Università di Pisa

Dipartimento di Fisica "Enrico Fermi"
Corso di Laurea Magistrale in Fisica

Real-time luminosity and detector monitoring using FPGAs at LHCb experiment

Candidato:
Daniele Passaro
Mat. 624808

Relatore:
Prof. Giovanni Punzi

Anno Accademico 2021/2022

”La natura è prima dell’uomo,
ma l’uomo è prima delle scienze naturali”

– Carl Friedrich von Weizsacker in Heisemberg,
Fisica e Filosofia

Abstract

For the LHC Run-3, just started in July 2022, the LHCb experiment has adopted for the first time a trigger system based on the complete reconstruction of all collision events. The real-time reconstruction of complex events arriving with an average rate of 30 MHz is a very significant computational challenge, which the collaboration has decided to tackle with a heterogeneous computing system, in which CPU, GPU and FPGA cards are employed at the same time. Many R&D projects were established to increase the computing power of the experiment by realizing specialized computing solutions to carry out the most repetitive and low-level reconstruction steps in the most effective way. One of these projects concerns the implementation on FPGA of a real-time bi-dimensional cluster reconstruction algorithm, realized on the readout boards of the VELO, the pixel detector of LHCb, and has already provided significant savings of computing resources.

However, the real-time accessibility of all hits on a complex detector such as the VELO also allows to probe the further potential offered by FPGA reconstruction in real time. The goal of this thesis work is to explore the feasibility of some of such new opportunities.

This thesis describes the implementation of a FPGA-based system for real-time luminosity measurement and monitoring of the beams at the LHCb interaction point, transparently embedded within the readout cards of the detector. Additional functionalities for monitoring the operational status of the VELO sensors are also discussed, addressing spillover effects and sensors detection efficiency. The results and performances of these realizations are reported, calibrated and measured through detailed simulations, and validated through tests in the actual LHCb experimental setup and in data taking conditions.

Contents

1 Introduction

1.1 Physics opportunities at LHCb

The Standard Model (SM) of particle physics is the theory describing the electroweak and the strong interactions. The SM has been constantly tested over the last 50 years. The observation of the Higgs boson [1, 2] finally completed the picture described by the SM. While remarkably successful in providing many and various precise predictions, the Standard Model leaves a number of fundamental questions unresolved. Many of them reside in the flavor sector of the theory: why are there only three fermions generations (both in the leptonic and in quark sectors) and which mechanism is responsible of their mass hierarchy? which mechanism determines the Cabibbo-Kobayashi-Maskawa (CKM) quark-mixing matrix structure? what is responsible of the observed baryon asymmetry between matter and antimatter? how to deal with non-zero neutrino masses?

The HEP community is addressing these (and many other) questions looking for New Physics (NP) effects that could provide experimental evidence for *Beyond the Standard Model* (BSM) theories, such as SuperSymmetry scenarios, leptoquark models and extended gauge bosons theories including W' and Z' bosons.

Tests of the SM are performed both in the *energy* and the *precision* frontiers: new particles can be directly searched for in high-energy collisions through on-mass-shell production, or indirect evidences of their existence can be asserted by precision measurements of flavor-changing transition in heavy-hadrons physics. Given its peculiar structure, in which effects of new particles contributing at loop level can be measured, the flavor physics has been historically relevant to probe the SM structure. Flavor physics gave rise for instance to the so-called *November Revolution* following the discovery of charm quark, previously postulated by the GIM mechanism, and confirmed the existence of three generations of fermions as postulated by the CKM mechanism thanks to the discovery of the beauty quark. Within the heavy-flavor sector of the SM, measurements of beauty and charm hadrons decay rates or $\mathcal{CP}$ violation asymmetries can probe for effects outside SM predictions, even with typical energy scales much higher than those involved in the process, because of subtle contributions - both at tree and loop level - from yet unseen, heavy (or weakly-coupled) new particle states. Channels often regarded as best suited for NP discoveries include heavy neutral-meson oscillations

($D^0 - \overline{D^0}$ for instance) and rare decays such as $B^0_S \longrightarrow \bar{l}l$ and $b \longrightarrow s\bar{l}l$ (l is a lepton), as well as SM-forbidden processes (*e.g* LFV processes like $\tau \longrightarrow \mu\mu\mu$ or $\tau \longrightarrow \mu\phi$ decay). It has been shown that measurement of these observables, happening at the GeV energy scale, can actually probe mass scales from 10^3 to 10^4 TeV, depending on the detailed structure assumed for the new interactions [3], while current direct searches for new states (for instance at LHC), although potentially offering more obvious experimental signatures, are limited to mass scales in the TeV range.

Precision studies of the $\mathcal{CP}$ asymmetry in elementary processes are another probe into the fundamental physics at high mass scales, and are also crucial for understanding the mechanism producing the observed cosmological imbalance of matter over antimatter. Sakharov [4] in 1967 pointed out that one of the required ingredient to explain the baryon asymmetry is precisely the violation of $\mathcal{CP}$ symmetry in elementary processes. The quark sector of the SM incorporates the CP asymmetry within a single complex non-vanishing phase in CKM matrix. Many experimental results confirm the CKM-based description [5–7], but remarkably, its observed size is not enough to explain the cosmological abundance of matter over antimatter, and additional CP-violating effects are very actively being searched.

These brief examples should be enough to indicate why Quark-flavor phenomenology is a powerful experimental tool in the exploration for BSM effects, and why a dedicated experiment was designed with this specific purpose to operate at the LHC - the most abundant source of heavy quarks available today.

