Università di Pisa

Dipartimento di fisica "E. Fermi"

Corso di laurea magistrale in fisica

A 2D FPGA-based clustering algorithm for the LHCb silicon pixel detector running at 30 MHz

Candidate:
Luca Giambastiani

Advisor:
Dr. Michael J. Morello

Anno accademico 2019/2020

Abstract

Starting from the next LHC run, the upgraded LHCb data acquisition system will read and process events at the full LHC collision rate (averaging 30 MHz) by means of a large CPU farm. In order to save the power and flexibility of CPUs for the higher level tasks, an effort is being made to address the lowest-level, more repetitive tasks at the earliest stages of the data acquisition, by means of specialized processors, generally called "accelerators". Modern FPGA devices are very well-suited to perform with a high degree of parallelism certain computations, that would be significantly time demanding if performed on general-purpose CPUs. This thesis describes a custom firmware implementation of a new 2D cluster-finder algorithm for the LHCb VELO pixel detector, that will run in the LHCb FPGA readout cards in real time during data taking at the unprecedented event rate of 30 MHz. The results and the performances achieved with this specialized system are reported after being measured in tests emulating realistic running conditions of the LHCb Upgrade and the operation of the clustering algorithm at low level.

Contents

Acronyms

SP *Super Pixel*, the unit in which pixels are packed. It is a 4×2 pixel matrix.

LUT *Lookup table*, a memory filled with the results of a calculation for each possible input datum.

VELO *VErtex LOcator*, the detector closest to interaction point in LHCb.

MUX *Multiplexer*, logical device that connects its output port to one of its inputs based on the state of selector signals.

FIFO Memory of the *First In First Out* type in which data are written and read back in the same order.

EE *End Event*, a special empty data packet indicating the end of an event.

ECS *Experiment Control System*: The system by which the experiment is monitored and configured by the control room.

FE *Front-End*, an electronic management and data transfer system installed on detector boards.

TFC *Timing and Fast Control*, system that distributes clock signals, commands and configurations to FE, and calibration commands to detectors.

BXID *Bunch Crossing ID*, counter of bunch collisions.

SOP and EOP *Start Of Package* and *End Of Package* respectively, they are used to mark the beginning and the ending of the transmission of data of one event.

FSIZE Signal carrying the number of SPs or clusters to be sent for that event.

FTYPE Signal carrying information on the type of data transmitted.

READY Signal generically used by some entity to tell the previous one that it is able to accept new data.

HOLD Signal generically used by some entity to tell the previous one that it cannot accept other data because it is busy.

ICF *Isolated Cluster Flag*, a flag indicating whether a SP is surrounded by only empty SPs.

rdempty, wrempty Signals output by FIFOs indicating emptiness of respectively its read and write side.

rdreq Signals sent to FIFOs to ask for data.

wrreq Signals sent to FIFOs to store new data.

GEC *Global Event Cut*, a selection that discards events with too many clusters in UT and SciFi.

FSM *Finite State Machine*, abstract machine that can be in exactly one of a finite number of states at any given time and it can change from one state to another in response to some inputs.

Introduction

The LHCb detector, operating at a luminosity of 4×10^{32} cm^{-2} s^{-1}, collected about 9 fb^{-1} of integrated luminosity of good data during the LHC Run 1 and Run 2, starting from 2010 until the end of 2018, and it has demonstrated that the LHC is an ideal laboratory for the heavy-flavour physics. During the years LHCb has produced some of the most significant achievements of the field. These include the discovery of the ultra-rare decay $B_s^0 \to \mu^+\mu^-$ [1, 2], the first single-experiment observation of charm mixing [3], the first observation of *CP* violation in the charm sector [4], the worlds most precise measurements of both the CKM angle γ [5] and the B_s^0 weak-phase φ_s [6, 7], and tantalizing hints of Lepton Flavour Non-Universality in the B meson decays [8, 9].

During the current Long Shutdown 2 (LS2) LHCb is being replaced by an upgraded experiment, referred as the Phase-I Upgrade (or the so-called LHCb-Upgrade). In order to read out the experiment at the crossing rate of the LHC (30 MHz of visible collisions) and to operate at a higher luminosity of 2×10^{33} cm^{-2} s^{-1}, most of the sub-detectors will be replaced, as will all of the frontend electronics and data-acquisition system. In particular, a pixel Vertex Locator (VELO) [10], a silicon tracking station before the magnet (UT) and a large-scale downstream Scintillating Fibre (SciFi) tracker system will be installed [11]. A new trigger system will also allow the experiment to function effectively at the higher luminosity, and will provide significantly increased efficiency, particularly for hadronic final states. By the end of LHC Run 4, in 2030, the experiment will have accumulated a data sample of around 50 fb^{-1}.

The Phase-I Upgrade will greatly improve the sensitivity of many flavour studies, however, the precision on a host of important, theoretically clean, measurements will still be limited by statistics, and other observables associated with highly suppressed processes will be poorly known. There is therefore a strong motivation for building a Phase-II Upgrade, which will fully realize the flavour potential of the HL-LHC during the LHC Run 5 ($\geq$ 2031) at a luminosity larger than 10^{34} cm^{-2} s^{-1} with the aim of collecting > 300 fb^{-1} of total integrated luminosity [12, 13]. The challenges of performing precision flavour physics at such high luminosities are daunting. The mean number of interactions in each event will be around $50 - 100$. The increased particle multiplicity and rates will present significant problems for all detectors, as will the increased radiation damage for certain components. In particular the data handling demands will achieve unprecedented levels, and as a consequence the Trigger and DAQ (TDAQ) system will be the major cost item and it may amount to a major technical limitation to the experiment performance. Last but not least, the scale of the needed offline computing resources is expected to grow significantly, much more than the expected growth in CPU, disk and network resources [14], and this is particularly true for heavy flavour physics experiments at hadron collisions, as LHCb. A lot of R&D will be needed to find viable solutions to the challenges of the

Phase-II Upgrade.

The LHCb-Upgrade during LHC Run 3 represents an unique opportunity to try out solutions, even before the start of the High Luminosity Phase II of ATLAS and CMS experiments (scheduled in the LHC Run 4) Because of the higher luminosity, the detector must be able to record and analyze the greater rate of data that is produced in the collisions. In addition, to improve the overall efficiency, particularly for hadronic channels, the limitation on the maximum readout rate (1.1 MHz) of most sub-detectors have been removed, so the new High Level Trigger (HLT) will have to process data at the full collision rate, equal to 30 MHz on average. This means that most of the work needed to select and reconstruct events will be done in real-time, hence the necessity of new and powerful TDAQ systems. The produced data throughput (40 Tbit/s) will be so high that the usage of heterogeneous computing systems has been considered as a necessary addition already for the upcoming LHC Run 3. These systems are composed by a large farm of CPUs, versatile and general-purpose devices, and FPGAs and GPUs accelerators, that are specialized hardware very fast at executing simple and repetitive tasks.

The work performed in this thesis has been developed in this context and aims at moving one of the most parallelizable task of the LHCb VELO pixel subdetector, the 2D cluster-finding, to the FPGA readout cards, freeing the HLT farm from the burden of this computation. Thus, the thesis presents a custom FPGA firmware implementation of a new 2D cluster-finder algorithm, that can run in real time during data taking at the input rate of 30 MHz. The firmware of the whole project is written in VHDL hardware description language, without the usage of high-level synthesis tools,[1] to take full advantage of all the features of modern and powerful FPGA chips, and in particular of the high level of parallelization achievable with this specialized hardware. The LHCb Collaboration is currently reviewing the proposal of the LHCb-Pisa Group of adopting such FPGA algorithm as the baseline for the LHC Run 3 and Run 4.

The clustering algorithm and the firmware were developed and tested on a Stratix-V FPGA chip in Pisa during the very early stage of the project. The work performed to port the core firmware to the Arria-10 chip, mounted on the readout cards, along with the development of the all functional firmware for a full integration into the VELO software is described in the first part of the thesis. All the ancillary components, whose purpose was to feed clustering and to read its output for development and testing, have been removed; the remaining clustering core software has been modified in order to be compatible with the rest of the VELO firmware. The standard LHCb control systems, the ECS (*Experiment Control System*) and the TFC (*Timing and Fast Control*), have been implemented, along with a new entity to manage internal and external errors. Additional signals, read at input side, indicating the type and the size of data packets, have been placed at the output side. An entity to output both clusters and corresponding raw data has been added for debugging and testing. The clustering algorithm has also been modified in order to reduce the number of lost and duplicated clusters.

The system has been extensively tested using fully realistic simulated samples at the LHCb Upgrade running conditions, and this work is described in the second part of the thesis. Various quantities related to tracking and clustering quality have been analyzed

[1] High-level synthesis tools provide a direct path to generate register transfer level firmware implementations from algorithm specifications written in high-level programming languages, such as C/C++.

and compared for FPGA and CPU algorithm, in order to determine the accuracy and the robustness of the FPGA-based algorithm. A high level simulation, written in C++ programming language, capable to emulate all the features of the clustering algorithm was integrated in the official software of the experiment in order to reconstruct realistic simulated *pp* collisions events at LHCb-Upgrade conditions. Clustering and tracking efficiencies have been studied in details as a function of several kinematic and topological variables. The clustering efficiency is found to be excellent, very close to that one obtained with the CPU algorithm, particularly for clusters originated by the type of tracks most useful for physics analysis. Tracking efficiencies for the two algorithms are almost indistinguishable. Thanks to FPGA algorithm, a gain in event processing throughput of about 8% is measured for the first stage of the HLT. More importantly, the success of this implementation represents a first, if small, step in the direction of moving non-trivial reconstruction functions from traditional CPUs to heterogenous systems running at a very early level of the data taking, that will hopefully be further expanded in the future.

Chapter 1

Heavy flavour physics and heterogeneous computing

This chapter presents a short introduction to the physics opportunities of precision measurements in flavour physics at the high intensity frontier, with a particular focus on the current hot topics: CP violation, lepton universality, and exotic hadrons. The challenges of future high luminosity environment at hadron colliders are also discussed, with particular reference to the case of Future Upgrades of the LHCb experiment, in which new forefront techniques for data acquisition and triggering will be employed for the first time.

1.1 Introduction

The Standard Model (SM), the theory that describes all particle physics measures as of today, has survived a large number of tests. Measurements of the Higgs boson's properties, including its quantum numbers and production and decay modes, are consistent with SM predictions. Electroweak precision constraints on variables such as the W boson, top quark masses and the effective weak mixing angle agree with the SM. Studies of *CP* violation in the quark sector are consistent with the predictions based on there being a sole source of matter-antimatter asymmetry, encoded in the Cabibbo-Kobayashi-Maskawa (CKM) quark mixing matrix [15, 16]. All determinations of the properties of the so-called Unitarity Triangle fit together within the SM. Searches for candidate dark matter particles, either in the Large Hadron Collider (LHC) or in extremely low background detectors have yielded null results.

Nevertheless, there remain clear arguments to continue to search for physics beyond the SM. There are strong reasons to believe that the hierarchy problem may be resolved in a theoretically attractive way, at a higher energy scale than has never been probed as of today. There is a compelling need to understand the constitution of the 95% of the energy density of the Universe that is not baryonic matter (68% dark energy, 27% dark matter). The origin of the observed excess of matter over antimatter in the Universe also cannot be explained in the SM.

The method of testing the SM through precision measurements in flavour physics is fully complementary to that of searching for on-shell production of new particles in high energy collisions. Mixing and decay of beauty and charm hadrons occur through weak

interactions, but other, currently unknown particles could also contribute, in which case measured parameters such as decay rates and *CP* violation asymmetries would be shifted from the SM predictions. While direct searches for new particles are carried out in the TeV mass range at the LHC, flavour physics measurements can probe much higher mass scales: it has been shown that, considering a generic effective Lagrangian, mixing and *CP* violation observables in the beauty and charm system probe scales of up to 1 000 to 10 000 TeV [17–19].The reach of measurements of flavour physics observables is limited only by precision, both experimental and on theoretical predictions. Rare processes, in particular processes for which the SM contribution occurs through loop diagrams (i.e. flavour changing neutral currents), are often considered golden channels for potential discoveries of physics beyond the SM. These include $B^0 - \bar{B}^0$, $B_s^0 - \bar{B}_s^0$ and $D^0 - \bar{D}^0$ mixing processes, as well as rare decays including $B \to ll$, $B \to Xll$ and $B \to X\gamma$ (where l is a lepton and X is a hadronic system).

The LHC and its high luminosity upgrade will continue, for about the next 20 years, to be our most powerful tool to improve our fundamental understanding of nature. Figure 1.1 shows the roadmap of the LHC, showing runs for data acquisition and shutdown periods for detector and accelerator upgrade. LHC will undertake a major upgrade in LS3 that will give it a substantial increase in luminosity. The upgraded accelerator is called High Luminosity LHC (HL-LHC) and Phase-II is the name of the following physics program, including Runs 4 and 5[1]. For this reason, the 2013 update of the European Strategy for Particle Physics stressed that "Europe's top priority should be the exploitation of the full potential of the LHC," noting that this will "provide further exciting opportunities for the study of flavour physics". This confidence in the potential for flavour physics at the HL-LHC was justified by the successful operation of the LHCb experiment during Run 1 (2011-12) that also indicated the concept and design of a dedicated heavy flavour physics experiment at a hadron collider. Among the several hundreds of publications based on the Run 1 data sample, highlights include the first observation, together with CMS, of the very rare decay $B_s^0 \to \mu^+\mu^-$ [20, 21], world-leading results on *CP* violation in beauty and charm hadrons, significant improvements in precision on Unitarity Triangle angles, and observations of new hadronic states of matter including pentaquarks [22]. LHCb has continued its successful operation during Run 2 (2015-18) thanks to the increased cross-sections, due to the higher collision energy, and the implementation of novel online data processing strategies. The original LHCb detector was designed to be able to collect 8 fb^{-1} at an instantaneous luminosity of up to 2×10^{32} $\text{cm}^{-2}\,\text{s}^{-1}$. By the end of Run 2 this target has been exceeded, with the majority of the data collected at 4×10^{32} $\text{cm}^{-2}\,\text{s}^{-1}$, and therefore in an environment with higher pile-up. Several anomalies recently reported by LHCb, including angular distribution in $B^0 \to K^{*0}\mu^+\mu^-$ [8, 23], branching fractions of $B \to X\mu^+\mu^-$ decays particularly with respect to $B \to Xe^+e^-$ and hints of lepton universality violation in charged-current weak interactions [9, 24], have led to speculation that a discovery of physics beyond the SM may be not far off. None of these measurements is individually above the 5σ threshold, so only a cautious interpretation can be made, however, even if these anomalies are not confirmed, they are a good example of the potential of flavour physics at Future Upgrades of the LHCb experiment.

In order to be able to continue the LHCb physics programme, a first LHCb upgrade

[1]The nomenclature for the Future Upgrades of the LHCb experiment is a little bit different and will be clarified later in the text.

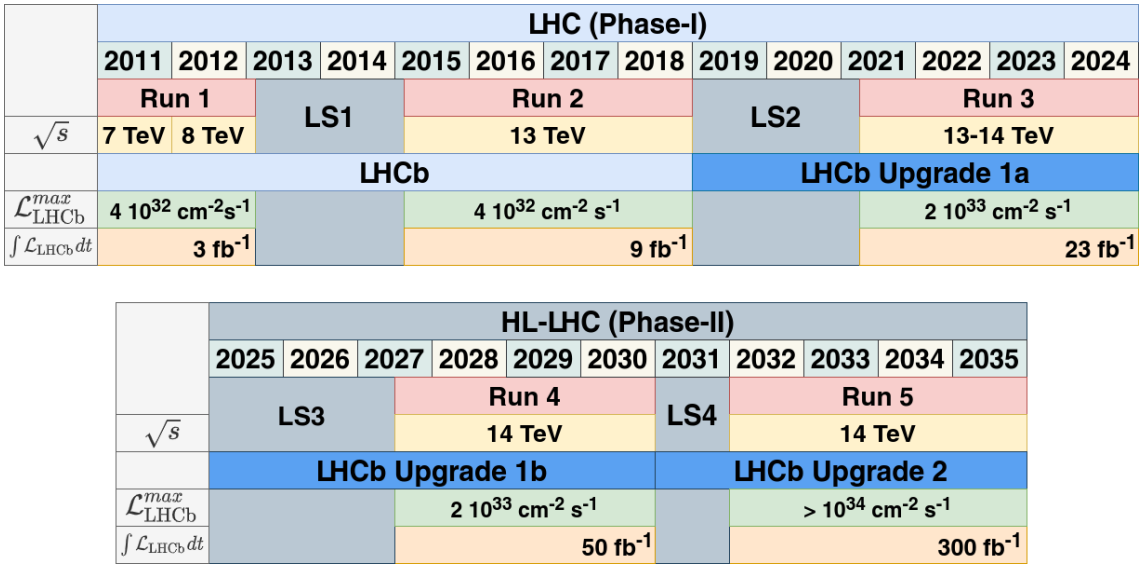


Figure 1.1: LHC roadmap. Phase-II includes runs with the so called High Luminosity LHC (HL-LHC), that will be installed during LS3. Integrated luminosities are cumulative [13].

was approved in 2012 [25]. The key concept is the upgrade of slow detectors, like the VErtex LOcator (VELO), that necessitated the usage of the hardware L0 trigger to reduce the rate at which they were read. The purpose is to read out the full detector at the average LHC bunch crossing rate of 30 MHz and implement the trigger decisions using all available information even at the earlier trigger stages. In this way it is possible to increase the luminosity without suffering of efficiency loss. By increasing the instantaneous luminosity by a factor of five, to 2×10^{33} cm^{-2} s^{-1} and improving the trigger efficiency for most modes by a factor of two, the annual yields in most channels will be an order of magnitude larger with respect to Run 2. Integrated luminosities of 23 fb^{-1} and 50 fb^{-1} by the end of Run 3 and Run 4 respectively are expected. The upgraded LHCb detector (the so called LHCb Upgrade 1a or Phase-I Upgrade) has been designed to meet these specifications. It will be installed during LHC Long Shutdown 2 (2019-20) and will start operation in 2021. During Run 3, it will be collecting data at the same time as the Belle II experiment [26] at the SuperKEKB asymmetric e^+e^- collider. Although it has a similar flavour physics programme to that of LHCb, the rather different collision environments lead to two complementary experiments. For final states composed of only charged tracks, LHCb will in general have much larger yields and lower backgrounds; Belle II on the other hand tends to have better capability for channels involving neutral particles or missing energy. LHCb can study all species of beauty hadrons, and the large boost and excellent vertexing allow B^0_s oscillations to be resolved; Belle II can exploit the $e^+e^- \to \tau^+\tau^-$ production mode to study all properties and decays of the τ lepton. Belle II is expected to collect 50 ab^{-1} around the $\Upsilon(4S)$ resonance by 2025, when it will conclude operation.

The precision on several important, theoretically clean, measurements will still be limited by statistics, and other observables associated with highly suppressed processes will be poorly known, at the end of LHC Run 4 (2027-2030). Therefore, the LHCb Collaboration is currently proposing an ambitious plan of Future Upgrades: a consolidation of the Phase-I

Upgrade in view of the LHC Run 4, the so-called Phase-Ib Upgrade [2], and for building a major Phase-II Upgrade, which will fully realize the flavour potential of the HL-LHC during the LHC Run 5 ($\geq$ 2031) at a luminosity $\mathcal{L} > 10^{34}\text{cm}^{-2}\text{s}^{-1}$ [12, 27]. The time schedule of the LHCb experiment is reported in Fig. 1.1 for clarity.

1.2 Physics opportunities

CP violation is the non-invariance of weak interactions under the joint application of charge conjugation (C) and parity (P). It was first discovered in 1964 through the observation of K_S^0 and K_L^0 mesons decaying in two or three pions [28]. This is a manifestation of the indirect *CP* violation, caused by the fact that mass eigenstates are not *CP* eigenstate. In 1999, it has been discovered the direct *CP* violation by NA48 and KTeV collaborations [29, 30], again in the kaon system, concerning the different decay amplitudes of *CP* conjugates states. Belle and BaBar collaborations discovered *CP* violation also in B mesons [31, 32] in 2001, specifically in the decay $B^0 \to J/\psi K_S^0$, caused by the interference of amplitudes of decay through $B^0 - \bar{B}^0$ mixing and that of direct decay. Recently, in 2019, *CP* violation has been discovered in D^0 mesons for the first time by the LHCb collaboration [4]. One of the requirements to explain the baryon asymmetry observed in the Universe, $\mathcal{O}(10^{-10})$, is the presence of *CP* violation in elementary interactions. Nonetheless, SM *CP* violation seems to be too small to explain the large baryon asymmetry of the Universe. Precision measurements of *CP* violation are interesting to study new physics beyond the SM because incompatibilities with the SM predicted *CP* asymmetries could mean the presence of new particles.

CP violation is encoded in the SM by means of the quark mixing matrix of weak charged current.

$$d \to W^- u,$$

where d and u are, respectively, down-type and up-type quarks, and W^- indicates the charged weak interaction gauge boson. To account for the differences in coupling strength between different families of quarks, the Cabibbo-Kobayashi-Maskawa mixing complex matrix has been introduced [15, 16]. It describes a transformation conventionally applied to down-type quarks

$$\begin{pmatrix} d' \\ s' \\ b' \end{pmatrix} = V_{\text{CKM}} \begin{pmatrix} d \\ s \\ b \end{pmatrix} = \begin{pmatrix} V_{ud} & V_{us} & V_{ub} \\ V_{cd} & V_{cs} & V_{cb} \\ V_{td} & V_{ts} & V_{tb} \end{pmatrix} \cdot \begin{pmatrix} d \\ s \\ b \end{pmatrix}.$$

The lagrangian term of weak interacting charged current is

$$\mathcal{L}_{int}^{cc} = -\frac{g_2}{\sqrt{2}} \cdot (\bar{u}, \bar{c}, \bar{t}) \cdot \frac{\gamma^\mu (1 - \gamma^5)}{2} \cdot V_{\text{CKM}} \cdot \begin{pmatrix} d \\ s \\ b \end{pmatrix} \cdot W_\mu^\dagger + \text{h.c.},$$

where g_2 is the coupling constant and W_μ is the field corresponding to the charged W boson. This theoretical structure allows for flavour mixing in weak decays of hadrons. Furthermore, the CKM matrix is not only a pure rotation but it has also a complex phase

[2] The LHCb-Upgrade in Run 3 is also named as Phase-Ia Upgrade.

responsible for the violation of the *CP* symmetry. This can be better understood if the $V_{\rm CKM}$ matrix is written using the "standard" parameterization:

$$V_{\rm CKM} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_{23} & s_{23} \\ 0 & -s_{23} & c_{23} \end{pmatrix} \cdot \begin{pmatrix} c_{13} & 0 & s_{13}e^{-i\delta} \\ 0 & 1 & 0 \\ -s_{13}e^{i\delta} & 0 & c_{13} \end{pmatrix} \cdot \begin{pmatrix} c_{12} & s_{12} & 0 \\ -s_{12} & c_{12} & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

where c_{ij} and s_{ij} are, respectively, cosine and sine of the three mixing angles θ_{ij} (note that θ_{12} is the Cabibbo angle), and δ is the *CP* violating complex phase. Another useful parameterization of the $V_{\rm CKM}$ matrix was introduced by L. Wolfenstein [33], that uses four parameters, λ, A, ρ and η, defined as:

$$\lambda = s_{12}$$

$$A\lambda^2 = s_{23}$$

$$A\lambda^3(\rho - i\eta) = s_{13}e^{-i\delta}.$$

The $V_{\rm CKM}$ matrix can be, therefore, written in terms of Wolfenstein parameters, up to the λ^3 order, as

$$\begin{pmatrix} 1 - \frac{\lambda^2}{2} & \lambda & A\lambda^3(\rho - i\eta) \\ -\lambda & 1 - \frac{\lambda^2}{2} & A\lambda^2 \\ A\lambda^3(\rho - i\eta) & -A\lambda^2 & 1 \end{pmatrix},$$

where the hierarchy of the different elements can be easily visualized, along with the position of the complex phase. The $V_{\rm CKM}$ matrix is unitary,

$$V_{\rm CKM} \cdot V_{\rm CKM}^{\dagger} = V_{\rm CKM}^{\dagger} \cdot V_{\rm CKM} = I,$$

this conditions translates into six normalization and six orthogonality relations, where the latters can be represented in the (ρ,η) complex plane as triangles. Only two of these triangles are non-squashed, with all sides of the same order of magnitude

$$V_{ud}V_{ub}^* + V_{cd}V_{cb}^* + V_{td}V_{tb}^* = 0 \quad \text{and} \quad V_{ud}V_{td}^* + V_{us}V_{ts}^* + V_{ub}V_{tb}^* = 0,$$

that are identical if only the λ^3 order is taken, and called unitarity triangle. It is a common practice to rescale the unitarity triangle by dividing the equation (let us take the first one) by its second additive term

$$\frac{V_{ud}V_{ub}^*}{V_{cd}V_{cb}^*} + 1 + \frac{V_{td}V_{tb}^*}{V_{cd}V_{cb}^*} = 0,$$

obtaining a final triangle where a unit length is assigned to one of its sides and place it onto the x-axis, as shown in Fig. 1.2. As mentioned above, at the order of λ^3, in the Wolfenstein parameterization the two non-degenerate orthogonality relations become identical

$$A\lambda^3[(\rho + i\eta) - 1 + (1 - \rho - i\eta)] = 0,$$

and the rescaled unitarity triangle is obtained by dividing by $A\lambda^3$. Figure 1.3 shows the triangle obtained using all the currently available experimental information, while Fig. 1.4 shows how it will appear at the end of the LHCb Phase-I and Phase-II, respectively. The fact that the apex of the triangle is not on the x-axis, i.e. the triangle is not degenerate, means that there is *CP* violation in the weak interaction.

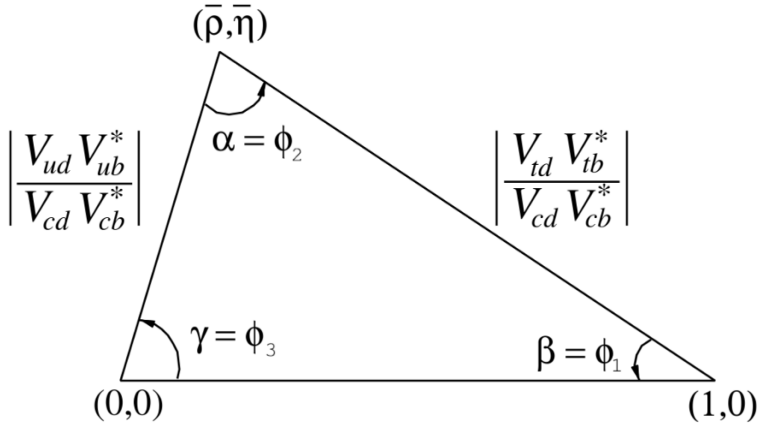


Figure 1.2: Rescaled unitary triangle.

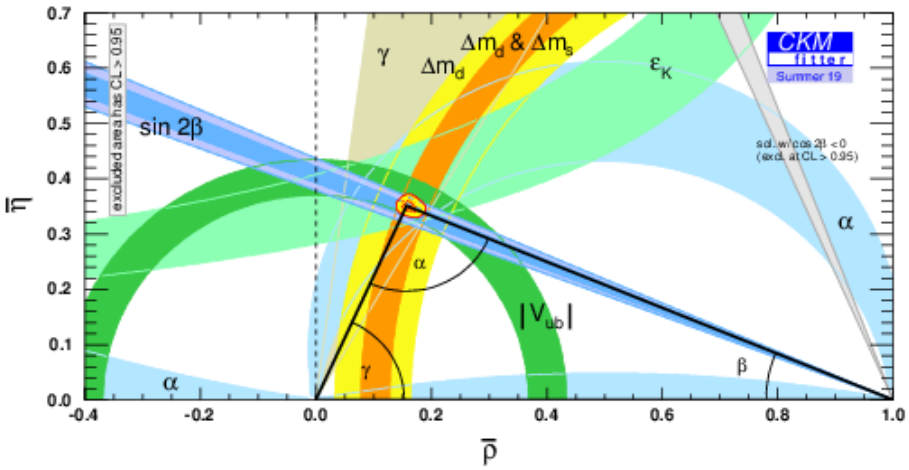


Figure 1.3: Rescaled unitary triangle and experimental constraints [34].

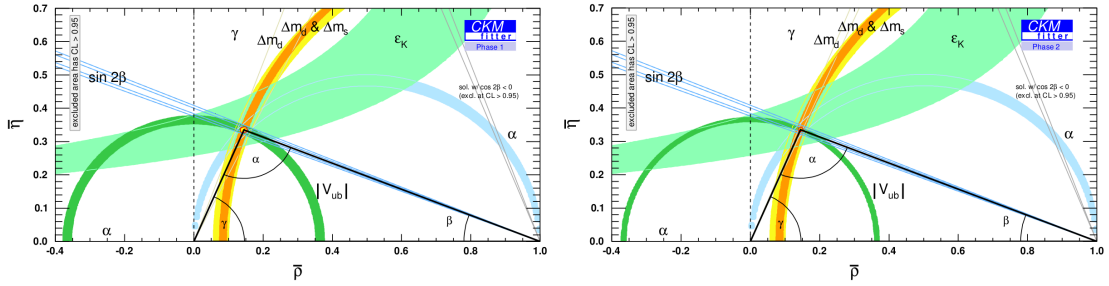


Figure 1.4: (Left) Expected rescaled unitary triangle at the end of LHC Phase-I. (Right) Expected rescaled unitary triangle at the end of LHC Phase-II [35].

The elements of the CKM matrix are closely linked to *CP* violation and many experimental efforts have been done to measure them with more and more precision, as well as those of the unitarity triangle. Nonetheless, they still remain poorly known if compared

with the high precision electroweak tests of the SM. A further effort is needed to better investigate such poorly known parameters in order to test the SM.

The SM does not predict the values of the $V_{\rm CKM}$ matrix elements, and so they must be experimentally measured. However, the unitary nature of the CKM matrix and the SM impose relations between them. The angle γ of the unitarity triangle, for instance, is of particular interest, because it can be determined extremely cleanly from measurements of *CP* asymmetries and relative rates in the $B^- \to Dh^-$ ($h = K, \pi$) family of decays. Its value is currently measured to be $\gamma = (72.1^{+5.4}_{-5.7})^\circ$ [34]. The theoretical uncertainty in predicting the value of this parameter in the SM is five orders of magnitude smaller than the attainable experimental precision at the end of the Phase-II period [13, 36]. For this reason, ever more precise measurements of γ will remain very well-motivated in the future, and will provide a reference determination of the apex of the unitarity triangle against which all other measurements of the triangle can be compared. In parallel to these studies of tree-level processes, it will be essential to improve the measurement of γ in channels where virtual loops play a significant role, for example $B^0_s \to K^0_S\pi^+\pi^-$, $B^0_s \to K^-\pi^+\pi^0$ and the family of $B \to h^+h^-$ modes.

The study of semileptonic $b \to sl^+l^-$ transitions, in which the dilepton pair is not produced in the decay of a hadronic resonance, offers a rich set of observables that probe and constrain [37] new physics models in a complementary way to the study of $B^0_s \to \mu^+\mu^-$. LHCb has already reported important results for many of these modes [23, 38–43], most notably $B^0 \to K^{(*)}\mu^+\mu^-$ and $B^0 \to K^{(*)}e^+e^-$. The main systematic uncertainties are expected to scale with integrated luminosity, motivating continued study of these decay channels at the Phase-II Upgrade. Since the analyses will be performed for both electron and muon final states, all measurements will also be naturally interpretable as tests of $e - \mu$ lepton universality.

LHCb will take full advantage of the enormous production rate of charm hadrons at the LHC, and collect the largest sample of charm decays ever recorded. During the Phase-II Upgrade, LHCb will collect over two orders of magnitude more $D^0 \to h^+h^-$ and $D^0 \to K^0_S h^+h^-$ decays ($h = \pi$, K), than Belle II, allowing it to probe indirect *CP* violation with a precision of 10^{-5} and measure charm-mixing parameters with a precision of 10^{-4}. *CP* violation in mixing-related phenomena is predicted to be very small, $\mathcal{O}(10^{-4})$ or less [44], and therefore improved measurements have excellent discovery potential for observing NP contributions. Effects of *CP* violation in decay are less cleanly predicted, and can be as large as $\mathcal{O}(10^{-3})$, as recently discovered in 2019 [4]. The Phase-II Upgrade will provide the opportunity to search for these effects in many other channels in order to shed light on the standard or non standard origin of the recent discovery. In addition, LHCb trigger will have the possibility to study a wide range of three- and four-body decay modes, along with a deep exploration of the charm-baryon sector.

LHCb has a unique reach for rare decays of strange hadrons, and has already produced world-best results in the search for $K^0_S \to \mu^+\mu^-$ [45, 46] and made studies of the decay $\Sigma^+ \to p\mu^+\mu^-$ [47]. The Phase-I/-II Upgrade will allow LHCb to observe $K^0_S \to \mu^+\mu^-$ down to its SM decay rate and make similarly sensitive measurements for the decays of other charged hadrons. The LHC is also an extremely rich laboratory for the study of exotic hadrons and this opportunity has been exploited by LHCb to great effect during Run 1 and Run 2. Highlights include the demonstration of the four-quark nature of the $Z_c(4430)^+$ resonance [48] and the observation of the pentaquark states [49]. The Phase-II Upgrade will have an unprecedented sensitivity for exotic hadron studies, whether produced

directly in pp collisions or in the decays of beauty hadrons where high-statistics amplitude analyses will allow the resonant nature of these states to be determined.

The Phase-II Upgrade (and the Phase-I) of LHCb will both enable a significant improvement in statistical reach for observables already under study, and will also allow complementary observables in highly suppressed processes to be measured with good precision for the first time. The wide programme of measurements will provide high sensitivity in the search for effects beyond the SM, and allow for a detailed characterization of any NP that is discovered.

1.3 TDAQ architecture considerations

The challenges of performing precision flavour physics at very high luminosities are daunting. The mean number of interactions in each event, foreseen for the LHCb Phase-II Upgrade, will be around 55 at a luminosity of 2×10^{34} cm^{-2} s^{-1}. The increased particle multiplicity and rates will present significant problems for all detectors, as will the increased radiation damage for certain components. A huge amount of R&D work is carried out for developing new technologies and new particle physics subdetectors (tracking, calorimeters, particle identification, etc.) to cope with this extreme crowded environment. In addition to these already difficult challenges, the data handling demands will achieve unprecedented levels, and as a consequence the trigger and data acquisition systems will be the major cost item and it may amount to a major technical limitation to the experiment performance. Last but not least, the scale of the needed offline computing resources is expected to grow significantly, much more than the expected growth in CPU, disk and network resources [14].

Trigger and Data Acquisition (TDAQ) are important elements of the technology required by the experimental study of particle physics. Most experiments use detectors producing large flows of data, sustained for periods from months to years, generally exceeding the capabilities of any practical permanent storage system. The word "Trigger" is used in a wide sense to indicate any methodology for regulating and reducing this flow of data to a manageable level, by deciding which parts of the detector information should be acquired and at what time. This function most often requires performing substantial computations at high speed, in order to select the most useful information. The word "DAQ" is generally used to indicate the whole process of carrying data from the detector to the permanent storage, often also including the flow to and from the trigger system.

TDAQ systems of some kind have always been part of particle physics experiments. While electronic technology has undergone huge advancements over the years, becoming cheaper and more powerful by large factors, the requirements of HEP experiments have been growing at an even faster rate. As a consequence, the TDAQ system still represents today a major cost item in modern experiments, and in the most data-hungry experiments, like hadronic collisions at high luminosity, it may be a major technical limitation to the experiment performance. Most modern TDAQ systems have a multi-layered structure, in which the decision of selecting a particular collision event is taken in a sequence of stages, with progressively decreasing data flow and increasing computational cost per event. The lowest-level trigger (generally called L1) decision is usually based on a low-complexity processing of a limited amount of data, within low latencies, while the last level before permanent storage is now universally performed by CPU systems running high-level code. All these features were already present in the pre-LHC generation of experiments, but in

moving to the LHC the overall DAQ bandwidth has been increased, taking advantage of the fast internet growth that pushed telecommunication technology quickly forward, allowing large data handling systems, based on commercial products, to be built at reasonable prices.

