My experience at LHCb: an ultra-fast clustering algorithm

Giovanni Bassi

SNS@CERN - 27/05/2023

My journey: from Trieste to Genève

2013

Bachelor@UniTS
(2013-2016)
ALICE experiment

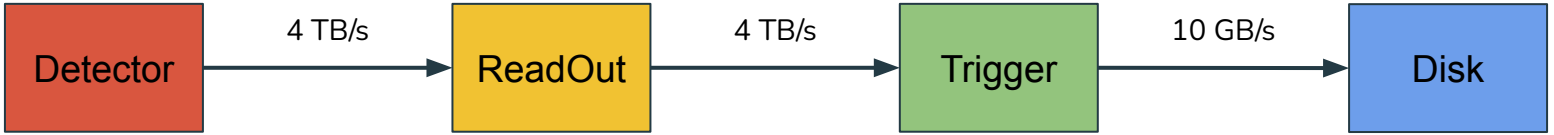
MSc@UniTS
(2016-2018)
Belle II experiment

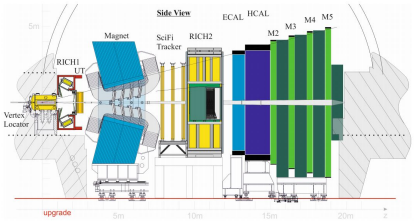
2018

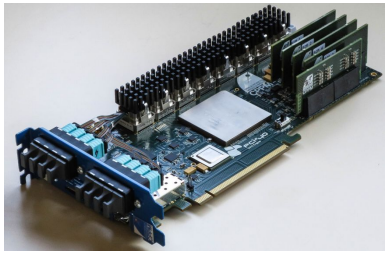
PhD@SNS
(2018-2022)
LHCb experiment

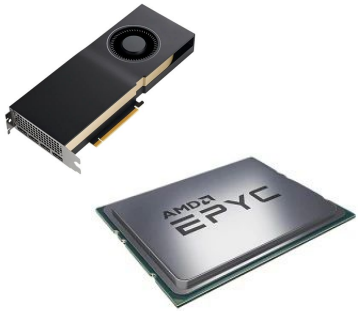
PostDoc@SNS
(2022-ongoing)
LHCb experiment

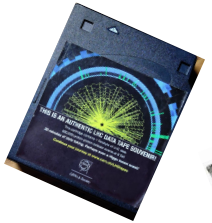
LHCb data path in a nutshell











Prospects in data handling

- Designing and validating a data handling system takes several year ⇒ it is important to define constraints as early as possible
- High Energy Physics experiments are following a common trend towards collecting more and more data to probe the underlying theory with ever-increasing precision
- It becomes important to investigate all the available possibilities in terms of architectures (CPU, GPU, FPGA) and programming models ⇒ **heterogeneous computing**

How SNS and INFN Pisa are involved?

- SNS and INFN Pisa are active contributors to the LHCb experiment for more than 10 years, both in the physics analysis side and in the “hardware” side

- One of the main projects in which the Pisa group has been involved is Retina, a R&D project aimed at developing a specialized processor allowing the reconstruction of events with hundreds of charged-particle tracks using modern, high-speed, high-bandwidth FPGA devices

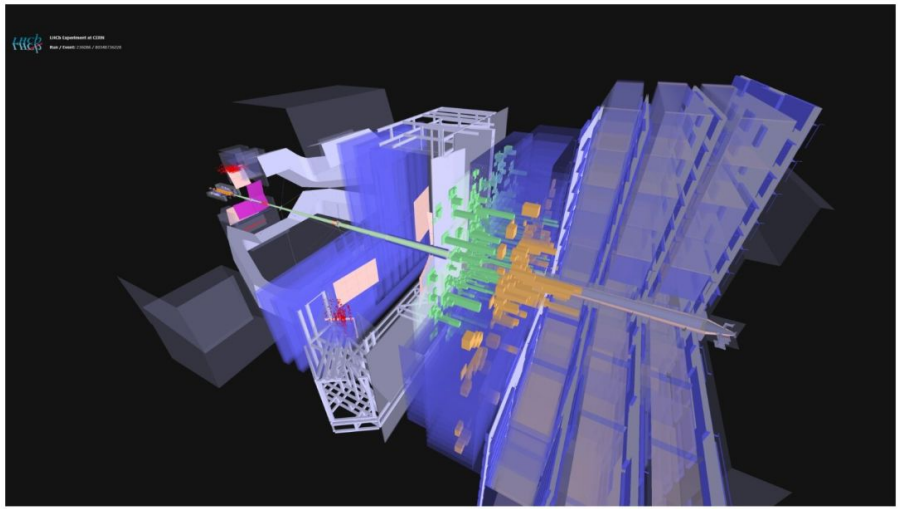
How SNS and INFN Pisa are involved?

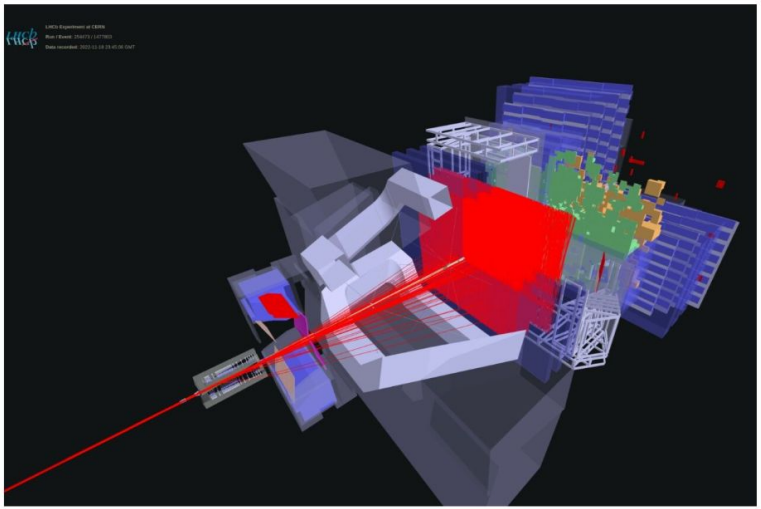
- SNS and INFN Pisa are active contributors to the LHCb experiment for more than 10 years, both in the physics analysis side and in the “hardware” side

- One of the main projects in which the Pisa group has been involved is Retina, a R&D project aimed at developing a specialized processor allowing the reconstruction of events with hundreds of charged-particle tracks using modern, high-speed, high-bandwidth FPGA devices

What’s an event?

