowing for an extremely fast processing of isolated SPs, with a very limited amount of logic resources within the FPGA
- LUT connects each of the 256 (2^8) possibile pixel configurations inside a SP to the center of mass of the cluster/s (if two clusters are generated)





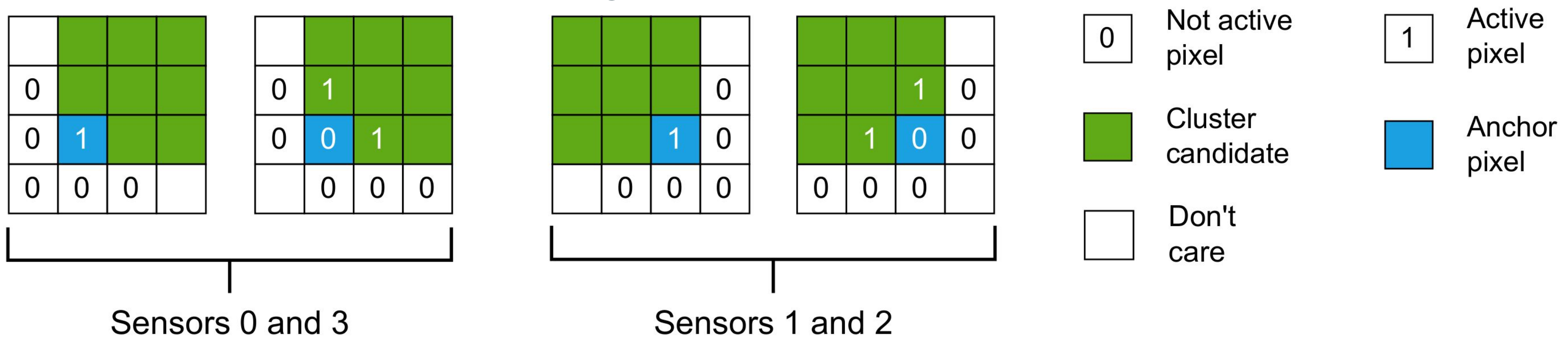
Algorithm overview

- SPs with neighbors fill a set of matrices, 3x3 SPs each (6x12 pixels)
- First SP filling a matrix determines → position of the matrix in the detector
  → set of coordinates of SPs that can fill the matrix
- If a SP belongs to a matrix it fills it, otherwise it moves forward, checking the next matrix or filling a blank one in the center



Algorithm overview

- At the end of each event, in a fully parallel way, each pixel checks if it belongs to one of the following patterns, if so a cluster candidate is identified
- Each cluster candidate is resolved using a LUT

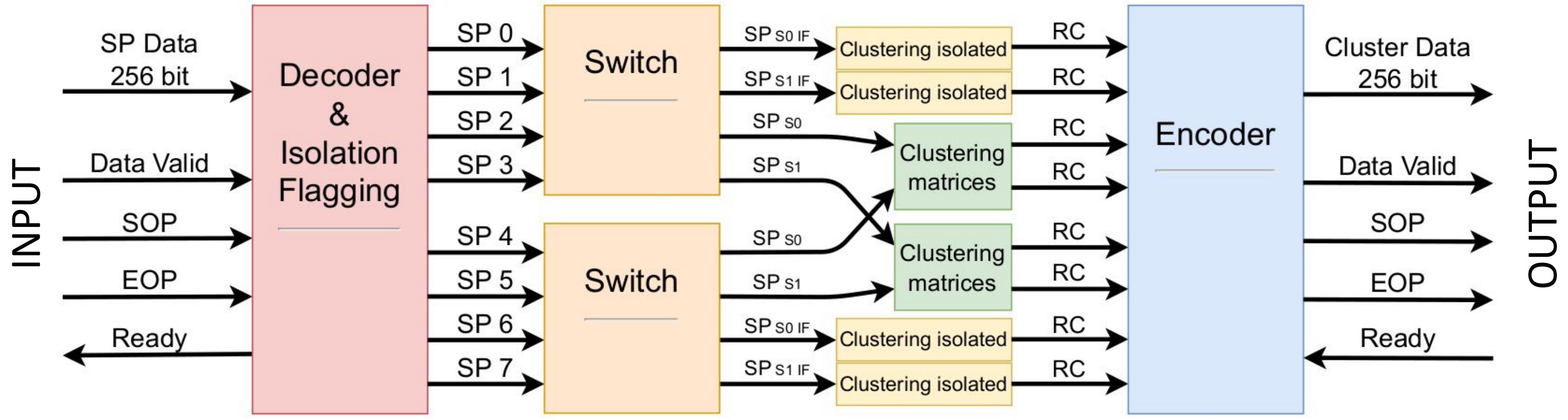


- Algorithm parameters:
  - Matrix shape and size ⇐ average number of neighbor SPs and their arrangement
  - Number of matrices ⇐ distribution of total number of not isolated SPs per event
  - Cluster maximum dimension (3x3 pixels) ⇐ distribution of cluster sizes

Open Access Repository

Firmware overview

Having defined the algorithm behavior, the corresponding firmware has been developed in VHDL