The first upgrade of the LHCb experiment is planned for an earlier timescale (LHCb-Upgrade in Run 3, 2021-2023 and in Run 4, 2027-2030). This is motivated by the need to move beyond the current TDAQ system, that has already reached plateau performance. LHCb is not pursuing an improved L1 trigger for this short-term upgrade since it already works at the highest L1 accept rate amongst all LHC experiments. The LHCb chosen approach has been to limit its operating luminosity to 2×10^{33} cm^{-2} s^{-1} so that data flow can be handled by the High Level Trigger (HLT) with no help from L1, which will be eliminated. The upgraded HLT will use a much bigger processor farm. The implication for the DAQ system is that it must be able to move the entire data flow of 40 Tbit/s from the front-end to the farm with no reduction. This is done by connecting the fiber optic readout on the FE directly to a system of about 500 PC nodes, each of them equipped with a FPGA card for data formatting, communicating with the PC with a 100 Gbit/s connection. These PCs perform the event building by exchanging data amongst themselves, and then each of them transfers the assembled events to the HLT farm at a 80 kHz event rate[3]. LHC is an enormous source of heavy-flavoured hadrons, that will continue for many years in the future with growing intensity. In the High-Luminosity phase planned for 2025 and beyond, at instantaneous luminosities of about 10^{35} cm^{-2} s^{-1}, the LHC will produce about 10^{14} beauty and 10^{15} charm hadrons per year. There is, therefore, a strong motivation for building a Phase-II Upgrade, which will fully realise the flavour potential of the HL-LHC during the LHC Run 5 ($\geq$ 2031) at a luminosity larger than 10^{34} cm^{-2} s^{-1} with the aim of collecting > 300 fb^{-1} of total integrated luminosity. The physics opportunities offered by such unprecedented sample call for a serious study of the feasibility of such an "extreme" flavour experiment. Given the above-mentioned limitations on data-processing, the key to such an experiment seems to be to perform the data analysis in real time, rather than off-line. Beam crossings at the LHC will occur at a frequency of 40 MHz (30 MHz of visible collisions), for each beam crossing many elementary pp collisions occur, resulting in thousands of tracks traversing tracking detectors. However, not all the collisions in a crossing are interesting for the studied processes, and, more importantly, not all the tracks in a collision event. If heavy-flavour decays of interest could be identified in real time, only the relevant part of the event will be recorded so that a large reduction in the amount of data to be transmitted and stored will be achieved.

In the following is assumed that the detector will produce an aggregate output data of the order of 0.5 PB/s. This estimate comes from scaling the already mentioned 40 Tbit/s output of the upgraded LHCb detectors by a factor of 100 for luminosity. A first challenge arises from the fact that, with current technologies, the resulting number of serial data links needed to read out the whole detector would be prohibitively huge. The fastest radiation-hard serialiser/deserialiser devices are the GigaBit Transceivers (GBTs), which are designed and built by the CERN micro-electronics team for the upgrades of the LHC experiments. These devices operate at a transmission rate of 4.8 Gbit/s, thus the number

[3]The described configuration was the baseline until very recently, when the Allen GPU project has become the new baseline for HLT1. Since Allen runs on GPUs installed in EB nodes, HLT1 will be performed in the EB farm, transmitting data to HLT2 at a much lower rate. This fact makes no difference for the purposes of this thesis.

of GBT links that would be required to read out the whole detector if running at 1 PB/s would be of the order of 2×10^6 while, for comparison, the readout system for the LHCb upgrade will require about 9×10^3 optical data links. This highlights the fact that such experiments need real time processing of the raw data at the lowest possible level ("on-detector") in order to reduce the data flow, based on more complex data treatment than simple zero-suppressed hits. Improvement in serial communication speed must also be achieved in order to reduce the number of optical links down to a manageable level.

The reconstruction of tracks in real time has been a distinctive feature of the most advanced trigger systems of the pre-LHC generation of experiments. However, the challenges to be overcome for the first LHC run have forced the experiments to concentrate on other aspects of the TDAQ system, like increase in dimensions and standardization, and specialized track reconstruction systems have disappeared. The development of an experiment capable of flavour data processing in the future HL-LHC runs requires sophisticated tracking capabilities at the earliest possible processing level. First, a reconstruction of the majority of tracks is required, down to the lowest p_T of interest and including all charged tracks. Second, the reconstruction must be of sufficient quality to be used for full physics analysis, allowing the raw data to be discarded.

1.4 The LHCb-Upgrade and the use of accelerators

The LHCb experiment already recorded an unprecedented dataset of beauty and charm hadron decays during Run 1 (2011-2012) and Run 2 (2015-2018) of the LHC. A key computing challenge was to store and process these datasets, because of the limitations of the LHCb trigger. In order to increase the size and the purity of the collected data samples (charm physics was particularly limited in Run 1 because of trigger output rate constraints) LHCb pionereed the real time analysis, with the implementation of a new streaming strategy (Turbo) including the possibility to perform the physics analysis with candidates reconstructed in the trigger, thus bypassing the offline reconstruction and discarding the raw event [50]. In the Turbo stream the trigger writes out a compact summary of physics objects containing all information necessary for analyses, and this allowed an increased output rate and thus higher average efficiencies and smaller selection biases. This idea was commissioned and developed during 2015, at the beginning of Run 2 data taking, with a selection of physics analyses. The turbo stream has been immediately after adopted by an increasing number of analyses during the remainder of Run 2, and ultimately will be the dominant stream for the upcoming Run 3 (starting in 2021) with the upgraded LHCb detector.

From this point of view the LHCb-Upgrade during LHC Run 3 represents an unique opportunity to face the challenge of the high luminosity, even before the start of the High Luminosity Phase-II of ATLAS and CMS experiments (scheduled in the LHC Run 4) to take the full advantage of the abundant production of charm and beauty hadrons and prepare the more challenging Phase-II Upgrade of the experiment (LHC Run 5). The reconstruction of tracks, and of more complex objects, in real time already is the most relevant feature of the LHCb trigger system, and the success of the current and the future upgrades of the experiment will crucially depend on the capability of moving sophisticated tracking algorithms at the earliest possible processing level. The rate at which data will be

produced already in Run 3 (40 Tbit/s) will be so high that the usage of heterogeneous computing systems has been considered as a viable and cost-effective solution nowadays, and not only for the Future Upgrades (Run 4 and 5). These systems are composed by a large farm of CPUs, versatile and general-purpose devices, and by FPGAs and GPUs (*Graphics Processing Unit*), that are specialized hardware, being very fast at executing simple and repetitive tasks, leaving to CPUs only the more complex ones. Parallel programming is the key feature to fully exploit their performances.

The LHCb experiment is, therefore, already facing the use of these emerging technologies, generally called *accelerators*. An important R&D line is the usage of GPUs in the TDAQ systems of HEP experiments. GPUs are a kind of commercial microprocessor specialized in the manipulation of digital graphics, recently used also for data processing. They are powerful and efficient devices available at affordable prices thanks to the thriving gaming industry. An important project about computing on GPUs is Allen [51], an HLT software designed to run on GPUs installed in the Event Builder PC server nodes, that has been recently adopted as baseline HLT1 option by the LHCb Collaboration. Many other R&D developments involve modern high-end FPGAs, including many projects on real time tracking, a computing intensive and parallelizable task. Some of them involve the use of neural network, with machine learning algorithms trained to perform tracking and whose final models are exported to run on FPGA, while others are specifically designed for tracking through a massive and intelligent pattern matching, as the LHCb-RETINA project [52], a parallel, high throughput, low latency tracking algorithm inspired to the natural vision of mammals brain. In this regard, the collaboration has decided to implement a parasitic testbed, already in Run 3, in which new architectures based on accelerators will be developed and optimized, by feeding them with a copy of real data coming from the readout boards. Also other LHC experiments are building systems based on hardware accelerators. For instance CMS is designing an FPGA tracker for Run 4 [53] and ATLAS will use an FPGA-based card for the new FELIX system, already in Run 3, as data router [54].

The work performed in this thesis has been developed in this context and aims at moving one of the most simple and repetitive task of the LHCb VELO pixel subdetector, the 2D cluster-finding, to the FPGA readout cards, freeing the HLT from the burden of this computation. The work is a relevant part of the RETINA project for the real time reconstruction of the VELO tracks at an event rate of 30 MHz, which includes the VELO clustering, as the first data preparation stage for the implementation of the full tracking algorithm.

Chapter 2

The LHCb-Upgrade experiment

This chapter briefly describes the Large Hadron Collider complex, along with the LHCb-Upgrade experiment, with a particular focus on the aspects that are the more relevant for the work described in the thesis, namely the new Vertex Locator (VELO) pixel subdetector and its readout system. Further information on the LHC and the LHCb-Upgrade experiment can be found in the references cited throughout the chapter.

2.1 The LHC accelerator

LHC (Large Hadron Collider) is the largest and most powerful accelerator in the world and is located at CERN (Conseil Européen pour la Récherche Nucleaire) in an underground tunnel, placed about 100 meters under the surface, approximately circular and long 27 km. It is a proton-proton and Pb ions collider that has four points of interaction where particles collide and where there are the main experiments: CMS, ATLAS, LHCb and ALICE. ATLAS and CMS are general purpose experiments, whose main aims are the direct search of new non-standard particles and the discovery of extra-dimensions and dark matter. The ALICE experiment is developed specifically for heavy ions collisions and studies the property of quark-gluon plasma, a state of strong-interacting matter at very high energy density, very similar to the condition of the universe just after the Big Bang. The LHCb experiment, which is the subject of this thesis, is instead a single-arm spectrometer specifically designed to study the properties and the rare decays of heavy quark hadrons (beauty and charm). There are also three minor experiments installed at the LHC: LHCf, that exploits particles escaped outside ATLAS at very narrow angles with respect to beam line to simulate cosmic rays; TOTEM, that shares the interaction point with CMS and that measures total, elastic and diffractive p-p cross section; MoEDAL, placed in LHCb cavern, searches for magnetic monopoles.

Inside LHC protons reach an energy of 6.5 TeV per beam, giving a total energy in the center-of-mass frame of 13 TeV; this energy cannot be reached directly by LHC but a series of acceleration stages is required and is provided by the older accelerators at CERN. Figure 2.1 shows a map of CERN accelerating facilities and major experiments. The first stage of acceleration is under improvement in the LS2 (Long Shutdown 2, 2019-2020). In the previous runs LINAC2 was the first accelerator of the chain, creating proton beams from hydrogen molecules (contained in a gas bottle) to which an electric field strips off the electron and accelerating them to an energy of 50 MeV. To increase the luminosity

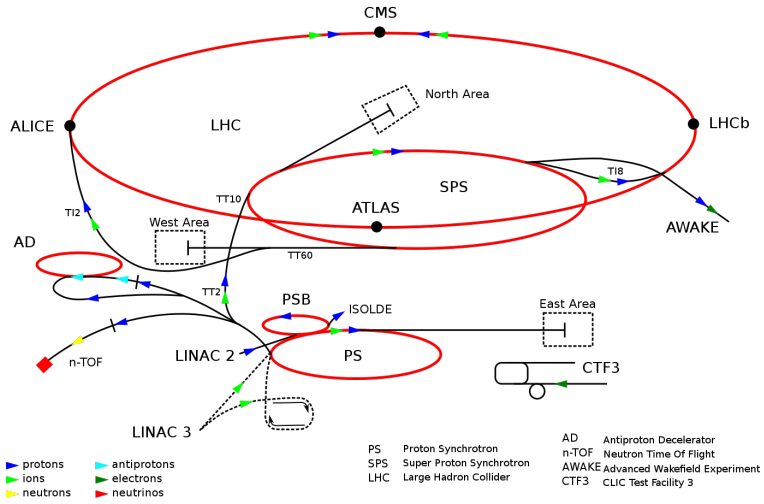


Figure 2.1: Accelerator complex at CERN showing major experiments and type of particles accelerated.

of proton beams the new LINAC4 is required as proton source; it accelerates H^- ions to 160 MeV and the two electrons are stripped off only at the moment of the injection in the following accelerator. LINAC2 and LINAC4 injects their beams in the Proton Synchrotron Booster, a storage ring that accelerates beams up to 1.4 GeV before injecting them in the Proton Synchrotron (PS). PS has been the first synchrotron at CERN and now is used to accelerate protons up to 25 GeV before injection in the Super Proton Synchrotron (SPS) that boosts them to 450 GeV when they eventually are sent to LHC.

The proton beams run in two separate pipes kept at an ultra-high vacuum consisting of a pressure of 10^{-13} bar. A number of 1 232 superconducting dipole magnets (15 m long) bends protons direction with a magnetic field of 8.3 T; a cryogenic system is required to maintain their superconducting properties, keeping them at a temperature of 1.9 K. Quadrupoles magnets are also important in all accelerators for squeezing particles in a tight beam before collisions. Particles are accelerated by means of radiofrequency cavities. The electromagnetic field inside cavities locates virtual regions on LHC circumference, called buckets, in which particles are kept during the runs of the accelerator. A particle exactly synchronised with the radiofrequency is called synchronous particle. Instead of being spread uniformly around the circumference of the accelerator, the particles get "clumped" around the synchronous particle in a bunch. Not all LHC buckets need to be filled with bunches (only 2808 out of 3557 buckets are filled in LHC). Each bucket can hold a bunch of protons or can be empty. The "abort gap" is a number of buckets in a row which are supposed to be never loaded with protons and form a gap in the circumference. The purpose of this gap is that in the dump process it takes a short, but significant time to switch on the magnets which divert the beam from the LHC into the dump [55].

2.2 The LHCb-Upgrade experiment

The LHCb (Large Hadron Collider Beauty) experiment is a single-arm spectrometer designed to study hadrons containing heavy quarks. Its geometrical acceptance covers only

the forward direction, since heavy quarks are produced at high pseudorapidity. During the current Long Shutdown 2 (LS2) the LHCb experiment is being replaced by an upgraded experiment, referred as the Phase-I Upgrade (or the so-called LHCb-Upgrade). In order to read out the experiment at the crossing rate of the LHC (30 MHz of visible collisions) and to operate at a higher luminosity of 2×10^{33} $\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$, most of the sub-detectors will be replaced, as will all of the front-end electronics and data-acquisition system. The increase of luminosity is achieved also with a better LHC filling, leading to a 25 ns time between consecutive collisions. Collision rate is therefore 40 MHz but, since there are empty buckets, the average rate of visible collisions is about 30 MHz. The average number of primary interactions per collision is estimated to be 7.6. A new trigger system will allow the experiment to function effectively at the higher luminosity, by accepting the full LHC crossing rate of 30 MHz instead of 1.1 MHz of Run 2, and will provide significantly increased efficiency, particularly for hadronic final states. By the end of LHC Run 4, in 2030, the experiment will have accumulated a data sample of around 50 fb^{-1}.

LHC Run	1	2	3	4	5
Years	2010–12	2015–18	2021–24	2027–30	2032–35
LHCb Phase			Ia	Ib	II
E_{cm} (TeV)	$7-8$	13	14	14	14
LHC $\mathcal{L}_{\mathrm{peak}}$ ($\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$)	$7.7 \cdot 10^{33}$	$1.7 \cdot 10^{34}$	$2 \cdot 10^{34}$	$7 \cdot 10^{34}$	$7 \cdot 10^{34}$
LHCb $\mathcal{L}_{\mathrm{peak}}$ ($\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$)	$2-4 \cdot 10^{32}$	$2-4 \cdot 10^{32}$	$2 \cdot 10^{33}$	$2 \cdot 10^{33}$	$> 10^{34}$

Table 2.1: LHC parameters of pp runs from 2010 to 2035. The LHCb experiment is not limited by the number of $c\bar{c}$ and $b\bar{b}$ produced in the pp interactions but by the trigger and reconstruction efficiencies and by the amount of data that can be saved to be analysed. LHCb is, therefore, able to limit the luminosity level by shifting the colliding beams in the plane transverse to their direction, thus reducing their overlap at the collision region. Thanks to this choice, the number of pp interactions per bunch crossing can be kept lower than general purpose experiments as ATLAS and CMS, limiting the detector occupancy and facilitating the trigger selection and reconstruction.

The nominal LHC luminosity at LHCb interaction point during the LHC Run 3 and Run 4 is planned to be 2×10^{33} $\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$, as mentioned above. Note that luminosity at LHCb is smaller than the accelerator nominal luminosity. This is obtained with collisions of bunches displaced apart transversely. A technique called luminosity-leveling is also used, in which bunch displacement is reduced with time as beam current naturally decreases with collisions, in order to maintain the same luminosity. Table 2.1 reports both LHC and LHCb peak luminosities during the different LHC runs.

Almost all of the LHCb subdetectors are being replaced with new and more performant ones for the beginning of the Run 3 during the current LS2. The upgraded experiment includes a completely new tracking system made of the silicon pixel VErtex LOcator (VELO), the silicon strips upstream tracker (UT) and the scintillating fibers (SciFi). A dipole magnet with a bending power of 4 T·m is placed between the UT and SciFi, allowing the measurement of momentum. Two Ring Imaging Cherenkov detectors, RICH1 before the magnet and RICH2 after, allows the distinction of different charged hadrons. The Scintillating Pad Detector and the Preshower, used until Run 2 for the L0 trigger, has

been removed since L0 trigger is no more used. Electromagnetic and hadronic calorimeters measures energy and helps to distinguish photons, electrons and hadrons. Muons are identified by the muon stations, the detector further from interaction point. A schematic drawing of the LHCb-Upgrade detector, operating during the LHC Run 3 and 4 is shown in Fig. 2.2. The LHCb coordinate system is righthanded, the z axis is along the beam pipe going from the VELO to muon detectors, x axis is horizontal going in the opposite direction to the center of LHC, y axis is vertical upward.

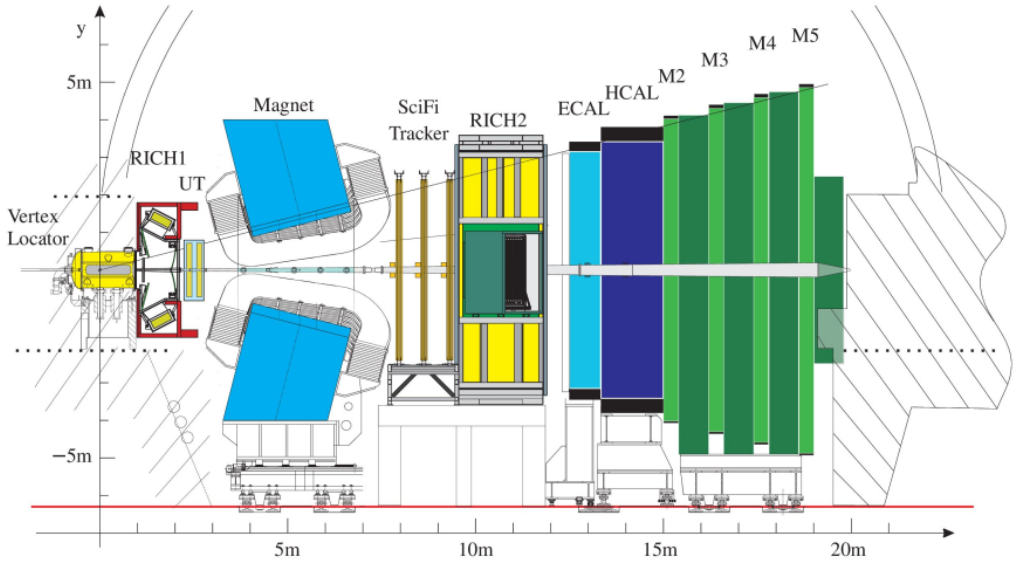


Figure 2.2: Layout of the Run 3 and Run 4 LHCb experiment, the so-called LHCb Upgrade experiment.

The work described in this thesis is mainly focused on the tracking system, and in particular on the new VELO (Vertex Locator) subdetector. The tracking system is the most important part of the experiment because it allows the reconstruction of charged particles and a precise measurement of their momentum. The VELO, in particular, is crucial also for the primary vertex reconstruction, and for the detection of secondary decay vertices. Next sections are, therefore, devoted to provide an overview of such a tracking system, and in particular to the VELO subdetector and its readout system.

2.3 The LHCb upgraded VELO

The new VELO has to be capable to operate at the LHC crossing-rate of 40 MHz and must be sufficiently radiation-hard to bear the conditions of Run 3 and Run 4. The collaboration opted for a silicon pixel detector, with radiation-hard readout ASICs called VeloPix and an evaporative CO_2 cooling system able to absorb the heat produced by the detector itself and by radiation exposure. The VELO is made of 26 *stations*, 19 in the forward region and 7 in the backward region with respect to the interaction point; each of them comprises two *modules*, one on the left hand side of the detector, the other on the right hand side; the LHC beam pipe is placed between the two modules. Figure 2.3 shows a diagram of

spatial arrangement of VELO stations and the layout of a single module in closed and open configurations. A module contains channels for coolant flow, in addition to electronic support for ASICs and the bias voltage for sensors. Upon each module four sensors are installed, two on each side; see Fig. 2.4 for the layout and numbering scheme of sensors on each module. Each sensor is made of three 256×256 pixel matrices bump-bonded to the three underlying VeloPix ASICs. Pixels are square with a surface of 55×55 μm^2, giving a raw hit resolution varying from 9 to 15 μm, depending on the angle of the particle [10]. All the components of the VELO have been optimized in order to reduce the material that particles have to pass through, given that every component falls, at least partially, under the acceptance.

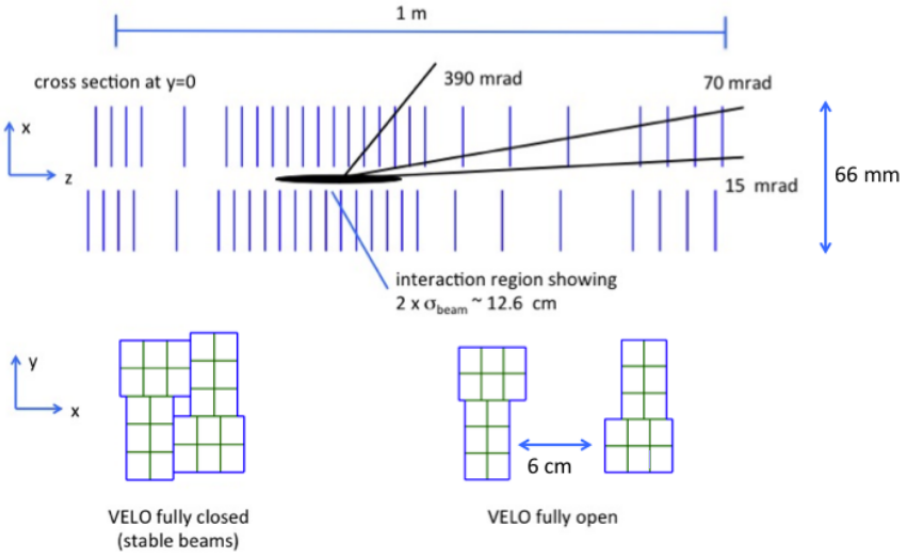


Figure 2.3: (Top) Spatial distribution of VELO stations. (Bottom) Layout of VELO modules in closed and open configurations.

Like the Run 2 VELO, it will be placed inside a vacuum tank that is directly connected to the LHC beam pipe. The tank is cylinder shaped, with a diameter of 1.1 m and a length of 1.4 m, coaxial with the beam pipe, that provides the vacuum conditions necessary for beam operation, as well as the space for housing the VELO modules. Other components inside the vacuum tank are the RF foils, aluminum-magnesium alloy boxes that separates the VELO regions from the beam region, in which a much more strong vacuum is required. Their purpose is also to isolate the VELO modules from the image current of the beams and the electromagnetic interference due to the discontinuous nature of particle bunches. An RF foil can be described as a rectangular box of dimensions 100 cm $\times$ 20 cm $\times$ 40 cm, of which one of the long faces is absent and the opposite face is corrugated, with steps corresponding to the positions of VELO modules. The thickness required for the corrugated face is less than 250 μm, in order to reduce multiple scattering of particles to be detected, and its structure closely matches that of the other detector half. The RF foil must be as close as possible to the beam pipe in order to minimize the distance between the particle beam and the VELO modules, because this is a key element for enhancing vertex resolution; the achieved distance for the new VELO is 5.1 mm, to be compared with 8.2 mm of the old detector.

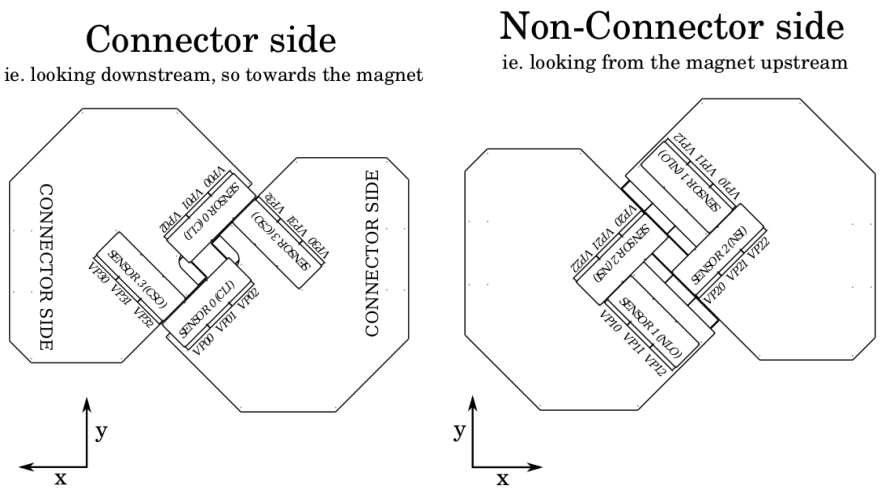


Figure 2.4: Look of one VELO station: on the left the two sensors on the connector side (looking towards the magnet) installed on each module, numbered 0 and 3; on the right the two sensors on the other side of each module, numbered 1 and 2.

Similarly to Run 2 VELO, during LHC fill cycles the modules of the VELO are pulled away from the beam line by a mechanical system, in order to prevent accidental damage due to protons escaping from the beams; when the accelerator is ready they are restored in their position for data taking, closer to the beam pipe.

2.4 The Upstream Tracker

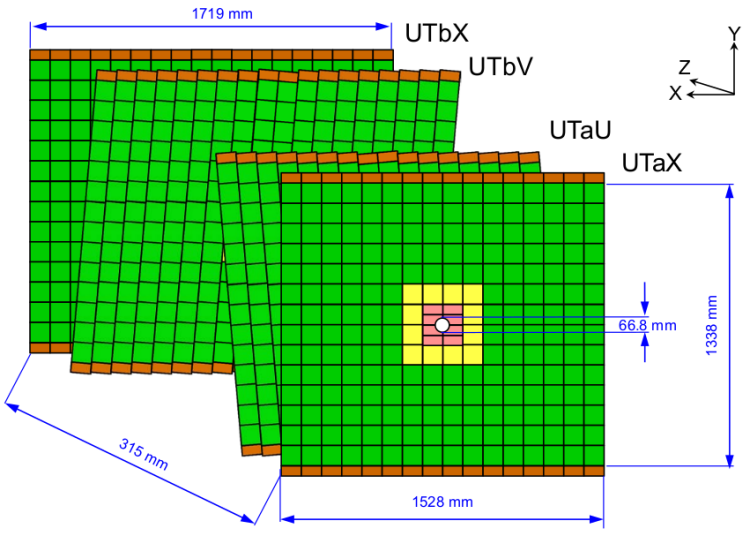


Figure 2.5: Layout of the UT. Green region has 190 µm width 10 cm length strips. Pink region has half-width half-length strips, while yellow region has full-length half-width ones.

The Upstream Tracker (UT) is the replacement for the old Tracker Turicensis (TT). It

is located upstream the dipole magnet and its purpose is to detect low-momentum particles that are swept out by the magnet. It is a silicon detector made of four layers in *x-u-v-x* arrangement: the first and the last have vertical staves and the two central silicon sensors have staves tilted by $\pm 5^\circ$ in the xy plane. Each layer has a stave on each side of the frame structure. These staves contains 14 square silicon sensors on each side, about 10×10 cm in dimension and 250 μm thick, and corresponding read-out ASICs. Sensors of the two sides are staggered in order to cover the whole space in the $x - y$ plane without holes. Such sensors are made of 512 silicon strips each, with a pitch of 190 μm and 97.28 mm in length. In the innermost region there are sensors with full-length half-pitch strips and half-length half-pitch ones (yellow and pink in Fig. 2.5). The two upstream layers contains 16 staves, the others 18. The hole in the center is a housing for beam pipe. With respect to the older TT, the new UT has half-thick sensors and sensors pitch varying from 95 μm to 190 μm, while TT had only strips 183 μm pitch.

2.5 Dipole magnet

The magnet allows to measure the momentum of particles with an integrated magnetic field of about 4 T$\cdot$m. It is made of two identical coils disposed as shown in Fig. 2.6. The region involved in the magnetic field goes from $z \approx 3$ m to $z \approx 8$ m but considering the fringing field it covers from $z \approx 0$ m to $z \approx 10$ m. Each coil is made of 15 low-carbon steel plates 10 cm thick installed on an iron yoke. It dissipates 4.2 MW of electric power with a current of 5.85 kA in normal operating condition. Coil current is periodically inverted with the intent to reduce systematical errors in precision *CP* measurement. 180 Hall probes detect the magnetic field in the entire tracking region with a relative resolution of 4×10^{-4}.

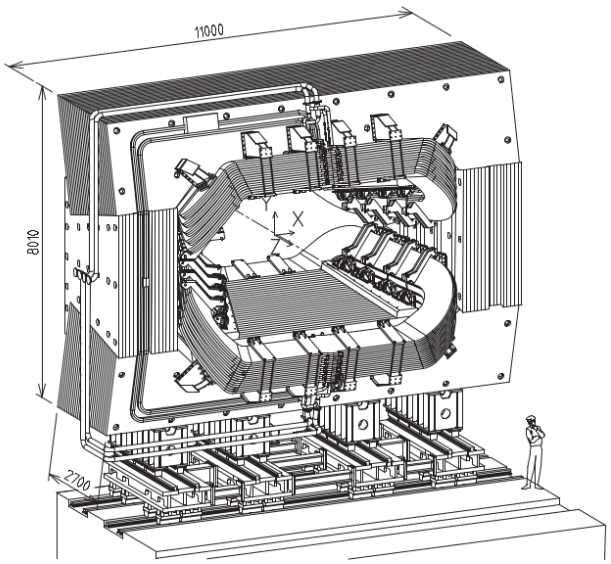


Figure 2.6: 3D view of the dipole magnet.

2.6 Scintillating Fiber detector

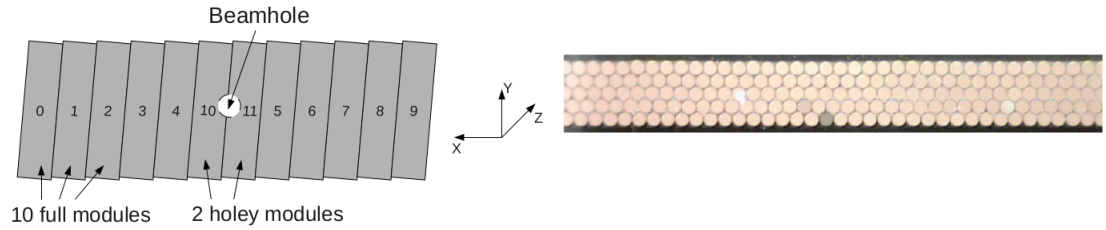


Figure 2.7: (Left) Layout of the 12 modules making one SciFi layer. (Right) Section view of the scintillating fibers of one SciFi module.

The Scintillating Fiber (SciFi) detector is downstream the magnet. It's the replacement for both IT and OT. Similarly to the old OT it will consist of three stations, called *T-stations*, each made of four layers in the usual *x-u-v-x* configuration (see section 2.4). Each layer is made of 12 modules 5 m high and 52 cm wide. The two modules surrounding the beam pipe have a hole to accommodate it and contain six fiber layers. The remaining modules have five fiber layers because of lower radiation exposure. Each scintillating fiber has circular section 250 µm in diameter and are made of a polymer core with the addition of 1% in weight of fluorescent dye. Photons are produced by excitation of polymer core and are propagated internally by total reflection to SiPM, at a speed of 6 ns/ cm. There is a 3 mm gap between each module, leading to an expected inefficiency of 1%. The simulated hit efficiency of the detector at the end of its lifetime is estimated to be above 97%. A picture of SciFi and a section view of fibers are shown in Fig. 2.7. Raw hit resolution is 42 µm [27].

2.7 Tracking in LHCb

Tracking is the process of linking hits of consecutive detector layers to reconstruct the paths of particles. In LHCb software, a track is a data structure containing a series of vectors called *track states*. A track state is related to a z_i position and is defined as

$$\overrightarrow{S_i} = (x, y, t_x, t_y, q/p)$$

where x and y are the coordinates of the track, t_x and t_y are the slopes in the $x - z$ and $y - z$ longitudinal plane respectively,

$$t_x = \frac{\partial x}{\partial z} \quad \text{and} \quad t_y = \frac{\partial y}{\partial z},$$

q is the charge of the particle and p its momentum. Track uncertainty is expressed by the corresponding 5×5 covariance matrix. Tracks are divided into different types depending on the subdetectors in which they are reconstructed:

- **long track**: a track reconstructed both in VELO and SciFi (or T-stations) subdetectors;
- **upstream track**: a track reconstructed both in VELO and UT subdetectors;

- **downstream track**: a track reconstructed on UT and SciFi (or T-stations) subdetectors;
- **T-track**: a track reconstructed with the information of the SciFi only;
- **VELO-track**: a track reconstructed with the information of the VELO only.

Figure 2.8 shows a representation of this track classification. The most valuable tracks for physics analysis are the so-called long tracks which are reconstructed in the VELO and the T-stations. They have excellent spatial resolution close to the primary interaction and a precise momentum information due to the combined information of the track slope before and after the magnet. T-tracks are not used in physics analyses, but they are used as inputs to reconstruct downstream tracks, important for the reconstruction of the daughters of long-lived particles such as K_S^0 mesons or Λ baryons which decay outside the VELO.

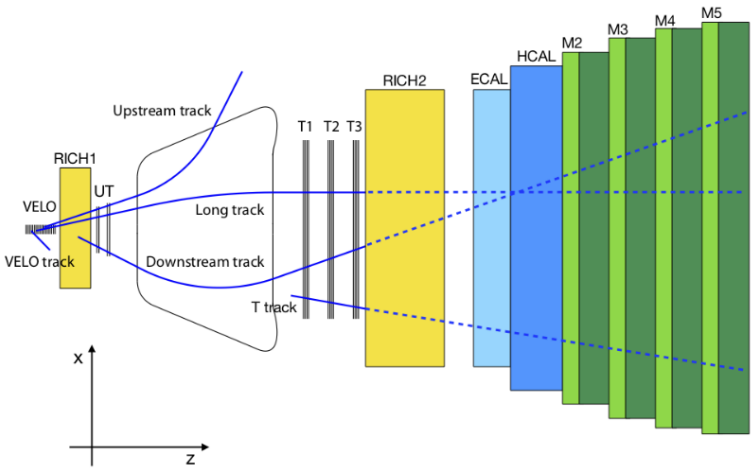


Figure 2.8: All the five track types considered in LHCb tracking. Long tracks are the most useful in data analysis since they leave information in all detectors.

The reconstruction of tracks in the VELO is performed by the `PrPixelTracking` algorithm. Given that the magnetic field in the VELO is almost negligible, the tracking is done by simply searching for straight lines in both $x-z$ and $y-z$ planes. It starts looking for a pair of unused clusters in two consecutive stations that makes slopes less than 0.4 in absolute value, for both longitudinal planes, and continues looking for a compatible cluster in the adjacent station, considering a maximum scattering angle. The algorithm searches in the upstream direction. The minimum number of hits per track is three, in which case all of them must be "not shared", i.e. belonging only to that track. In case of more than three hits, "shared" hits must be less than 50%.

`PrForwardTracking` is the main one of the two algorithms responsible for tracking of long tracks. It uses the Hough transform, with VELO or upstream tracks as input. This consists in projecting hits of a VELO track on a reference plane and searching local clusters of projected hits [56]. The other way to perform long tracking is with the matching algorithm, that takes as inputs VELO tracks and T-tracks, extrapolates them to the focal plane and searches for matching couples. The focal plane derives from an approximation of the magnetic field bending action, in which it is approximated as a kick in a given z

position, identifying the focal plane. UT hits are also used for long tracking, this gives a better momentum resolution, reduces significantly the ghost rate[1] and speeds up the `PrForwardTracking` algorithm providing it a preliminary estimate of momentum [11].