Retina, a R&D project aimed at developing a specialized processor allowing the reconstruction of **events** with hundreds of charged-particle tracks using modern, high-speed, high-bandwidth FPGA devices

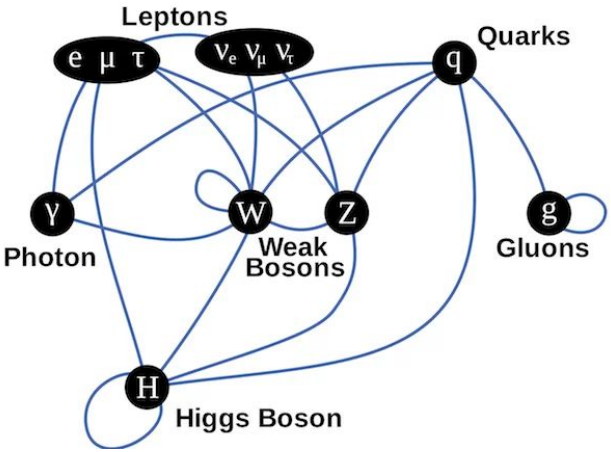




Why charged particles?

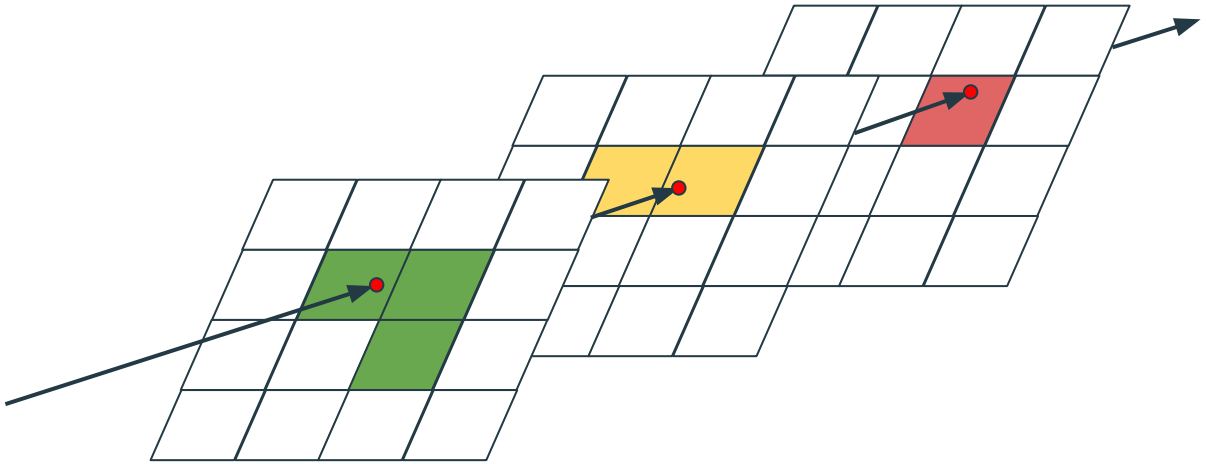
Retina, a R&D project aimed at developing a specialized processor allowing the reconstruction of events with hundreds of **charged-particle** tracks using modern, high-speed, high-bandwidth FPGA devices

- Charged particles interact with matter in many ways (ionization, Cherenkov radiation, ...)
- We can exploit them to study all particle properties (mass, momentum, charge, energy, ...)
- We can study neutral particles also by identifying their charged decay products



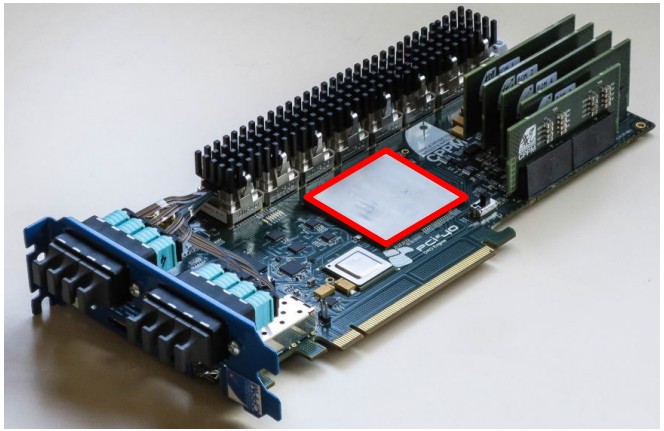
What's a track?

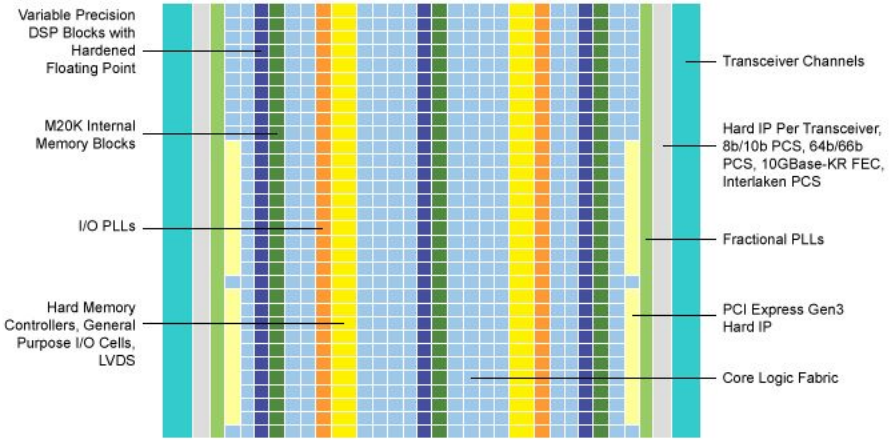
Retina, a R&D project aimed at developing a specialized processor allowing the reconstruction of events with hundreds of charged-particle **tracks** using modern, high-speed, high-bandwidth FPGA devices



What/Why FPGAs?