Spits 256-bit input bus into 8 32-bit busses and computes isolation

Arranges SPs by sensor and isolation flag

Reconstructs clusters from isolated SPs

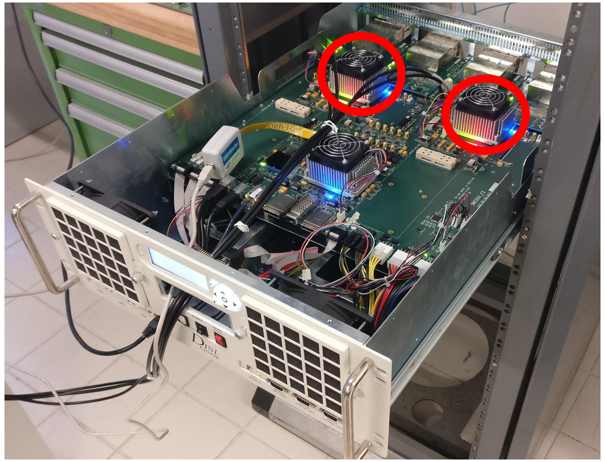
Reconstructs clusters from non isolated SPs

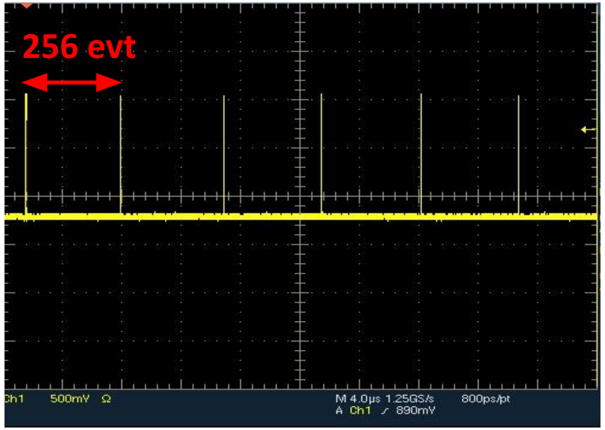
Encodes 8 32-bit busses into 256-bit output bus

Prototyping

- The first functioning prototype of the clustering firmware was developed and tested within the LHCb-Pisa INFN laboratory
- To run as a real-time algorithm, the firmware has to sustain the LHC average bunch crossing rate of 30 MHz
- Firmware tested on a prototyping board to measure:
  - Amount of logic required on the chip
  - Average frequency of events clusterized (**throughput**)
- We measured:
  - **26% logic occupancy** of the chip
  - Average throughput of **38.9 MHz**

- The low amount of logic required and the achieved throughput (>30 MHz) ease the full integration of the clustering firmware inside existing DAQ boards, without extra costs





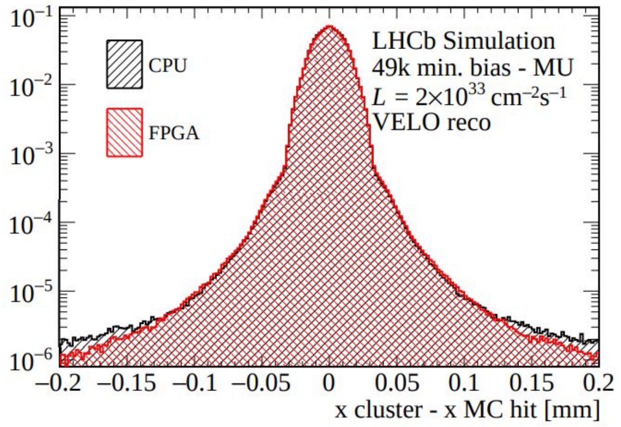
10.1109/TNS.2023.3273600

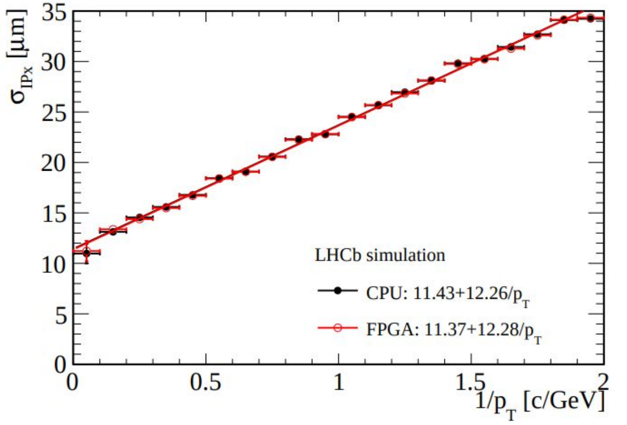
Physics performance

- Moving from a full-fledged software implementation of the VELO clustering to a FPGA-based one required a careful evaluation of possible impacts on physics performances in terms of
  - Cluster efficiency → find hit on detector
  - Cluster residual → match hit position
  - Track efficiency → find track
  - Track resolution → match track parameters

Track type	Quantity	CPU clusters [%]	FPGA clusters [%]
All VELO tracks	efficiency	98.254 ± 0.007	98.254 ± 0.007
	clone	1.231 ± 0.006	1.234 ± 0.006
Long tracks	efficiency	99.252 ± 0.006	99.252 ± 0.006
	clone	0.806 ± 0.006	0.806 ± 0.006
	ghost	0.848 ± 0.003	0.928 ± 0.003