Tracking in the SciFi is performed by the *seeding* (or also called *hybrid seeding*) algorithm. At first it searches for tracks in the $x - z$ plane, then u and v hits are added, giving information on the $y - z$ slope. T-tracks found by this algorithm are used by the matching algorithm for long tracks reconstruction and as a starting point for downstream tracking.

Downstream tracking is, indeed, performed starting from T-tracks and extrapolating them to the UT, searching for matching hits. This is used for tracking of daughters of neutral long-living particles. Upstream tracking is performed adding UT hits to already found VELO tracks. VELO tracks are extrapolated through the UT and hits inside a tolerance window are a track candidate. Since a small fringing field is present in the UT, the tolerance window defines the minimum momentum cut. A track candidate must have at least three hits, no more than one per UT layer. A single VELO track can have more UT track candidates, one will be selected on the basis of the χ^2 of a simple fit and the number of hits. Upstream tracking is used to identify and measure the momentum of slow particles that are swept out of LHCb acceptance from the magnet.

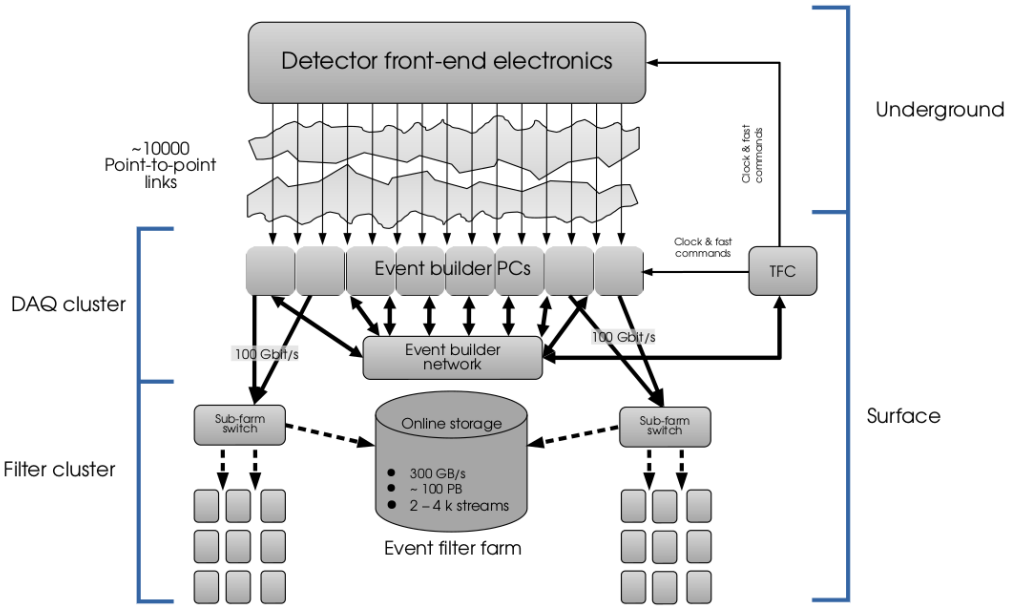
The algorithms described above are *tracking finding* ones, i.e. algorithms to combine hits of different layers to form a track. Tracks are then fitted using a Kalman filter approach [57, 58]. This gives the best estimate of track parameters and their covariance matrix. The tracking system provides a measurement of momentum of charged particles, p, with a relative uncertainty that varies from 0.6% at 10 GeV/c to 0.9% at 50 GeV/c. The minimum distance of a track to a PV, the impact parameter, is measured with a resolution of $(12.5 + 13.5/p_{\mathrm{T}})$ µm, where p_{T} is the component of the momentum transverse to the beam, in GeV/c. See Chap. 5 for more details about performances of the LHCb experiment.

2.8 The LHCb upgraded readout and trigger

The upgraded readout system is designed to run at LHC bunch crossing rate of 30 MHz (visible collisions), and the new trigger system will process such an high input rate and it will be able to take decisions employing information from all subdetectors, including raw hits from tracking subdetectors, as VELO, UT and SciFi. This would allow to implement "upfront" selections based on a precise measurement of the momentum and of the impact parameter of the reconstructed tracks, in addition to the exploitation of the information from muons subdetectors and calorimeters (already available upfront during the LHCb Run 1 and Run 2). The downside of this approach is the difficulty of dealing with much larger data rates, and, therefore, the new readout system (along with the new trigger system), is the most important upgrade of the experiment. A schematic diagram of the architecture of the Run 3 data acquisition and trigger system is shown in Fig. 2.9, along with the organization of the trigger levels in Run 3 and bandwidth involved.

The new trigger will be executed by the Event Filter Farm, a computer cluster specifically designed for this purpose. This is divided in two stages named HLT1 and HLT2, where the

[1] A ghost track is a reconstructed track that shares less than 70% of hits with one MC track, in at least one of the three tracking detectors.



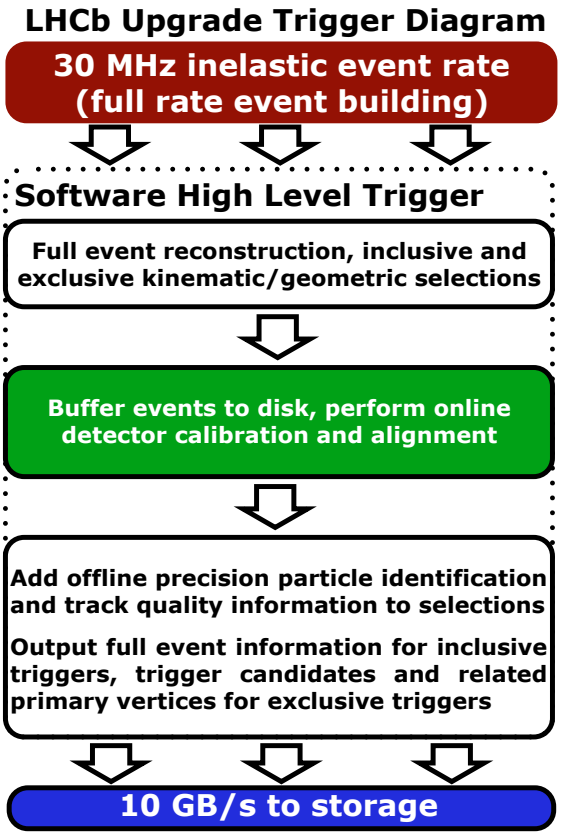


Figure 2.9: Run 3 data acquisition and trigger architecture.

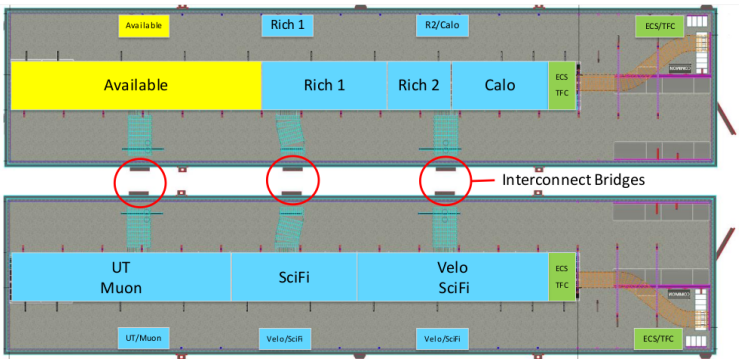


Figure 2.10: Distribution of detector data lines to EB nodes. In each region servers read data from a specific detector.

first performs the simpler and basic selections to reduce data rate by a factor of about 30, while the latter executes more sophisticated and complex algorithms analysis. Between the two stages, data are stored in disk buffers and are used to effectuate detector calibrations and alignments. The trigger output will have an offline quality (Turbo stream [59]) and no further offline reprocessing will be performed for the majority of the charm and beauty reconstructed hadron decays.

Another computer cluster is necessary to perform the *Event Building* and is thus called *Event Builder* (EB). As can be seen in Fig. 2.10, each sub-detector is assigned to a given amount of EB nodes so that an event is read by a large number of nodes, and therefore each of them will own only a little amount of the data relative to that event. Hence an exchange of data is necessary in order to transfer all the information of an event to one EB node; at that point the event is considered reconstructed and can be analyzed. To perform this data exchange the Event Builder nodes in the farm must be connected to each other by means of a network, called Event Builder Network. The detector front-end performs *zero suppression* and sends data to EB farm, located on earth surface, through 300 m long optical fibers with Versatile Link technology. Each node has a PCI board through which it receives data; thanks to PCI express gen3 technology the board can directly write data to computer memory via DMA (Direct Memory Access) at a rate of up to 100 Gbit/s. This design has the advantages of placing the readout boards directly inside the computer, minimizing data links length, and using the node's memory, thus removing the need of on-board memory. A total of about 12 000 links is required, with 24 links per board, resulting in a number of 500 boards. Each link will transfer at most 112 bit per bunch crossing occurring at an average of 30 MHz, giving a total bandwidth of $112 \times 12\,000 \times 30 \cdot 10^6 \simeq 40$ Tbit/s, so $40\,000 \div 500 = 80$ Gbit/s, less than the available bandwidth.

The readout board to be used is the PCIe40, a custom built board equipped with an Arria-10 FPGA, one of the most powerful available nowadays on market. Figure 2.11 shows a diagram of data path inside a farm node. Each node consists of a server-PC with two CPU sockets, each with its own memory bank array. Each socket is equipped with a PCIe40 board, an Infiniband network card and an Ethernet network card. While the purpose of the first is to receive raw data from detectors, the second is needed to connect EB nodes each other in the EB network and the third for communication between EB

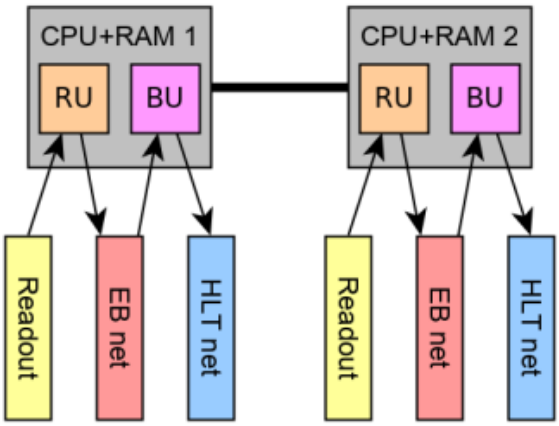


Figure 2.11: Diagram of data flow inside an EB node and its architecture. RU and BU are respectively the readout-unit and building-unit.

and the Event Filter Farm. Data path inside the Event Builder begins with the reading process, performed by PCIe40 readout board, of data sent by detector front-end with Versatile Link optical fibers; the FPGA of which it is equipped temporarily orders data, performs preliminary data processing and finally writes them on server memory with DMA. Subsequently, if necessary, data are sent via EB network, implemented with Infiniband technology, to the node designed to collect all data of that event, then when the event is complete it is sent to the Event Filter Farm via the Ethernet based HLT network. The two CPUs of each node will run the readout-unit (RU) and building-unit (BU) processes, whose purpose is respectively to manage data transfer from PCIe40 to server memory and from server memory to EB network, and perform event building. In particular, one of the tasks of RU is to collect received data in buffers before sending them on EB network. Since a single data fragment is too little for transmission, about 1 000 data fragments are sent together.

Very recently, the LHCb collaboration decided to adopt a GPU-based implementation for the first stage of the HLT trigger as baseline for the Run 3. This project, called Allen [51], foresees to run the HLT1 trigger on PCIe boards equipped with GPUs, installed in the EB nodes, leaving to the event filter farm only the task of executing on CPUs the HLT2 stage. This choice allows to drastically reduce the number of links required to move data from the EB to the event filter farm since data transmitted are only the output of the HLT1 stage. The insertion of the GPUs in the EB nodes drastically changes the baseline configuration of the event builder and of the trigger system, as described in Ref. [27] and in Fig. 2.9. Nonetheless, such modifications have no impact on the work described in this thesis since it deals with the software of the readout PCIe40 cards, whose implementation does not depend on the configuration of the other boards installed in the EB nodes, and on the specific HLT1 implementation.

2.8.1 The PCIe40 board

The PCIe40 board is a custom built board made specifically by the Collaboration [27] as interface between front-end electronics and server memory. It is equipped with an

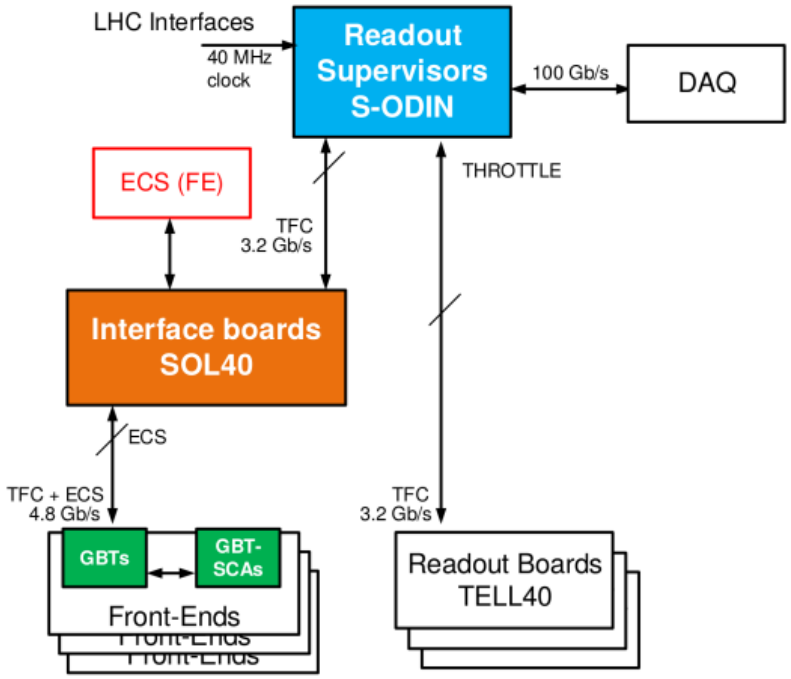


Figure 2.12: TFC and ECS system logical architecture, showing supervisor S-ODIN, SOL40 and TELL40 firmware in their respective roles.

Arria-10 FPGA (mod. 10AX115S4F45E3SG), one of the most powerful FPGA on market with 1.2 millions of logic elements, 2 713 M20K memory cells (20 kbit each) and 20 774 of MLAB memory cells (640 bit each). Since an FPGA can be freely re-programmed, one board can not only be used for data acquisition, but also for other tasks vital for the experiment such as *Timing and Fast Control* (TFC) and *Experiment Control System* (ECS) management. TFC is the system responsible for the distribution of clocks, timing and trigger information and commands to FE and readout. In details, the information it propagates are the following:

- 40 MHz clock synchronized to LHC master clock;
- synchronous commands to control event processing in FE and readout;
- calibration commands for detectors;
- distribution to FE of ECS-generated electronics configuration.

The ECS is the system used to propagate controls from control room to all of the electronics in the experiment. Figure 2.12 shows a diagram of TFC and ECS architecture.

Three PCIe40 flavours exist. They are three different firmware that can be used to program the FPGA. The S-ODIN is the readout supervisor and centrally manages the readout process of a partition of sub-detector, to which it's associated. Each S-ODIN board corresponds to an array of SOL40 and TELL40 boards which it drives. The purpose of a SOL40 board is to propagate clocks, fast and slow control signals from the S-ODIN supervisor and ECS from the experiment control room to its associated array of TELL40 and front-end boards. TELL40 is the firmware for readout boards.

From the point of view of the hardware, in addition to the FPGA, one PCIe40 is equipped with 48 bidirectional optical links rated 10 Gbit/s each, an SFP+ optical connector for TFC signals and a PLX chip as interface between the FPGA and PCIe connector. The layout described can be seen in Fig. 2.13. The purpose of the PLX switch is to merge two x8 links from the FPGA pins to make a single x16 PCIe link. The optical links are connected to the external side of the server thanks to 8 MPO connectors.

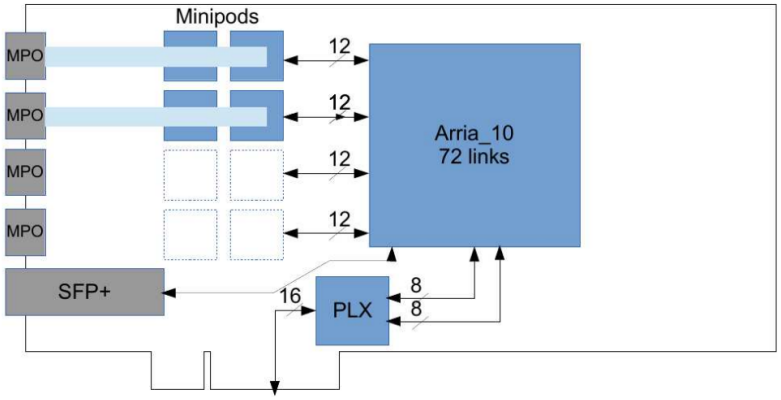


Figure 2.13: Scheme of the hardware equipment of a PCIe40 board.

DMA allows the board to directly write data into server memory. To fully exploit this feature of PCIe v3 the implementation of a DMA controller inside the FPGA firmware is necessary. A set of buffers is used to store data and are flushed when full, thus optimizing transfer speed. As soon as a transaction has been confirmed by the PCI system of the server the flushed buffer is considered empty and can accept new data. Different configurations have been tested, with different number of buffers and different dimensions, showing an optimal configuration with 16 buffers 4 kiB each. With this configuration a transfer rate of 55 Gbit/s has been measured; this value is related to one x8 link so the output x16 link has a bandwidth in the order of magnitude of 100 Gbit/s.

2.8.2 The Field Programmable Gate Array

One of the key features of the PCIe40 board and of the project described in this thesis is the *Field Programmable Gate Array*, FPGA. These devices are a type of *Programmable Logic Device* (PLD), integrated circuits consisting of a matrix of interconnected logic gates. Unlike CPUs, that executes consecutively instructions loaded in memory, these devices have a special circuitry that can be configured by user to directly perform the required calculations. Such configuration is stored in a volatile SRAM so that each time the FPGA is turned on it must be reprogrammed. They are more energy-efficient and have lower latencies than CPUs because they don't have to read and decode instructions to be executed and, most of all, many of the calculations can be performed at once, since also very complex logic functions can be implemented. ASICs (*Application Specific Integrated Circuits*) would be the best devices because they contain the exact number of logic gates and connections required for a specific task, but they have higher prices because of the long time needed for development and, being non reprogrammable, cannot be produced in large

volumes. Therefore they are typically used when a very large number is needed, otherwise FPGA are chosen as they are reprogrammable and can thus be produced industrially at low cost.

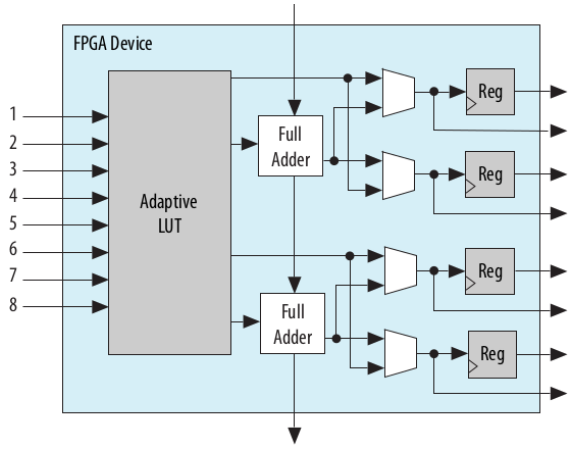


Figure 2.14: Internal architecture of Arria-10 ALM.

The FPGA used in the PCIe40 board is the ALTERA Arria-10 GX. Its building block is the ALM (*Adaptive Logic Module*), whose internal design is shown in Fig. 2.14. It is made of an eight-input LUT, two adders, four registers and a set of MUXes. The LUT can be split in two parts not necessarily equals, if needed. The purpose of the program loaded into SRAM is just to configure the LUT and the MUXes, to perform the desired calculations and correctly route data inside ALM. 427 200 ALMs are available in the FPGA model chosen by the LHCb Collaboration. Other components are present in the FPGA, first of all memories: 2 713 M20K memory blocks and 20 774 MLAB blocks. The first are 20 kbit memory blocks with ECC, while the latter are 640 bits blocks without ECC. Other internal components are hard IP (Intellectual Properties), circuits placed outside the programmable fabric that implement common high-level functions, such as external memory controller, PLL, DSP blocks for complex arithmetical calculations, etc.

2.9 The LHCb-RETINA project

An R&D work is currently ongoing [60–63], within the LHCb Collaboration, for the realization of an innovative tracking device capable of reconstructing in real time tracks of particles in the context of the future upgrades (beyond LHC Run 3) of the LHCb experiment. Such a specialized processor is expected to obtain a copy of the data from the readout system, reconstruct tracks, and insert them back in the readout chain before the event is assembled, in order to be sent to the high level trigger in parallel with the raw detector information. This approach, where tracks can be seen as the output of an additional "embedded track detector" is based on the *artificial retina* algorithm [60, 61, 64], which is a highly-parallel pattern-matching algorithm, whose architectural choices, inspired to the early stages of image processing in mammals brain, make it particularly suitable for implementing a track-finding system in present-day FPGAs.

The track parameter space is divided into cells. For a given input, only a few units are active. The portion of the input space where a cell has non-zero response is called its *receptive field*. This local representation mirrors the locally sensitive, orientation-selective neurons in the mammals visual system: cells in the visual cortex are tuned to recognize a specific pattern and to respond maximally when the retinal stimulus and the pattern are closest. Local connectivity allows to exploit spatially local correlations to increase scalability and reduce the total number of cells required, and is particularly important when dealing with high-dimensional inputs, as it would be impractical to connect all cells to all regions of the input space. The continuous level of matching featured in the artificial retina represents therefore an advantage over algorithms relying on a binary response, such as the Hough transform [65] or the Associative Memory [66, 67].

First small prototypes of the track-processing unit, able to reconstruct two-dimensional straight-line tracks in a 6-layers realistic tracking detector, based on the artificial retina algorithm have been designed, simulated, and built [52, 68–70] using commercial boards, equipped with modern high-end FPGAs. Throughputs for processing realistic LHCb-Upgrade[2] events above 30 MHz and latencies lesser than 1 µs have been achieved running at the nominal clock speed, demonstrating the feasibility of fast track-finding with a FPGA-based system. A detailed description of the artificial retina architecture, along with an early evaluation of its performance, can be found elsewhere [52, 68–70]. Only a brief summary is reported here. The tracking process is made with two main stages. In the first one (switching), all hits received from the different tracking detector layers are coordinate-transformed, and delivered to the appropriate location in a processing array, through a large custom-built switching network. During this stage a significant duplication of information can occur, requiring the use of a large bandwidth. The second stage is performed by a large array of cells (processing engines), mapping the track parameter space. Each cell evaluates an appropriate weighting function, similar to the analog excitation response of biological neurons, related to the distance of each hit from a set of reference tracks. The array is endowed with the capability of performing local cluster finding to determine the location of tracks and their parameter estimates in a completely parallel fashion over the entire device, without wait states, thus ensuring high throughput and low latency. The size of a such envisioned device it is affordable with already commercially available off-the-shelf FPGAs; a full realistic tracking system requires about 10^5 processing engines [52, 69].

Two implementations of RETINA tracker have been currently developed, one for the VELO tracking [63] and the other for the *seeding* algorithm, i.e. the reconstruction of standalone T-tracks using only hits coming from the SciFi subdetector (the so-called Downstream Tracker) [62]. The purpose is to relieve the CPUs from the pattern recognition task, one of the the most time-demanding to be executed by the HLT, that, being low-level and parallelizable, can be effectively transferred to specialized hardware accelerators, at the earliest possible processing level ("on-detector") in order to reduce the data flow as much as possible and as soon as possible.

The Retina Tracker has to be integrated inside the DAQ architecture of the LHCb Upgrade, and precisely into the Event Builder (EB) [27] which receives data from the detector readout. In the current foreseen layout, the VELO subdetector will send raw hits to about 50 EB nodes, while the SciFi subdetector to about 150 EB nodes. Gathering

[2]Running conditions in Run 4 will be the same as in Run 3, the so-called LHCb-Upgrade.

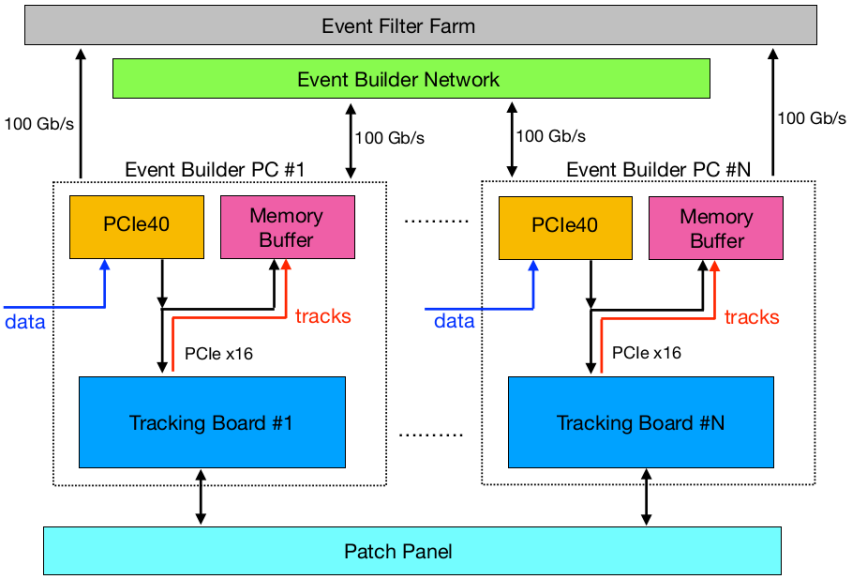


Figure 2.15: Diagram of the RETINA tracker integration into the Event Builder farm, showing the interconnections between EB nodes, PCIe40 cards and tracking boards. The patch panel implements the full-mesh network for data exchange between all the tracking boards.

data from such a large number of nodes requires an equally large number of devices to perform the switching function. For this reason each EB node of the subdetector of interest (VELO, SciFi, or both) must be instrumented with a standard commercial PCIe card equipped with FPGAs (tracking board) receiving a copy of data from the readout before the beginning of the event building stage. Each tracking board will host both functionalities of the switching network and of the processing engines. Each portion of data, distributed in different tracking boards, will be sent to all relevant engines, through a mesh network (patch panel) allowing the exchange of the hits between different tracking boards. Each tracking board will return a subsample of reconstructed track candidates to the EB node to which it is connected, and the reconstructed tracks candidates will be added to the raw data collected by that specific node.

At this point the event building process will proceed as usual and reconstructed track candidates will be treated by the system as raw track-hits from an additional "embedded track detector", namely a virtual subdetector providing tracks instead of hits. An illustration of the integration of the RETINA tracker inside the Event Builder farm is shown in Fig. 2.15. The tracking boards are PCIe boards equipped with an FPGA chip executing the RETINA algorithm, both the switching network and processing engines stages. Each EB node is linked, via a PCIe bus, to one tracking board.

2.9.1 This thesis: the FPGA-based VELO clustering

The upgraded readout system is designed to run at LHC bunch crossing rate of 30 MHz, providing the opportunity to develop and test such highly-parallel tracking systems, as the VELO tracker and the downstream tracker, in view of future LHCb Upgrades (beyond LHC

Run 3). While the SciFi subdetector will already have a clustering algorithm[3] running on the readout FPGA cards at a speed of 30 MHz during the upcoming Run 3, the 2D clustering of the silicon pixels of the VELO subdetector, being much more complex and timing consuming, is foreseen to be performed into the High Level Trigger, occupying resources that could be used for higher level tasks. Furthermore, the implementation of any VELO tracking algorithm, both in real time or at offline level, requires pre-processed input clusters, therefore it is of fundamental importance the development and implementation of a new 2D clustering algorithm for the VELO subdetector, running at a speed of 30 MHz at the conditions of the LHCb Upgrade. In this regard, the work performed in this thesis is developed within the LHCb-RETINA project and aims at moving the 2D VELO cluster-finding to the FPGA readout cards, freeing the HLT farm from the burden of this computation, and paving the way for a first test bench of the VELO tracking algorithm, hopefully already during the LHC Run 3 with real data, in preparation of the installation of the whole system for the LHC Run 4.

The structure of this thesis is as follows. Chapter 3 describes the FPGA-based 2D clustering algorithm specifically developed for the upgraded LHCb VELO pixel subdetector, along with its hardware implementation. Chapter 4 illustrates the integration of the clustering firmware into the TELL40 firmware, with a detailed overview of the new programmed entities and all modifications made to the already existing entities specifically for this purpose. Lastly, chapter 5 presents detailed studies on the performance of the LHCb tracking, obtained using clusters provided from the FPGA-based clustering algorithm described in the thesis, along with the comparison with baseline CPU algorithm. These studies are a necessary step, since the algorithm, designed to be run on FPGAs, is intrinsically different from the CPU-based baseline algorithm.

As a master thesis student I had the responsibility of the integration of the FPGA-based clustering firmware into the VELO readout card firmware. I designed and wrote all the VHDL software that allowed the porting of the core clustering firmware, developed and tested on a Stratix-V FPGA chip in Pisa during the very early stage of the project, into the the Arria-10 chip, mounted on the readout LHCb cards. I have also significantly contributed to the test of the final algorithm implementation, using fully realistic simulated samples at the LHCb Upgrade running conditions, by writing and optimizing many parts of the high level simulation of the whole system.

[3]The 1D clustering is a well known and simple task to be performed on a FPGA chip, even at high event rate of 30 MHz.

Chapter 3

A 2D FPGA-based clustering algorithm

The chapter describes a new FPGA-based 2D clustering algorithm specifically developed for the upgraded VELO pixel subdetector, and its hardware implementation. The design of the algorithm is fully based on the LHCb-RETINA R&D project aiming at performing the full VELO-tracking on FPGA cards in real time at a speed of 30 MHz*.*

3.1 Algorithm overview

A 2D clustering algorithm is much more complex than a 1D algorithm, since the latter can be done with a sequential scan of the information, while the first one needs a more complex approach, like the RETINA clustering design described in this chapter.

In order to reduce the output bandwidth, the VELO pixels are read in groups of four rows and two columns, and they are called Super Pixels (SP). The SPs are sent from sensors to the DAQ PCIe40 boards, and pass through the FPGA Arria-10 chip mounted on the board. In the flow inside the FPGA, SPs are flagged as "isolated" ("non-isolated") if none (at least one) of its eight closest neighbours has any active pixel. This distinction leads to a performance optimization of the algorithm, since isolated SPs are much easier to clusterize and, therefore, require less usage of both logic and memory. The isolation flag is very valuable also for the CPU baseline algorithm, that otherwise could not do it very quickly.

SPs are, therefore, split according to their isolation flag. Isolated SPs are resolved using a LUT: for each possible configuration of hit pixels the center of mass is stored in FPGA's memory. Since there are eight pixels in each SP the dimension of the LUT is of 256 entries. This approach is very fast and requires small amount of memory and logic elements.

The algorithm for non-isolated SPs is more complex, since it requires the parallel processing of an ensemble of SPs. For this purpose a chain of 20 matrices that can host 15 contiguous SPs each in three rows and five columns is defined; only contiguous SPs of the same bunch crossing and the same sensor can fill a matrix (one PCIe40 handles a VELO module consisting of four sensors). The first SP arriving is placed in the center of the first available matrix. Each time the first SP enters a matrix, the set of coordinates of each SPs that can fit inside the same matrix is calculated. The subsequent SP is placed

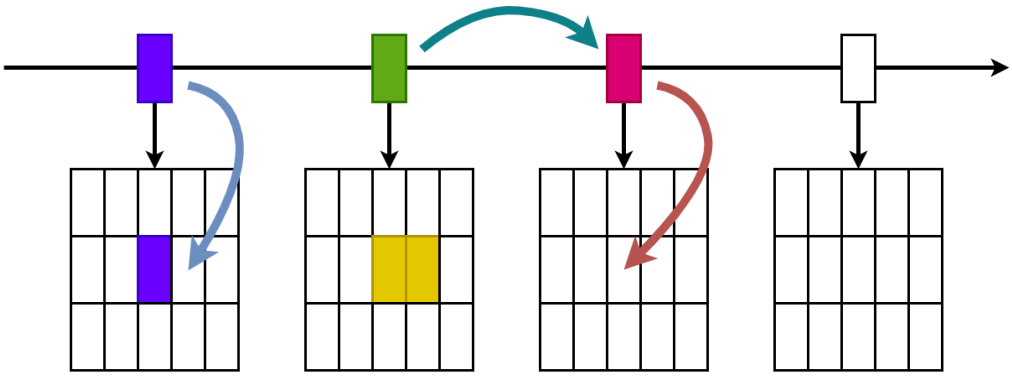


Figure 3.1: The process of filling the matrix chain is illustrated. The blue SP goes inside the first matrix because is a neighbour of the SP already at the center. The green SP cannot fit inside the matrix because is not a neighbour of yellow SPs, and it goes to the next matrix. The red SP couldn't fit inside any of the previous matrices therefore is placed at the center of the first empty one.

in that matrix if its coordinates matches those calculated, otherwise it goes ahead and checks the next matrix. Such a process continues until all the matrices are filled with the corresponding SPs. If there is no available matrix the SP is treated as isolated though it is an approximation and the position of the returned cluster could be biased.[1] A diagram of the filling process of the matrices is shown in Fig. 3.1.

When SPs of a given event are finished, each matrix looks, in a parallel way, for a pattern of pixels, which identifies the seed (also called checking pixel) of a 3×3 sub-matrix of pixels used to calculate the center of the cluster. Since the majority of the clusters are made by a small number of pixels (see later), the more efficient patterns to look for returned to be a "L" shaped sequence of inactive pixels with two different configurations of active pixels (called "L-patterns" from now on), as shown in Fig. 3.2, where an explicative sketch of them is illustrated. Once the checking pixel is found, only the pixels contained in the corresponding 3×3 sub-matrix are considered for the determination of the cluster position. As the isolated SPs the sub-matrix is resolved by means of a LUT, twice the size of the isolated one ($2^9 = 512$ entries). Finally the global coordinates of the cluster are obtained as vector sum of the position of the matrix inside the sensor, the position of the checking pixel inside the matrix, and the position of the cluster inside the 3×3 sub-matrix.

Various design parameters of the algorithm need to be adjusted according to the features of simulated SPs, like the size of the sub-matrix, the shape of the matrices, and their number. Since the algorithm is designed to run on FPGA, these parameters cannot dynamically change and, therefore, they must be carefully chosen. The distribution of cluster size obtained with the official LHCb Upgrade simulation is shown in Fig. 3.3.[2] providing useful information. Since the majority of the clusters is made up of a small

[1] This occurs rarely and the bias due to the small amount of non-isolated SPs, resolved as isolated, results to be negligible (see Chap. 5). The length of the matrices chain is chosen to reduce as much as possible the usage of FPGA resources keeping under control any possible resulting bias in the determination of the cluster position.

[2] More details on the official LHCb Upgrade simulation can be found in Sec. 5.2.

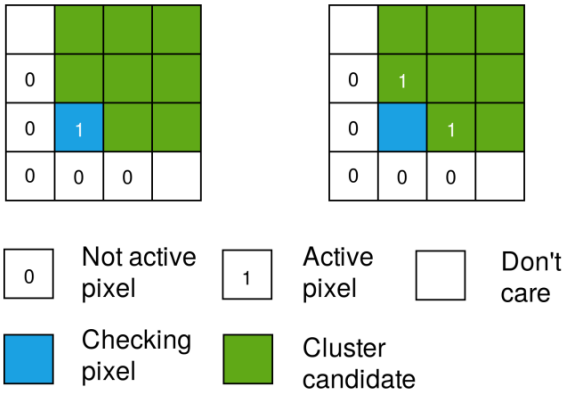


Figure 3.2: Patterns searched by the firmware to find the "checking pixel". The 3×3 green matrix contains the pixels used to determine the position of the cluster.