Retina, a R&D project aimed at developing a specialized processor allowing the reconstruction of events with hundreds of charged-particle tracks using modern, high-speed, high-bandwidth **FPGA devices**





Why high speed and bandwidth?

Retina, a R&D project aimed at developing a specialized processor allowing the reconstruction of events with hundreds of charged-particle tracks using modern, **high-speed, high-bandwidth** FPGA devices

- p-p collisions happens at 40 MHz (25 ns between two collisions) ⇒ we need to be very fast at reconstructing particle tracks
- 4 TB/s of data out of the detector ⇒ we need a system that can swallow all of them

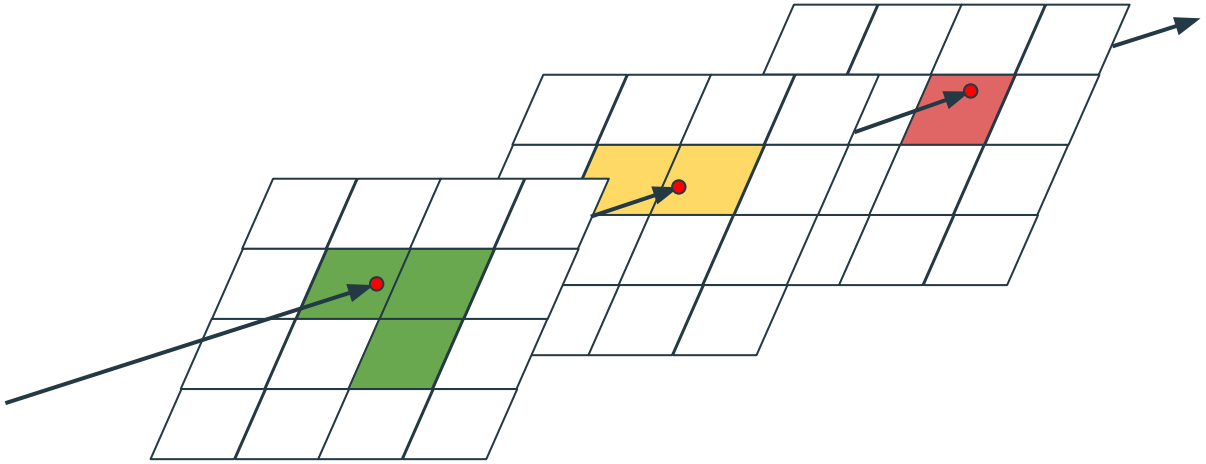
Why high speed and bandwidth?

Retina, a R&D project aimed at developing a specialized processor allowing the reconstruction of events with hundreds of charged-particle tracks using modern, **high-speed, high-bandwidth** FPGA devices

- p-p collisions happens at 40 MHz (25 ns between two collisions) ⇒ we need to be very fast at reconstructing particle tracks
- 4 TB/s of data out of the detector ⇒ we need a system that can swallow all of them

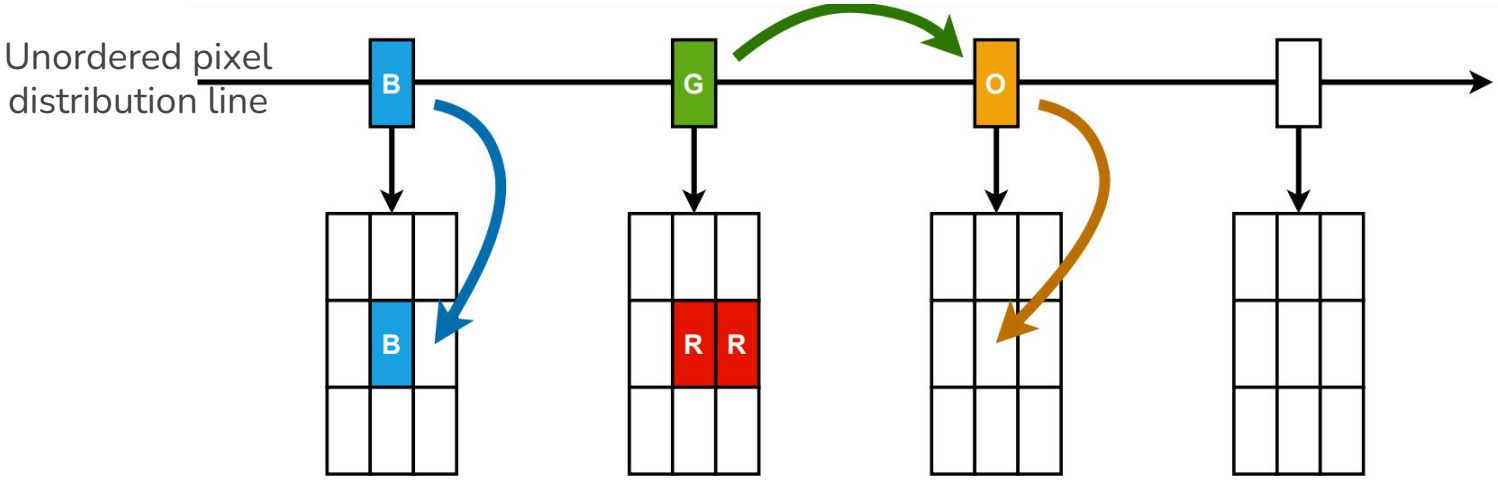
Let’s get our hands dirty

- A charged particle hitting a (silicon-pixel) detector layer might light up multiple parts (pixels) of the layer, due to the particle interaction itself
- The first step in track reconstruction is putting together pixels on the same detector layer and left by the same particle ⇒ **clustering**



FPGA clustering in a rush

- Isolated pixels are clusters made of a single pixel ⇒ no processing required
- Pixels with neighbors fill a set of matrices ⇒ pixels close together fill the same matrix



From prototyping to production

- The clustering project was born as a spin-off the the Retina project
- During my PhD I have:

From prototyping to production

- The clustering project was born as a spin-off the the Retina project
- During my PhD I have:
  - Developed the first prototype