- **FPGA algorithm tracking performance is nearly indistinguishable from CPU/GPU clustering**

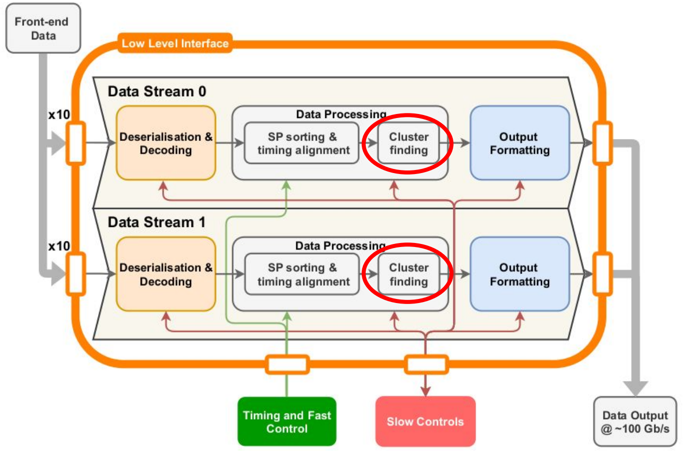




Integration

- Having verified that the clustering has a high enough throughput, while satisfying the very demanding LHCb physics requirements, **the clustering firmware has been fully integrated within the VELO DAQ firmware**
- LHCb DAQ cards are equipped with an Intel Arria 10 FPGA:
  - Detector data received on optical links
  - SP data are decoded, time aligned and sent to the clustering block
  - Clusters are sent to the host server via PCIe
- Clustering requires:
  - **31% of the logic elements**
  - **11% of the M20K** memory blocks
  - **350 MHz** clock
- The entire VELO firmware, including clustering, requires 73% of logic elements and 71% of M20K memory blocks

JINST 11 P02013 (2016)



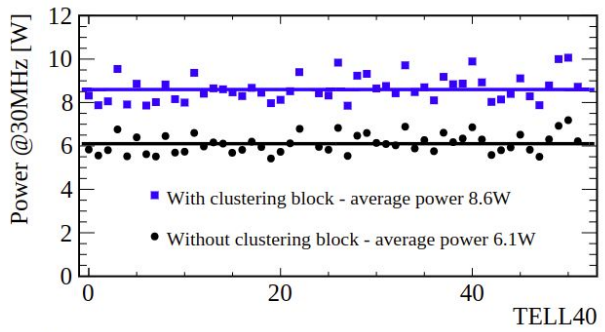
IEEE TNS 68 (2021) 2472

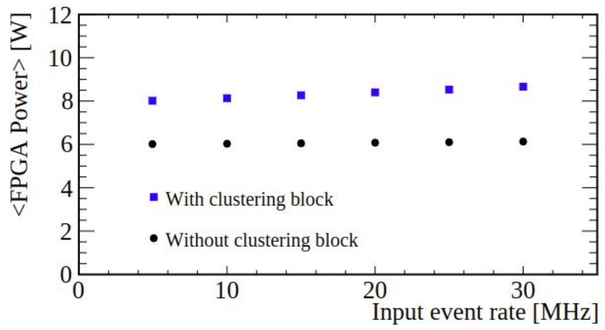
10.1109/TNS.2023.3273600

Throughput, bandwidth & power consumption

- Moving VELO clustering from the HLT1 sequence (GPUs) to early data processing (FPGAs) allows HLT1 to accept a **11% higher rate of events**, since clusters are already available
- Moreover, outputting clusters instead of SPs leads to an additional benefit of a **14% bandwidth reduction**, as reconstructed clusters are less than input SPs

- With the addition of cluster reconstruction, VELO readout cards need to perform more operations. We measured the additional FPGA power consumption to be 2.5W, comparing SP and cluster firmwares (+130W on all 52 VELO DAQ cards)
- As a comparison, performing clustering on GPUs would have required roughly 6kW ⇒ **clustering on FPGAs requires O(50x) less electrical power wrt the GPU implementation**