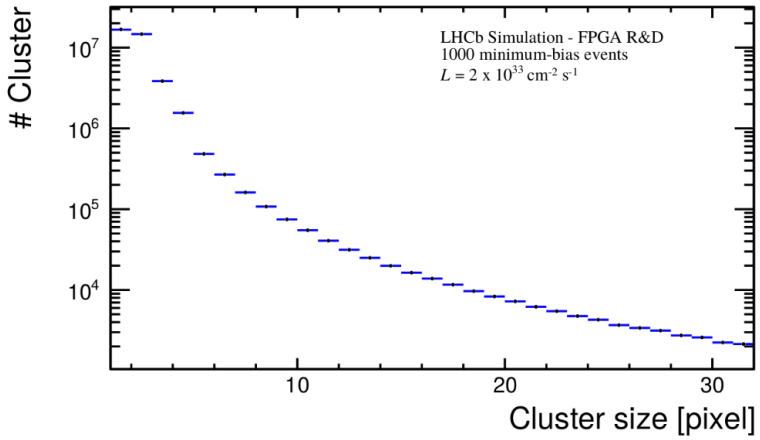


Figure 3.3: Distribution of cluster size. Since the majority of clusters is small, it is worth clusterize isolated SPs separately with a LUT and keep only a 3×3 sub-matrix for non-isolated SPs.

number of pixels (very few of them), it results very convenient to treat isolated SPs apart, because of the limited usage of logic and memory FPGA resources. Moreover, non-isolated SPs can be successfully resolved with 3×3 sub-matrices, since a very large fraction of clusters has a number of pixel lesser than nine. Although bigger sub-matrices would be better in resolving very large clusters, the FPGA resources (memory and logic of the used LUT) necessary to solve them grows exponentially with the number of pixels, therefore 3×3 sized sub-matrices are definitely the optimal choice. A number of 20 matrices is chosen as a trade-off between reconstruction quality and resource usage, as already mentioned above. Events with too many non-isolated SPs (a small tail in the distribution) can overflow the chain of 20 matrices, and in that case SPs are "wrongly" resolved as isolated ones. The final performance of the algorithm will be extensively studied in Chap. 5, where the impact of all these approximations will result to be negligible with respect to baseline CPU algorithm, where all available information is used.

3.2 The hardware implementation

The described implementation is designed to operate on data coming from two VELO sensors, and in order to process a whole VELO module (made of two sensors) two identical instances of the same firmware are needed. Figure 3.4 shows a scheme of the entire firmware. It has a modular design, being made of many independent entities. Input data are caught by the `decoder` that converts them from a single 256-bit word to eight 32-bit words, each carrying a SP. It sends them to two instances of the `switch` that select SPs by their isolation flag and sensor. The `switch` sends them to the appropriate clustering entity, according to isolation flag. SPs from different sensors go to different instances of clustering entities and are never mixed. Finally the `encoder` converts back eight 32-bit words into a single 256-bit word.

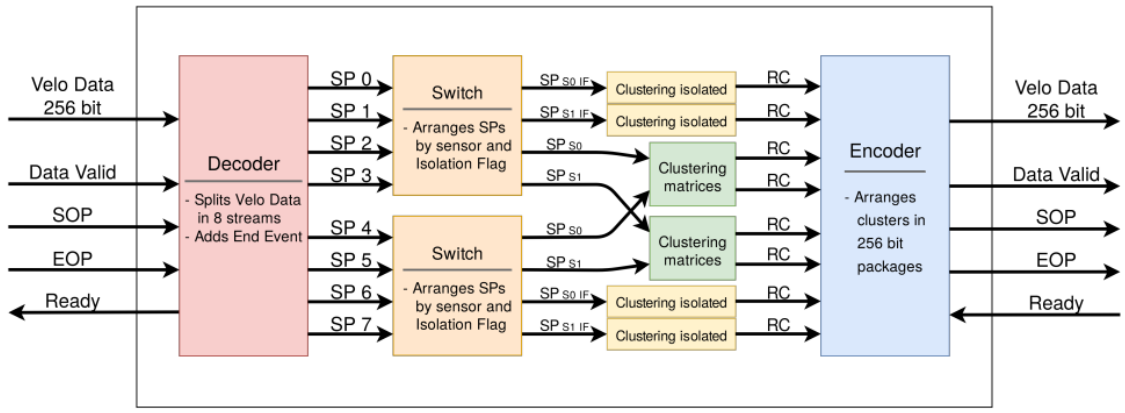


Figure 3.4: Overview of the first working clustering firmware with its main components.

3.2.1 Data format

Fig. 3.5 shows both SPs and clusters data format. They are both encoded in 32-bit words. Starting from the available 32 bits for the SP word, eight bits are used for the pixmap (map of hit pixels), 15 bits for the position of the SP inside the sensor, two bits to identify the position of the sensor in the module, and two bit for the isolation flag. Clustering firmware checks only bit 31 to discriminate isolated SPs. Bit 30 is used to tag SPs for particularly busy events, for which the Isolated Cluster Firmware entity cannot flag all input SPs. In those cases SPs are all considered not isolated and resolved into clusters using matrices. Since a sensor has 256 rows and 768 columns, six bits are needed for encoding the SP row index, and nine bits for SP column index, for a total of 15 bits, as said before. Clusters are also encoded in 32-bit words. A number of 18 bits is needed for the center-of-mass position, and one bit for sensor. Since data processing for sensors 0-1 and 2-3 are separated, a single bit is enough to distinguish clusters from sensors in a couple. Additional 6 bits are used for fractional parts of the row (three) and of the column (three), providing a resolution of ⅛ of pixel size.

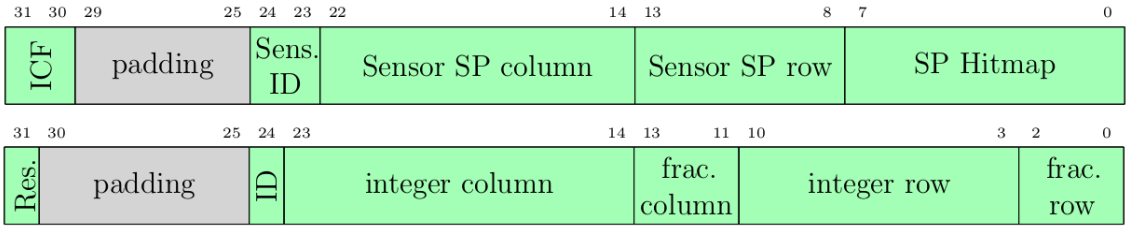


Figure 3.5: Data format used for encoding (top) SPs and (bottom) cluster center-of-mass.

3.2.2 Input-output interfaces

SPs are arranged in 256-bit words containing eight 32-bit words, as described in Sec. 3.2.1; a corresponding valid signal is also sent. *Start Of Package* (`SOP`) and *End Of Package* (`EOP`) signals are also provided. The `SOP` comes together with the first 256-bit word of the event, while the `EOP` with the last word of the event; if an event is empty or is made of only one word, `SOP` and `EOP` signals are simultaneously sent. The `READY` signals are used by all components of PCIe40 firmware to ask the preceding component for more data. As far as the output side, there are the corresponding signals: data words have the same size and carries clusters instead of SPs.

Input data are stored in a vector FIFO buffer composed of nine 32-bit FIFOs. Going from the least to the most significant, the first eight contain the eight SPs transmitted simultaneously in the same 256-bit word, while the last one contains (still from the least to the most significant bit) `SOP`, `EOP`, `BXID`, `FTYPE` signals, and a final padding. An intermediate entity, the `feeder`, sends read requests to the input FIFO, while sending the corresponding valid signals to the decoder.

3.2.3 Decoder

The `decoder` is the first components of the clustering firmware. Its job is to unpack data from one 256-bit word to eight 32-bit word for parallel processing. Additionally, it converts the `SOP-EOP` signal used to separate different events to the End-Event (`EE`) logic: in this approach a special word (the End-Event) is placed between two consecutive events to separate them. This is the standard used internally by the clustering firmware, and will be converted back to `SOP-EOP` before outputting data. The four least significant bits of an event counter are also included as data identifier in each `EE` word in order to track data flow and ensure synchronization. The `decoder` writes its output data to the `transf_FIFO`, that is read by the `switch`.

3.2.4 Switch

The purpose of the `switch` is to divide SPs according to their origin sensor and isolation flag. The eight SP lines exiting from the decoder are sent to two `switch`, four-inputs each. As can be seen in Fig. 3.4 the four output lines of one switch unit transfer SPs divided by sensor and isolation flag, the non-isolated ones go to the matrix units (one per each sensor) and the isolated ones to two instances of the isolated clustering LUT (two per each sensor).

The `switch` is made of two logic units: the `splitter` and the `merger`. The connection

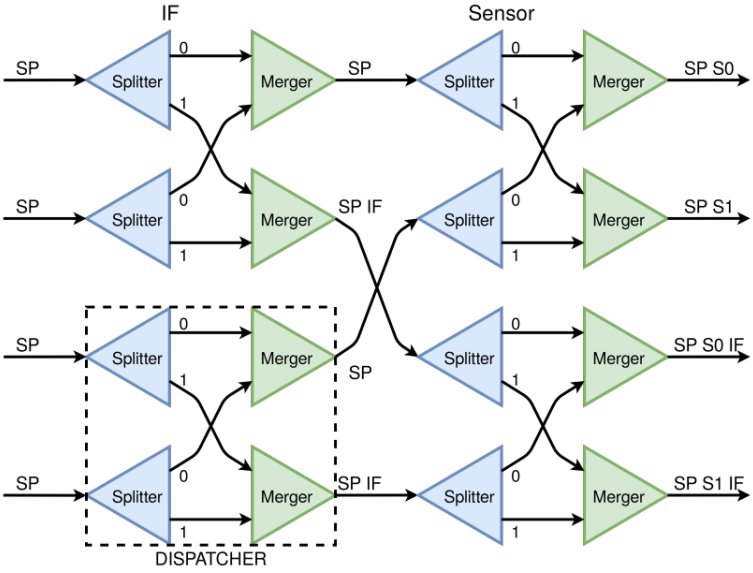


Figure 3.6: Logic architecture of the `switch`.

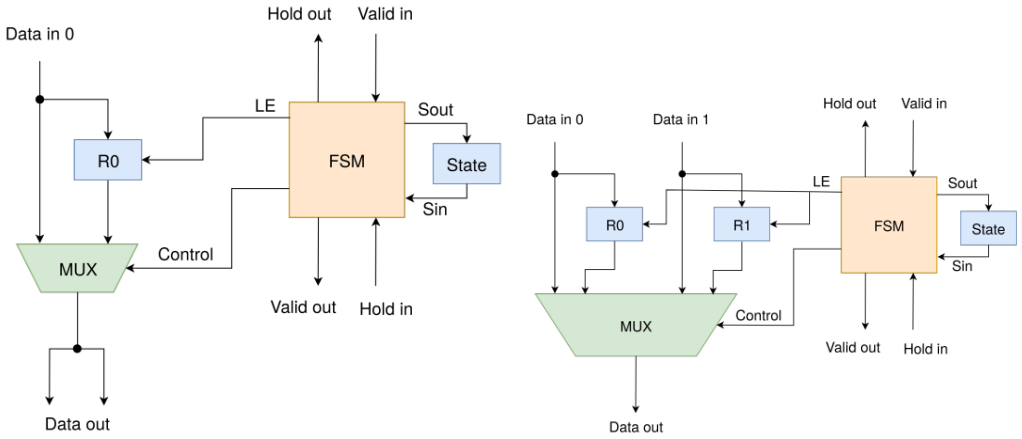


Figure 3.7: Logic architecture of `splitter` on the left and `merger` on the right.

diagram between these two units is shown in Fig. 3.6. The ensemble of two `splitters` and two `mergers` is called `dispatcher`, it gets two SPs as input and outputs them according to one flag, in our case the isolation flag or the sensor flag; for example the `dispatcher` in the left-lower corner of Fig. 3.6 outputs in the upper line non-isolated SPs (and isolated ones in the other line). Four `dispatchers` are needed in order to split SPs in both isolation and sensor flags.

Figure 3.7 shows the logic diagram of the `splitter` on the left. It is based on a Finite State Machine (FSM) that univocally assigns the new state of the entity and generates the output signals according to the current status and input signals. `R0` is a register where an input SP can be placed if the `HOLD_IN` signal is high, meaning that one of the following `mergers` is not able to accept any data because it's full; in this case the FSM asserts a write request (LE in Fig. 3.7) and the register `R0` is written to, then the `HOLD_OUT` signal is

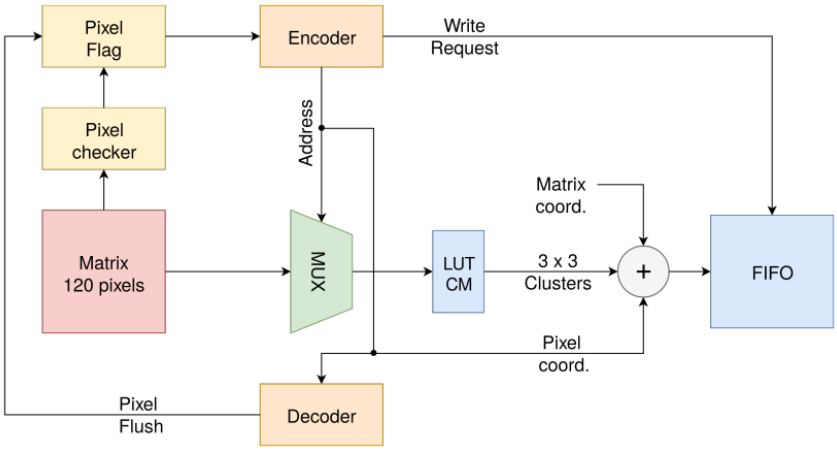


Figure 3.8: Logic architecture of the `cluster_finder`. This entity operates on one matrix of the chain.

asserted meaning that the `splitter` cannot accept further SPs. The FSM controls the output of the `splitter` with a mux; if data can be sent and `R0` is full then its content is output before any incoming SP. The output SP is sent on one of the output lines according to the flag checked by the `dispatcher` of which is part of; if an `EE` arrives then it is sent to both output lines.

Figure 3.7 shows the diagram of the `merger` on the right. It is conceptually similar to the `splitter`, except for the fact that it does not have to check for a flag and that it has two input and one output. Having two inputs means that there must be two auxiliary registers `R0` and `R1` with two write requests and a 4:1 mux. If an `EE` is received on one of the input lines, it is stored and it waits for an `EE` on the other line; if the two `EE` are associated with different event-identifier an error is raised meaning that data synchronization has been lost.

The `switch` writes its output data inside the `postswitch_FIFO`, that is read by both subsequent clustering entities.

3.2.5 Clustering for isolated SPs

There are four instances of the clustering for isolated SPs, each of them reads one line of the `postswitch_FIFO` carrying isolated SPs. Considering that a SP is composed by only eight pixels then the best way (in terms of throughput and resources usage) to clusterize isolated ones is to use a LUT, where the center of mass for each possible configuration is stored. It has been chosen to return the center of mass coordinates with the resolution of ⅛ of pixel, therefore three bits are used for the fractional part of each coordinate for a total of six bits. The final cluster coordinates are obtained as vector sum of the coordinates returned by the LUT, that are referred to the lower-left corner of the SP, and the coordinates of the SP inside the sensor. Output data are written inside four lines of the `out_FIFO`.

3.2.6 Clustering for non-isolated SPs

There are two instances of the non-isolated clustering, each reading two lines of the postswitch_FIFO corresponding to the same sensor. The key element for this task is a chain of 20 matrices to be filled with contiguous SPs. One matrix can accommodate 15 SPs on three rows and five columns. The process is performed in two stages: in the first one the SPs fill the matrices of the chain, while in the second one the filled matrices, after the arrival of the EE signal, are sent to the *cluster finder* entity to look for the L-pattern and calculate the position of the cluster candidates.

Each matrix is filled starting from its center. As soon as a SP is accommodated inside an empty matrix, the coordinates of this SP are associated to the matrix, and only a well defined subset of other SPs, the neighbours of the SP placed in the center, can fill the free slots of the matrix around the central SP. If a SP does not match any possible slot of the matrix, it checks the subsequent matrix of the chain, filling the central position in case of empty matrix, and so on until all the available matrices of the chain are filled. If all the 20 matrices are already busy, the SP is resolved by a LUT, the same as for the isolated SPs.

The second stage starts when the clustering for non-isolated entity reads two EE from its input lines; the content of its matrices is transferred to the cluster_finder entity that checks in a parallel way every pixel for one of the two patterns shown in Fig. 3.2. If a pattern match is found the corresponding 3×3 sub-matrix is used to determine the position of the cluster candidate. The logic diagram of the cluster_finder is shown in figure 3.8: if a checked pixel matches the required pattern the corresponding pixel_flag is set, and an encoder generates the address for the MUX to selects the 3×3 sub-matrix, that will be resolved by a 512-entries LUT. A decoder catches the address generated by the encoder and sends a flush signal to reset the pixel_flags already processed. The global coordinates of the cluster are obtained as a vector sum of the coordinates of the cluster with respect to the 3×3 sub-matrix, the position of the sub-matrix inside the matrix, and the position of the matrix inside the sensor. The cluster candidates are then written in a FIFO, one for each matrix. After that, cluster candidates from all matrices are merged by a 20:1 merger. Finally, they are written in the out_FIFO, two data lines for each matrix chain.

3.2.7 Encoder

The encoder reads data from the out_FIFO. It has the purpose to convert eight 32-bit words to one 256-bit word, and to convert back the EE logic to the SOP-EOP logic. It is composed recursively by the basic block which merges two data lines into one line with a double width. Figure 3.9 shows the logic diagram of the basic block: a FSM managing the signals, three buffer registers (R0, R1 and R3), and three MUXes. If two data are simultaneously received on the input lines they are sent directly to the output, while if only one data is received it is stored in R3 register. The purpose of R0 and R1 is to store data to be output when an HOLD_IN signal is asserted, similarly to the switch (see Sec. 3.2.4). If an odd number of clusters is received then the output word is zero-padded: for this purpose MUX0 and MUX1 have an input hard-wired to 0. An EE signal is propagated only when two EE are received on the input lines and their event count is the same, otherwise an error is generated.

Another entity is necessary to generate the SOP and EOP signals. The EOP signal is

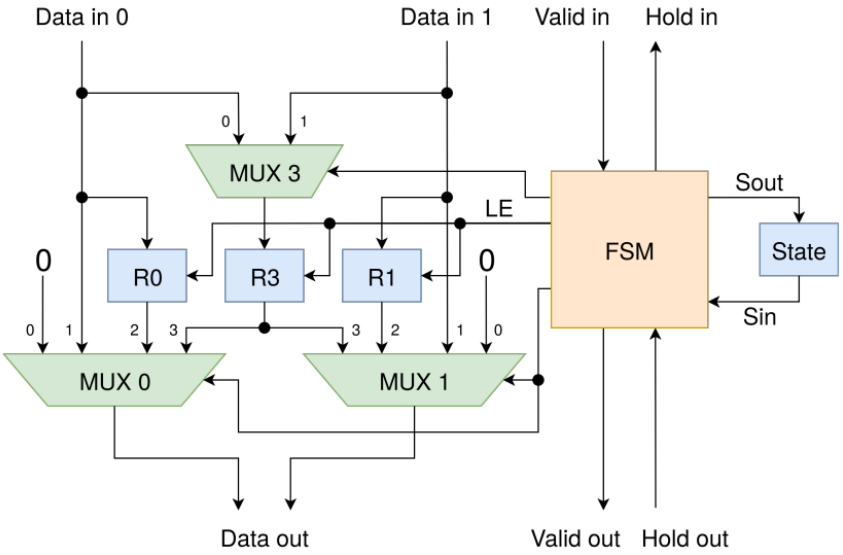


Figure 3.9: Logic architecture of the basic block that makes up the `encoder`.

generated when an `EE` is received, then an internal signal is asserted as trigger. It fires when the first word of the next event is received, thus generating a `SOP` signal. Output data is written in the `postencoder_FIFO` from where they are read for the transmission to the PC.

3.2.8 Control logics

As explained in previous sections, a large number of FIFOs has been inserted between entities shown in Fig. 3.4 and also inside the entities as decoupling buffers, very useful in absorbing local fluctuations in processing speed. An ECS subsystem is also implemented to allow reading the information of the state of the various entities, and changing their configuration, if needed.

A large number of entities can generate an error signal, generally to indicate data loss or mixing of data from different events. Data loss can affect each component with internal buffer registers and occurs when an already full register is going to be written. Data mixing occurs when an entity with more input lines (like `switch merger` and `encoder` basic block) receives two `EE` with different event identifiers. Error signals are propagated to the top entity that simply disable the entire firmware. By the type of error signal received one can determine which component has crashed.

3.3 The hardware prototype

In the early stage of the project, the firmware of the clustering was designed and tested, in Pisa, on a DINI prototyping board [71] (model DNS5GX_F2) purchased by the INFN-Pisa within the INFN-RETINA project funded by the CSN5. The board is equipped with three Stratix-V FPGA chips (model 5SGXEABN2F45C3), a model similar to the Arria-10 mounted on the PCIe40 boards. At that time, the core firmware of the clustering was not

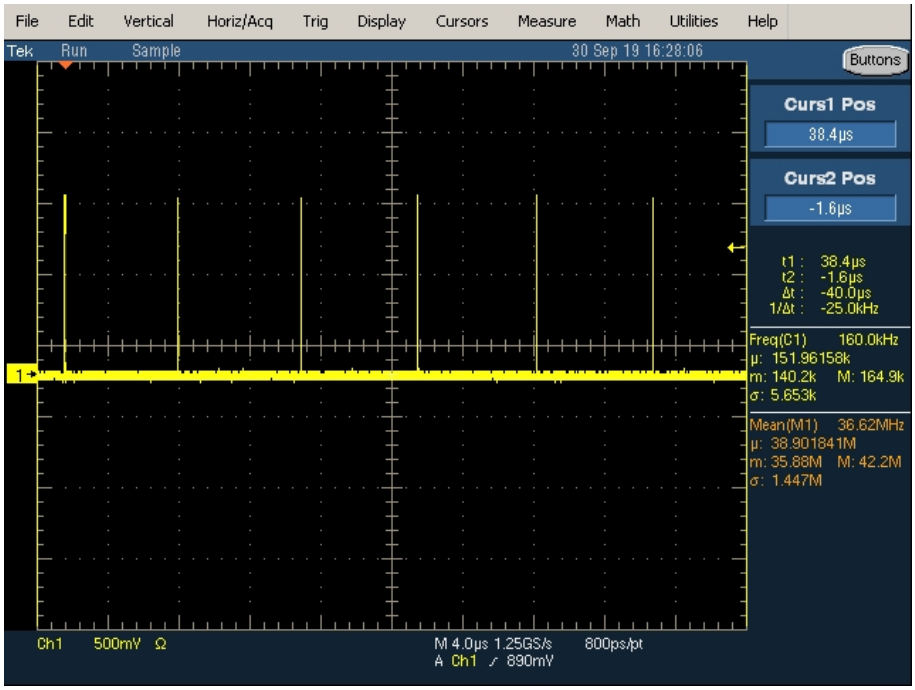


Figure 3.10: Oscilloscope screenshot showing the reached throughput of 38.9 MHz.

integrated into the VELO readout software (which was yet to be developed at that time), therefore, both an input and an output stage were included in order to feed the system, and send the output to the PC for the later check.

For the input stage a loop-ram is implemented, which is loaded on startup using an ensemble of 1999 SPs from a simulated sample of minimum bias data, generated at the conditions of the LHCb Upgrade. The reading of the loop-ram is started again each time the end is reached. The output stage, instead, consists of a prescaler which selects one event every N events, and an acquisition component, called `retinaspy`. The latter is necessary for the communication with the PC, in order to store and check data. The number of events stored in the loop-ram and the number N must be chosen coprime in order to collect the entire event sample in more than one cycle. This prescaling is necessary because the PC cannot read clusters at a speed of about 30 MHz, but it is desirable to check every single returned cluster. Throughput and correctness of output have been extensively checked. Figure 3.10 shows an oscilloscope screenshot illustrating the reached event throughput of 38.9 MHz. This is well above the minimum allowed rate of 30 MHz.

The core firmware of the clustering algorithm was ready and functional at the beginning of this thesis work. It was the very first implementation, developed to run on a Stratix-V FPGA chip (different from the Arria-10 chip mounted on the PCIe40 boards) and was not conceived at that time for the integration into the TELL40 firmware. Such a porting and integration process is the first part, and maybe the most relevant one, of my thesis work and it is described in detail in the next chapter.

Chapter 4

Integration in the LHCb-Upgrade DAQ system

The clustering algorithm and the firmware were developed and tested on a Stratix-V FPGA chip in Pisa during the very early stage of the project. This chapter describes the work performed to port the core firmware to the Arria-10 chip, mounted on the readout cards, along with the development of the all functional firmware for a full integration into the TELL40 firmware of the VELO readout cards.

4.1 Introduction

The integration of the clustering algorithm in the TELL40 firmware consists of several stages. At first all the ancillary components, whose purpose is to feed clustering, and to read its output is removed; the remaining clustering core firmware is adapted to allow a full compatibility with the TELL40 firmware. Additionally, all IPs, mainly FIFOs and LUTs, are recreated, accounting for a different FPGA architecture. A full integration also requires programming of new entities, as it will be described in the following sections. For brevity the first firmware implementation, as described in Chap. 3, is referred as the *Stratix-V* version, while the new one, developed in this thesis, as the *Arria-10* version. The firmware is written in VHDL language, a Hardware Description Language. Words like *entity*, *port*, *process*, *signal*, etc. are meant as the corresponding keywords of this language.

Figure 4.1 shows a detailed diagram of the firmware, as it was at the beginning of the work of this thesis. Its purpose is to show the path followed by data and all entities instantiated in the top entity of the firmware. The clustering box comprises the clustering entities for both isolated and non-isolated SPs.

4.2 Isolation of clustering firmware

At the beginning of this work the implementation of the clustering firmware included input and output stages, in order to feed in simulated SPs and read out reconstructed clusters, for a later check. These stages are no longer needed in the PCIe40 board since the feeding process is managed by the TELL40 firmware, and output to PC is not desired. A reorganization of the all firmware is, therefore, necessary to integrate it in the new

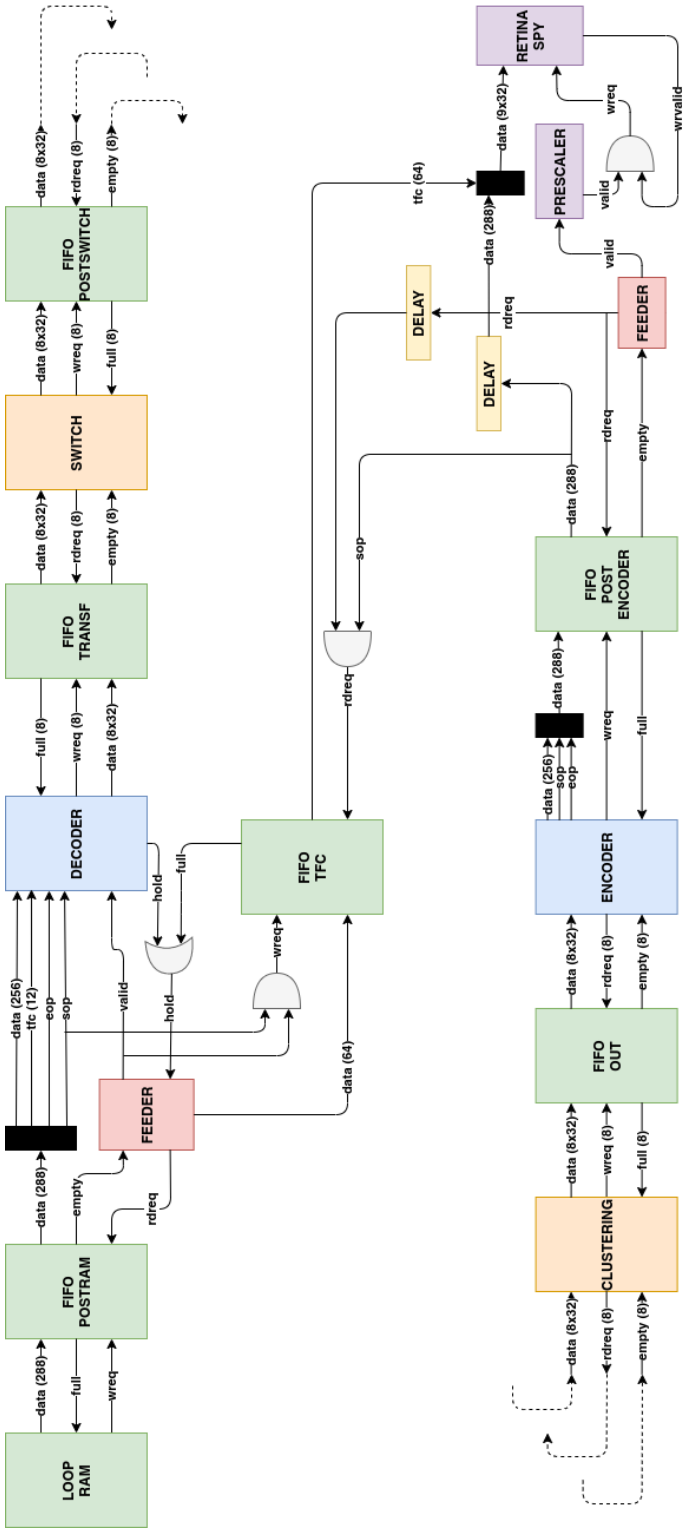


Figure 4.1: Detailed diagram of the *Stratix-V* version firmware.

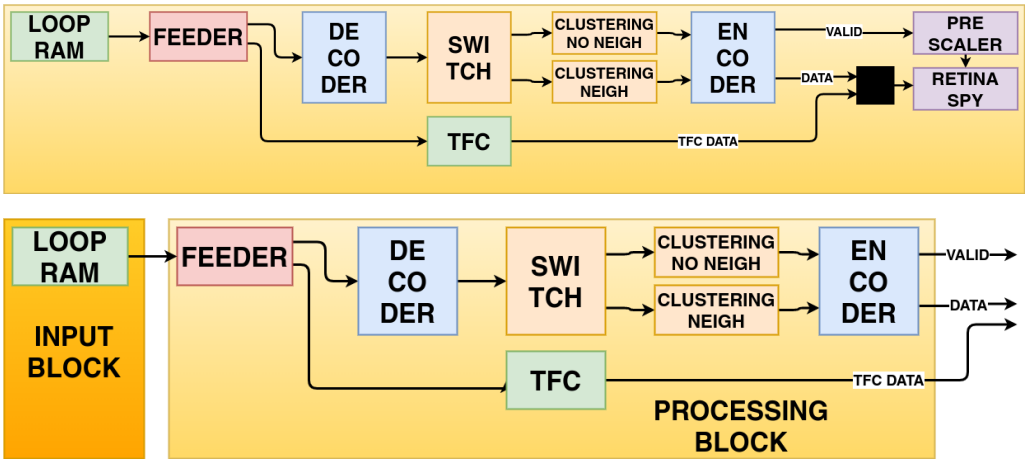


Figure 4.2: A general scheme of (top) the firmware at the beginning of this work and (below) after the division. Note that the latter doesn't have the output stage.

environment, as a first step. This task consists in the separation of the core firmware in three different parts that are called `input_block`, `clustering_block` and `output_block`. They are hosted in three separated source files, corresponding to as many entities. These entities are instantiated within a top entity, called `test_bench`, that has been written from scratch; in truth, since the Arria-10 version will be used only in simulation, the `output_block` is not instantiated because it cannot be used, but the `input_block` is still used for SPs feeding. The test bench hosts the two blocks and feeds them with various signals. A graphical comparison of the firmware at the beginning and after the reorganization is shown in Fig. 4.2, while a more detailed description of the `input_block` and `output_block` can be found in Sec. 3.3. Clocks are generated in the `test_bench`, without the use of dedicated IP (PLLs) [1]. The Stratix-V version already used two clocks, at 50 MHz and 350 MHz for the ECS system (see Sec. 4.3) and as a common clock, respectively; in the newer version another clock is introduced, with a frequency of 250 MHz, for the entities that don't require a fast clock; this hopefully will allow the use of longer combinatorial paths without problems [2]. The mixed usage of different clocks requires proper FIFOs, already present as decoupling buffers, that must be able to read and write data at different clock speeds. Reset and enable signals are also managed by the `test_bench`. These signals will be used to check the robustness of firmware to errors (see section 4.13 for more details). Since the `input_block` delivers SPs with only an event counter as additional information, the `test_bench` also generates all the others signals that the TELL40 firmware really sends, to simulate more accurately the true environment in which the clustering will be fitted in. These signals are the topic of section 4.6.

As previously mentioned, the TELL40 firmware will run on an Arria-10 chip, so hard IPs must be recreated to be compatible with this different model of FPGA chip. The

[1] In the final placement inside the Arria-10 chip, clocks are provided externally.

[2] A combinatorial path consists of a complex logic inside the FPGA between two registers, that usually comes from a complex VHDL combinatorial piece of code. The maximum allowed clock frequency of the register receiving data from the combinatorial path is inversely proportional to the length of the path itself. Longer combinatorial paths restrict clock frequencies to lower values.

type of IP used by the clustering firmware are only FIFOs and LUTs; additionally the `input_block` uses a loop-ram, in which simulated SPs are stored. It contains a RAM IP which cyclically outputs SPs to feed `clustering_block`. IPs are generated with the Intel Quartus Prime software [72]. It provides a wizard tool for selecting all the properties desired and finally generates the IP. For each IP type a VHDL entity is needed as abstraction layer useful for firmware programming, but these were already available in the Stratix-V version and don't need to be modified.

4.3 ECS addresses remap

The purpose of Experiment Control System (ECS) [3] is to allow controlling and monitoring the electronics of the entire experiment during data taking by qualified personnel from the control room. Every electronic board can be controlled or monitored with its own specific set of addresses. The Stratix-V version uses a 32 bit space, allocating specific bit fields to each category of entities. In the TELL40 firmware ECS addresses are 25 bits long, with (going from most to least significant bit) the first bit selecting one of the two processing units (corresponding to one side of a VELO module) and the following eight bits selecting a specific processing block; `0xB4` is the code used to address the clustering firmware. The following 16 bits are used to identify single processing blocks and can be freely used. Therefore, a remapping is needed to shrink the previous used 32 bit space into the new one at 16 bits. This is possible since the used addresses were spread over all 32 bit in the Stratix-V version, with much space available between consecutive ECS blocks.

The most significant two bits are used to address the top entity and the `clustering_block`. The top entity is instantiated by TELL40 firmware and contains `clustering_block`. The following two bits are used to address FIFOs, registers, and loop-ram (the last one kept for simulation but will not be included in TELL40 firmware). The last 12 bits are used in different ways, depending on the category. They are split in three groups of four, so that each group is an hex digit, as follows.

- FIFOs: the first digit is used by vectorized FIFOs to allocate each vector component; the middle digit identifies the FIFO; the last one select a content inside the FIFO.
- Registers: all 12 bits select a register: address corresponding to `0x300` and `0x301` refers to the `overflow` registers, in which each matrix chain stores the number of overflowed SPs (i.e. those resolved as isolated though they are not); address `0x400` refers to the `double_output` register; `0x038` refers to the `enable_register`; `0x01C` to the `error_register`. Addresses starting with 2 are `snapshot` registers, where the second digit specifies which FIFO they belong to, and the latest digit is always set to 0. Addresses starting with 1 are loop-ram registers, and the following two digits specify the single register.
- Loop-ram: the space is not split, and only one address (`0x2000`) is used to access the content currently read. The content of loop-ram settings register specifies the 32-bit word to be read.

The new ECS address mapping is shown in table 4.1.

[3] See Sec. 2.8.1 for more details.

13:12	11:8	7:4	3:0	
00	01C			Error_register
	038			Enable_register
	1	00		Loop-ram loopsnum register (number of loops to do)
		04		Loop-ram loop register (loop index)
		08		Loop-ram locsnum register (unspecified)
		0C		Loop-ram loc register (reading index)
		10		Loop-ram delaynum register (number of delay cycles)
		14		Loop-ram delay register (delay cycle index)
		18		Loop-ram settings register (selector for reading content)
	2	x	0	FIFO snapshot registers
	30		0	Overflow register matrix chain sensor 0
			1	Overflow register matrix chain sensor 1
	400			Double_output register
01	x	x	0	Offset (rdata, not rdempty, not wrfull)
	x	x	4	Monitor (number of words currently stored)
	x	x	8	Maxfill (maximum number of stored words)
	x	x	C	EFMonitor (empty-full monitor)
	x	0	x	Unused
	x	1	x	Unused
	x	2	x	Transf_FIFO
	x	3	x	Out_FIFO
	x	4	x	Bypass_FIFO
	x	5	x	Post_switch_FIFO
	x	6	x	Post_RAM_FIFO
	x	7	x	Post_encoder_FIFO
	x	8	x	TFC_FIFO
	x	9	x	Post_LUT_FIFO
	x	A	x	FSIZE_FIFO
	x	B	x	Command_FIFO
	x	C	x	Double_output_FIFO
	x	D	x	SP_FSIZE_FIFO
	x	E	x	Unused
	x	F	x	Unused
10	000			Loop-ram (read 32-bit word selected by settings register)
11	xxx			Unused

Table 4.1: View of new ECS address mapping. All values are hex except for the first column (bit 13:12) where they are binary.