From prototyping to production

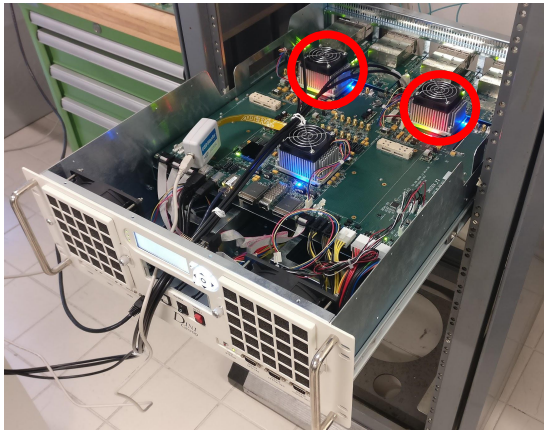
- The clustering project was born as a spin-off the the Retina project
- During my PhD I have:
  - Developed the first prototype
  - Studied its performances

From prototyping to production

- The clustering project was born as a spin-off the the Retina project
- During my PhD I have:
  - Developed the first prototype
  - Studied its performances
  - Integrated and deployed it to production

FPGA clustering - development

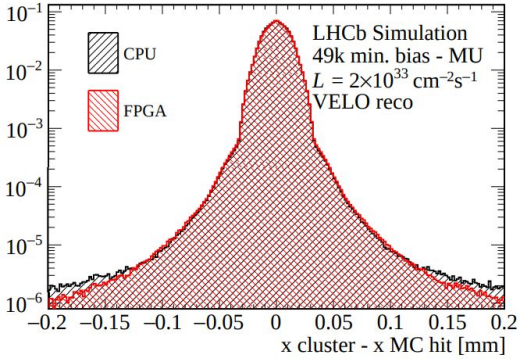
- The first functioning prototype of the clustering firmware was developed and tested within the LHCb-Pisa INFN laboratory
- The main goals of the development stage were the following:
  - Develop a first working firmware that reconstructs clusters correctly
  - Test the firmware on a prototyping board, to check that it is fast enough
  - Measure the amount of FPGA resources required

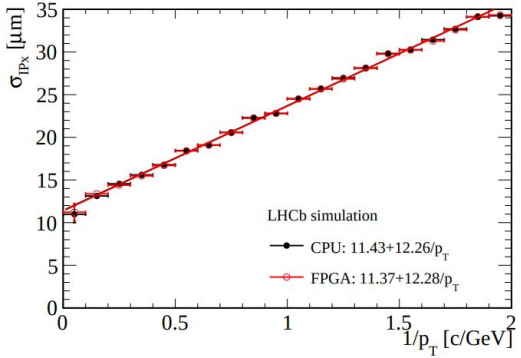


- Since the firmware was running fine, at a high speed and requiring a low amount of resources, the collaboration asked us to integrate it within the readout cards ⇒ reduce computational load on trigger
  ⇒ reduce bandwidth
  ⇒ reduce power consumption
  ⇒ no extra costs

FPGA clustering - physics performances

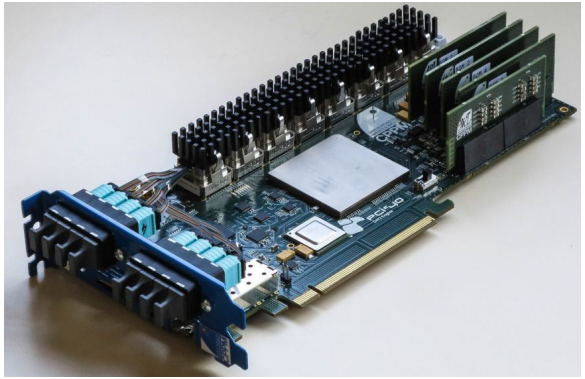
- We have developed a new clustering algorithm, running on an FPGA platform ⇒ we need to make sure it has good physics performances
  - Check that FPGA clusters reconstruct the position of the impinging particle precisely
  - Check that tracks produced from FPGA clusters have the same quality wrt the ones produced by more standard algorithms
- These checks have been performed using LHCb simulation where the particles produced by a simulated p-p interaction interact with a simulated detector

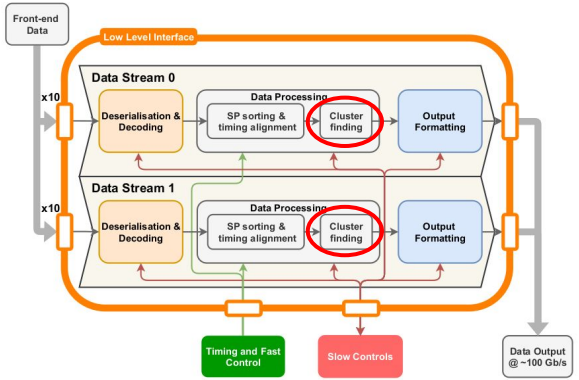




FPGA clustering - integration

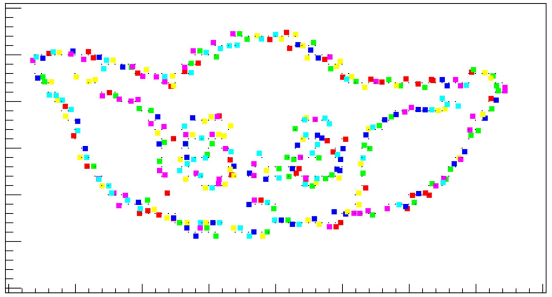
- Having verified that the clustering has a high enough throughput, while satisfying the very demanding LHCb physics requirements, the clustering firmware has been fully integrated within readout cards
  - Extract the cluster-reconstruction part of the firmware
  - Make input-output interfaces LHCb-compliant
  - Add monitoring and error handling
- The trigger software had to be updated to accept clusters as input instead of single detector pixels

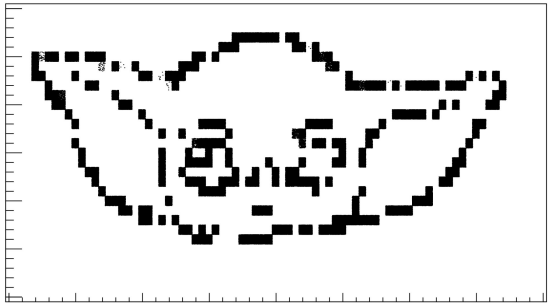




FPGA clustering - deployment

- Deploying the firmware to the production system poses its own issues and challenges:
  - Deal with real detector data
  - Scale up to full system
  - Achieve high data taking efficiencies
  - Coordinate with other LHCb subsystems
  - Wait for beam from LHC
  - ...