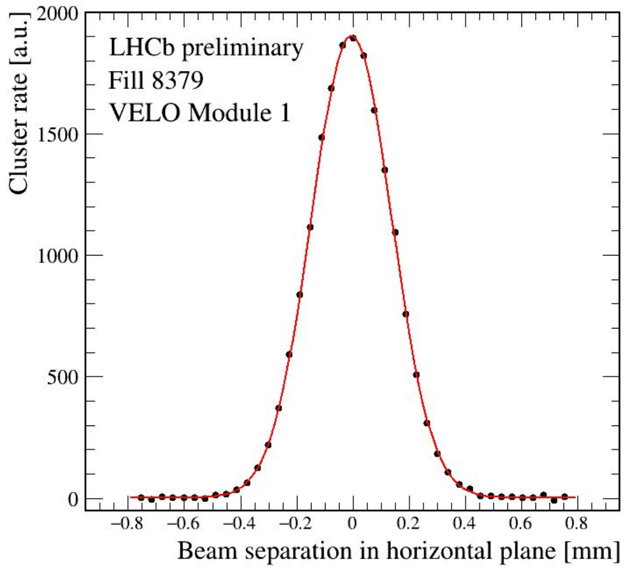
CERN-THESIS-2022-231

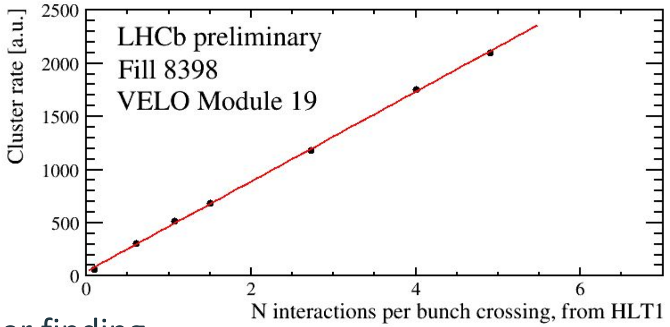
New opportunities

- With cluster reconstruction occurring inside VELO DAQ cards, it becomes possible to perform beam parameter measurements, such as:
    - Luminosity
    - Spillover
    - Beam position

- These parameters can be measured at a very high rate in the firmware, and accessed via slow control, even when HLT1 is not running

- **Luminosity counters have been implemented, calibrated and tested during luminosity and Van der Meer scans**





Summary

- Despite being a conceptually simple task, clustering requires a non-negligible amount of the time needed for the entire HLT1 sequence and consumes a non-negligible amount of power
- Being simple and highly parallelizable makes clustering an ideal candidate to be moved from HLT1 to a preprocessing stage, with benefits for the entire data acquisition chain
- We have developed, implemented and commissioned, for the first time, a 2D FPGA-based clustering algorithm that processes events from VELO-pixel sensors in real time, at the unprecedented speed of 30 MHz
- Given the limited amount of logic (31%) and memory (11%) required, the firmware has been fully integrated in the existing DAQ readout cards, without extra costs, leading to
    - 11% increase in HLT1 event rate
    - 14% reduction of the required bandwidth
    - O(50x) reduction in power consumption

- **The FPGA-based clustering algorithm has been fully commissioned at LHCb in 2022, and is now the adopted solution for physics data taking for the Run 3**

Reference

- More details about the clustering algorithm and the related firmware can be found in the paper *A FPGA-based architecture for real-time cluster finding in the LHCb silicon pixel detector*

- The paper has been accepted for publication by IEEE Transactions on Nuclear Science and it is available as a preprint on [10.1109/TNS.2023.3273600](https://doi.org/10.1109/TNS.2023.3273600)

IEEE TRANSACTIONS ON NUCLEAR SCIENCE, VOL. XX, NO. XX, XXXX 2023 1

A FPGA-based architecture for real-time cluster finding in the LHCb silicon pixel detector

G. Bassi, L. Giambastiani, K. Hennessy, F. Lazzari, M. J. Morello, T. Pajero, A. Fernandez Prieto, G. Punzi

***Abstract*—This article describes a custom VHDL firmware implementation of a two-dimensional cluster-finder architecture for reconstructing hit positions in the new vertex pixel detector (VELO) that is part of the LHCb Upgrade. This firmware has been deployed to the existing FPGA cards that perform the readout of the VELO, as a further enhancement of the DAQ system, and will run in real time during physics data taking, reconstructing VELO hits coordinates on-the-fly at the LHC collision rate. This pre-processing allows the first level of the software trigger to accept an 11% higher rate of events, as the ready-made hit coordinates accelerate the track reconstruction and consumes significantly less electrical power. It additionally allows the raw pixel data to be dropped at the readout level, thus saving approximately 14% of the DAQ bandwidth. Detailed simulation studies have shown that the use of this real-time cluster finding does not introduce any appreciable degradation in the tracking performance in comparison to a full-fledged software implementation. This work is part of a wider effort aimed at boosting the real-time processing capability of HEP experiments by delegating intensive tasks to dedicated computing accelerators deployed at the earliest stages of the data acquisition chain.**

***Index Terms*—Clustering, Connected Component Labelling, FPGA, LHCb, VHDL**

I. Introduction

THE LHCb experiment has collected data over the past decade, during the Run 1 and Run 2 of the LHC, and recently underwent a major update for the current Run 3. In addition to replacing most of the subdetectors, the front-end electronics and data-acquisition system were completely renewed [1], to read out and process the complete information of the detector at the full LHC beam crossing rate of 40 MHz (30 MHz averaged over the LHC cycle). This change is motivated by the needs of the LHCb physics programme, which requires the collection of low transverse momentum events that need high-level processing to be distinguished from background events [2]. This evolution puts a large computing toll on the new real-time processing system, motivating the deployment of innovative features, with a general trend of increasing customisation, parallelisation, and early data pre-processing. A new trigger system [1], [3] was designed to allow the experiment to collect data effectively at an instantaneous luminosity of 2×10^{33} cm^{-2}s^{-1}, five times higher than during Run 2, corresponding to a bandwidth of about 32 Tb/s. The subsequent event-building stage and software high-level-trigger (HLT) processing lead to a data storage flow of 80 Gb/s.

