EPJ Web of Conferences **245**, 10001 (2020) https://doi.org/10.1051/epjconf/202024510001
CHEP 2019

Reconstruction of track candidates at the LHC crossing rate using FPGAs

Giulia Tuci[1,*] and *Giovanni* Punzi[1], on behalf of the LHCb RTA project

[1]Universitá di Pisa and INFN-Pisa, Largo Bruno Pontecorvo 3, Pisa (IT)

Abstract. In 2021 the LHCb experiment will be upgraded, and the DAQ system will be based on full reconstruction of events, at the full LHC crossing rate. This requires an entirely new system, capable of reading out, building and reconstructing events at an average rate of 30 MHz. In facing this challenge, the system could take advantage of a fast pre-processing of data on dedicated FPGAs. The results of an R&D on these technologies, developed in the context of the LHCb Upgrade I, are presented in this document. In particular, the details and potential benefits of an approach based on producing in real-time sorted collections of hits in the VELO detector (pre-tracks) are discussed. These pre-processed data can then be used as seeds by the High Level Trigger (HLT) farm to find tracks for the Level 1 trigger with much lower computational effort than possible by starting from the raw detector data, thus freeing an important fraction of the power of the CPU farm for higher level processing tasks.

1 Introduction

The LHCb detector [1] is a single-arm forward spectrometer covering the pseudorapidity range $2 < \eta < 5$, designed for the study of particles containing b or c quarks. The silicon-strip vertex detector surrounding the pp interaction region allows c and b hadrons to be identified from their characteristically long flight distance and the tracking system provides a precise measurement of the momentum of charged particles ($\sigma_p/p \sim 0.5\% - 1\%$).

The detector has successfully taken data during Run 1 (2011-2012) and Run 2 (2015-2018), and it is currently being upgraded to start taking data in 2021 with an instantaneous luminosity 5 times larger than before. To take full advantage of the increased luminosity, the software trigger will be upgraded to fully reconstruct events at the full LHC collision rate (30 MHz on average), for an aggregate data flow of 40 Tb/s. For most of the events, only the most relevant part of the data will be preserved; this implies that the majority of the analysis and data selection will have to occur in real time. Performing the analysis of this huge flux of complex data in real time, effectively reconstructing and selecting a rich array of known, and possibly unknown, physics phenomena is an enormous challenge .

A large effort is being made by the collaboration to keep a high reconstruction efficiency for all decay channels even in this new and more difficult regime. This includes R&D studies on new devices capable of performing part of the track reconstruction before the software trigger ("hardware accelerators"). The majority of the CPU time-budget available at the first stage of the high-level trigger (HLT1) is in fact employed not in making intelligent physics

*e-mail: giulia.tuci@cern.ch

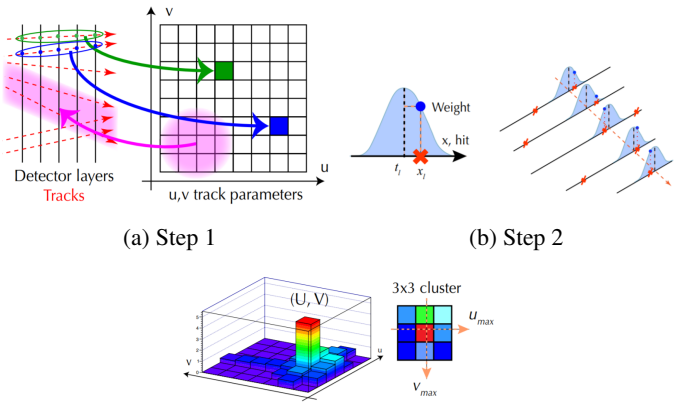
© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (http://creativecommons.org/licenses/by/4.0/).

selections, but rather in highly repetitive and parallelizable track-reconstruction tasks, that can be easily offloaded to an FPGA-based system. This system can operate in a pipeline before the events are loaded in the CPU farm that executes the HLT1, freeing up time that can be used, for example, to process tracks with low p_T or to run the second stage of the trigger (HLT2). Reconstruction of tracks in the vertex detector (VELO) is the first step in the HLT1 sequence [2], is a necessity for every tracking sequence considered in the Run 3 plan, and it burns a large fraction of the available time. For this reason, it is a natural target for offloading to a specialized device.

2 The "artificial retina" algorithm

The technology used as the main tool at the core of this system (from now on it will be called the *retina* system) is a highly-parallel architecture for pattern recognition going under the name of "artificial retina" [3]. It owes its name to the effort towards mimicking some structural features and general principles found in the organization of the natural neural network responsible for fast vision processing in living organisms [4].