Figure 4.3: Example of isolated SP containing two clusters.

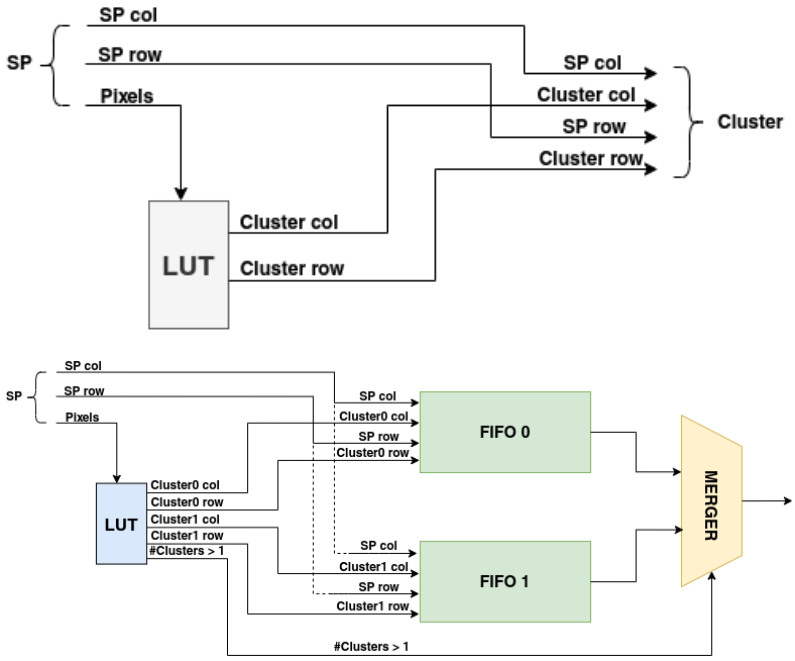


Figure 4.4: Logic architecture of the old (top) and newer (bottom) version of the isolated SPs clustering unit.

4.4 Double cluster in isolated SPs

Isolated SPs are resolved by a LUT containing the center-of-mass coordinates for each possible configuration of hit pixels. Since a SP is a 4×2 pixel matrix the depth of the LUT must be $2^8 = 256$. The returned center row can vary from zero to three and center column from zero to one, resulting in three bits; six additional bits comes from the fractional part, given that both row and column are calculated with a precision of ⅛ of the pixel size (three bits each). Therefore the resulting width has 9 bits. This 256×9 sized LUT was already implemented in the Stratix-V version, but for Arria-10 version it has been rewritten to account for cases where a single SP contains two different clusters, as shown in Fig. 4.3. Note that a single SP cannot contain more than two clusters. Although these cases are rare, it has been decided to add the necessary firmware to implement this feature in the final version of the firmware.

Figure 4.4 shows a comparison between the old and the new version. The new LUT

has a width equal to 19, because each cluster takes nine bits plus one additional bit flag indicating the presence of a second cluster. A new FIFO, called `Post_LUT_FIFO`, is also added to store the clusters, along with a merger which reads the FIFO and serializes the clusters. The `Post_LUT_FIFO` is a vector entity containing two FIFOs. The merger takes as input the clusters from the vector FIFO and outputs them in a serialized way. In most cases only one FIFO contains reconstructed SPs which are passed through the merger to the output. In case a SP with two clusters is processed, each FIFO accommodates one of the two clusters that are then sent out sequentially by the merger.

4.5 `I/O buffer` and changed position of `ICF`

Two minor modifications are applied to input and output interfaces, dealing with data format and buffering. The `I/O_buffer` consists in an entity which acts as a decoupling buffer at the output side of `clustering_block`, storing all signals to be output and sending them the following clock cycle. This will help in timing closure of the overall firmware thanks to the division of the long path between `clustering_block` and the following entity.

The *isolated cluster flag* (`ICF`) entity, implemented before the clustering block, checks if an input SP is isolated, setting an isolation flag within the SP word, accordingly. Although it has been considered in Stratix-V version, it turned out that its position must be changed, in Arria-10 version, from bit 25 to 31, and consequently the `EE` (End Event) must be shifted from bit 31 to 25. This swap is implemented inside the decoder. In the output stage, it does not need to be re-swapped because clusters do not have any isolation flag.

In case the double output is active, and both clusters and SPs are sent to the Event Builder (see section 4.10 for more details about `double_output`), SPs are caught before the decoder and sent directly to the output.

4.6 Required input and output signals

The adjustment of input and output ports to TELL40 standard signals drives the majority of the work done in this thesis, therefore it's useful to understand their specifications. Figure 4.5 shows a diagram of the clustering input and output signals.

- `BXID`: 12 bit wide signal, it is a counter carrying the bunch crossing number corresponding to data transmitted simultaneously to it. PCIe40 specifications assure that the `clustering_block` will receive SPs ordered by `BXID`, so that it can only increase until overflow.

- `FTYPE`: eight bit wide, it identifies the transmitted data type. Table 4.2 shows `FTYPE` values and their meanings. Note that only `0x33` and `0x3C` are valid at input: if a different value is found then the corresponding event is silently dropped.

- `FSIZE`: 16 bit wide, it counts the number of 32-bit data packets sent with a specific `BXID`. At the output it has the same meaning but data packets will contain clusters.

- `DATA`: 256 bit signal carrying eight 32-bit words. These contain SPs or clusters respectively at input and output. Some of the 32-bit words can be padded, in which case they are filled with zeroes. Such paddings do not have to be at the end or at

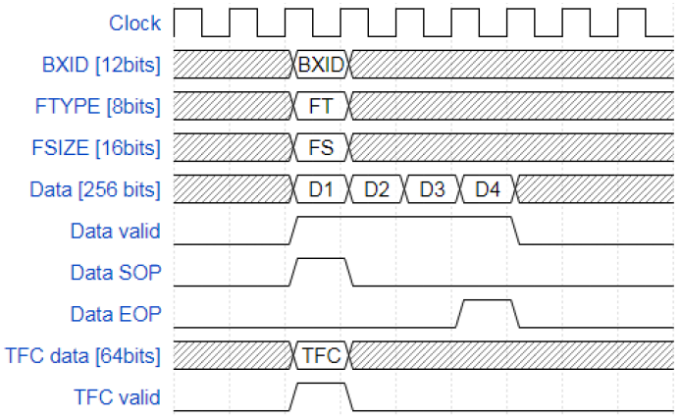


Figure 4.5: Diagram of the main signals at both input and output of the clustering firmware.

the beginning 256 bit word, for example a data packet can have valid SPs or clusters both in the most and least significant position with padding among them.

- `DATA_VALID`: one bit signal, if it is high the simultaneously transmitted `DATA` packet is to be considered valid. A `DATA` packet must be read each clock cycle the `DATA_VALID` signal is high. `DATA_VALID` does not necessarily have to stay high until the end of the event, but can also change its status during the transmission of the event.
- `SOP`: one bit, it is the *Start Of Package* signal which matches the first valid `DATA` packet of the single event. To be valid, also `SOP` needs a simultaneous `DATA_VALID` signal.
- `EOP`: one bit, it is the *End Of Package* signal, and it is transferred with the last valid `DATA` packet signal. This also needs a simultaneous `DATA_VALID`. If the event is contained within one `DATA` packet (because it has eight SPs or less) or is empty then `EOP` and `SOP` are simultaneous.
- `TFC`: 64 bit, it carries commands and the `BXID` of the event to which the commands have to be applied. It can be simultaneous or preceding the start of the corresponding event transmission, but it cannot follow that. The meaning of the single bits is reported in table 4.3.
- `TFC_VALID`: one bit, valid signal for the TFC system.
- `READY`: one bit, signal used to ask the preceding component for more data. This is not shown in Fig. 4.5. Note that this signal has the opposite direction with respect to all the others. For example, at the input side of clustering firmware the `READY` is an output port.

`FSIZE`, `FTYPE` and `BXID` signals must have the same value for the duration of the transmission of a single event, but they can assume different values before the `EOP` signal if the `DATA_VALID` signal is not active.

FTYPE value	Meaning
0x33	SuperPixels
0x34	Clusters
0x3C	Error word

Table 4.2: Values that FTYPE signal can assume at both input and output of the clustering firmware. Note that only 0x33 and 0x3C are valid at input. If a different value is found then the corresponding event is silently dropped.

Command name	Bit field	Length	Used in TELL40
	51	1	No
Multi event packet accept	50	1	No
	49	1	No
	48:43	6	No
	42:36	7	No
	35:34	2	No
Multi event packet destination address	33:18	16	No
Trigger type	17:14	4	No
Calibration type	13:10	4	No
Synch	9	1	Yes
Status and counter snapshot	8	1	Yes
Trigger	7	1	No
BX-veto	6	1	Yes
Non-zero suppression mode	5	1	No
Header only	4	1	Yes
TELL40 reset	3	1	Yes
FE reset	2	1	No
Event identifier reset	1	1	No
BXID reset	0	1	Yes

Table 4.3: Details of the TFC word showing all bit fields and if they are used by the TELL40 firmware. The BXID occupies the bit positions from 63 to 52 of the TFC word, carrying the BXID on which other commands are referred to. Commands with no name do not have a specification at the time of writing this thesis.

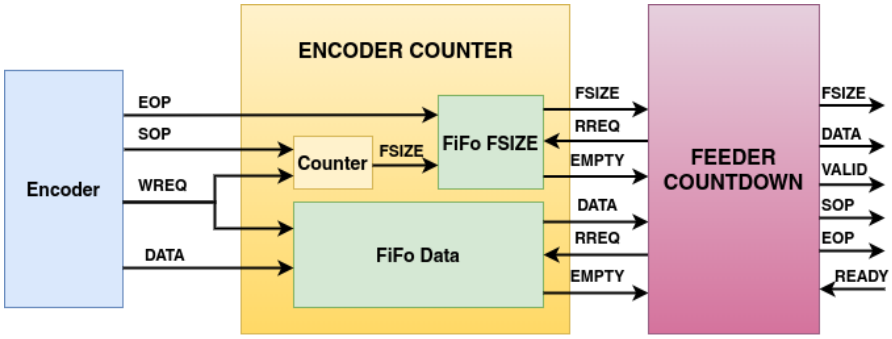


Figure 4.6: Logical architecture of the whole FSIZE system. Write request from encoder acts as write request for data FIFO and as signal to count for the counter. EOP is the write request for FSIZE FIFO and SOP is the reset of the counter.

4.7 FSIZE management implementation

The clustering firmware has a FSIZE signal at its input, carrying information about the number of SPs. Since this information is not currently used by the algorithm it is not considered. However, as far as the output side, the FSIZE signal must be correctly sent out by the clustering firmware. Therefore, a new entity is required to count, for each event, the number of reconstructed clusters. Since such a count has to be available at the beginning of each event, a new FIFO is needed, in which data packets are stored until the end of the event; at that point FSIZE is output with data packets. This entity must also perform a number of logical operations in order to handle the counter and, most of all, the output of FSIZE and data with the correct timing. Two new entities are therefore written, called encoder_counter and feeder_countdown. Their architecture is graphically represented in Fig. 4.6, along with their interconnecting signals.

- encoder_counter contains a counter that counts the number of 256 bit data packets flowing. The entity also contains two FIFOs, data_FIFO and FSIZE_FIFO storing the data packets and the counter values, respectively (see Fig. 4.6). The arrival of an EOP signal triggers the write request of the count into the FSIZE_FIFO, while the subsequent SOP signal causes the reset of the counter. The returned value must be multiplied by eight in order to have a cluster count instead of a packet count. Note that the first packet is not counted, since the reset of the counter happens with the arrival of a SOP signal. A piece of logic code is included to add non-padding words in the last packet. Since the last packet is also counted by the counter, this piece of code solves both the problems of the missing first packet and of empty words in the last packet. Note that this does not include padding before the last cluster, FSIZE could be, therefore, bigger than the actual number of clusters.

- feeder_countdown is a feeder performing logical operations in order to manage the output of the two FIFOs, data_FIFO and FSIZE_FIFO. It is a FSM (*Finite State Machine*) built with the use of its truth table, which contains the values of the output signals for each combination of input signals. There are various output signals: three read request signals for the FIFOs of the encoder_counter and for the TFC_FIFO, two different valid signals, and the EOP signal. The input signals, instead, are

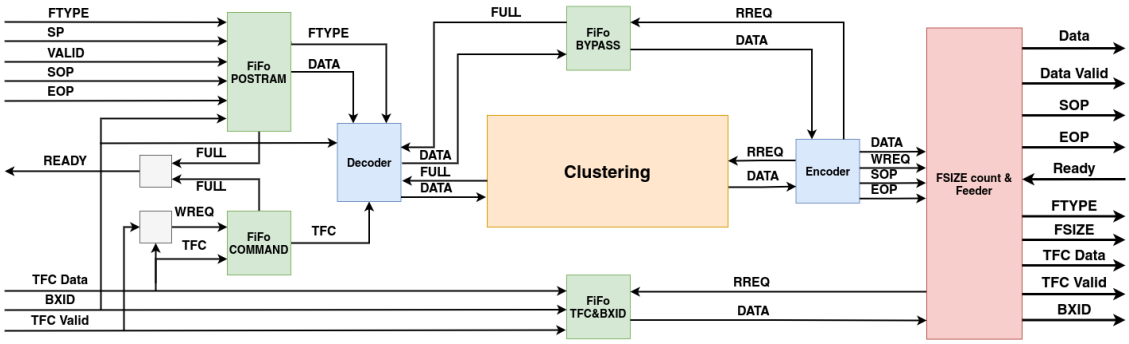


Figure 4.7: Logical architecture of the clustering firmware with TFC management implementation, showing `command_FIFO`, `TFC_FIFO` and `bypass_FIFO`. `Command_FIFO` is of the type *ahead*.

three: the `HOLD` signal coming from the output side and two `empty` signals from the `encoder_counter` FIFOs. A set of logic functions has been deduced, one for each output, with the use of a software program capable to generate combinatorial logic functions once a truth table is provided. The use of these type of functions leads to a faster component without any delay, while a process, much easier to write in VHDL, would have introduced an undesired delay affecting the overall throughput.

It has been verified that the value of `FSIZE` is correct and returned with the correct timing by the `feeder_countdown` using ModelSim simulation [73].

4.8 TFC management implementation

This section includes several modifications of the original firmware, whose purpose is to implement a system to handle the various TFC commands, reported in table 4.3. The TFC commands are attributes of the corresponding data packets [4], and those involving the TELL40 firmware of the PCIe40 boards are: Synch, Snapshot, BX-veto, Header-only, TELL40 reset, and `BXID` reset. Only the Snapshot command needs a management system in the clustering firmware, for all the others it is sufficient to bypass the corresponding data packets from input to output leaving them without any modification. The TFC commands are not necessarily simultaneous to the corresponding data words, therefore, a new system is needed to store all the interesting commands and execute them when the correct `BXID` comes. This is done with the addition of a new FIFO, called `command_FIFO`. Since the TFC commands could be simultaneous to the corresponding `BXID`, a special FIFO is needed, capable to send out immediately the written value. This type of FIFO is commonly called *ahead* and interprets a read request as a read acknowledge. When it receives a read request, it assumes that data have been already read and shows the next available datum.

The logical architecture of TFC management system is shown in Fig. 4.7. A TFC word is written into the `command_FIFO` if there is a `TFC_VALID` signal and a TFC command of interest. The `BXID` is used by the `decoder` to decide whether the data word has to be

[4]The data packets have in general a different content depending on the case, and they do not contain SPs.

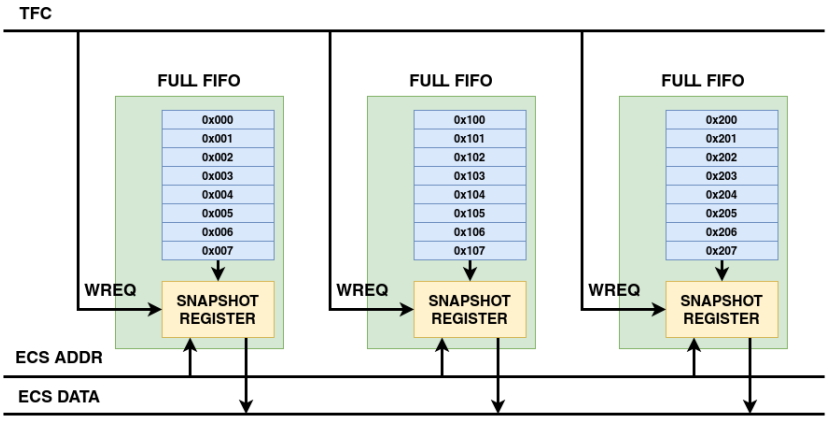


Figure 4.8: Diagram of the snapshot system. Each FIFO contains a `snapshot` register which is written each time a `snapshot` TFC command is received. Stored data can be read with the ECS system.

processed or bypassed (this is discussed in more detail later), and it is also written into the `TFC_FIFO` altogether with the coming TFC word. The TFC word written inside the `TFC_FIFO` also enters the `command_FIFO`. The `TFC_FIFO` is used to transmit the TFC words and the `BXID` from input to output, since they must not be modified. Such a FIFO was already present in the Stratix-V version but it carried only the TFC words, while in the Arria-10 version it has been recreated larger in order to carry both the TFC words and the `BXID`.

TFC commands requiring only bypass

In order to bypass the data words, a new FIFO is instantiated as shown in Fig. 4.7. It carries the data words, already unpacked, from the `decoder` to the `encoder`, preventing them from being clusterized. When the `decoder` finds a `BXID` corresponding to one of the commands requiring bypass (those of table 4.3 involving TELL40, except for the snapshot command) the associated data packet is written in the `bypass_FIFO`, and a peculiar `EE` signal is sent to the clustering chain. This `EE` has an additional bit set indicating that a data packet is inside the `bypass_FIFO`, and it is propagated inside the clustering machinery. The `encoder` finds such a special `EE`, and it reads the data packet from the `bypass_FIFO`.

The `encoder` is also modified, since a new entity is implemented, called `bypass_merger`, based on the `merger` described in section 3.2.4. Depending on the presence of the peculiar `EE`, as described above, the `encoder` outputs data exiting from the clustering machinery or those from the `bypass_FIFO`.

Snapshot command

The snapshot command is a central request to all the components of the TELL40 firmware asking for information about their current status. As soon as this command is received, the corresponding data packet is bypassed, as described in the previous sub-section, and information about the occupancy level of all FIFOs of the clustering is provided. This does require the instantiation in the `full_fifo`, which is the general FIFO entity, a new

register, called snapshot register, capable to read the number of stored words inside the FIFO each time a snapshot request is received. This information can be accessed by the ECS system (see section 4.3 for details about the ECS protocol). The functional diagram of the snapshot system is shown in Fig. 4.8.

Simulation

In order to simulate the TFC management system it is useful to feed the clustering_block with some TFC commands, and see if it correctly bypasses them. With this aim the loop-ram is enlarged from nine to ten 32-bit words with the last being filled with the low half of the TFC words. In the majority of cases such TFC semi-words are all zeroes, meaning that no command is sent. In order to simulate a realistic behavior some words are modified to codify a synch command [5] requiring the bypass. Such words are those corresponding to unit-length data, namely those with simultaneous SOP and EOP signals, as it is in the real world. Two different configurations are tested, one with all the unit-length event with a synch command associated, and a second configuration in which only one word out of two has the associated command.

4.9 FTYPE management system

As reported in Tab. 4.2 the three values of the FTYPE signal involving the clustering firmware are 0x33, 0x34 and 0x3C. They indicate the type of data transmitted: SPs, clusters, or errors, respectively. At the input side, the FTYPE signal is used to generate the write request signals for the postram_FIFO, the command_FIFO, and the TFC_FIFO. For the first two, only a values of FTYPE = 0x33, simultaneously to a DATA_VALID signal, can enable the write request, while for the latter both values, 0x33 and 0x3C, are good. In this way if a different (non valid) value of FTYPE is received then the corresponding data is dropped. At the output side instead, the FTYPE signal can also be equal to FTYPE =0x34, namely the system outputs clusters instead of SPs. Its value is determined by the double_output and the error_manager entities, as described in Sec. 4.10 and Sec. 4.11.

4.10 Double output

The double_output entity is implemented to output both clusters and corresponding SPs. This feature is required from the collaboration to check and test the functioning of the clustering firmware during the commissioning period at the beginning of the data taking. This feature cannot be maintained during the full operations at high luminosity because of bandwidth limitations.

The double_output entity receives clusters from the feeder_countdown entity and the corresponding SPs from the postram_FIFO. An additional signal tells if an event is bypassed because of a TFC command, and the associated data packet is not read again by the postram_FIFO. Its internal structure replicates the functionalities of the encoder_counter and the feeder_countdown, indeed it contains a SP_FIFO and a SP_FSIZE_FIFO analogous to the data_FIFO and FSIZE_FIFO, respectively, placed inside encoder_counter. An

[5] The "synch command" is one of the TFC commands reported in Tab. 4.3.

additional instantiation of `feeder_countdown` managing the two FIFOs is also contained. This allows the `double_output` to output the SPs with their own `FSIZE`.

This `double_output` entity is disabled by default in the clustering firmware and it can be enabled by writing in the double output register (address `0x0400`, see table 4.1) via ECS protocol, if desired. By default it propagates clusters received from input to output, instead if it is enabled it outputs clusters and `FTYPE` = `0x34` followed by the corresponding SPs with their own `FSIZE`, same `BXID` and `FTYPE` = `0x33`. Clusters and SPs are identified solely by the change in `FTYPE`, no `EOP-SOP` signal is placed to separate them.

4.11 Error managing

A unified error managing standard in the firmware is of fundamental importance to report problems to the control room during data taking runs, in order to allow qualified personnel to react in case of problems. Therefore, a new entity, called `error_manager`, is designed to manage errors coming from outside and those internally generated. When an error is received, the `error_manager` drives the clustering firmware in the process of outputting all intact data inside the FIFOs, if they are present. After that it locks the firmware, whose normal operation can be restored with a reset.

All parts of the TELL40 firmware, in case of error, send out `FTYPE` = `0x3C`, and `DATA` signal contains an error word carrying information about the error itself. The `error_manager` gets information about the empty FIFOs, error signals from all entities, and error words from outside, in case `FTYPE` = `0x3C` is received. It drives enable signals and outputs all the required signals. It cannot handle multiple errors at the same time, therefore if an error rises the whole `clustering_block` has to terminate any processing without further errors. A logic scheme of the interconnections between the `error_manager` and the other components of the clustering firmware is shown for clarity in Fig. 4.9.

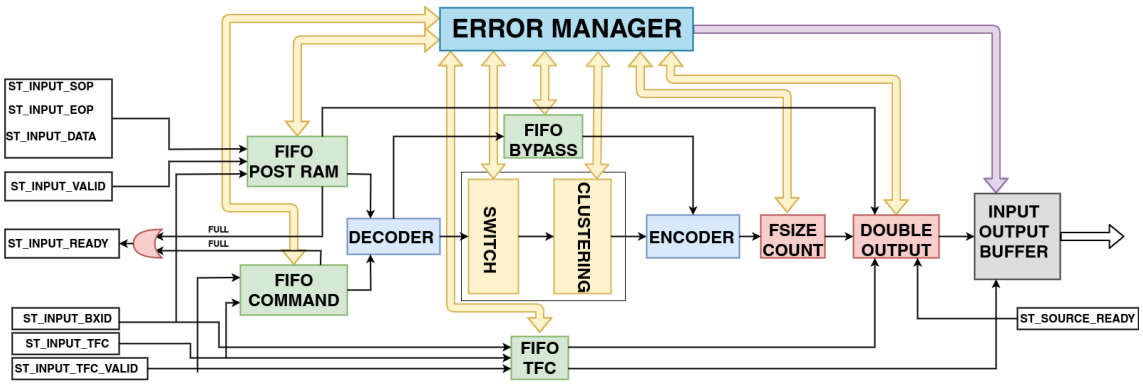


Figure 4.9: Interconnections between `error_manager` and all other components of clustering firmware. Yellow big arrows stand for empty, error and enable signals, the purple big arrow is the data output by the `error_manager`.

If a valid external error from the TELL40 firmware is received, the `clustering_block` sends a signal triggering the `error_manager`. At first, the `error_manager` closes the input side by disabling any writing into the `postram_FIFO`, `TFC_FIFO` and `command_FIFO`, and, subsequently, it monitors the state of all other FIFOs in the chain, waiting for them to

empty. The empty signals of the FIFOs are those of the read side, and therefore they are also referred to as `rdempty` signals. These signals come four clock cycles late with respect to the actual emptiness of the relative FIFO, and this delay causes an issue when all `rdempty` signals are high, but there is still some data inside the FIFOs [6]. In order to avoid this problem the `error_manager` simply waits for 100 clock cycles, from the instant when all the `rdempty` signals are high, to declare that all the clustering firmware is actually empty. At that time it outputs the error word, `BXID`, `FSIZE`, `SOP`, `EOP`, and `FTYPE` = `0x3C`, where the first three (error word, `BXID`, `FSIZE`) were previously read and stored by the `error_manager`. Nothing will happen until a reset signal is received.

If an internal error is generated in the clustering firmware, all data still available inside the FIFOs are processed before sending out the error word and locking the whole firmware. There are two possible cases to be analyzed: the error message comes from the `double_output` entity or from any component preceding the `double_output` entity. In the first case, the `error_manager` does not have to check any FIFOs (no other FIFOs are present in the firmware chain after the `double_output`), and it outputs a trailing `EOP` signal with `FTYPE` = `0x33` if the last event is not finished; the error word with all other signals are sent with the subsequent clock cycle. Instead, if the error is not generated by the `double_output`, the situation is more complex, and all FIFOs preceding the crashed entity are disabled at the write side. The last event is likely to be not complete, and this fact has a major consequence for some vectorized FIFOs, namely the `postswitch_FIFO` and `out_FIFO`. The subsequent entities cannot read the components of the vectorized FIFOs in a parallel way because some of them are empty, although there are still some data in the others. This would cause a data loss in the last event, but since it is not complete it will be discarded afterwards by the `feeder_countdown` [7]. The `error_manager` uses, therefore, a logic function of empties of single FIFO components to determine if the vectorized FIFO can be considered actually empty. `transf_FIFO` and `bypass_FIFO` are also vectorized FIFOs, but there is no entity before them that can generate errors. Lastly, the `error_manager` waits for all the FIFOs to empty, with a safety margin of 100 clock cycles, and then outputs all signals as in the previous case of external errors. The error word is a 32-bit error code, repeated eight times, whose bits indicate which entity has crashed.

4.12 Pattern reversal

As shown in Fig. 3.3 the size of the clusters can span a large range, and, therefore, the associated topology can be very different, even if the large majority of the clusters is made of a very limited number of pixels. It is possible to have large clusters, most likely produced in the sensors placed in the vicinity of the interaction point, which are released by particles passing nearly parallel to the sensors, as it can be seen in Fig. 2.3. Those particles produce swipes of active pixels, and, unfortunately, the clustering algorithm described in this thesis can only partially resolve them. As shown in Fig. 3.2, the algorithm looks for the L-pattern

[6] The same issue also arises when all FIFOs are empty but some processing entity has still data inside (note that only FIFOs have an empty signal).

[7] The `feeder_countdown` is the last entity before the `double_output`, and it cannot generate errors. The last event coming out is, therefore, complete because if it was not, the data of the event will remain inside the `data_FIFO`. This comes out by the fact that the `error_manager` checks, in this case, the emptiness of `FSIZE_FIFO` only, without paying attention to the state of `data_FIFO`.

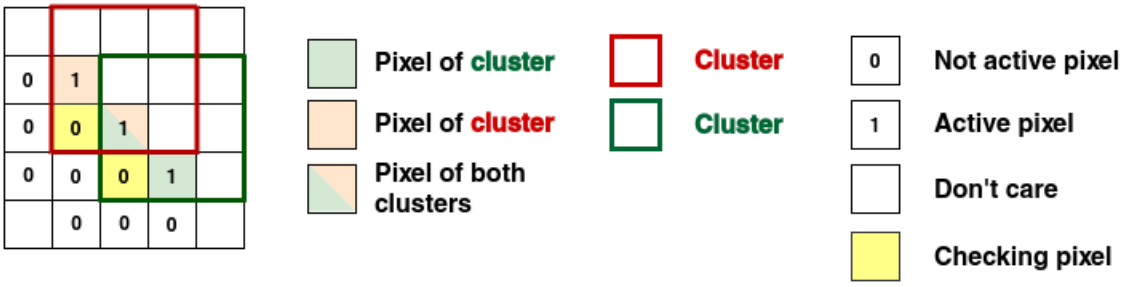


Figure 4.10: Pathological situation in which a big cluster going from bottom-right to top-left is resolved as two separate clusters, although only three pixels are active. A longer swipe would be even worse.

having a bottom-left corner of inactive pixels. If a big cluster goes from bottom-left to top-right in the considered matrix of SPs, the algorithm will find only a single cluster at the bottom-left end of the swipe. If the cluster is so big to be split in many matrices, the algorithm may return different reconstructed cluster candidates, one for each matrix. On the contrary, if the swipe of active pixels goes from bottom-right to top-left, the algorithm will probably find many different clusters, even in the same matrix of SPs, maybe partially overlapping. Such a pathological situation is illustrated in Fig. 4.10.

Since particles come from the center of the VELO subdetector, the type of swipe is determined by sensor position and, therefore, by its identification number. The sensors labelled as 0 and 3 will likely have to process bottom-left-to-top-right big clusters, while sensors labelled as 1 or 2 will likely have to handle big clusters of the other type, causing pathological situations similar to that shown in Fig. 4.10.

While for the bottom-left-to-top-right big clusters the algorithm cannot be easily improved, unless changing the whole architecture, for the bottom-right-to-top-left ones it is possible to largely improve the situation by "reversing" the L-patterns for the sensors 1 and 2, and leaving unchanged those for the sensors 0 and 3. This makes the response of the algorithm to the different type of big clusters almost perfectly symmetrical, improving the robustness of the algorithm in presence of such (very rare) big cluster swipes. The reversed L-patterns are shown in Fig. 4.11, in analogy to the standard patterns shown in Fig. 3.2, as described in Sec. 3.1.

With this purpose a new version of the `clustering_neighbour` is written and implemented in the clustering firmware. Since the checking pixel is at the bottom-right corner of the 3×3 sub-matrix a new LUT is needed, not only with a different content but also with a different width. This is due to the fact that the integer part of both the column and the row, of the reconstructed cluster candidates, is at most 1 for the standard L-patterns, while it can also assumes the value of 2 (only for column) for the new reversed pattern. Some examples of different cluster configurations and the associated L-patterns are shown in Fig. 4.12. The a new LUT, for the reversed patterns, has a size of 512×9, instead of 512×8 used for the standard patterns. Only the column needs an additional bit, while the row has the same width.

During the implementation of the the reversed L-patterns, as described above, the LUT used to determine the cluster center-of-mass is improved in order to account for the presence of "detached" pixels in the 3×3 sub-matrix identified by the checking pixel. An

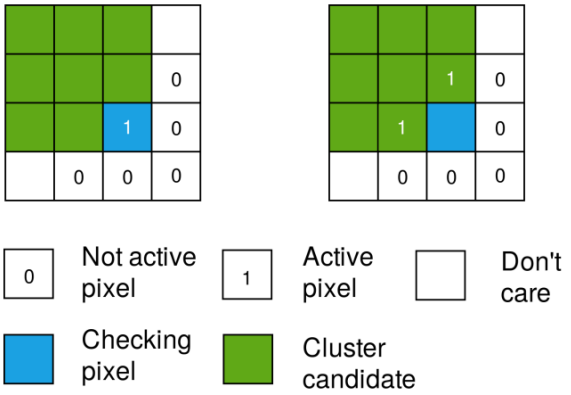


Figure 4.11: Reversed L-patterns searched by the firmware to find the "checking pixel" in the sensors 1 and 2. The 3×3 green matrix contains the pixels used to determine the position of the cluster.

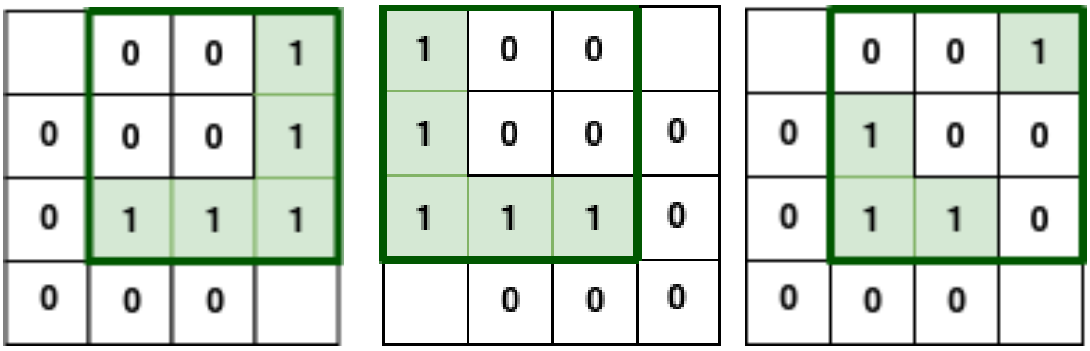


Figure 4.12: (Left) The cluster that has the biggest column (7/5) in the case of the normal pattern. (Center) The cluster that has the smallest column (3/5) in the case of the new reversed pattern, note that integer part is 0. (Right) Double cluster inside a 3×3 sub-matrix, the pixel in the top-right corner is the checking pixel of another 3×3 sub-matrix.

example of such a configuration is shown in Fig. 4.12 on the right pad. The detached pixels do not enter the calculation of the the center-of-mass, and they will be recovered with different L-patterns providing the correct clusters.

The design of the LUT for the new reversed pattern also accounts for the different available 3×3 sub-matrix positions inside the matrix of the chain, and, therefore, for the possible slots that the checking pixel can fill inside the matrix. The comparison with the standard and the reversed patterns is shown in Fig. 4.13.

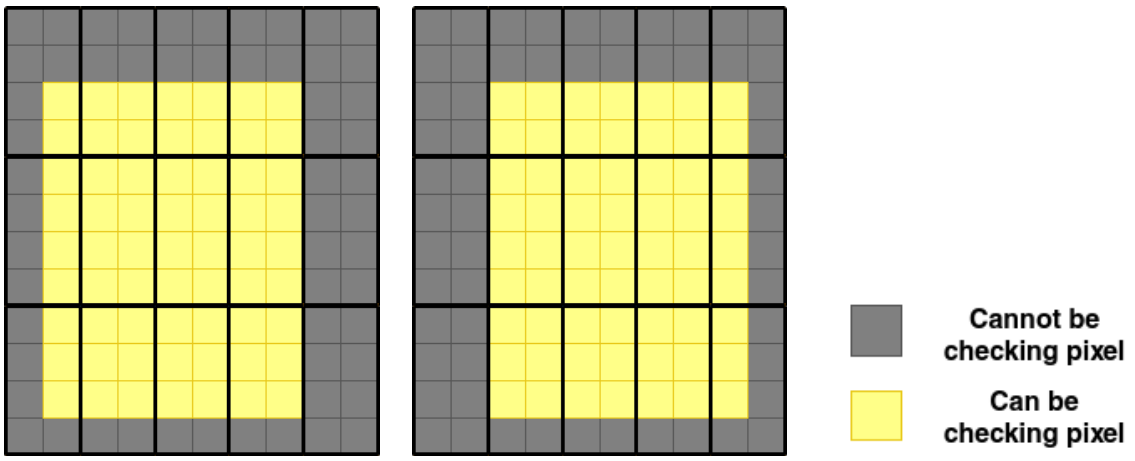


Figure 4.13: Pixels inside the matrix that are checked for 3×3 cluster candidates, with the standard pattern (on the left) and with the reversed pattern (on the right). Since the positions are different a reversed version of the LUT is needed.