The triggering process is divided into two main stages, named HLT1 and HLT2. The HLT1 uses an array of GPU servers to perform a faster event reconstruction, with the only purpose of reducing the event rate, while retaining as much signal as possible, to a level acceptable for HLT2. The HLT2, based on an array of CPU servers, performs a complete reconstruction of events with an offline-level quality, that is permanently stored for subsequent analysis. To perform its function effectively, the HLT1 needs to perform a nearly complete event reconstruction. First, it finds track segments in the VErtex LOcator detector (VELO), attaching to them hits from the further tracking stations upstream and downstream of the magnet to obtain complete tracks; then, the positions of the primary vertices of the proton-proton (*pp*) collisions are found, as well as those of displaced vertices that constitute the main signature of heavy-flavour particle decays.

The feasibility of implementing several parts of this sequence in a specialised architecture, using programmable digital electronics co-processors (FPGAs), has been studied with the aim of achieving a faster and cheaper reconstruction, especially in view of future runs, moving parts of it before the event-building stage [4].

In this article, we address the very first step in the HLT1 event reconstruction, that is the search for clusters of active pixels in the VELO [5]. Grouping contiguous pixels in clusters is a conceptually simple but computationally demanding task, due to the two-dimensional (2D) geometry and the large number of pixels of the VELO detector (approximately 40 million). In the preliminary version of the HLT1, designed to run entirely on CPUs, this task alone consumed 17% of the time required by the complete HLT1 reconstruction sequence. We address here this issue and describe an efficient architecture of this functionality, requiring a very modest amount of FPGA resources, while providing the throughput and the performance required for its use within the LHCb DAQ system. The core ideas underlying the design of this architecture are based on studies of an FPGA-based track-finding system, performed within the INFN-RETINA R&D project [4]. The overall structure of our algorithm and its main

Submitted on xx/yy/2023.

G. Bassi and M. J. Morello are with INFN Sezione di Pisa, Pisa, IT and Scuola Normale Superiore, Pisa, Italy (e-mail: giovanni.bassi@cern.ch).

L. Giambastiani was with INFN Sezione di Pisa, Pisa, IT and Università di Pisa, Pisa, IT. Now with Università degli Studi di Padova, Padova, Italy.

K. Hennessy is with the Department of Physics, Liverpool University, Liverpool L69 7ZE, United Kingdom.

F. Lazzari is with INFN Sezione di Pisa, Pisa, IT and Università degli Studi di Siena, Siena, Italy.

T. Pajero was with INFN Sezione di Pisa, Pisa, IT and Scuola Normale Superiore, Pisa, IT. Now with University of Oxford, Oxford OX1 3RH, United Kingdom.

A. Fernandez Prieto is with the Instituto Galego de Física de Altas Enerxías (IGFAE), Universidade de Santiago de Compostela, E-15782 Santiago de Compostela, Spain.

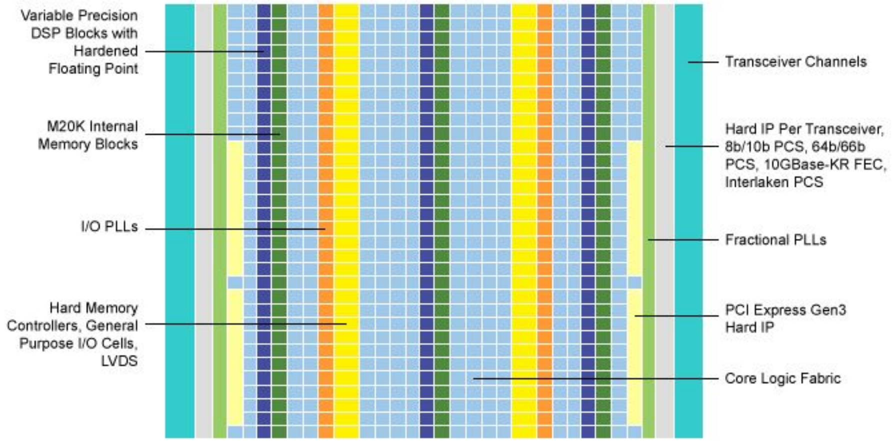
G. Punzi is with INFN Sezione di Pisa, Pisa, IT and Università di Pisa, Pisa, Italy.

BACKUP

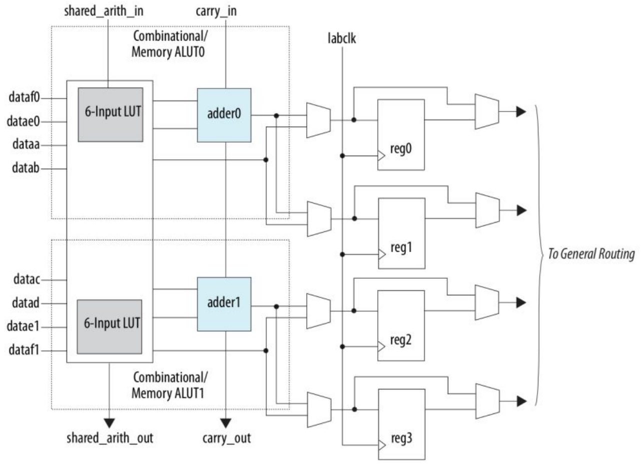
Intel Arria 10 Device Overview

Field Programmable Gate Arrays

- FPGAs are integrated circuits that can be **configured by the user**
- FPGAs contain an **array of programmable logic blocks**, that can be programmed to perform different logic functions
- I/O ports, PLLs, memory blocks and clock distribution lines are integrated within the FPGA
- Configuration is done using a **hardware description language** (HDL)