In brief, the space of possible value of track parameters is divided into small cells (Step 1 in Figure 1). An independent cellular processor ('engine'), is associated to each cell, accumulating weights (excitation levels) for every detector hit compatible with the local parameters, dependent on the hit distance from the cell center point (Step 2 in Figure 1). Tracks are then reconstructed by the matrix of cellular processors, operating in a fully parallel fashion, exchanging information with their neighbors to find clusters of excitation in the parameter space (Step 3 in Figure 1).



(a) Step 1 (b) Step 2

(c) Step 3

Figure 1. Processing steps of pattern recognition with the retina algorithm. The parameter space is mapped into a matrix of cells (a) and hits are delivered to appropriate cells, which accumulate weights depending on the distance between the hit and the center of the cell (b). Then for each cell all the weights are summed and a track is reconstructed identifying a local maxima over a threshold (c).

The FPGAs output is a list of hits associated to each reconstructed track-candidate, which can be used as an input of the CPU HLT1 sequence, where the hits are fitted to determine the track parameters.

The hardware architecture allowing the processing just described to be carried out at a very high speed is schematically shown in Figure 2 [5, 6]. Hit data from a tracking detector

flow into the system along a number of parallel lines; a number of blocks, each mapping a section of the track parameter space, hosts a matrix of cellular processors; a large, custom fast switching network takes care of delivering the hits only to the locations of the parameter space where they are needed.

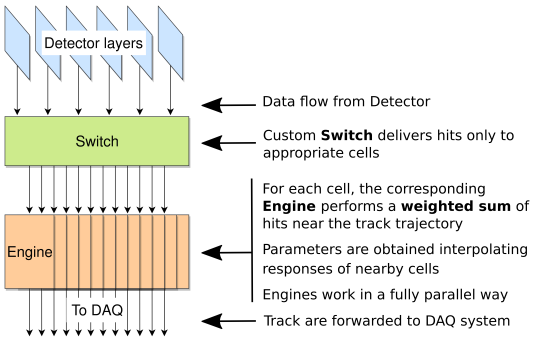


Figure 2. General structure of a "retina system".

3 Integration in the DAQ system

The general structure of the Upgraded LHCb Data Acquisition, as described in the LHCb online TDR [7], is shown in Figure 3. The LHCb Event Builder (EB) will be composed of about 500 rack-mounted PC boxes ("nodes"), each equipped with an optical data acquisition card (PCIe40). Each PCIe40 card receives data from the front-end over up to 48 optical links, reading out from a specific piece of the detector located in the experiment cavern. Data from several bunch-crossings are merged into a multi-event fragment packet (MEP) to reduce the message rate and ensure efficient network usage. Event-building is then performed by combining all MEP containing data from the same bunch-crossings in a single piece of data. Complete events are then sent via a LAN to the CPU farm , running the HLT. In order

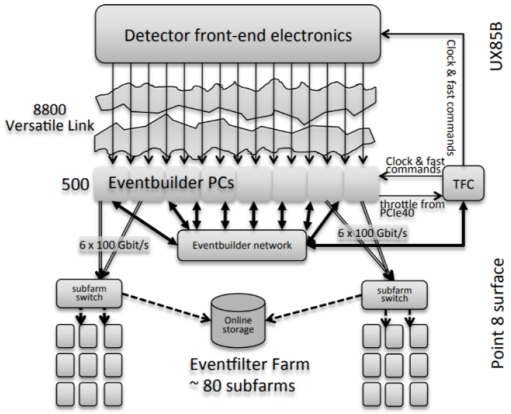


Figure 3. General structure of upgraded LHCb data acquisition system. The PCIe40 cards lie in the eventbuilder PCs.

to fit the *retina* project within this system, data has to be intercepted at the exit of the PCIe40 card, then tracks have to be reconstructed and the results have to be injected in the MEP before it is sent over for building. In this way a fully-embedded reconstruction, that remains transparent to the rest of the system, can be realized. The system would be implemented as a set of identical PCIe-hosted FPGA cards, each supporting a fragment of both the switch and the processing blocks necessary for track reconstruction, appropriately broken down and distributed across nodes. This leads to a conveniently uniform and scalable system. Cards with the needed features are commercially available today, they have the low-latency response needed to operate within the EB requirements, and are easily integrated in the EB by insertion in an available PCIe slot in each EB node connected to the detector of interest.