Looking forward

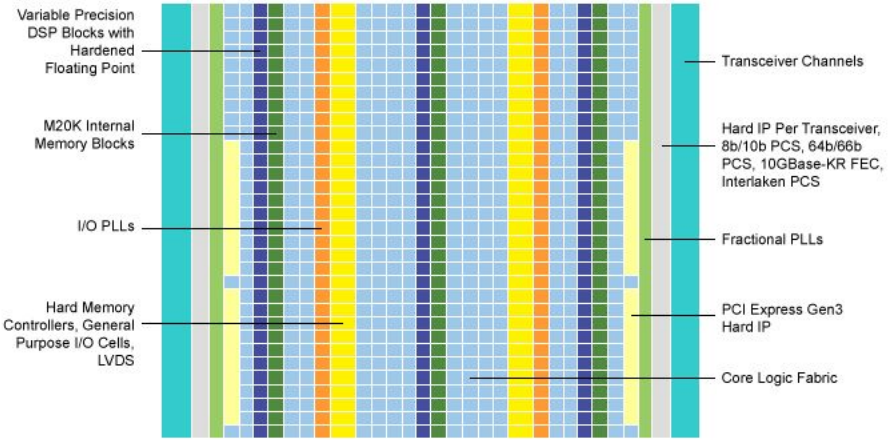
- LHCb offers the unique opportunity to learn and actively contribute to leading edge particle physics research

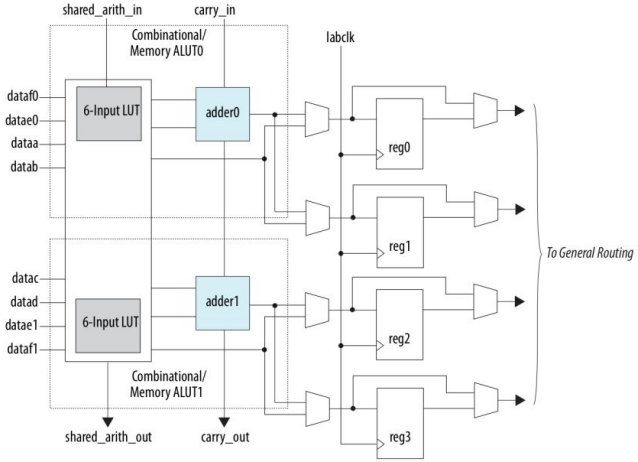
- LHCb is currently on an upgrade path towards High Luminosity LHC (2030s) ⇒ many opportunities to contribute for every taste, from detector design and construction, to readout planning and development, from coding fast and efficient trigger to detailed simulations

QUESTIONS?

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)

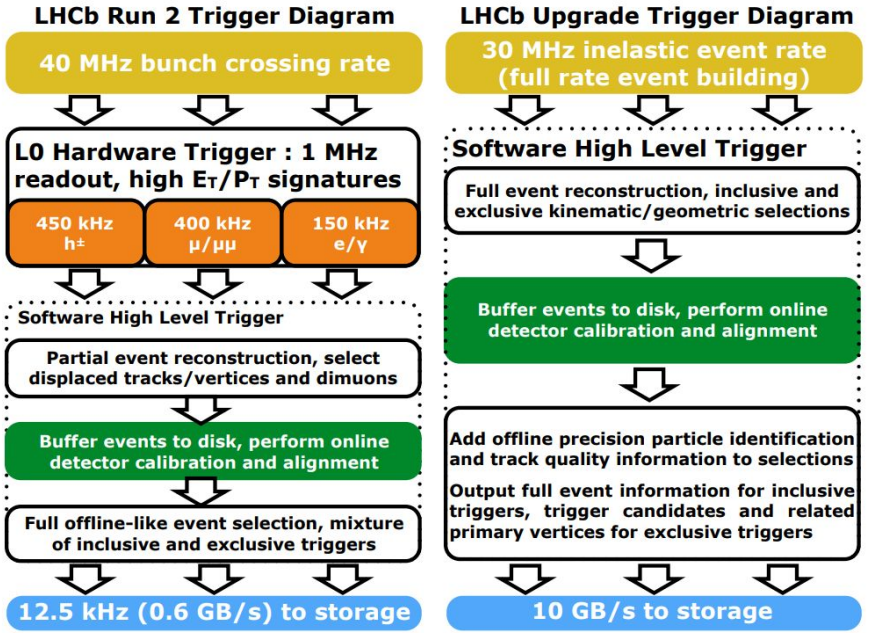




LHCb trigger

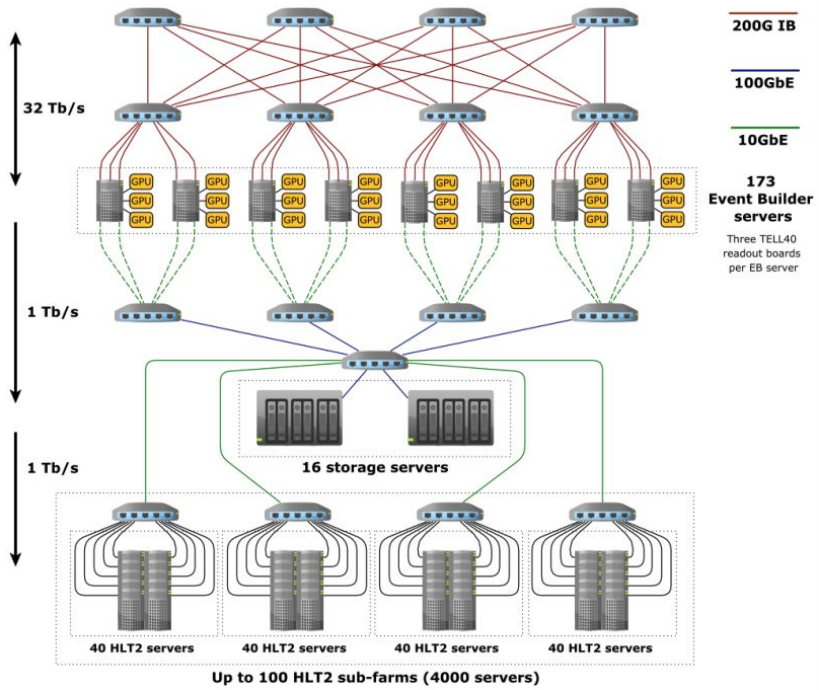
LHCB-FIGURE-2020-016

- 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



Event building (EB)