4.13 Testing and debugging

The integration of the original clustering firmware required several modifications and the addition of many new entities, as described in the previous sections. Therefore, an extensive and accurate work of testing and debugging of the all clustering firmware is carried out. The `test_bench`, providing all the signals required by the firmware, is modified in order to perform the following tests.

- TFC command execution. The clustering firmware is fed with various types of TFC commands to check that each command is executed appropriately on the right event, avoiding the clusterization of non-SP input data. TFC commands are directly written in the loop-ram files in the same way as described in Sec. 4.8.

- `double_output` synchronization. As described in Sec. 4.10, the `double_output` allows to send both clusters and the corresponding SPs as an output of the clustering firmware. A series of tests are carried out to check that the correct synchronization between clusters and SPs is met, while avoiding both overlapping of output signals or delays between the end of the cluster data packet and the subsequent SPs. For this purpose, the `double_output` is cyclically turned on and off by writing in the double output register with ECS protocol (see table 4.1). A counter is implemented inside the `test_bench`, triggering an ECS write when it reaches a predefined value.

- `HOLD` signal responsiveness. The data flow inside the TELL40 firmware is controlled by the exchange of hold signals between entities, ensuring that the data-receiving block is ready to accommodate new data. While receiving hold signals from the subsequent firmware component, the clustering must stop data transmission, avoiding data loss. To test the entity responsiveness to those requests, an intermittent `HOLD` signal is generated, using a system similar to the `double_output`. A counter keeps the hold high when its value is below a given threshold, periodically injecting halt requests into the firmware.

- Internal and external error managing. The correct managing of error signals is a crucial step during data processing and it requires a separate entity, as described in Sec. 4.11. As an error is generated, the firmware must act appropriately and quickly to avoid mis-reconstruction of data, while providing the required debugging information.

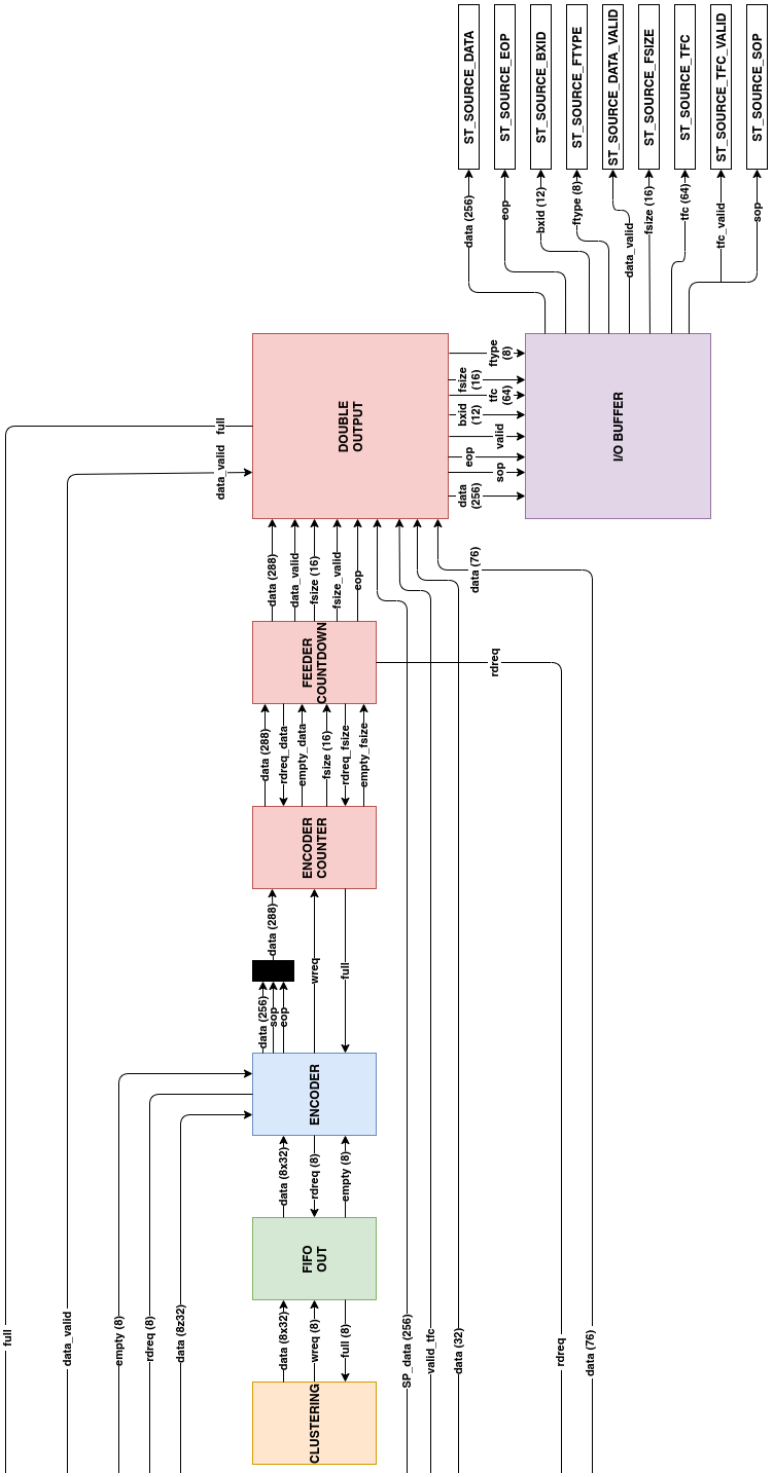
  In order to simulate an error coming from inside clustering firmware, the `test_bench` sets a bit in the `error_register` with the ECS protocol. All types of error are tested. A counter is instantiated inside the `test_bench`; when it reaches a predefined value, it triggers an ECS write. A system to generate a random error is also developed.

  External errors are based on the same type of counter as of the internal errors, but they are sent only if the simultaneous datum has unit length (that is, `SOP` and `EOP` are simultaneous).

  If an error is generated, both internal or external, the `test_bench` writes `FTYPE` = `0x3C` and a corresponding error word is sent, carrying the information about the type of error and its origin.

4.14 Final clustering firmware

Figure 4.14 shows the final diagram of the whole clustering firmware, after the integration within the TELL40 firmware, as described in this chapter. The resource usage in the Arria-10 FPGA amounts to the 10% of M20K memories and the 26% of ALMs. The entire FPGA clustering project is contained in the repository in Ref. [74], including also all the work of integration and optimization performed in this thesis.


ST_SOURCE_DATA
ST_SOURCE_EOP
ST_SOURCE_BXID
ST_SOURCE_FTYPE
ST_SOURCE_DATA_VALID
ST_SOURCE_FSIZE
ST_SOURCE_TFC
ST_SOURCE_TFC_VALID
ST_SOURCE_SOP
data (256)
eop
bxid (12)
ftype (8)
data_valid
fsize (16)
tfc (64)
tfc_valid
sop
I/O BUFFER
DOUBLE OUTPUT
full
data_valid
data (288)
fsize (16)
eop
rdreq
data (76)
FEEDER COUNTDOWN
rdreq_data
empty_data
rdreq_fsize
empty_fsize
ENCODER COUNTER
wreq
ENCODER
sop
data (8x32)
rdreq (8)
empty (8)
FIFO OUT
wreq (8)
full (8)
CLUSTERING
empty (8)
rdreq (8)
data (8x32)
SP_data (256)
valid_tfc
data (32)

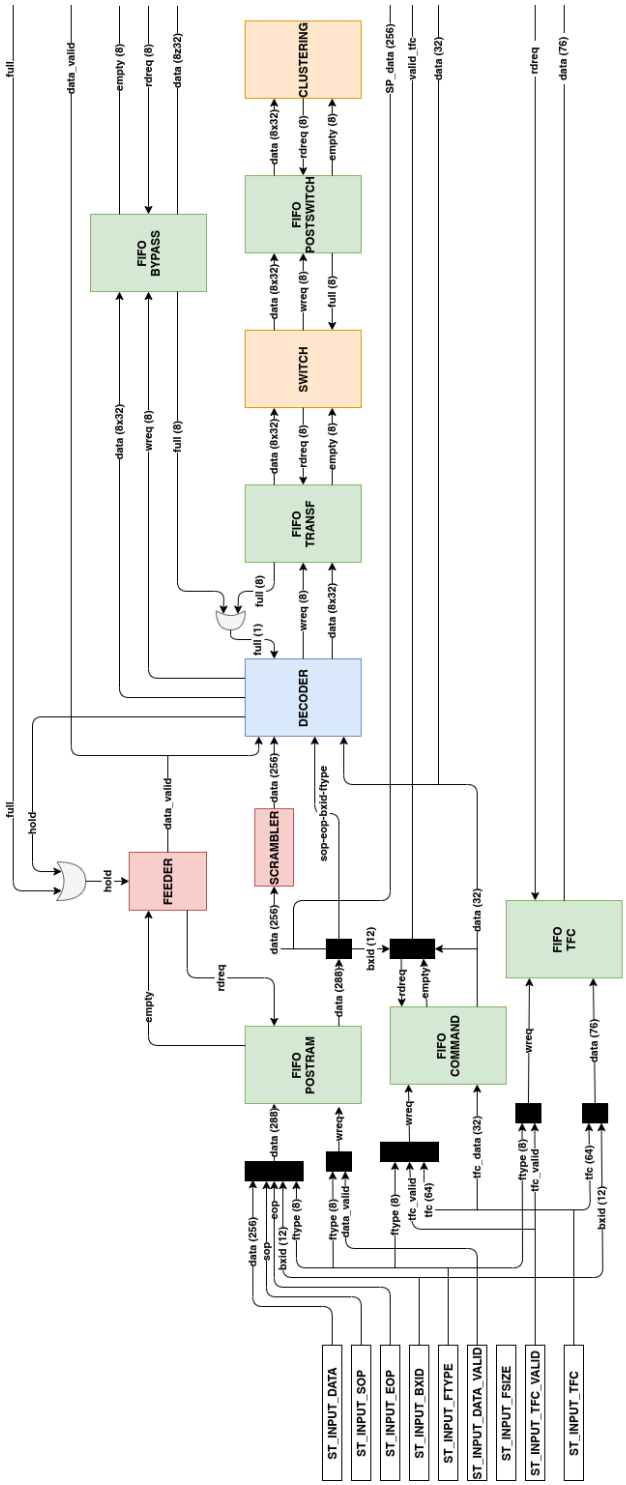


Figure 4.14: Detailed diagram of the clustering firmware after the work of this thesis.

Chapter 5

Physics performance and throughput

This chapter reports all the extensive studies and tests performed to determine the accuracy and the robustness of the FPGA-based clustering algorithm using fully realistic simulated samples at the LHCb Upgrade running conditions. Various quantities related to clustering and tracking quality are analyzed and compared with the CPU baseline algorithm. A high level simulation, written in C++ programming language, capable to emulate all the features of the clustering algorithm is integrated in the official software of the experiment in order to reconstruct realistic simulated pp collisions events at LHCb-Upgrade. The gain of the event processing throughput of the first stage of the HLT is also measured and presented.

5.1 Introduction

The baseline for the LHCb Upgrade foresees to run the VELO clustering in the Event Filter Farm, at the HLT1 stage, as it was previously done in the past LHC Runs. Since the two algorithms, the FPGA-based and the baseline, are different, a comparison of the performance is needed to verify that FPGA-based algorithm provides clusters with the same quality, or very close, of those returned from the baseline algorithm, which is executed on CPUs. Here, the main differences of the two algorithms are briefly summarized; they are mainly due to the approximations done in the FPGA-based algorithm to make it highly parallel.

- Clustering of non-isolated SPs is done by selecting a 3×3 sub-matrix, therefore, if a cluster does not fit entirely inside such sub-matrix, some of its pixels are not used to calculate the position of its center-of-mass.
- Big clusters may be split up between different matrices of the chain, leading to the reconstruction of fake clusters from parts of the original cluster.
- Events containing many non-isolated SPs may not fit inside the chain of matrices, and overflow SPs are resolved as isolated though they are not.
- Center-of-mass coordinates of the cluster candidate are rounded down, while the baseline algorithm does not contemplate any rounding. FPGA clusters are encoded in a fixed-point format with three fractional digit, giving a resolution of ⅛ of pixel.

The impact on the final performance of all these approximations are studied with the help of the official LHCb Upgrade simulation, written in C++, that can simulate the operation of both clustering algorithms at low level.

5.2 The official LHCb Upgrade simulation

In LHCb the task of modeling the behavior of the spectrometer for the different type of events occurring in the experiment is carried out by two separate applications called GAUSS and BOOLE [75]. GAUSS generates the initial particles and simulates their transport through the LHCb detector, whilst BOOLE reproduces the different subdetectors responses and their digitization converting the data in the same format provided by the experiment electronics and the DAQ system. After digitization real data and Monte Carlo data follow the same path through trigger, reconstruction and analysis procedures. GAUSS is customized by choosing and configuring the appropriate set of algorithms to execute in a given sequence and other suitable components for the various tasks. In LHCb the production of particles coming out of the primary pp collision of the LHC beams is handled by default with PYTHIA [76], a general purpose event generator, whilst the decay and time evolution of the produced particles is delegated to EVTGEN [77] package. Lastly, in GAUSS the simulation of the physics processes undergone by the particles traveling through the detector, is delegated to the GEANT4 toolkit [78, 79]. A general and more detailed description of the whole framework can be found in Ref. [80].

MOORE is the LHCb high-level trigger (HLT) application [81]. It is responsible for filtering an input event of 30 MHz of visible collisions down to an output rate of around 100 kHz. It is made of two sequential stages: the HLT1, which performs a fast track reconstruction and makes a decision based on one- and two-track objects, and HLT2, which performs a high-fidelity reconstruction and makes a decision based on the full detector read-out information. During the data taking periods MOORE will be executed by the Event Filter Farm in real time. As far as the simulation, instead, it is executed at offline level processing the output of the BOOLE application. MOORE has been modified, in the context of this thesis work, in order to insert the code responsible for the simulation of the new clustering algorithm, and to run the HLT1 reconstruction algorithms by using the FPGA VELO clusters as input.

In order to design the FPGA-based algorithm and to study its performance three different simulated samples are used[1]. They differs one from another by the event topology. All of them are produced at the LHCb Upgrade conditions, as follows: $\sqrt{s} = 14\,\text{TeV}$, LHC bunch spacing = 25 ns, $\mathcal{L} = 2 \times 10^{33}\ \text{cm}^{-2}\,\text{s}^{-1}$, $\nu = 7.6$[2]. The first one is a sample of generic inelastic events, the so-called Minimum Bias sample, while the others two are filtered samples containing an hard collisions in each event, which has produced a $B^0 \to K^{*0}e^+e^- \to [K^+\pi^-]e^+e^-$ decay, and a $B_s^0 \to \phi\phi \to [K^+K^-][K^+K^-]$ decay. A summary of the three samples is reported in Tab. 5.1. The sample containing the Minimum Bias events, is examined in detail in the following sections. No relevant differences between the three available samples are observed, apart from the fact that the events containing b-hadron decays show a little higher occupancy on average, into all LHCb

[1] Actually, many other simulated samples are used, but for the purpose of this thesis results only the most important ones are shown.

[2] ν is the mean number of primary vertices per bunch crossing.

Nb. Events	Decay Mode
50 000	Minimum-Bias
10 000	$B^0 \to K^{*0}e^+e^- \to [K^+\pi^-]e^+e^-$
1 000	$B_s^0 \to \phi\phi \to [K^+K^-][K^+K^-]$

Table 5.1: Simulated data samples.

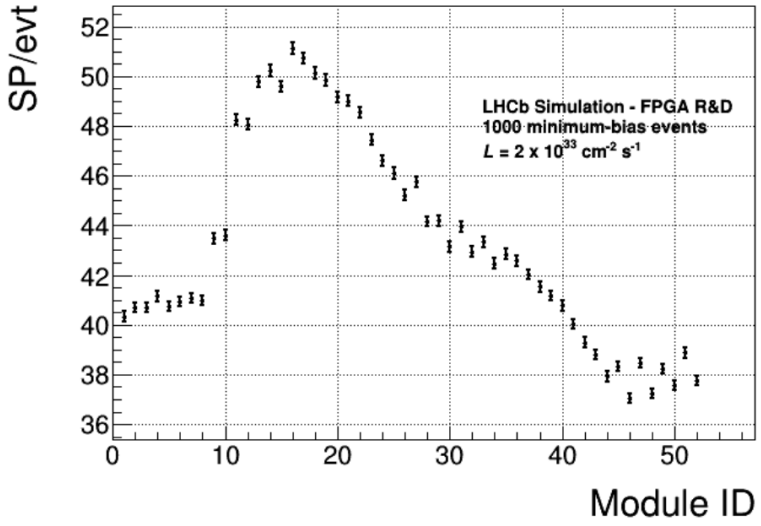


Figure 5.1: Distribution of the average number of SPs per event as a function of module number. 1 000 minimum bias events have been used.

subdetectors, and, therefore also into the VELO. This is expected, since a b-hadron is produced by an hard pp collision. The distribution of the size of the simulated clusters for the Minimum Bias events is shown in Fig. 3.3, while Fig. 5.1 reports the distribution of the average number of SPs per event as a function of the VELO module number. It can be noted that VELO modules near the interaction point (module 16 is the closest to the nominal interaction point), have a larger average number of SPs, about 50 SPs per event. This region is of particular interest for the performance of the FPGA clustering since the larger average number of SPs can cause, more likely, overflows of SPs for matrix chains (see also Sec. 5.8).

The simulated sample enriched with the presence of electrons from B decays is used to study in detail the reconstruction of clusters produced from "good" electrons, that are in general more complex (see section 5.7 for more details) than other clusters produced by different track particles. The simulated sample filtered with the presence of the $B_s^0 \to \phi\phi$ decays, instead, is used to verify the robustness of clustering algorithm when crowded events are processed. The event throughput must stay above 30 MHz on average and the physics performance must not degrade. This sample is also very useful to study possible issues that can raise because of very close (in the space) real tracks (kaons from the $\phi \to K^+K^-$ decay). The FPGA-based algorithm returned to be very robust in this case, for all the tested scenarios and/or configuration, with a final performance almost indistinguishable

from the baseline algorithm.

The performance achieved with the minimum bias sample, the most abundant, are illustrated in the following sections, unless otherwise stated.

5.3 The FPGA clustering emulator

An emulator for the FPGA clustering is written by the LHCb-Pisa group in C++ programming language. It intercepts SPs in the simulation chain and converts them to clusters with the same logical operations performed by the FPGA-based algorithm. The output clusters are processed by the Moore application, which is able to apply the same reconstruction algorithms on clusters coming from both FPGA and CPU emulators. Great care is taken to ensure a perfect correspondence between the high level simulation and the real FPGA operations. Nonetheless, some differences cannot be completely removed. For instance, the output of the clustering algorithm slightly changes if the same ensemble of SPs is fed but with a different order. The simulation provides SPs ordered by sensor column and row, while in the real implementation the order is unpredictable. This is relevant for the clustering algorithm since the output, the reconstructed clusters and their position, slightly depends on how the matrices are filled. Major differences are due to events with many clusters, some of which overflows the matrix chain, or due to long swipes of active pixels that are split between different matrices. However, as it is shown in the following sections, these events are very rare and differences due to the order of the input SPs are completely negligible for the assessment of the final performance.

5.4 Clustering efficiency

The clustering efficiency is defined as

$$\epsilon_{\text{clu}} = \frac{N_{\text{linked\_MC\_hits}}}{N_{\text{MC\_hits}}}$$

where $N_{\text{linked\_MC\_hits}}$ is the number of MC hits [3] with a linked reconstructed cluster. An algorithm included in the official LHCb simulation determines whether a MC hit and a reconstructed cluster are linked, running on every pixel of the cluster and looking for associated hits [4]. $N_{\text{MC\_hits}}$ is the total number of MC hits. Figure 5.2 shows the clustering efficiency as a function of different quantities, both for the baseline and for the FPGA algorithm. The plots on the left side are for all clusters associated to the reconstructible VELO tracks, while the plots on the right side are made by selecting only clusters associated to reconstructible long tracks (see Sec. 2.7 for all LHCb track types). All plots in this sections are made with 1 000 minimum bias events. As can be seen, the FPGA-based algorithm returns a response almost indistinguishable with respect to the baseline algorithm.

[3] The MC hit, associated to a simulated particle, is a set of information about the interaction of the particle with the detector material, like the position of the intersection and the amount of energy released.

[4] The association between a pixel and a MC hit is performed at an earlier stage, inside BOOLE, considering the passage of the particle through the detector material and the amount of energy released.

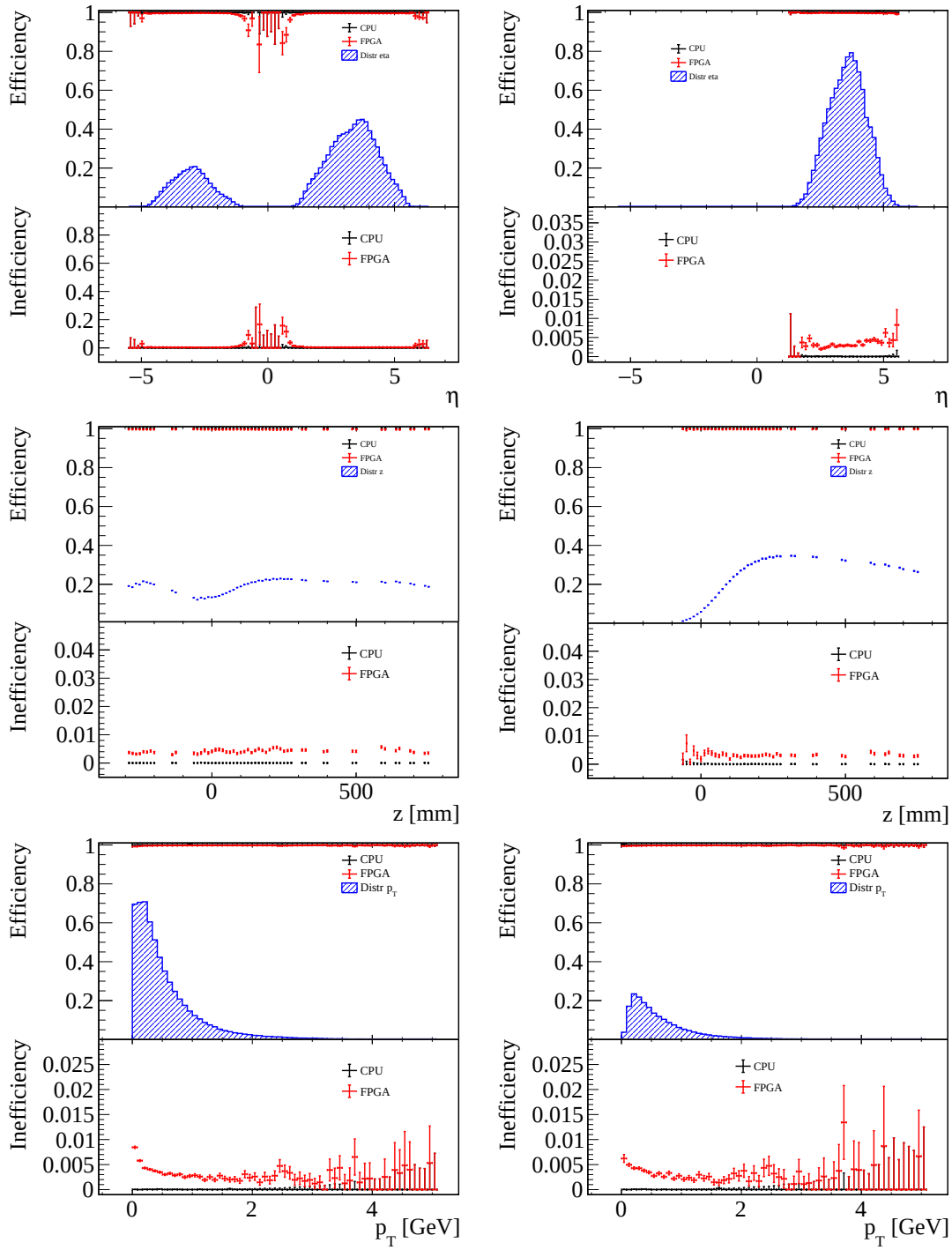


Figure 5.2: Clustering efficiency (and inefficiency) as a function of η, z, and p_T of the associated track. Results for the reconstructible VELO tracks (on the left), and for the reconstructible long tracks (on the right). The distribution of the η, z, and p_T variables is also superimposed (blue hatched histogram).

5.5 Robustness to split-up of large clusters

One of the features of FPGA clustering algorithm that may decrease its tracking performance is the fact that large clusters may be split up and more than one cluster center can be found. In order to study this issue 50 000 minimum bias simulated events are used, selected on the base of the fraction of large clusters they contain. Large clusters are here defined as those with more than nine hit pixels. Figure 5.3 shows the distribution of the fraction of large clusters per event. Such distribution is divided into four equally populated quantiles, additionally the last one is further split in two to better investigate the effect of the highest percentage of large clusters. Clone and ghost rates are calculated in the five regions just presented. The clone rate is defined as

$$\text{clone rate} = \frac{N_{\text{clones\_tracks}}}{N_{\text{tracks}}},$$

where two tracks are considered clones if they share at least 70% of hits in each tracking detector (VELO, UT and SciFi), and N_{tracks} is the number of reconstructed tracks. The ghost rate is defined as

$$\text{ghost rate} = \frac{N_{\text{ghost\_tracks}}}{N_{\text{tracks}}},$$

where a ghost track is defined as a track that share less than 70% of hits with one of the MC tracks, in at least one of the three tracking detector. N_{tracks} is still the number of reconstructed tracks. A ghost track is a track obtained connecting the hits of different tracks. Ghost rate is generally larger in conditions of higher detector occupancy. Figure 5.4 shows the clone rate and the ghost rate calculated in the five regions discussed above for both baseline and FPGA clustering.

As can be seen, the difference of performances between the two algorithm is generally at the permille level. This difference doesn't get worse as the fraction of large clusters increases, therefore also running conditions with lots of large clusters should not be an issue for the FPGA.

In addition, Fig. 5.5 shows a comparison of the distributions of cluster residual for the x coordinate, $x_{\text{cluster}} - x_{\text{MC}}$, for both clustering reconstruction algorithms. The two

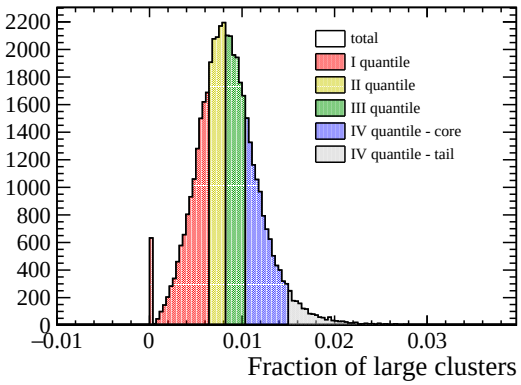


Figure 5.3: Distribution of the fraction of large clusters per event. Each coloured region is an equally populated region, except for the last one that is divided in two to better investigate the effect of very large fractions of big clusters.

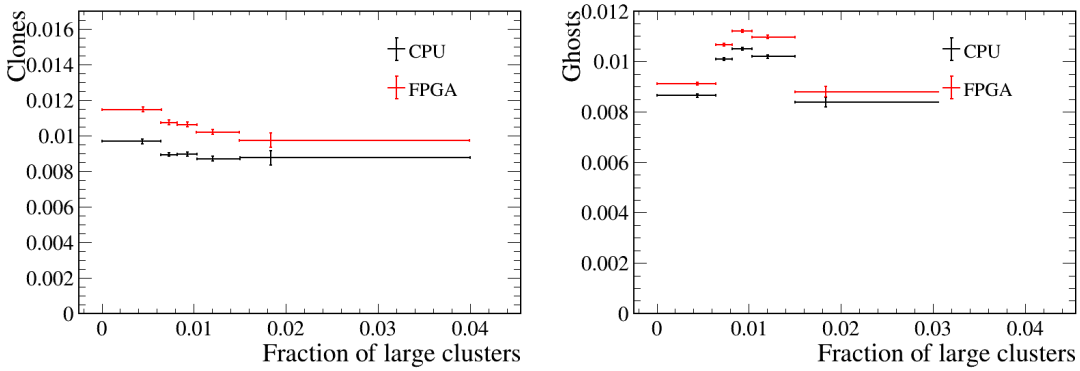


Figure 5.4: (Left) clone rate and (right) ghost rate, calculated in each of the coloured regions of Fig. 5.3. All reconstructible tracks of the 50 000 minimum bias sample have been used.

distributions are nearly indistinguishable, showing that the effect of large clusters is very small also for the FPGA-based algorithm. The standard deviation of both residual distributions is about 11.5 µm, to be compared with the single pixel surface of 55×55 µm^2.

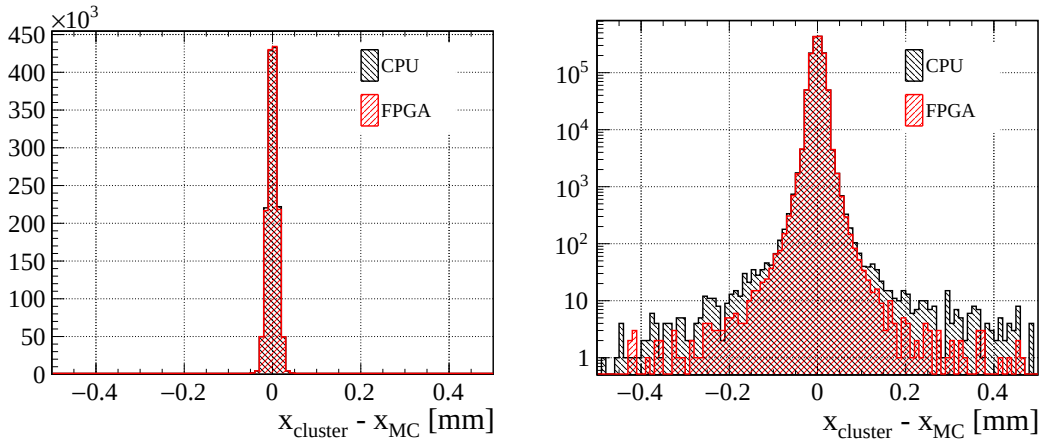


Figure 5.5: Residual distributions of cluster center coordinates, $x_{reco} - x_{\mathrm{MC}}$, for CPU and FPGA algorithms, in linear scale on the left and logarithmic scale on the right.

5.6 Primary vertex reconstruction

This section shows a comparison of the performance related to the primary vertex (PV) reconstruction. For these studies the set of 50 000 minimum bias events, already used in section 5.5, is used. Primary vertex reconstruction efficiency is defined as

$$\epsilon_{\mathrm{PV}} = \frac{N_{\mathrm{MC\_matched}}}{N_{\mathrm{MC\_reconstructible}}},$$

where a PV is considered reconstructible if it has at least four reconstructed tracks associated to it. A reconstructed PV is matched to a MC one when the distance on z axis between them is less than 2 mm or $5\sigma_{z_{\mathrm{PV}}}$, whichever is the smaller; $\sigma_{z_{\mathrm{PV}}}$ is the uncertainty in the PV position. This efficiency is studied as a function of the number of tracks associated to the PV (see Fig. 5.6 left) and the z coordinate of the PV (see Fig. 5.6 right). As can be seen the efficiency of the two algorithms are practically indistinguishable. A small drop can be seen for both as a function of z_{PV}, with a peak downward at about -20 mm.

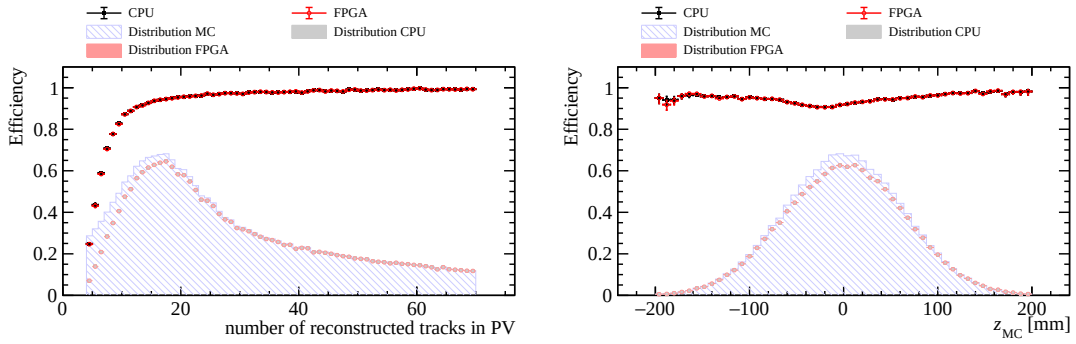


Figure 5.6: Primary vertex reconstruction efficiency as a function of the number of reconstructed tracks associated to the PV (left) and z of the MC matched PV (right). The two algorithms are practically indistinguishable.

The PV resolution and possible biases in the position are also studied, exploiting the distribution of the residuals $\Delta x = x^{\mathrm{PV}}_{reco} - x_{\mathrm{MC}}$. Its standard deviation can be considered as a reliable measure of the resolution, while its mean should be 0. A different value would indicate the presence of a bias in PV reconstruction. To avoid taking into account also outliers in the tails far from the center of the distribution, the following procedure has been adopted: first, the standard deviation of the core of the distribution is estimated from the 25^{th} and 75^{th} percentiles; then a fit with a gaussian function is performed in the range of $\pm 4\sigma_{estimated}$ from 0; the mean and σ returned by the fit are measures of bias and resolution. The results of this procedure, applied to all PV coordinates, are shown in Figs. 5.7 and 5.8, respectively for resolution and bias, as a function of z_{MC} on the left and of the number of associated reconstructed tracks on the right. The performance of the two algorithms in reconstructing the PV is almost indistinguishable.

It is worth mentioning that these studies initially showed a bias in the reconstruction of the z coordinate of the PV position. A non negligible bias of 2 µm was present, in the region of z_{MC} between -100 mm and 0 mm. A large amount of work has been done in this thesis to find out the cause of this issue, showing that it was caused by the truncation of

coordinates, applied by the FPGA algorithm, to one eighth of pixel. It has been shown that with the usage of rounding to the closest eighth of pixel such a bias disappears. In order not to break the flow of the chapter it is decided to present this work in Appendix B.

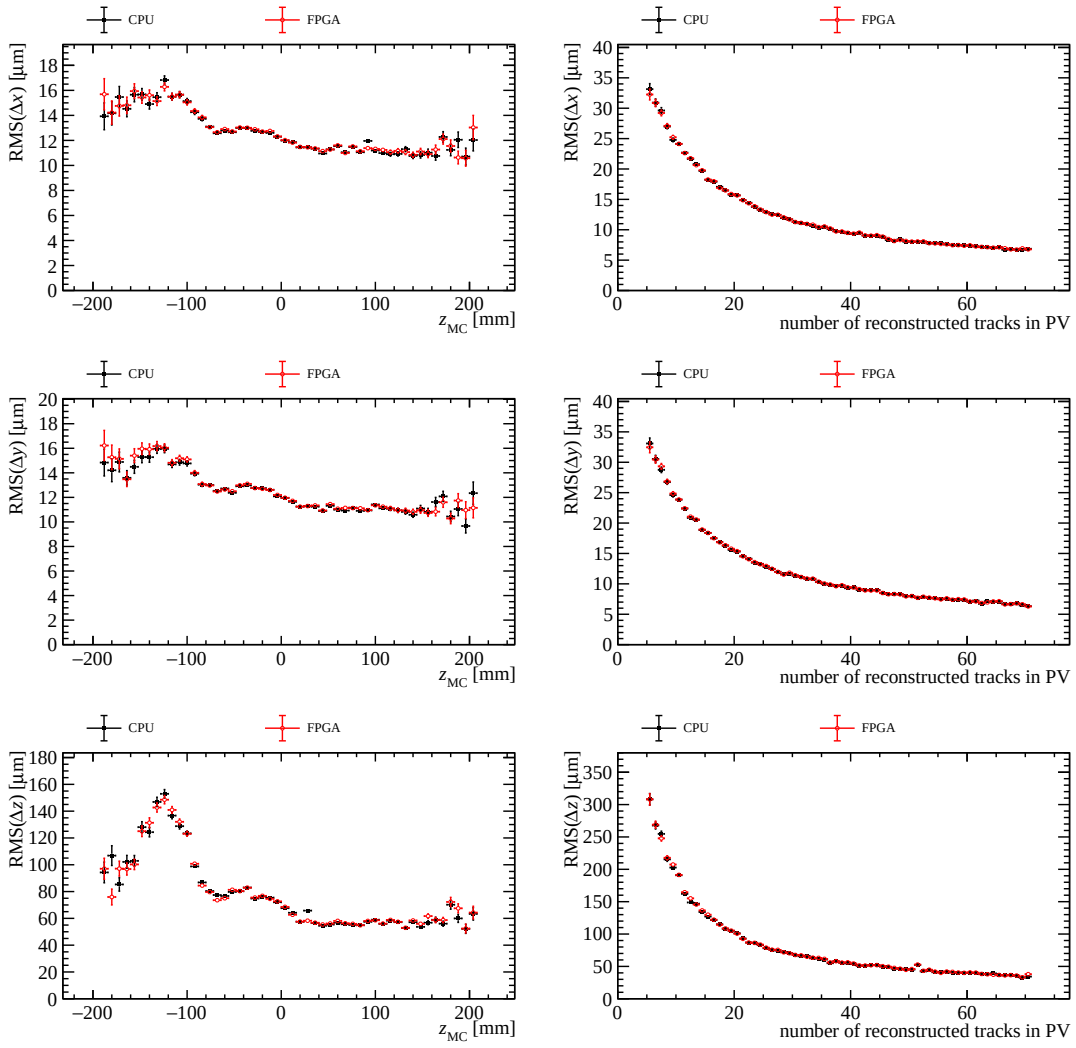


Figure 5.7: Primary vertex resolution as a function of z of the MC matched PV (left) and the number of reconstructed tracks associated to the PV (right).