4 Implementation and performance

4.1 The upgraded VELO detector

As part of the LHCb Upgrade, the current VELO will be replaced by a new detector, based on silicon pixel technology [8]. The new VELO will consist of 52 modules, arranged in 26 layers, positioned along the beam axis, both upstream and downstream of the nominal interaction point.

For the purpose of the first implementation it has been decided to use only data from layers 8 to 26. Those layers are located at positive values of z, where z = 0 corresponds to the nominal interaction point (see Figure 4), and they hold most of the data useful in the reconstruction of forward-going tracks produced in the vicinity of the primary interaction. The remaining layers are mainly used to refine the primary vertex position and for the data-taking with the gas injection system (SMOG2) active [9], therefore they can be treated separately. Each module will be read by a separate PCIe40 card, and for each PCIe40 an FPGA card will be added to realize the *retina* system; therefore a total of 38 cards will be needed.

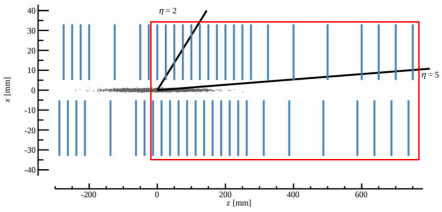


Figure 4. Layout of the upgraded VELO. The detector consists of two retractable halves, each housing an array of 26 silicon pixel detector modules. The red rectangle indicates the modules of interest for forward-track reconstruction

4.2 Emulator of VELO pattern recognition

Before buying the necessary hardware and writing the firmware, a software emulator of the system has been realized to study the performance and optimize the configuration of the pattern recognition algorithm.

To simulate the parameter space division in regions, a grid is built with a fixed number of cells, each of them corresponding to a pattern track, and is filled with all receptors, *i.e.* with the interceptions of pattern tracks with detector layers. Hits coming from the LHCb official Monte Carlo events, generated with the Run 3 conditions, are used as input to the simulation

and, sequentially for each hit, the distance with all the receptors is calculated. For each receptor having a distance lower than the fixed distance search d_s , its weight is computed and stored in a grid. Each cell of the grid corresponds to one cell of the parameter space. So each grid-cell accumulates weights of all receptors of the related track pattern. Once all the hits in the event are processed, the search of local maxima starts. For each maximum found, which corresponds to a track candidate, the list of the hits contributing to that maximum with a non-zero weight is saved in a file.

Track candidates produced by the emulator and associated hits are then imported in the LHCb framework thanks to an additional routine: the produced track is finally formatted and plugged into the standard LHCb HLT1 sequence, that from this point on continues with the rest of the reconstruction, using these tracks in place of the tracks produced by the CPU-based VELO reconstruction. This allows to run exactly the same performance diagnostics on exactly the same samples, switching between the two reconstruction methods.

4.3 Parameter space matrix

To perform pattern recognition with the artificial retina algorithm, a parameter space for the tracks needs to be defined. In the adopted configuration, the parameter space is defined by two parameters: the x and y intercept of the track with a fixed-position layer, located at $z = 70$ cm. However, two parameters are not enough to describe a straight line in a 3D space. The 3D track parameter space has been therefore segmented in 10 slices running along the z-axis. In this way we can have sufficient reconstruction acceptance over the entire (rather wide) distribution of primary vertex along the LHCb beam crossing point. Several *retina*-matrices have been instantiated, each tuned to a different z-slice, and each of them running the artificial retina algorithm in parallel, independently of the others.

Tracks with different z_{vtx} (the longitudinal coordinate of the origin vertex of the track) may happen to be reconstructed in more than one z-slice; the possibility of multiple solutions is avoided by having each matrix perform a check on the z of minimum distance to beam of every found track, to ensure it belongs to the correct fiducial region. As a final step, remaining ambiguities need to be sorted out and possible spurious hits rejected before performing the final fit of track parameters. This might be better left to the HLT1 with the benefit of the most accurate alignment information; a decision has not yet been made on this point. While in principle alignment parameters can be also loaded into the FPGA, this implies a more complex system only for performing a task which is already very fast on CPU.

The size of the acceptance region in z depends on the individual cell sizes, that in turn is constrained from below by the need of limiting the needed amount of FPGA resources, and constrained from above by the multiplicity of hit combinations/ambiguities within each cell. A matrix of 100x100 = 10000 cells has been allocate to each slice, as this produces a good uniformity of acceptance along z and, at the same time, a limited number of multiple combinations within the same cell.