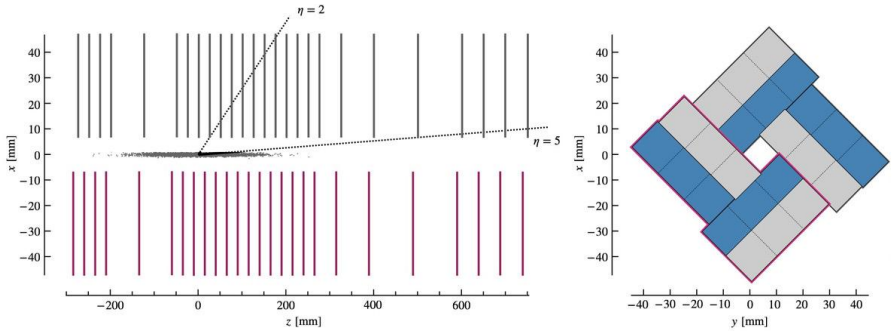
LHCb-DP-2021-003



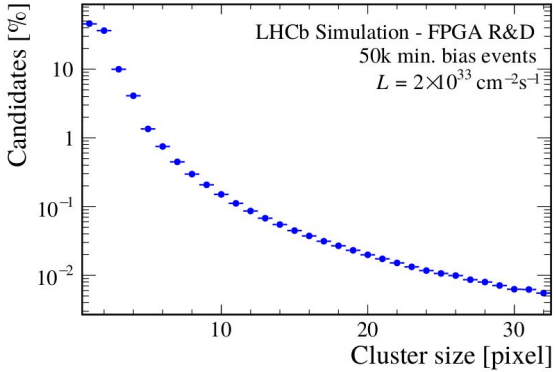
- Each sub detector sends raw data, asynchronously, to a unique EB node, that receives data through DAQ cards
- EB nodes exchange detector data using an EB network
- All detector data for an event are sent to a specific EB node that builds the event
- GPU cards run HLT1 within the EB servers, reducing 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

The VELO detector

- 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

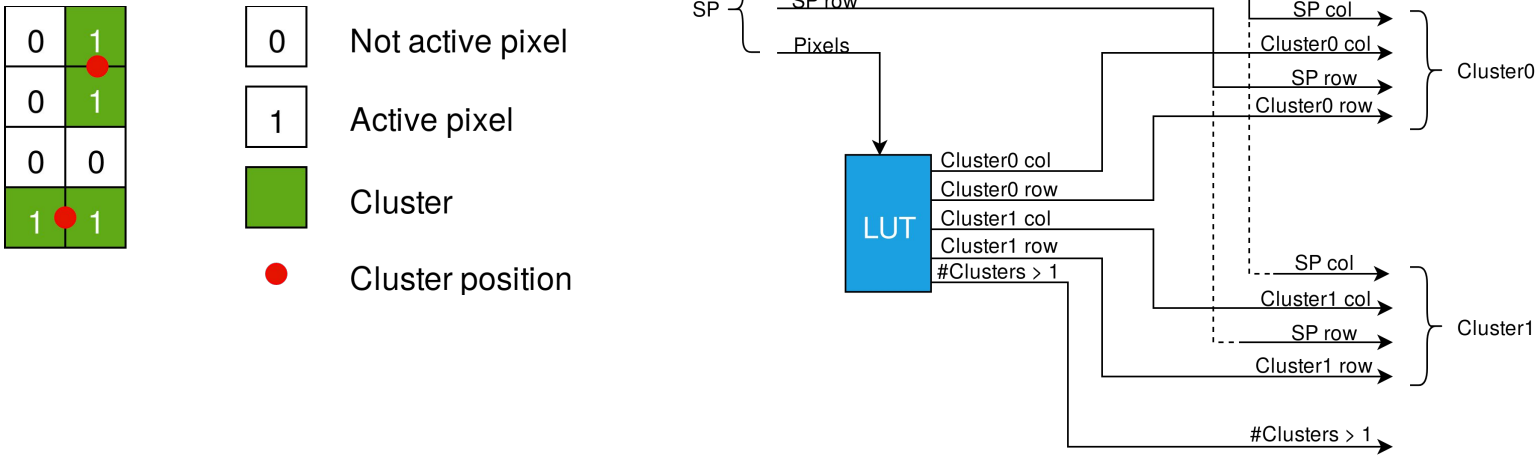


- Pixels are read in groups of 2x4 pixels (**SuperPixels**)
- VELO clusters are typically made of few pixels (1-4)
- The first step of the cluster reconstruction is to flag isolated SuperPixels (isolated = none of the 8 neighbors SPs have any active pixel)