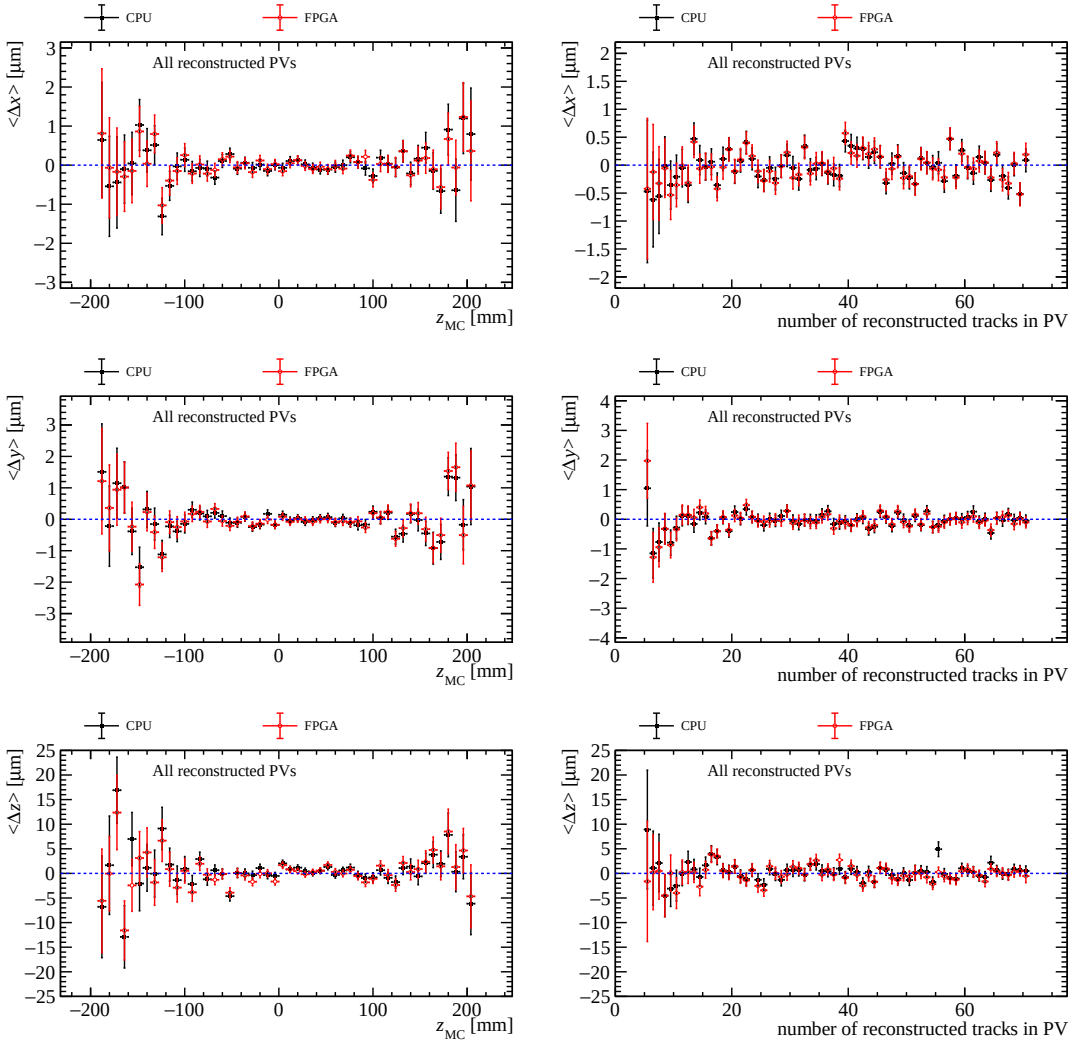


Figure 5.8: Primary vertex bias as a function of z of the MC matched PV (left) and the number of reconstructed tracks associated to the PV (right).

5.7 Tracking performance

In this section tracking reconstruction efficiency and ghost rate plots are shown. Tracking reconstruction efficiency is defined as

$$\epsilon_{track} = \frac{N_{\text{MC\_matched}}}{N_{\text{MC\_reconstructible}}}$$

where a track is MC-reconstructible if it has at least three hits in the VELO, and it is MC-matched if at least 70% of its hits are matched with the hits of the MC particle in all the three tracking detectors. The definition of the ghost rare is already discussed in section 5.5. The official HLT1 tracking algorithm is used. Plots in Fig. 5.9 show the tracking reconstruction efficiency as a function of various tracking quantities: η, momentum, transverse momentum and the number of PVs in the collision. 50 000 minimum bias events are used, and long tracks are selected. As can be seen, the performances of the two algorithms can barely be distinguished. Figure A.1 in Appendix A shows plots of the ghost rate, as a function of the same quantities. Tracking efficiency of tracks originated by b hadrons decay are also shown in Appendix A in Fig. A.2. Table 5.2, instead, shows

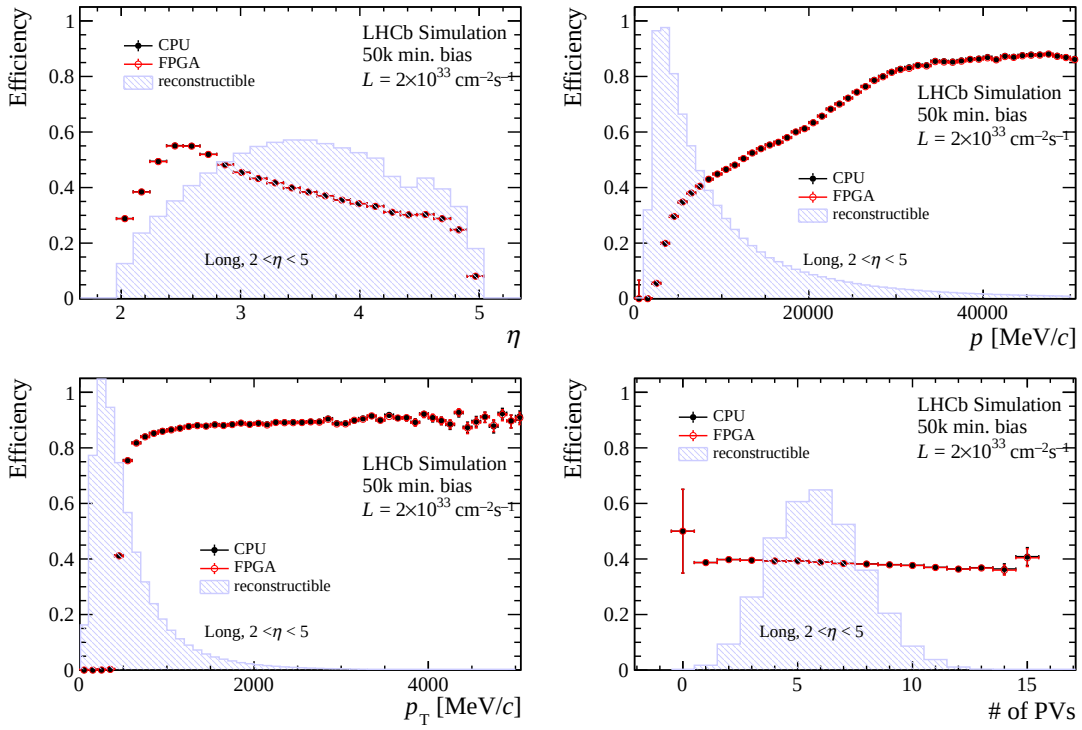


Figure 5.9: Tracking reconstruction efficiency for all long tracks. The 50 000 minimum bias events sample has been used.

the total tracking efficiency and the clone rate for various types of tracks, comparing the performance of the FPGA and CPU algorithms. The ghost rate is also reported. The 50 000 minimum bias events sample is used and official HLT1 tracking algorithms are used. VELO track type comprises all reconstructible tracks in the VELO subdetector, while long tracks are those reconstructible in the VELO, UT and SciFi subdetectors. The

Track type	Tracking efficiency		Clone rate	
	CPU	FPGA	CPU	FPGA
VELO	98.14%	98.03%	1.40%	1.54%
Long	99.29%	99.20%	0.90%	1.07%
Long, $p > 5$ GeV	99.57%	99.47%	0.65%	0.83%
Long, strange	97.43%	97.24%	0.85%	0.94%
Long, strange, $p > 5$ GeV	97.41%	97.21%	0.49%	0.59%
Long, fromB	99.18%	98.98%	0.68%	0.82%
Long, fromB, $p > 5$ GeV	99.32%	99.15%	0.55%	0.72%
Long, electrons	95.54%	92.71%	1.36%	3.04%
Long, fromB, electrons	96.84%	95.00%	1.87%	1.10%
Long, fromB, electrons, $p > 5$ GeV	97.65%	95.29%	1.97%	0.82%
Long, fromB, $p > 3$ GeV, $p_T > 500$ MeV	99.51%	99.36%	0.40%	0.69%
Ghost rate	CPU		FPGA	
	0.84%		0.90%	

Table 5.2: Table showing tracking efficiency and the clone rate for various types of tracks, comparing the performance of the FPGA and CPU algorithms. The ghost rate is also reported. The 50 000 minimum bias events sample is used and official HLT1 tracking algorithms are used.

labels *strange* and *fromB* indicate tracks belonging to a decay of a strange- and a b-hadron, respectively. The final HLT1 tracking efficiencies are nearly identical for the majority of all track types. The FPGA-based algorithm exhibits a little lower reconstruction efficiency. The difference never exceed the 0.2% level for all VELO and the long track types.

The only non negligible difference comes from electrons. The category "Long, electrons" shows a decrease in efficiency of 2.83%, which goes to the value of 0.15%, when the category "Long, fromB, $p > 3$ GeV, $p_T > 500$ MeV" is required. This is particularly relevant for the experiment since the study of semileptonic $b \to sl^+l^-$ transitions, in which the dilepton pair is not produced in the decay of a hadronic resonance, offers a rich set of observables that probe and constrain new physics models in a complementary way to the study of $B_s^0 \to \mu^+\mu^-$ and $B^0 \to \mu^+\mu^-$. As already mentioned in Sec. 1.2, the LHCb collaboration has already reported important results for many of these modes [23, 38–43], most notably $B^0 \to K^{(*)}\mu^+\mu^-$ and $B^0 \to K^{(*)}e^+e^-$, showing tantalizing hints of violation of the $e - \mu$ lepton universality. Those are, therefore, some of the more promising decay modes for the forthcoming LHCb-Upgrade and a strong motivation for the realization of the Phase-II Upgrade.

Electrons do interact with the detector material in a different way with respect to the heavier particles, such as pions, kaons, and muons. Like their corresponding heavy charged particles, electrons (and positrons) also suffers a collision energy loss when passing through matter. However, because of their small mass an additional energy loss mechanism comes into play: the emission of electromagnetic radiation arising from the scattering in the electric field of the nucleus, the so-called bremsstrahlung. At energies of a few MeV or less, this process is still a relatively small factor. However, as the energy increases, the probability of bremsstrahlung quickly shoots up so that at a few 10's MeV, loss of energy by

radiation is comparable or greater than the collision-ionization loss. At energies above this critical energy, as it happens for electrons in LHCb, bremsstrahlung dominates completely. Electrons (and positrons), therefore, can generate clusters with a different topology with respect to all other particles, such as pions, kaons, and muons, where the energy is lost with the ionization mechanism. It is, therefore, necessary to study with a greater detail the performance of the algorithm on "good" electrons, those coming from a b-hadron decay, such as the $B^0 \to K^{*0}e^+e^-$, selecting a more abundant sample of them. As above, the plots in Fig. 5.10 show the tracking reconstruction efficiency as a function of various tracking quantities: η, momentum, transverse momentum and the number of PVs in the collision. A 10 000 $B^0 \to K^{*0}e^+e^-$ event sample is used now, and long tracks are selected, including long electrons from a b-hadron decay with some minimal momentum requirement. Also in this case, the two algorithms can barely be distinguished. For completeness, Tab. 5.3

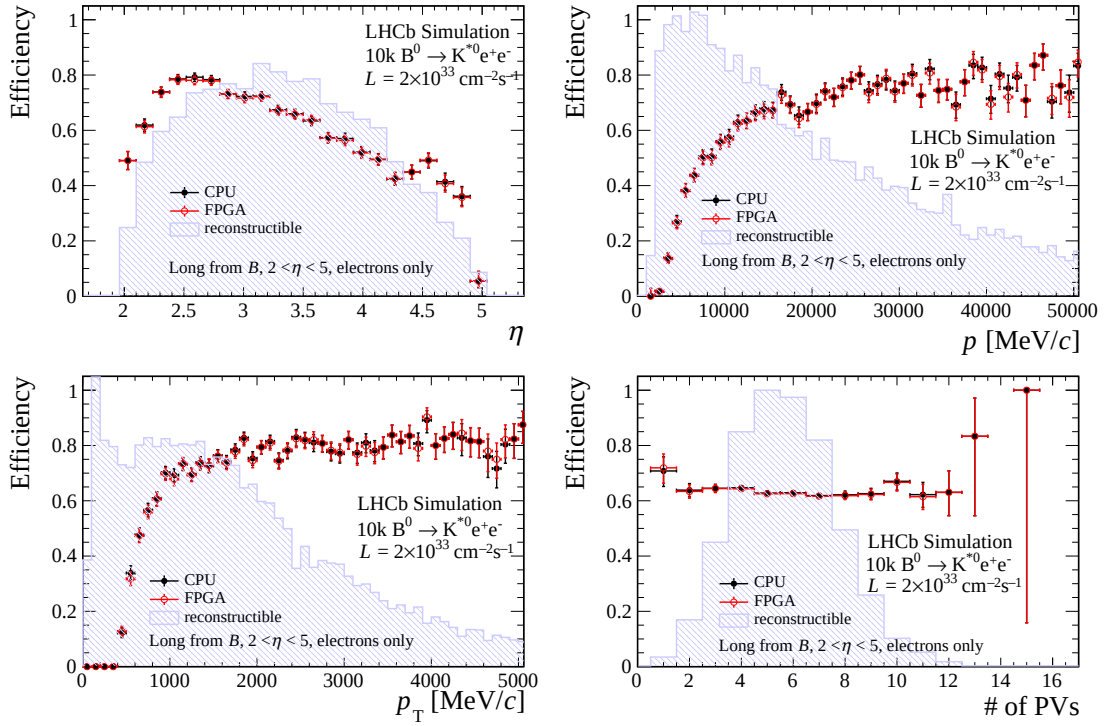


Figure 5.10: Tracking reconstruction efficiency for long electron tracks as a function of various kinematic variables. 10 000 events containing $B^0 \to K^{*0}e^+e^-$ decays have been used.

shows the total tracking efficiency and the clone rate for various types of tracks, comparing the performance of the FPGA and CPU algorithms. The ghost rate is also reported. The difference in efficiency for all "good" electrons categories never exceeds the value of 0.3%.

It is worth mentioning that similar studies are also performed with a sample of 10 000 $B^0 \to K^{*0}\gamma \to K^{*0}[e^+e^-]$, where electrons specifically come from a photon conversion, and similar results are found.

Track type	Tracking efficiency		Clone rate	
	CPU	FPGA	CPU	FPGA
VELO	98.23%	98.13%	1.41%	1.55%
Long	99.32%	99.24%	0.92%	1.09%
Long, $p > 5$ GeV	99.60%	99.49%	0.65%	0.85%
Long, strange	97.20%	97.09%	0.95%	1.01%
Long, strange, $p > 5$ GeV	97.46%	97.30%	0.53%	0.65%
Long, fromB	99.38%	99.26%	0.66%	0.79%
Long, fromB, $p > 5$ GeV	99.60%	99.51%	0.56%	0.70%
Long, electrons	96.00%	93.83%	1.34%	2.62%
Long, fromB, electrons	97.42%	97.14%	1.20%	1.65%
Long, fromB, electrons, $p > 5$ GeV	98.26%	97.93%	1.23%	1.66%
Long, fromB, $p > 3$ GeV, $p_T > 500$ MeV	99.59%	99.52%	0.50%	0.62%
Ghost rate	CPU		FPGA	
	1.02%		1.07%	

Table 5.3: Table showing tracking efficiency and the clone rate for various types of tracks, comparing the performance of the FPGA and CPU algorithms. The ghost rate is also reported. The 10 000 $B^0 \to K^{*0}e^+e^-$ sample is used and official HLT1 tracking algorithms are used.

5.7.1 Impact parameter and momentum resolution

One of the distinctive features of LHCb is the ability to precisely distinguish secondary vertexes from primary ones. This is done with the use of the track impact parameter, which is defined as the distance of the track trajectory from the PV. The impact parameter is decomposed in its x and y components, IP_x and IP_y, on the plane transverse to the z-axis. The resolution for these two components are calculated from the distributions of the residuals with a gaussian fit in the region between -300 µm and +300 µm. All tracks of the 50 000 minimum bias events sample have been used. Figure 5.11 shows the results as a function of η and $1/p_T$. The performance of the two clustering algorithms is almost indistinguishable, with a difference of the order of 1%. Momentum resolution are

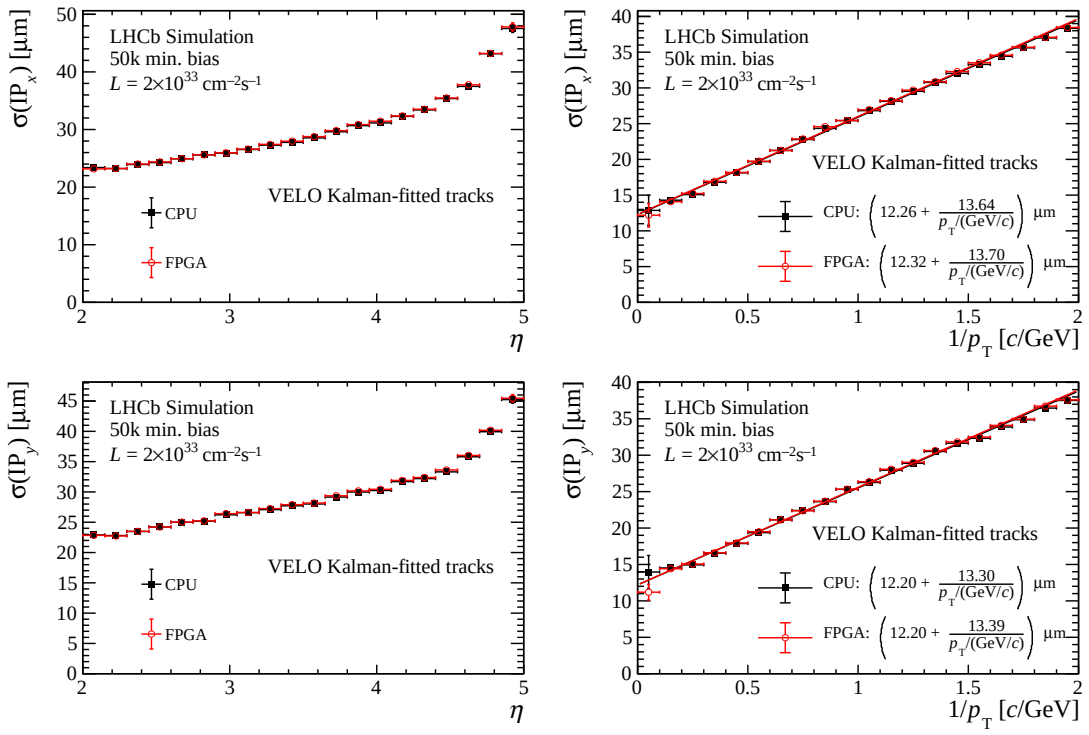


Figure 5.11: Impact parameter resolution, x component on top, y on bottom, as a function of η on the left and $1/p_T$ on the right.

also studied with the same 50 000 minimum bias events sample selecting long tracks with $2 < \eta < 5$. Resolution is obtained by a gaussian fit of the distribution of dp/p as a function of η, and as a function of the MC true p, respectively. Results are shown in Fig. 5.12. Also in this case no significant differences are observed.

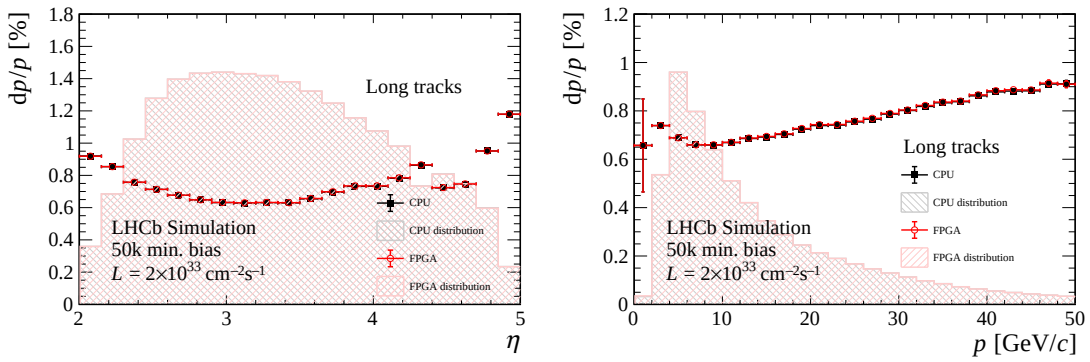


Figure 5.12: Momentum resolution of long tracks as a function of η on the left and the true MC momentum on the right.

5.8 SuperPixels overflowing matrix chains

Another peculiarity of the FPGA clustering algorithm is the static length of the matrix chain of each sensor, equal to 20 matrices. In case an event has more than 20 clusters per sensor made of non-isolated SPs, some of them will overflow from the chain because no free matrices are left. These SPs are resolved as isolated though they are not. In this section a study on this issue is presented, together with a possible mitigation of the problem. The 50 000 minimum bias events sample has been used in the simulations for this analysis, from which, at first, events with too many clusters in the UT and SciFi are removed; this is called *global event cut* (GEC) and, in our case, 46 380 events pass the selection. GEC has been used only because it is included in the official LHCb simulation but does not concern VELO.

Figure 5.13 shows the plots of the distribution of the fraction of overflow SPs per event, for the whole VELO and for sensor 60, one of the most crowded. For each event the ratio $N_{\text{ovfl}}/N_{\text{noniso}}$ is calculated, except for events without non-isolated SPs (this is why the number of entries is not 46 380). The majority of events have low fractions. The whole VELO has a shorter tail with respect to sensor 60: this is because in the first one for each event all modules are considered and modules farther from interaction point lower the overall fraction, since they typically have a lower number of SPs.

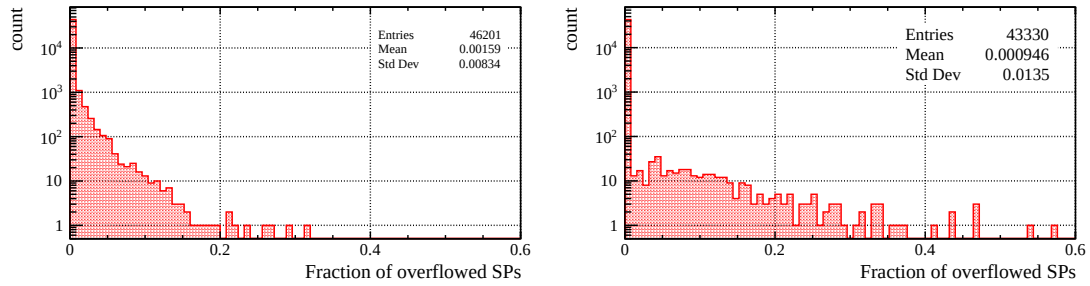


Figure 5.13: Distribution of the fraction of overflow SPs per event, for the entire VELO on the left and for sensor 60, one of the most crowded, on the right.

Figure 5.14 shows the plots of the fraction of overflow SPs per event as a function of the number of non-isolated SPs, again for the entire VELO and sensor 60, with superimposed distribution. This plots shows the dependency of the fraction of overflow to the total number of non-isolated SPs: as expected the more SPs there are in the event, the more overflow happens. As it can be seen, the fraction of overflow begins to grow when the distribution decreases, particularly for sensor 60 in which it reaches remarkable values but in very rare cases. In the right plot (related to sensor 60) the growth starts just after 40 non-isolated SPs: this is as expected because each chain is made of 20 matrices and at least two SPs will go inside each.

Figure 5.15 shows, in the black dot set, the fraction of overflow SPs per event as a function of sensor number. As it can be seen, even sensors, closer to beam pipe, have a higher peak near nominal interaction point. Odd sensors, being farther, have a much smaller peak. The red dots show a test made with asymmetric matrix chains: 24 matrices for even sensors, 16 for odd sensors. This trick allows to greatly reduce the fraction of overflow SPs in even sensors near interaction point, without penalizing significantly odd

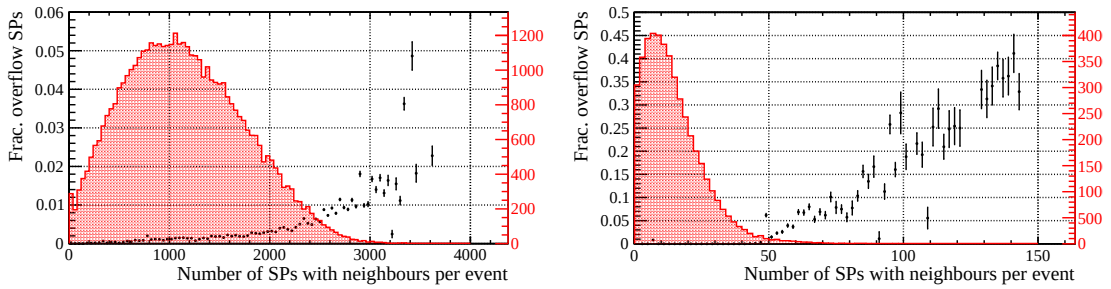


Figure 5.14: Fraction of overflow SPs per event as a function of the number of non-isolated SPs per event, for the entire VELO on the left and for sensor 60, one of the most crowded, on the right. The distribution superimposed is that of the number of non-isolated SPs.

sensors and, most of all, with the use of the same logic resources in the FPGA. The problem of overflow is at the level of permille with the standard symmetric chain configuration and is of the order of 10^{-4} in the asymmetric configuration. This modification will be adopted in the final implementation of the FPGA clustering algorithm.

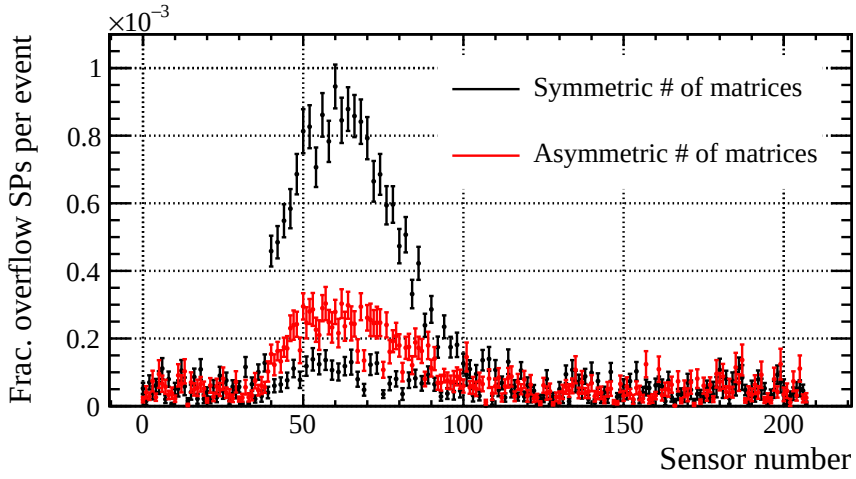


Figure 5.15: Fraction of overflow SPs per event as a function of sensor number. Black dots are obtained with the standard configuration of 20 matrices for each chain. Red dots are obtained with asymmetric matrix chains: 24 matrices in even sensor chains, 16 for the others. This modification does not increase logic resources utilization.

5.9 Throughput studies

The task of the clustering is assigned to the FPGA accelerators in order to relieve the CPU workload. It is, therefore, important to determine the gain in throughput of the HLT1 stage on a single node of the Event Filter Farm. Figure 5.16 shows how the computing power of a single node of the Event Filter Farm (running the HLT1 reconstruction algorithms) is spent on the different tasks. As it can be seen, the VELO tracking, which includes the VELO clustering, is the most expensive component.

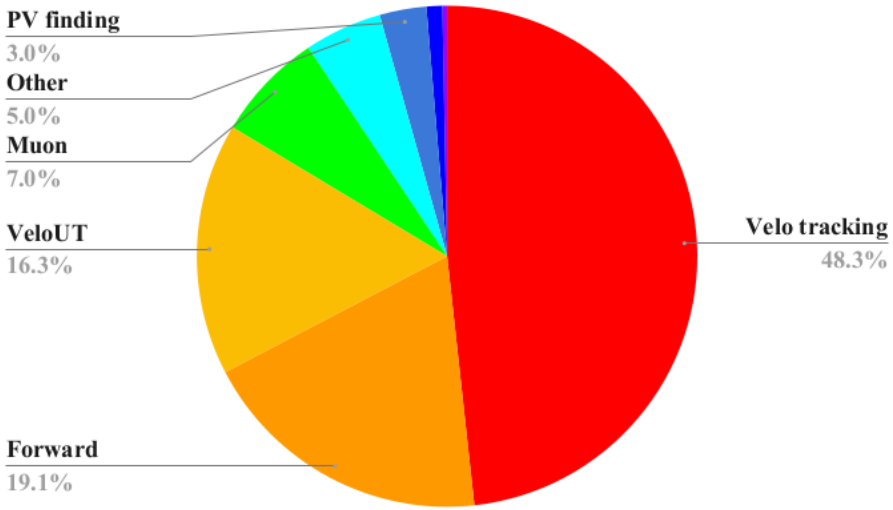


Figure 5.16: Computing power share of the average HLT1 node before FPGA clustering introduction.

Preliminary information obtained running FPGA clustering algorithm simulation shows a gain in the event rate throughput of about 8%. The HLT1 throughput of a single node of the Event Filter Farm is measured to be 168 kHz, reaching the value of 181 kHz if the FPGA clustering algorithm described in this thesis is used. This fact translates into a smaller number of nodes requested to reach the threshold of 30 MHz.

Very recently, the LHCb collaboration decided to adopt a GPU-based implementation for the HLT1 trigger as baseline for the Run 3, as described in Sec. 2.8, the so-called Allen project [51]. A first preliminary estimate of the throughput is also provided by the Allen group, which has implemented the possibility to process the FPGA clusters. Considering the Nvidia GeForce RTX 2080Ti GPU, likely the model to be used in Allen, a throughput of 132 kHz per GPU board was measured when processing SPs, reaching a value of 136 kHz. Therefore, the GPU-based HLT1 receives a gain in the event rate throughput, for a single GPU, of about 3.5% which is lower than the 8% found for the the CPU-based system, but still significant. It is worth mentioning that the GPU is a specialized device performing computations in a deep parallel way, and, therefore, it is not so straightforward to determine all the beneficial effects of removing the VELO clustering algorithm.

Chapter 6

Conclusions

Current and future experiments on *beauty* and *charm* physics have the potential to significantly improve the knowledge on the dynamics of the heavy flavours by means the study of the *CP* violation and the search of rare and very rare decays. However, typical interesting processes have a small signal-to-background ratio, and this is especially true for experiments installed at hadron colliders, as the LHCb experiment. The limited bandwidth available for storing data imposes the adoption of powerful and selective data acquisition and trigger systems. The most important discriminant for selecting b- and c-hadrons decays is their relatively long lifetime, requiring excellent tracking systems to disentangle interesting events from the huge background.

The LHCb experiment, during Long Shutdown 2 of LHC, is undergoing a major upgrade, called LHCb Upgrade (or Phase-I Upgrade). The upgraded experiment will read collisions data at the full crossing rate of 40 MHz (30 MHz of visible collisions) and the first event selection will be performed by the HLT1 trigger, running on a computer farm. The need of event processing at such a large rate led to the consideration of the use of specialized devices, generally called accelerators, as cost-effective solution to the challenge of the processing speed. These includes GPUs and FPGAs, widely used in many R&D projects.

The LHCb-RETINA is a R&D project developed by the INFN Pisa and aims at performing tracking in real time on FPGAs. Two implementations of the RETINA tracker, to be installed in the LHCb Event Builder, have been currently developed, and one is specifically dedicated to the VELO tracking. The purpose is to relieve the CPUs from the pattern recognition task, one of the most time-demanding to be executed by the HLT, that, being low-level and parallelizable, can be effectively transferred to specialized hardware accelerators, at the earliest possible processing level (on-detector). The implementation of the VELO tracking requires pre-processed input clusters, therefore it is of fundamental importance the development and implementation of a new 2D clustering algorithm for the VELO subdetector, running at a speed of 30 MHz at the conditions of the LHCb Upgrade. In this regard, the work performed in this thesis is developed within the LHCb-RETINA project and aims at moving the 2D VELO cluster-finding to the FPGA readout cards, freeing the HLT farm from the burden of this computation, and paving the way for a first test bench of the VELO tracking algorithm, hopefully already during the LHC Run 3 with real data, in preparation of the installation of the whole system for the LHC Run 4.

The core firmware of the clustering algorithm was ready and functional at the beginning

of this thesis work. It was the very first implementation, developed to run on a Stratix-V FPGA chip (different from the Arria-10 chip mounted on the readout PCIe40 boards) and was not conceived at that time for the integration into the TELL40 firmware. The main purpose of this thesis has been, therefore, the implementation of all control systems and signals necessary for a full and functional integration into the LHCb readout system.

All the ancillary components, whose purpose was to feed clustering and to read its output for development and testing, have been removed; the remaining clustering core software has been modified in order to be compatible with the rest of the VELO firmware. The standard LHCb control systems, the ECS (*Experiment Control System*) and the TFC (*Timing and Fast Control*), have been implemented, along with a new entity to manage internal and external errors. Additional signals, read at input side, indicating the type and the size of data packets, have been placed at the output side. An entity to output both clusters and corresponding raw data has been added for debugging and testing. The clustering algorithm has also been modified in order to reduce the number of lost and duplicated clusters. Other changes in the firmware have been made, including improvement and optimization of the original clustering algorithm. An example is the introduction of the reversed L-patterns, useful to mitigate the split up of long swipes of active pixels. Finally, an extensive and accurate work of testing and debugging of the all clustering firmware is carried out both in normal and harsh conditions.

The system has been extensively tested and optimized using fully realistic simulated samples at the LHCb Upgrade running conditions. A high level simulation, written in C++ programming language, capable to emulate all the features of the clustering algorithm was integrated in the official software of the experiment in order to reconstruct realistic simulated pp collisions events at the desired conditions. Various quantities related to clustering and tracking quality have been analyzed and compared for FPGA and CPU algorithms, in order to determine the accuracy and the robustness of the FPGA-based algorithm. Additional studies have been carried out to investigate more closely the most critical features of the FPGA algorithm, like the presence of large clusters or events with a large number of non-isolated SPs, to ensure that they are all well understood and do not impact the physics performance of the experiment. Moreover, a bias in the measurement of the z coordinate of the reconstructed PV has been found in the first implementation of the algorithm, and it has been determined to be caused by the truncation of cluster center coordinates to the smaller eighth of a pixel, while a rounding to the closest one makes it to disappear.