4.4 Physics performance

The performance measured on 1000 events containing a $B_S^0 \rightarrow \phi\phi$ decay is shown in Table 1 [10], and the efficiency variation as a function of z_{vtx} is shown in Figure 5.

The results obtained demonstrate that the HLT1 algorithm performance is only negligibly affected when running on tracks produced by FPGAs. Even if the marginal quality tracks with less than 5 hits are included, the efficiency of the FPGA reconstruction is still within less than a percent from the CPU reconstruction in the majority of cases. The small decrease at large z is not intrinsic to the methodology, but simply due to our choice of the acceptance

Table 1. Summary of efficiencies of the VELO tracking algorithm for different type of tracks using both the CPU-based and the FPGA-based pattern recognition algorithm. Numbers obtained on 1000 $B_s^0 \to \Phi\Phi$ events. The efficiency (defined as the number of reconstructed tracks divided by the number of reconstructible tracks, *i.e.* tracks with at least 3 hits in the VELO) is calculated using Long tracks, *i.e.* tracks crossing all the tracking sub-detectors, with $2 < \eta < 5$, $p > 5$ GeV/c and with more than 5 hits (Monte Carlo truth) in the VELO detector. Tracks belong to the fiducial region if the z coordinate of the origin vertex is located between -200 mm and 200 mm.

Track type	ε CPU pat-reco (%)	ε FPGA pat-reco (%)	
		all z	fiducial z-region
Long tracks with $p > 5$ GeV/c and hits in VELO> 5	99.84 ± 0.02	99.27 ± 0.06	99.45 ± 0.05
Long tracks from b with $p > 5$ GeV/c and hits in VELO> 5	99.61 ± 0.13	99.24 ± 0.21	99.41 ± 0.18
Long tracks from c with $p > 5$ GeV/c and hits in VELO> 5	99.89 ± 0.12	98.50 ± 0.53	98.62 ± 0.53

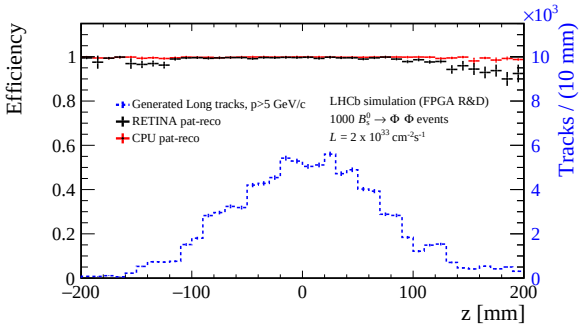


Figure 5. Comparison between VELO tracking efficiencies obtained with the standard CPU algorithm and the retina patter recognition algorithm. The efficiency is shown as a function of the z coordinate of the track origin vertex .

cutoff in the tails of the distribution. It has a small effect only on strange particle acceptance, and can be tuned if desired.

5 Conclusions

To summarize, detailed simulation studies show fairly good performance of FPGA tracking of the VELO, and feasibility of implementation in current commercial hardware. From a first estimate of the logic resources, the firmware will comfortably fit in 38 FPGA cards (needed to match the VELO readout boxes). This is a relatively small system and a good first test-case for other detectors.

Concerning computational performance, an actual hardware implementation of a generic 6-layer tracker had already demonstrated a throughput exceeding 30 MHz on prototype FPGA boards of a previous generation [6]. For the application proposed here the firmware need to be specialized for the VELO and ported to a sufficiently powerful FPGA card that is compatible with the LHCb DAQ. A modern commercial board with the needed characteristics has been

identified and is currently under testing, with the aim of producing a limited-size but complete demonstrator of the actual system functionality in the next few months.

References

[1] LHCb collaboration, R. Aaij et al., Phys. Rev. D **97**, 031101 (2018)
[2] LHCb collaboration, R. Aaij et al., LHCb-PUB-2017-005, (2017)
[3] L. Ristori, Nucl. Instrum. Methods A **453**, 425 (2000)
[4] N. Priebe and D. Ferster, Neuron **45**, 194 (2012)
[5] A. Abba et al., LHCb-PUB-2014-026, (2014)
[6] R. Cenci et al., PoS TWEPP-17 , 136 (2017)
[7] LHCb collaboration, R. Aaij et al., CERN/LHCC 2014-016, (2014)
[8] LHCb collaboration, R. Aaij et al., CERN/LHCC 2013-021, (2013)
[9] LHCb collaboration, R. Aaij et al. CERN/LHCC 2019-005, (2019)
[10] LHCb collaboration, R. Aaij et al., LHCb-FIGURE-2019-011, (2019)