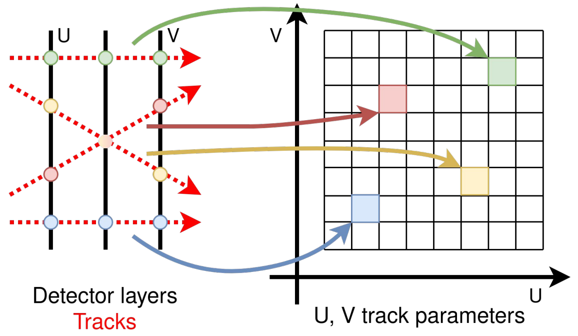


Variable Precision DSP Blocks with Hardened Floating Point
M20K Internal Memory Blocks
I/O PLLs
Hard Memory Controllers, General Purpose I/O Cells, LVDS
Transceiver Channels
Hard IP Per Transceiver, 8b/10b PCS, 64b/66b PCS, 10GBase-KR FEC, Interlaken PCS
Fractional PLLs
PCI Express Gen3 Hard IP
Core Logic Fabric



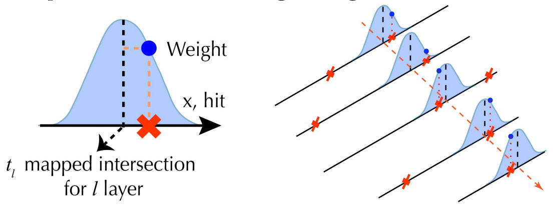
Retina algorithm

INFN-RETINA is an R&D project aimed at developing and implementing a specialized processor allowing the reconstruction of events with hundreds of charged-particle tracks in pixel and silicon strip detectors, at 40 MHz



The **track parameters** space is represented by a matrix of cells. The center of each cell corresponds to a reference track in the detector that intersects the layers in specific spatial points called **receptors**

Step 2: Accumulating weights (each cell)

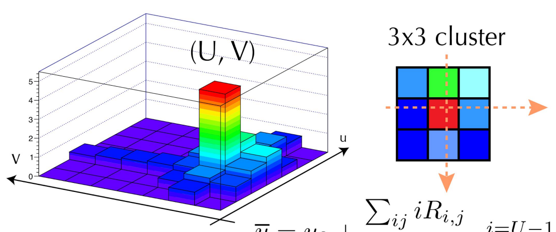


$$R = \sum^{\text{hits}} e^{-\frac{(x_l - t_l)^2}{2\sigma}}$$

R is close to N (# of layers) only if we have a set of hits near the mapped track

Each cell computes a **weighted sum** of hits nearby the reference track. The weights are proportional to the **distance** between the hits and the receptor

Step 3: Find the local maxima and compute centroid



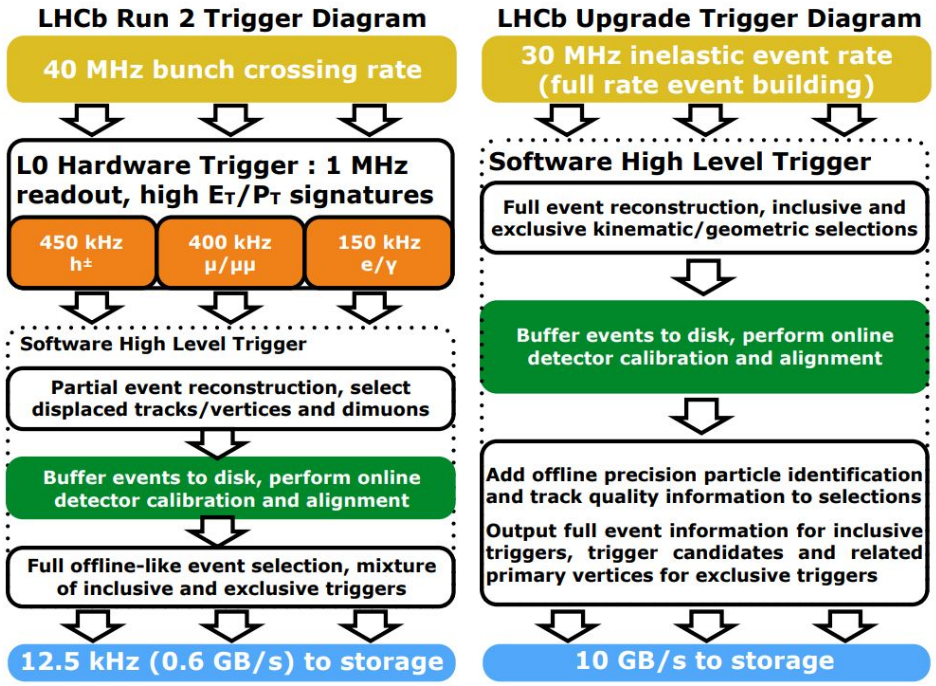
$$\overline{u} = u_0 + \frac{\sum_{ij} iR_{i,j}}{\sum_{ij} R_{i,j}} \qquad \overline{v} = v_0 + \frac{\sum_{ij} jR_{i,j}}{\sum_{ij} R_{i,j}} \qquad \begin{array}{l} i = U-1, U, U+1 \\ j = V-1, V, V+1 \end{array}$$