Clustering and tracking efficiencies have been studied in details as a function of several kinematic and topological variables. The clustering efficiency is found to be excellent, very close to that one obtained with the CPU algorithm, particularly for clusters originated by the type of tracks most useful for physics analysis. Tracking efficiencies for the two algorithms are almost indistinguishable. Thanks to FPGA algorithm, a gain in the event processing throughput of about 8% (3.5%) is measured for the first stage of the HLT implemented on CPUs (GPUs). More importantly, the success of this implementation represents a first, if small, step in the direction of moving non-trivial reconstruction functions from traditional CPUs to heterogeneous systems running at a very early level of the data taking, that will hopefully be further expanded in the future.

The LHCb Collaboration is currently reviewing the proposal of the LHCb-Pisa Group of adopting such FPGA algorithm as the baseline for the LHC Run 3 and Run 4.

Appendices

Appendix A

Tracking performance

Figure A.1 shows the ghost rate, as defined in Sec. 5.5, selecting long tracks. Since one

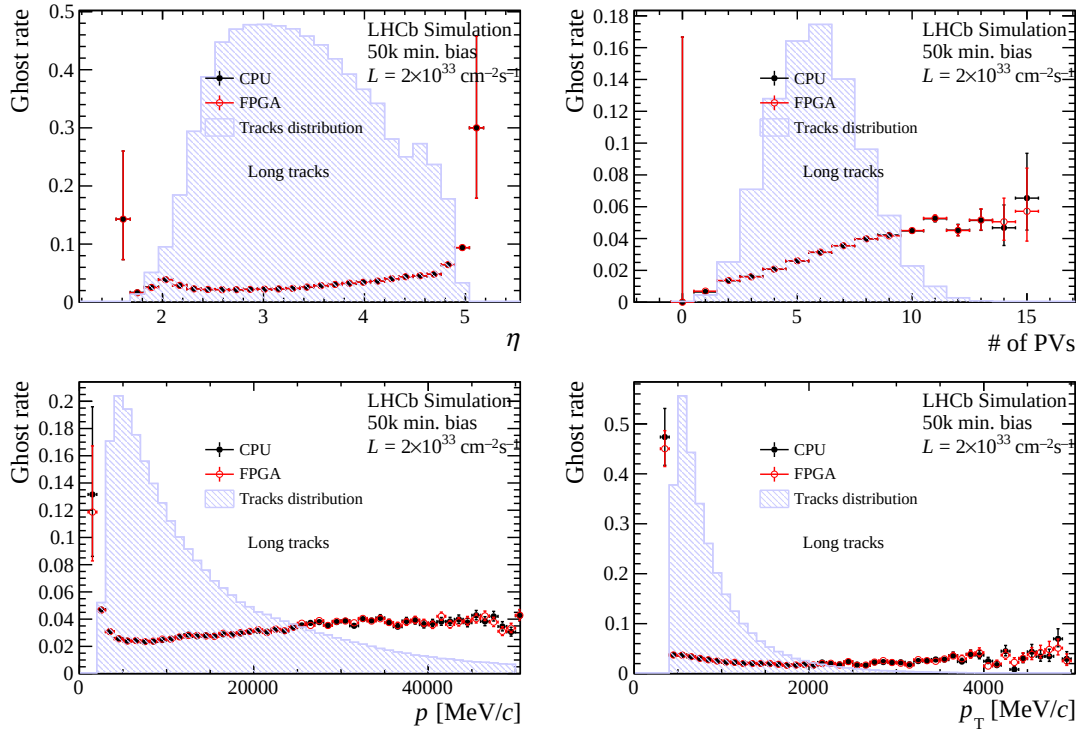


Figure A.1: Ghost rate for all long tracks. 50 000 minimum bias events.

of the main challenges of the LHCb experiment is to measure properties of b hadrons, it is useful to study tracking efficiency specifically for those particles. To this purpose the 50 000 minimum bias events sample are used, selecting long tracks originated by decays of b hadrons, with a momentum $p > 3$ GeV/c and a transverse momentum $p_T > 500$ MeV/c. Efficiency plots are shown in Fig. A.2.

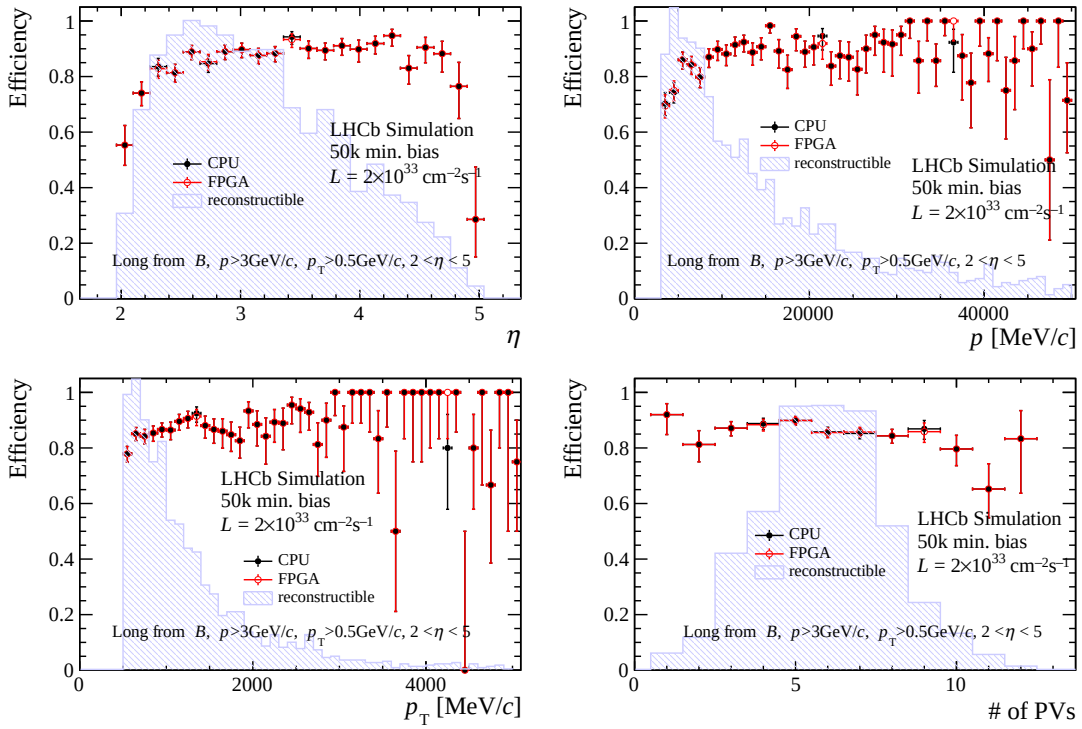


Figure A.2: Tracking reconstruction efficiency for long tracks originated by decays of b hadrons, with a momentum $p > 3$ GeV/c and a transverse momentum $p_T > 500$ MeV/c. The 50 000 minimum bias events sample has been used.

Appendix B

Studies on z_{PV} bias

As explained in Sec. 5.6, the studies performed to determine the performance of the FPGA based algorithm in reconstructing the position of PVs, have initially shown a bias in the reconstruction of z coordinate. A non negligible bias of about 2 µm was present, in the region of z_{MC} between -100 mm and 0 mm, as shown in Fig. B.1. On the left side of the figure it is shown the average bias $\langle \Delta z \rangle$ as a function of the z coordinate of the MC matched PV (in analogy with the same plot reported in Fig. 5.8), while on the right side it is shown the same observable but only four large bins are used in order to have a better statistical power to spot the issue. Since the bias is only visible in the FPGA algorithm, the

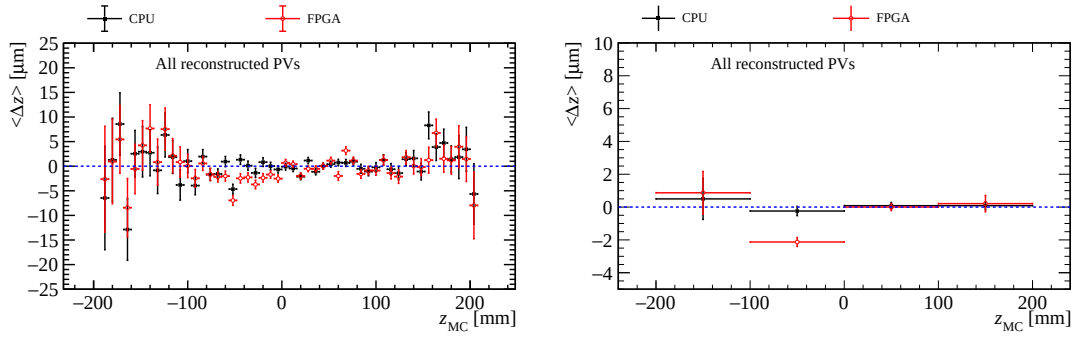


Figure B.1: Primary vertex bias as a function of z of the MC matched PV. The two plots report on the same observable, but a different numbers of bins is used.

cause must be related to one of the features that are different between the two algorithms. These, specifically, are:

- clusters that do not fit inside 3×3 sub-matrix;
- large clusters that are split between matrices;
- overflowed SPs;
- rounding down of cluster center-of-mass coordinates.

A large amount of work has been done in this thesis to find out the cause of this issue. The first possible source, which is investigated, is the first one of the list, that is clusters too

large to fit inside 3×3 sub-matrix. The reversed L-patterns were already implemented in the simulation (see section 4.12) at the time, avoiding the splitting of a big cluster in many 3×3 sub-matrices. However, it was noted that a small shift of coordinates, of the reconstructed cluster, towards higher values of the radial distance r (of the cluster) is generated by the algorithm, as it can be seen in Fig. B.2. Such a shift is very small (below

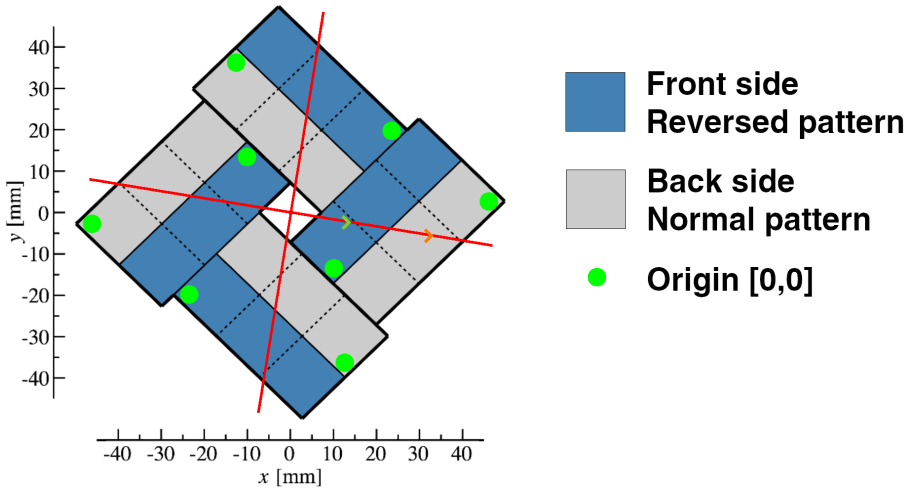


Figure B.2: The illustration of a VELO station is shown. The positioning of the sensors and some examples of tracks (coming out from the center) are also shown. Green circles indicate the origin of the coordinates of each sensor. The orange little "L" is the normal L-pattern for the associated back-mounted sensor, while the green little "L" is the reversed L-pattern for front-mounted sensor. The overall effect is a shift towards bigger values of the radial distance r.

10 µm), however it could be the cause of the $\langle \Delta z \rangle$ bias. In order to verify this hypothesis, a new type of L-patterns are used in the algorithm, the so-called *opposite patterns*: instead of the normal and reversed L-patterns, new L-patterns with a the top-right corner and with a the top-left corner are used, respectively. The introduction of these new L-patterns should result in the reconstruction of clusters with an equal but opposite $\langle \Delta z \rangle$ bias, assuming that the radial shift in measuring the cluster position is the source of the observed bias. Also a different set of L-patterns is used, the so-called *alternated patterns*: the normal L-patterns is used for sensors with an even identification number (sensors 0 and 2), while the opposite L-patterns are used for the other sensors, those with an odd identification number(sensors 1 and 3, see Fig. B.3 for a view of the used L-patterns). Figure B.4 shows the result obtained with the new L-patterns configurations. Unfortunately the same bias is present, with both opposite and alternated L-patterns.

An ensemble of only "small" clusters is processed by the simulation software of both algorithms (a small cluster is defined as a cluster from isolated SPs or otherwise containing at most two pixels). The result of this test would allow to exclude not only the second item of the list, but also the first one. However, overflowed SPs could be still the source of the observed bias (as well as the rounding down of cluster center-of-mass coordinates). Fig. B.5, on the left side, shows the average value of the residual $\langle \Delta z \rangle$ as a function of z_{MC}. Although the bias is reduced, it is still present. The same test is repeated by also

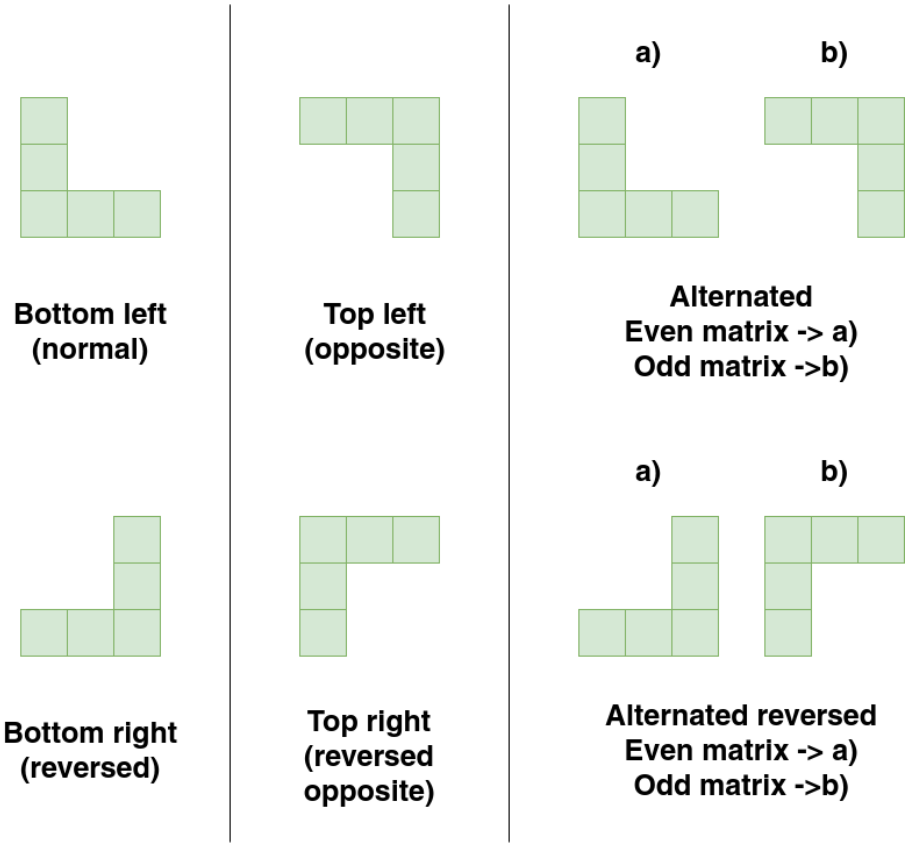


Figure B.3: Pixels configuration of "opposite" and "alternated" L-patterns. See the test for more details.

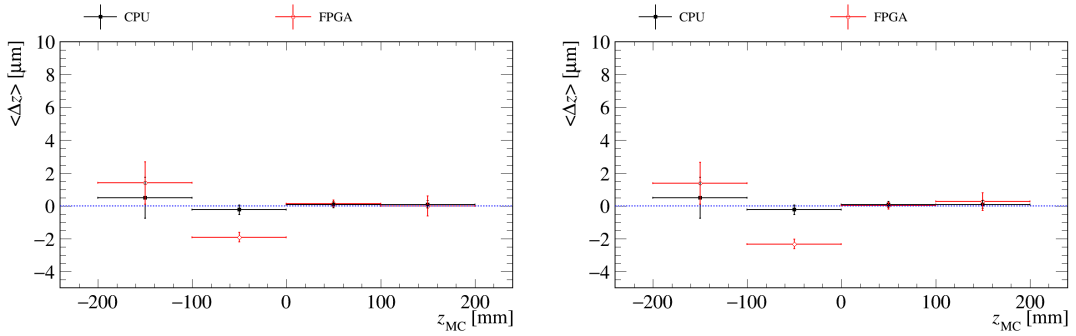


Figure B.4: Result obtained with the opposite L-patterns on the left, and alternated L-patterns on the right. The bias is still present and with the same sign in both cases.

excluding overflowed SPs (see Fig. B.5 on the right side). The result is very similar to the previous test using all small clusters.

Lastly, after the exclusion of all other options, the only possible source of the bias can be ascribed to the last item in the list: the truncation of coordinates, applied by the FPGA algorithm, to one eighth of pixel. By using a rounding to the closest eighth of pixel the

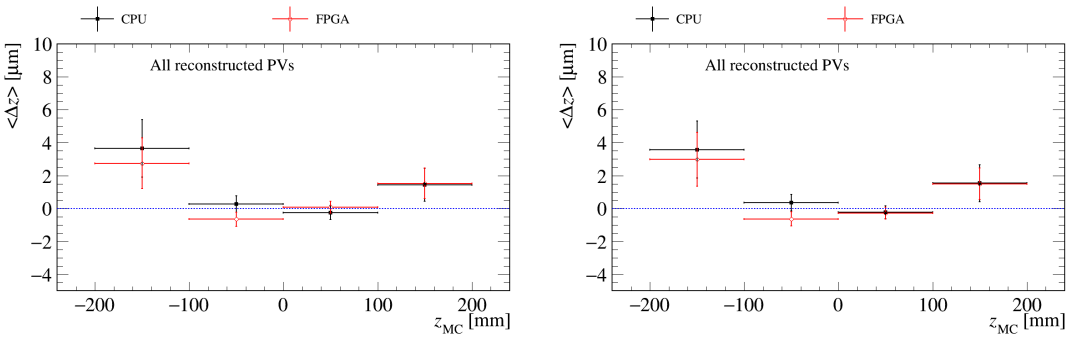


Figure B.5: (Left) Result obtained with only small clusters, i.e. those obtained from isolated SPs or otherwise from clusters containing at most two pixels. The bias is still present but it's reduced. (Right) Result obtained with small clusters excluding also clusters from overflowed SPs.

bias disappears, as shown in Figure B.6.

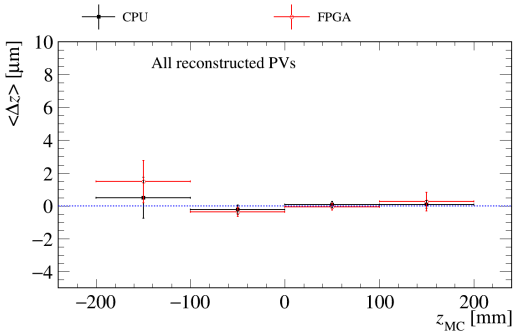


Figure B.6: Result obtained with the rounding of cluster center coordinates to the closest eighth of a pixel.

Ringraziamenti

Vorrei ringraziare Michael, per avermi offerto l'opportunità di realizzare il lavoro qui descritto, per averlo delineato con cura, e per il suo aiuto nella stesura di questo documento, soprattutto nella cura della grammatica e dell'inglese (cose in cui sono negato). Grazie al professor Punzi per l'importante contributo dato nella revisione finale di questa tesi.

Un ringraziamento speciale va a Giovanni, per aver passato insieme un anno lavorando a quattro mani ad un importante pezzo di firmware dell'esperimento. Instancabile guida durante tutto il lavoro, non si è mai tirato indietro alle mie numerose questioni. A lui devo le mie (ancora limitate) conoscenze nel campo del VHDL. Grazie anche a Federico per avermi fatto da guida nei primi mesi del lavoro, e soprattutto per avermi svelato i trucchi per massimizzare il rapporto quantità-prezzo della mensa del CERN.

Ringrazio i miei genitori e i miei zii per avermi supportato in questo percorso, soprattutto nell'ultimo periodo.

Ringrazio gli amici fisici: Pica, Bastiano, Alice, Silvia, Ale. Grazie a Gió per i momenti passati insieme e al grandissimo supporto morale. Grazie a Thomas e Davide per le interessanti e divertenti chiacchierate durante il pranzo. Grazie a Bianco per aver condiviso con me tutto il percorso universitario, le pene sofferte nella preparazione degli esami, e soprattutto per averli pianificati e avermi dato un ritmo, senza di lui oggi sarei ancora all'inizio della "raccolta crediti". Grazie a Paglia per aver condiviso, anche insieme a Bianco, le giornate spese a cercar di ottenere dei risultati dalle apparecchiature del laboratorio, e grazie anche per avermi motivato nelle mie battaglie a sostegno delle superbe qualità del C, in particolare se confrontato alla dispersività del Python. Tante grazie ad Andrea, perché è a lui che devo (quasi) tutto quello che so. Grazie agli altri amici fisici lucchesi per aver condiviso i momenti di pendolarismo.

Bibliography

[1] LHCb collaboration, R. Aaij *et al.*, *First evidence for the decay $B^0_s \to \mu^+\mu^-$*, Phys. Rev. Lett. **110** (2013) 021801, arXiv:1211.2674.

[2] LHCb collaboration, R. Aaij *et al.*, *Measurement of the $B^0_s \to \mu^+\mu^-$ branching fraction and effective lifetime and search for $B^0 \to \mu^+\mu^-$ decays*, Phys. Rev. Lett. **118** (2017) 191801, arXiv:1703.05747.

[3] LHCb collaboration, R. Aaij *et al.*, *Observation of D^0–$\overline{D}^0$ oscillations*, Phys. Rev. Lett. **110** (2013) 101802, arXiv:1211.1230.

[4] LHCb collaboration, R. Aaij *et al.*, *Observation of CP violation in charm decays*, Phys. Rev. Lett. **122** (2019) 211803, arXiv:1903.08726.

[5] LHCb collaboration, R. Aaij *et al.*, *Measurement of the CKM angle γ from a combination of LHCb results*, JHEP **12** (2016) 087, arXiv:1611.03076.

[6] LHCb collaboration, R. Aaij *et al.*, *Precision measurement of CP violation in $B^0_s \to J/\psi K^+K^-$ decays*, Phys. Rev. Lett. **114** (2015) 041801, arXiv:1411.3104.

[7] LHCb collaboration, R. Aaij *et al.*, *Measurement of the CP-violating phase ϕ_s in $\overline{B}^0_s \to J/\psi \pi^+\pi^-$ decays*, Phys. Lett. **B736** (2014) 186, arXiv:1405.4140.

[8] LHCb collaboration, R. Aaij *et al.*, *Measurement of CP-averaged observables in the $B^0 \to K^{*0}\mu^+\mu^-$ decay*, arXiv:2003.04831, submitted to Phys. Rev. Lett.

[9] LHCb collaboration, R. Aaij *et al.*, *Search for lepton-universality violation in $B^+ \to K^+\ell^+\ell^-$ decays*, Phys. Rev. Lett. **122** (2019) 191801, arXiv:1903.09252.

[10] LHCb collaboration, *LHCb VELO Upgrade Technical Design Report*, CERN-LHCC-2013-021.

[11] LHCb collaboration, *LHCb Tracker Upgrade Technical Design Report*, CERN-LHCC-2014-001.

[12] LHCb collaboration, *Expression of Interest for a Phase-II LHCb Upgrade: Opportunities in flavour physics, and beyond, in the HL-LHC era*, CERN-LHCC-2017-003.

[13] LHCb collaboration, *Physics case for an LHCb Upgrade II — Opportunities in flavour physics, and beyond, in the HL-LHC era*, arXiv:1808.08865.

[14] A. Andreazza *et al.*, *What Next: White Paper of the INFN-CSN1*, vol. 60, 5, 2015.

[15] N. Cabibbo, *Unitary symmetry and leptonic decays*, Phys. Rev. Lett. **10** (1963), no. 12 531.

[16] M. Kobayashi and T. Maskawa, *CP Violation in the Renormalizable Theory of Weak Interaction*, Prog. Theor. Phys. **49** (1973) 652.

[17] G. Isidori, Y. Nir, and G. Perez, *Flavor Physics Constraints for Physics Beyond the Standard Model*, Ann. Rev. Nucl. Part. Sci. **60** (2010) 355, arXiv:1002.0900.

[18] G. Isidori, *Flavor physics and CP violation*, in *2012 European School of High-Energy Physics*, pp. 69–105, 2014. arXiv:1302.0661. doi: 10.5170/CERN-2014-008.69.

[19] J. F. Kamenik, *Flavour Physics and CP Violation*, in *2014 European School of High-Energy Physics*, pp. 79–94, 2016. arXiv:1708.00771. doi: 10.5170/CERN-2016-003.79.

[20] LHCb collaboration, R. Aaij *et al.*, *Measurement of the $B^0_s \to \mu^+\mu^-$ branching fraction and search for $B^0 \to \mu^+\mu^-$ decays at the LHCb experiment*, Phys. Rev. Lett. **111** (2013) 101805, arXiv:1307.5024.

[21] CMS and LHCb collaborations, V. Khachatryan *et al.*, *Observation of the rare $B^0_s \to \mu^+\mu^-$ decay from the combined analysis of CMS and LHCb data*, Nature **522** (2015) 68, arXiv:1411.4413.

[22] LHCb collaboration, R. Aaij *et al.*, *Observation of $J/\psi p$ resonances consistent with pentaquark states in $\Lambda_b^0 \to J/\psi p K^-$ decays*, Phys. Rev. Lett. **115** (2015) 072001, arXiv:1507.03414.

[23] LHCb collaboration, R. Aaij *et al.*, *Angular analysis of the $B^0 \to K^{*0}\mu^+\mu^-$ decay using $3\,fb^{-1}$ of integrated luminosity*, JHEP **02** (2016) 104, arXiv:1512.04442.

[24] LHCb collaboration, R. Aaij *et al.*, *Test of lepton universality with $B^0 \to K^{*0}\ell^+\ell^-$ decays*, JHEP **08** (2017) 055, arXiv:1705.05802.

[25] LHCb collaboration, *Framework TDR for the LHCb Upgrade: Technical Design Report*, CERN-LHCC-2012-007.

[26] Belle-II collaboration, T. Abe *et al.*, *Belle II Technical Design Report*, arXiv:1011.0352.

[27] LHCb collaboration, *LHCb Trigger and Online Technical Design Report*, CERN-LHCC-2014-016.

[28] J. H. Christenson, J. W. Cronin *et al.*, *Evidence for the 2π decay of the K_2^0 meson*, Phys. Rev. Lett. **13** (1964), no. 4 138.

[29] NA48 collaboration, *A new measurement of direct CP violation in two pion decays of the neutral kaon*, Phys. Rev. Lett. **B465** (1999), no. 335.

[30] KTeV collaboration, *Observation of direct CP violation in $K_{S,L} \to \pi\pi$ decays*, Phys. Rev. Lett. **83** (1999), no. 22.

[31] BELLE collaboration, *Observation of large CP violation in the neutral B meson system*, Phys. Rev. Lett. **87** (2001), no. 9:091802.

[32] BABAR collaboration, *Observation of CP violation in the B^0 meson system*, Phys. Rev. Lett. **87** (2001), no. 9:091801.

[33] L. Wolfenstein, *Parameterization of the Kobayashi-Maskawa Matrix*, Phys. Rev. Lett. **51** (1983), no. 21 1945.

[34] CKMfitter Group, J. C. et al, *Updated results and plots available at: http://ckmfitter.in2p3.fr*, The European Physical Journal C - Particles and Fields **41** (2005) 1.

[35] A. Cerri *et al.*, in *Report from Working Group 4: Opportunities in Flavour Physics at the HL-LHC and HE-LHC*, A. Dainese *et al.*, eds., vol. 7, pp. 867–1158, 12, 2019. arXiv:1812.07638. doi: 10.23731/CYRM-2019-007.867.

[36] J. Brod and J. Zupan, *The ultimate theoretical error on γ from $B \to DK$ decays*, JHEP **01** (2014) 051, arXiv:1308.5663.

[37] W. Altmannshofer and D. M. Straub, *Implications of $b \to s$ measurements*, in *50th Rencontres de Moriond on EW Interactions and Unified Theories*, pp. 333–338, 3, 2015. arXiv:1503.06199.

[38] LHCb collaboration, R. Aaij *et al.*, *Angular analysis and differential branching fraction of the decay $B_s^0 \to \phi\mu^+\mu^-$*, JHEP **09** (2015) 179, arXiv:1506.08777.

[39] LHCb collaboration, R. Aaij *et al.*, *Differential branching fraction and angular analysis of $\Lambda_b^0 \to \Lambda\mu^+\mu^-$ decays*, JHEP **06** (2015) 115, Erratum ibid. **09** (2018) 145, arXiv:1503.07138.

[40] LHCb collaboration, R. Aaij *et al.*, *Angular analysis of the $B^0 \to K^*e^+e^-$ decay in the low-q^2 region*, JHEP **04** (2015) 064, arXiv:1501.03038.

[41] LHCb collaboration, R. Aaij *et al.*, *Angular analysis of charged and neutral $B \to K\mu^+\mu^-$ decays*, JHEP **05** (2014) 082, arXiv:1403.8045.

[42] LHCb collaboration, R. Aaij *et al.*, *Differential branching fractions and isospin asymmetries of $B \to K^{(*)}\mu^+\mu^-$ decays*, JHEP **06** (2014) 133, arXiv:1403.8044.

[43] LHCb collaboration, R. Aaij *et al.*, *Test of lepton universality using $B^+ \to K^+\ell^+\ell^-$ decays*, Phys. Rev. Lett. **113** (2014) 151601, arXiv:1406.6482.

[44] L. Silvestrini, *CHARM-2015 Theory Summary*, in *7th International Workshop on Charm Physics*, 10, 2015. arXiv:1510.05797.

[45] LHCb Collaboration, R. Aaij *et al.*, *Strong constraints on the $K_S^0 \to \mu^+\mu^-$ branching fraction*, arXiv:2001.10354.

[46] R. Aaij *et al.*, *Search for the rare decay $K_S^0 \to \mu^+\mu^-$*, Journal of High Energy Physics **2013** (2013) .

[47] LHCb collaboration, R. Aaij *et al.*, *Measurement of CP observables in $B^{\pm} \to DK^{*\pm}$ decays using two- and four-body D-meson final states*, JHEP **11** (2017) 156, Erratum ibid. **05** (2018) 067, arXiv:1709.05855.

[48] LHCb collaboration, R. Aaij *et al.*, *Observation of the resonant character of the $Z(4430)^-$ state*, Phys. Rev. Lett. **112** (2014) 222002, arXiv:1404.1903.

[49] R. Aaij *et al.*, *Observation of a narrow pentaquark state, $p_c(4312)^+$, and of the two-peak structure of the $p_c(4450)^+$*, Physical Review Letters **122** (2019) .

[50] S. Benson, V. V. Gligorov, M. A. Vesterinen, and M. Williams, *The LHCb Turbo Stream*, J. Phys. Conf. Ser. **664** (2015), no. 8 082004.

[51] LHCb collaboration, *LHCb Upgrade GPU High Level Trigger Technical Design Report*, CERN-LHCC-2020-006.

[52] R. Cenci et al. *Development of a High-Throughput Tracking Processor on FPGA Boards*, PoS **TWEPP-17** (2017) 136.

[53] CMS, T. James, *Level-1 Track Finding with an all-FPGA System at CMS for the HL-LHC*, in *Connecting the Dots and Workshop on Intelligent Trackers*, 10, 2019. arXiv:1910.12668.

[54] ATLAS TDAQ, W. Wu, *FELIX: the New Detector Interface for the ATLAS Experiment*, IEEE Trans. Nucl. Sci. **66** (2019), no. 7 986, arXiv:1806.10667.

[55] https://www.lhc-closer.es/taking_a_closer_look_at_lhc/0.buckets_and_bunches.

[56] Y. Amhis, O. Callot, M. De Cian, and T. Nikodem, *Description and performance studies of the Forward Tracking for a scintilating fibre detector at LHCb*, Tech. Rep. LHCb-PUB-2014-001. CERN-LHCb-PUB-2014-001, CERN, Geneva, Apr, 2014.

[57] D. H. Prez Cmpora, *LHCb Kalman Filter cross architecture studies*, J. Phys. Conf. Ser. **898** (2017), no. 3 032052.

[58] T. Basar, in *A New Approach to Linear Filtering and Prediction Problems*, pp. 167–179, 2001.

[59] R. Aaij *et al.*, *Tesla: an application for real-time data analysis in High Energy Physics*, Comput. Phys. Commun. **208** (2016) 35, arXiv:1604.05596.

[60] A. Abba *et al.*, *A specialized processor for track reconstruction at the LHC crossing rate*, Journal of Instrumentation **9** (2014) C09001.

[61] A. Abba *et al.*, *A specialized track processor for the LHCb upgrade*, Tech. Rep. LHCb-PUB-2014-026. CERN-LHCb-PUB-2014-026, CERN, Geneva, Mar, 2014.

[62] R. Cenci *et al.*, *Real-time reconstruction of long-lived particles at LHCb using FPGAs*, 2020. arXiv:2006.11067.

[63] G. Punzi *et al.*, *Real-time reconstruction of pixel vertex detectors with FPGAs*, PoS **Vertex2019** (2020) 047.

[64] L. Ristori, *An artificial retina for fast track finding*, Nucl. Instrum. Meth. A **453** (2000) 425.

[65] P. V. C. Hough, *Machine Analysis of Bubble Chamber Pictures*, Conf. Proc. C **590914** (1959) 554.

[66] M. Dell'Orso and L. Ristori, *VLSI structures for track finding*, Nucl. Instrum. Meth. A **278** (1989) 436.

[67] F. Morsani *et al.*, *The AMchip: A VLSI associative memory for track finding*, Nucl. Instrum. Meth. A **315** (1992) 446.

[68] F. Bedeschi *et al.*, *An Artificial Retina Processor for Track Reconstruction at the LHC Crossing Rate*, J. Phys. Conf. Ser. **898** (2017), no. 3 032038.

[69] F. Lazzari, *Development of a real-time tracking device for the LHCb Upgrade 1b*, Nov, 2017. Presented 13 Dec 2017.

[70] R. Cenci *et al.*, *Performance of a high-throughput tracking processor implemented on stratix-v fpga*, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **936** (2019) 344 , Frontier Detectors for Frontier Physics: 14th Pisa Meeting on Advanced Detectors.

[71] Dini Group. `https://www.dinigroup.com/web/DNS5GX_F2.php`.

[72] `https://www.intel.com/content/www/us/en/software/programmable/quartus-prime/overview.html`.

[73] `https://www.mentor.com/products/fv/modelsim/`.

[74] G. Bassi, F. Lazzari, M. J. Morello, and G. Punzi, *FPGA implementation of a fast 2D clustering algorithm (VHDL language)*, Nov., 2019. doi: 10.15161/oar.it/23524.

[75] LHCb collaboration, *LHCb computing: Technical Design Report*, CERN-LHCC-2005-019.

[76] T. Sjostrand, S. Mrenna, and P. Z. Skands, *PYTHIA 6.4 Physics and Manual*, JHEP **05** (2006) 026, `arXiv:hep-ph/0603175`.

[77] D. J. Lange, *The EvtGen particle decay simulation package*, Nucl. Instrum. Meth. A **462** (2001) 152.

[78] GEANT4, S. Agostinelli *et al.*, *GEANT4–a simulation toolkit*, Nucl. Instrum. Meth. A **506** (2003) 250.

[79] J. Allison *et al.*, *Geant4 developments and applications*, IEEE Trans. Nucl. Sci. **53** (2006) 270.

[80] LHCb, M. Clemencic *et al.*, *The LHCb simulation application, Gauss: Design, evolution and experience*, J. Phys. Conf. Ser. **331** (2011) 032023.

[81] `https://lhcbdoc.web.cern.ch/lhcbdoc/moore/master/index.html`.