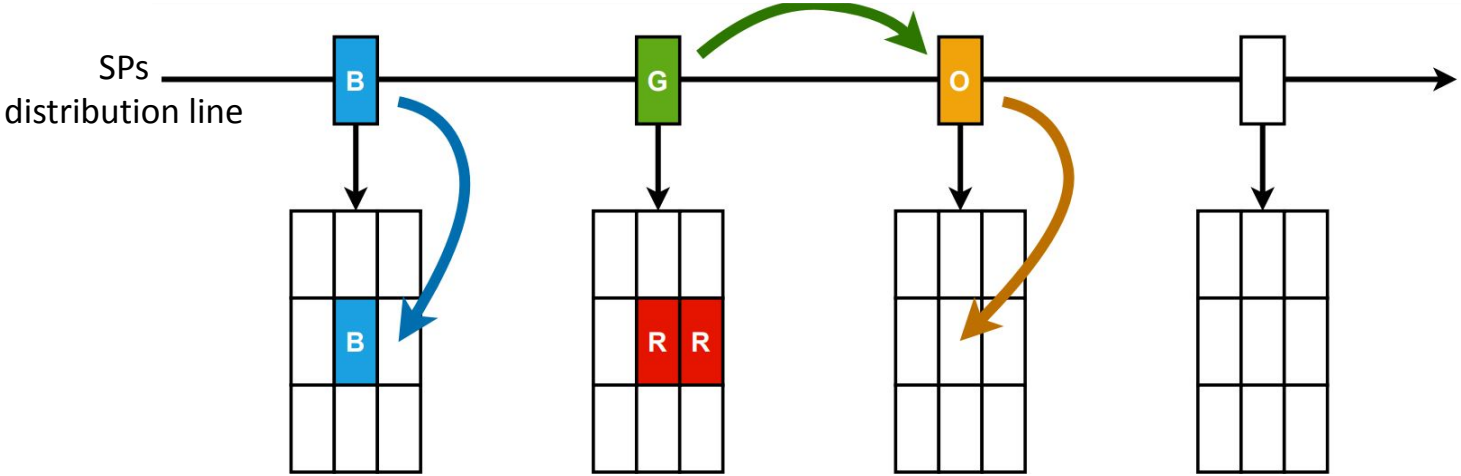
VELO clustering - algorithm overview

- 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)



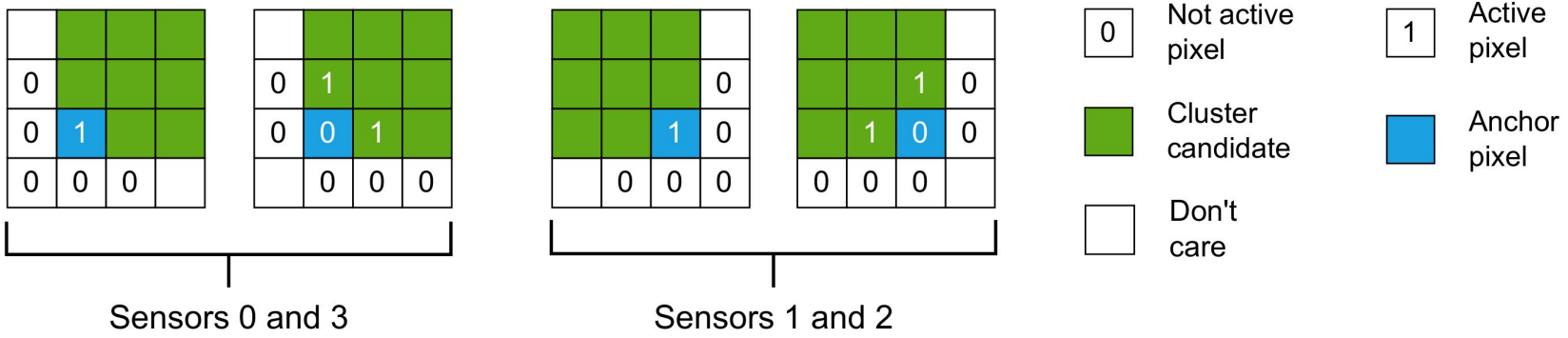
VELO clustering - algorithm overview

- 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



VELO clustering - algorithm overview

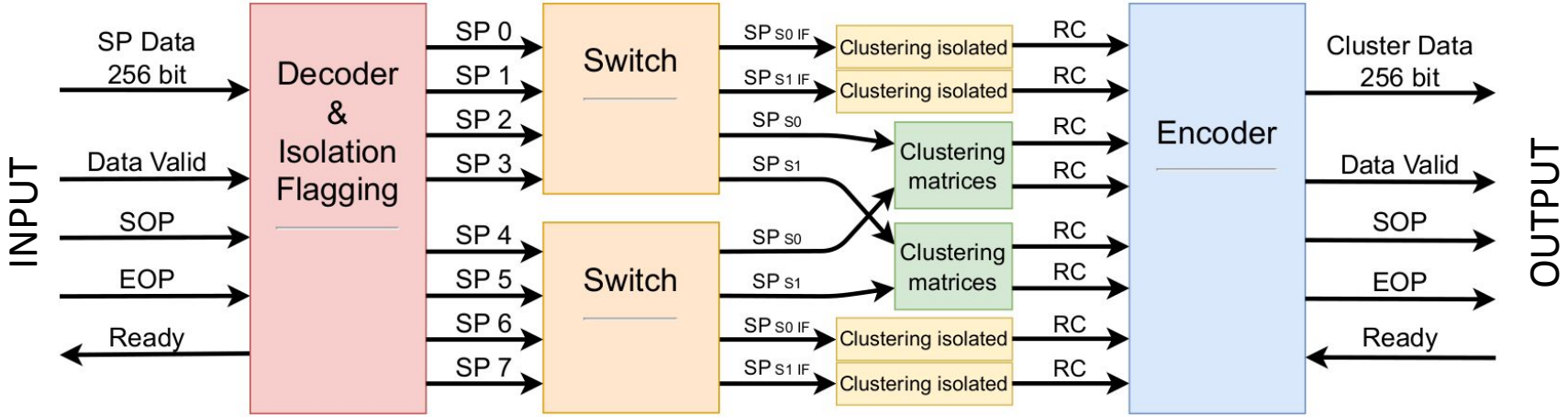
- At the end of each event, in a fully parallel way, each pixel checks if it belongs to one of the following patterns, if so a cluster candidate is identified
- Each cluster candidate is resolved using a LUT