Cells search **local maxima** in the matrix of weighted sums. **Track parameters** are reconstructed by interpolating the responses of cells nearby the local maxima

LHCB-FIGURE-2020-016

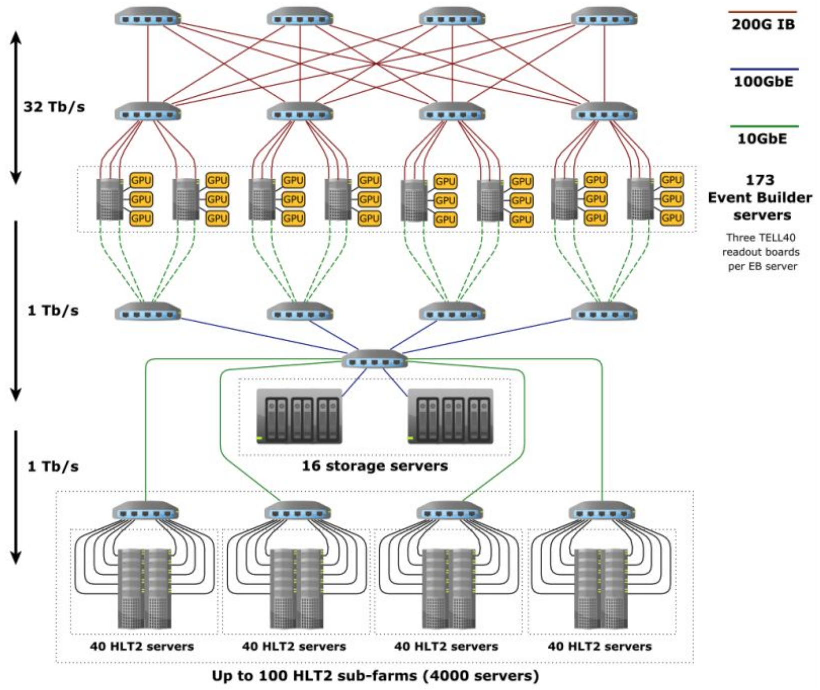
LHCb trigger

- Run 2 trigger
  - hardware Level-0 stage: 40 MHz ➝ 1 MHz
  - HLT1 fast tracking: 1MHz ➝ 100kHz
  - HLT2 full event reconstruction: 100 kHz ➝ 12.5 kHz
- Moving from Run 2 to Run 3 we need to categorise different "signals" ➝ access as much of the event as possible, as early as possible
- Run 3 trigger:
  - Full 30 MHz and x5 pileup
  - HLT1: 30 MHz ➝ 1 MHz
  - HLT2: 10GB/s to permanent storage


LHCb Run 2 Trigger Diagram
40 MHz bunch crossing rate
L0 Hardware Trigger : 1 MHz readout, high E_T/P_T signatures
450 kHz $h^\pm$
400 kHz $\mu/\mu\mu$
150 kHz e/γ
Software High Level Trigger
Partial event reconstruction, select displaced tracks/vertices and dimuons
Buffer events to disk, perform online detector calibration and alignment
Full offline-like event selection, mixture of inclusive and exclusive triggers
12.5 kHz (0.6 GB/s) to storage
LHCb Upgrade Trigger Diagram
30 MHz inelastic event rate (full rate event building)
Software High Level Trigger
Full event reconstruction, inclusive and exclusive kinematic/geometric selections
Buffer events to disk, perform online detector calibration and alignment
Add offline precision particle identification and track quality information to selections
Output full event information for inclusive triggers, trigger candidates and related primary vertices for exclusive triggers
10 GB/s to storage

Event building (EB)