- 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

VELO clustering - 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

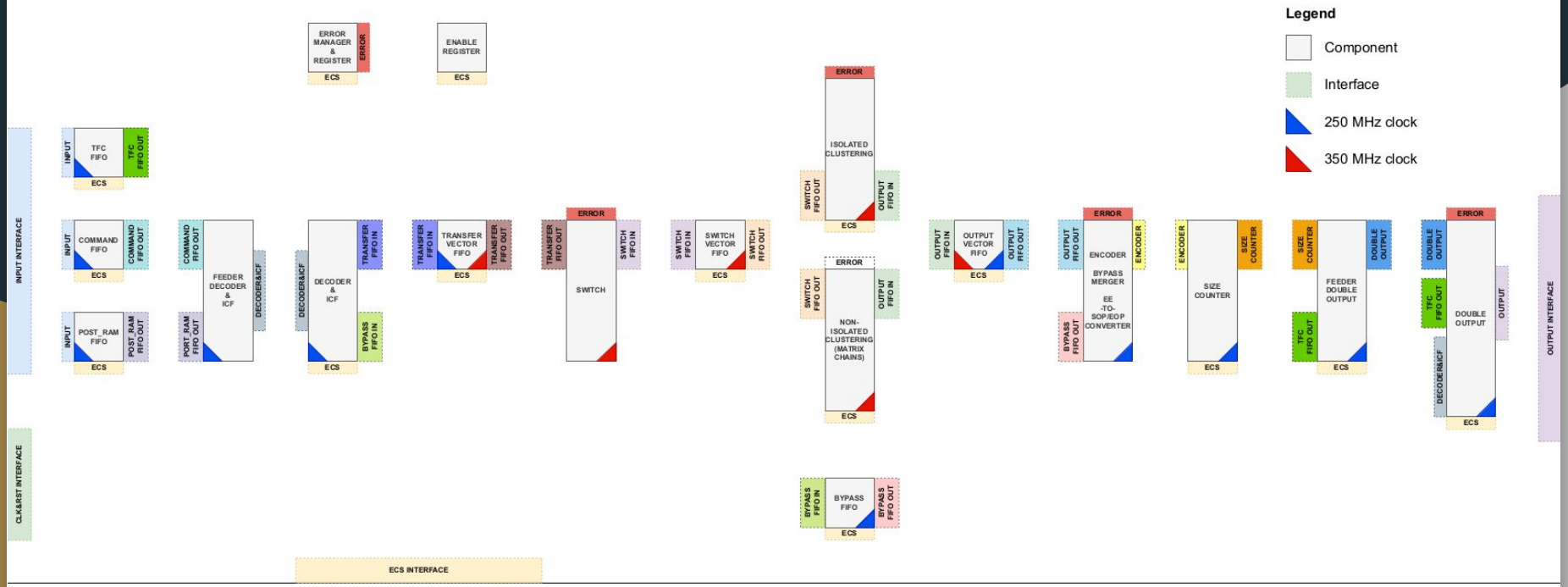
Arranges SPs by sensor and isolation flag

Reconstructs clusters from isolated SPs

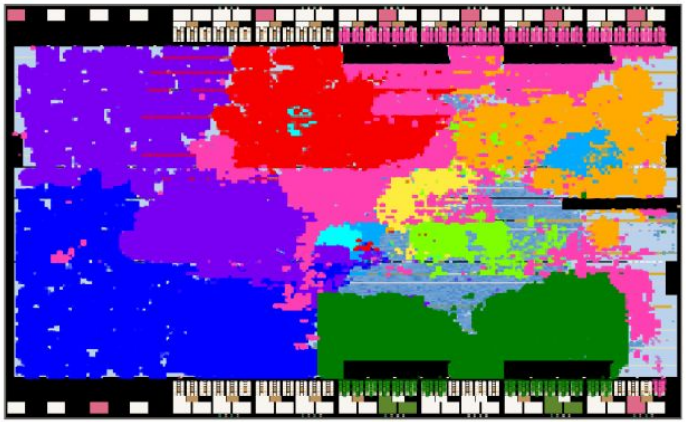
Reconstructs clusters from non isolated SPs

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

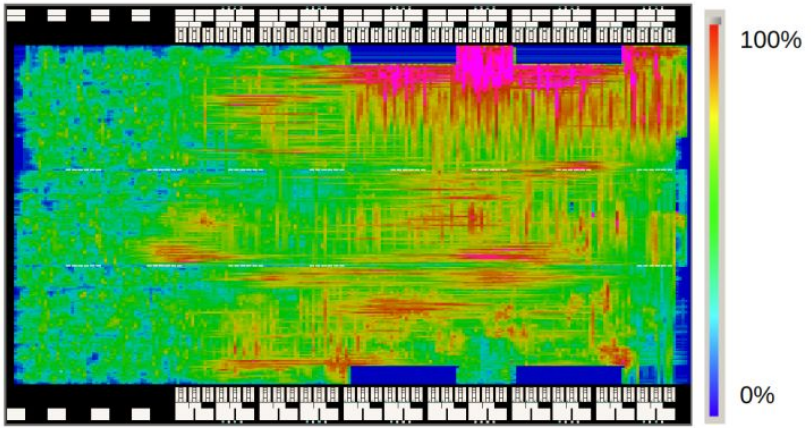
VELO clustering - firmware overview



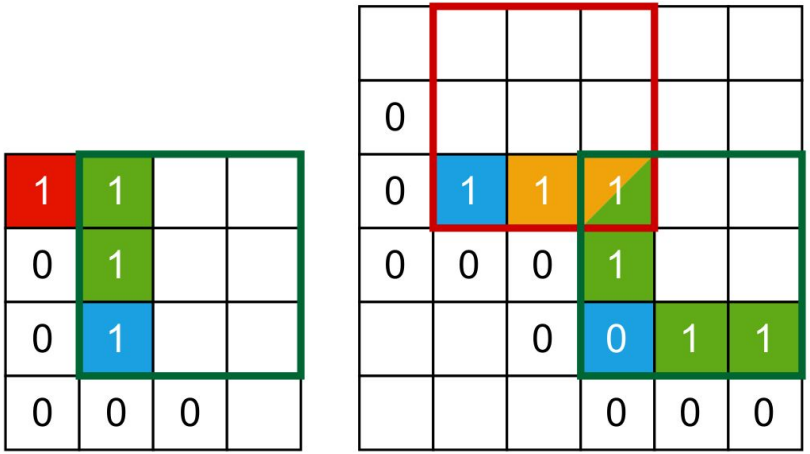
Fitting inside the FPGA



LLI:lli_inst
TELL40_0|PRE_ROUTER
TELL40_0|ROUTER
TELL40_0|POST_ROUTER
TELL40_0|CLUSTER
TELL40_1|PRE_ROUTER
TELL40_1|ROUTER
TELL40_1|POST_ROUTER
TELL40_1|CLUSTER
pcie_top_x8x8



Corner cases



Anchor pixel

Left out pixel

Cluster

Cluster

Pixel of **cluster**

Pixel of **cluster**