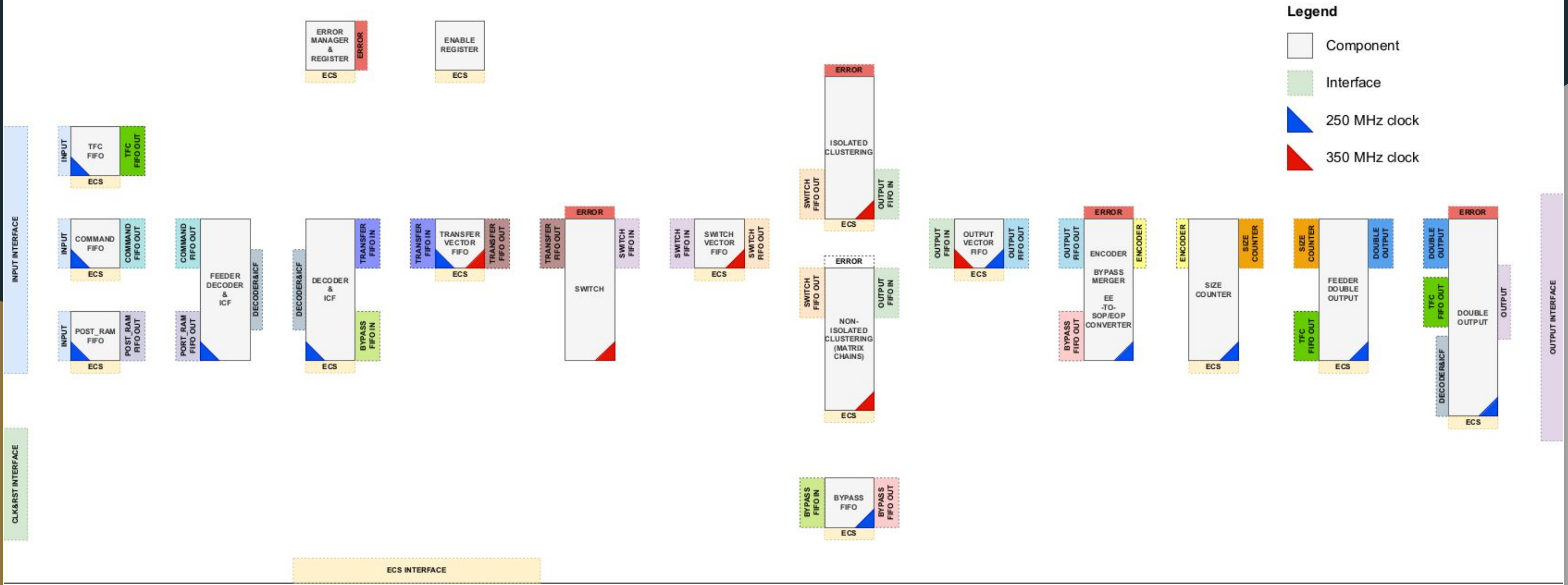
LHCb-DP-2021-003



- Each sub detector sends raw data, asynchronously, to a unique EB node, that receives data through DAQ cards
- EB nodes exchange detector data using an EB network
- All detector data for an event are sent to a specific EB node that builds the event
- GPU cards run HLT1 within the EB servers, reducing the data rate at the output of the EB by a factor of 30-60
- An array of disk servers buffers the HLT1 output data
- A separate server farm runs HLT2

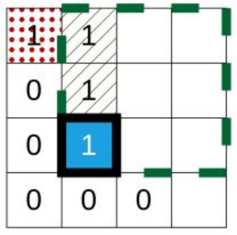
CERN-THESIS-2023-002

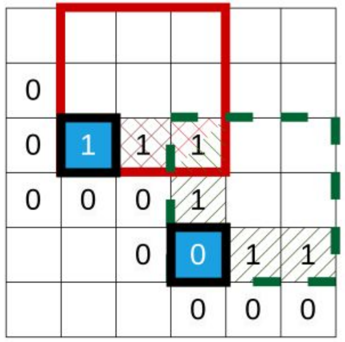
Firmware overview - detailed view

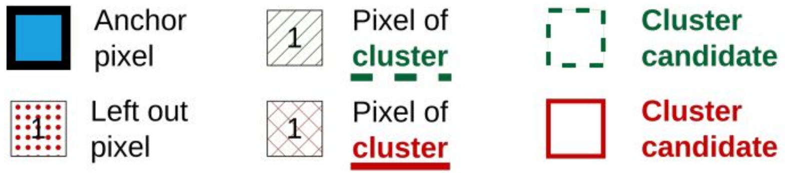


Algorithm peculiar behaviors

- Algorithm parameters:
  - Matrix shape and size ⇐ average number of neighbor SPs and their arrangement
  - Number of matrices ⇐ distribution of total number of not isolated SP per event
  - Cluster maximum dimension (3x3 pixels) ⇐ distribution of cluster sizes
- Examples of peculiar behaviors of the algorithm:


(a)

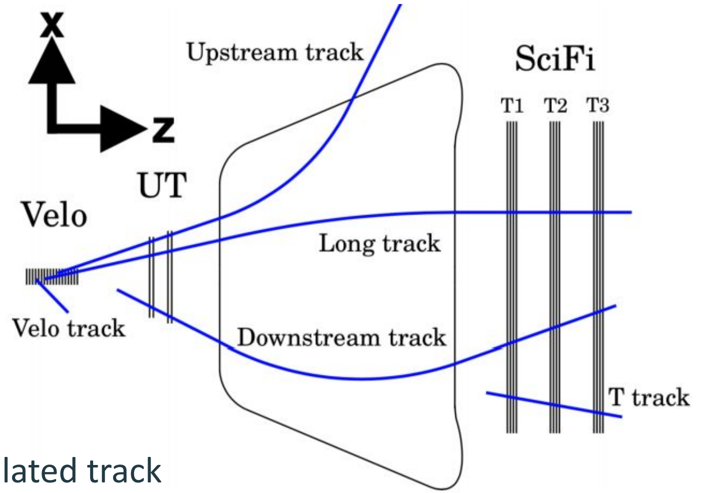

(b)



LHCb-PUB-2021-005

LHCb track classification

Tracking performance is evaluated using LHCb simulated events, comparing the output of the track reconstruction using FPGA and CPU clustering algorithms



LHCb track classification

- **Velo**: track with hits on the VELO
- **Long**: track with hits on the three LHCb trackers (VELO - UT - Scifi)

- **Ghost**: a reconstructed track not associated to a simulated track
- **Clone**: a second reconstructed track associated to the same simulated track