That experiment is LHCb, and was specifically optimized to address $\mathcal{CP}$ violation in b- and c-mesons and their decay rates and modes. Being the $b\bar{b}$ and $c\bar{c}$ production cross sections in the LHCb acceptance (covering pseudorapidity range $2 < \eta < 5$) respectively of $\sigma_{pp\rightarrow b\bar{b}} \approx 500\,\mu$b and $\sigma_{pp\rightarrow c\bar{c}} \approx 2000\,\mu$b at a center-of-mass energy of $\sqrt{s} = 14$ TeV, the LHCb produces a huge amount of both B and D mesons, thus being optimal to perform precision measurements of quark-flavor quantities.

The LHCb experiment with Run-1 and Run-2 data has already provided some of the most relevant measurement in the flavor-physics field: the discovery of the ultra-rare decay $B^0_S \longrightarrow \mu^+\mu^-$ [8], the first-ever observation of $\mathcal{CP}$ violation in the charm sector [9], the most precise measurement of CKM angle γ [10], the B^0_S $\mathcal{CP}$-violating weak phase [11] and measurements of lepton-flavor-universality observables $R(K)$, $R(K^*)$ and $R(D)$ [12] that showed some tensions with the SM predictions. Many other studies have been carried out, including the discovery of pentaquark and tetraquark states [13, 14] and several production cross section measurements in electroweak, QCD and heavy ion processes [15–17]. Production cross section measurements are also important since their knowledge is essential to make predictions on the expected rate of any process. The uncertainties associated to cross section measurements are dominated by the systematic uncertainties on the absolute luminosity, a parameter

Observable	Run 2 LHCb	Upgrade I		Upgrade II
	6 fb^{-1}	(23 fb^{-1})	(50 fb^{-1})	(300 fb^{-1})
CKM tests				
γ ($B \to DK$, *etc.*)	$4°$	$1.5°$	$1°$	$0.35°$
ϕ_s ($B_s^0 \to J/\psi\phi$)	32 mrad	14 mrad	10 mrad	4 mrad
$\lvert V_{ub}\rvert/\lvert V_{cb}\rvert$ ($\Lambda_b^0 \to p\mu^-\overline{\nu}_\mu$, *etc.*)	6%	3%	2%	1%
a_{sl}^d ($B^0 \to D^-\mu^+\nu_\mu$)	36×10^{-4}	8×10^{-4}	5×10^{-4}	2×10^{-4}
a_{sl}^s ($B_s^0 \to D_s^-\mu^+\nu_\mu$)	33×10^{-4}	10×10^{-4}	7×10^{-4}	3×10^{-4}
Charm				
ΔA_{CP} ($D^0 \to K^+K^-, \pi^+\pi^-$)	29×10^{-5}	13×10^{-5}	8×10^{-5}	3.3×10^{-5}
A_Γ ($D^0 \to K^+K^-, \pi^+\pi^-$)	11×10^{-5}	5×10^{-5}	3.2×10^{-5}	1.2×10^{-5}
Δx ($D^0 \to K_S^0\pi^+\pi^-$)	18×10^{-5}	6.3×10^{-5}	4.1×10^{-5}	1.6×10^{-5}
Rare Decays				
$\mathcal{B}(B^0 \to \mu^+\mu^-)/\mathcal{B}(B_s^0 \to \mu^+\mu^-)$	69%	41%	27%	11%
$S_{\mu\mu}$ ($B_s^0 \to \mu^+\mu^-$)	—	—	—	0.2
$A_T^{(2)}$ ($B^0 \to K^{*0}e^+e^-$)	0.10	0.060	0.043	0.016
A_T^{Im} ($B^0 \to K^{*0}e^+e^-$)	0.10	0.060	0.043	0.016
$\mathcal{A}_{\phi\gamma}^{\Delta\Gamma}(B_s^0 \to \phi\gamma)$	$^{+0.41}_{-0.44}$	0.124	0.083	0.033
$S_{\phi\gamma}(B_s^0 \to \phi\gamma)$	0.32	0.093	0.062	0.025
$\alpha_\gamma(\Lambda_b^0 \to \Lambda\gamma)$	$^{+0.17}_{-0.29}$	0.148	0.097	0.038
Lepton Universality Tests				
R_K ($B^+ \to K^+\ell^+\ell^-$)	0.044	0.025	0.017	0.007
R_{K^*} ($B^0 \to K^{*0}\ell^+\ell^-$)	0.12	0.034	0.022	0.009
$R(D^*)$ ($B^0 \to D^{*-}\ell^+\nu_\ell$)	0.026	0.007	0.005	0.002

Table 1.1: Extrapolated uncertainties for some key flavour observables at future upgrades of LHCb . From reference [18].

that is rather difficult to determine with some precision in spite of its conceptual semplicity, as will be discussed at some length in the present work.

In order to probe the SM predictions with increasing sensitivity with the "intensity" approach, very large data samples are required. The LHCb experiment is moving towards future upgrades aimed at increasing the amount of collected data. With the increased data samples it will be possible to further constrain the parameter space of BSM theories. The extrapolated uncertainties for some flavor quantities thanks to the future Upgrades of LHCb are shown in Table 1.1. In Run-3 (that is now starting) the instantaneous luminosity at which the experiment is going to operate will increase by a factor ×5 in respect of Run-2, reaching the value of $\mathcal{L} = 2 \times 10^{33}\ \mathrm{cm}^{-2}\mathrm{s}^{-1}$.

Other experiments at the LHC, namely ATLAS and CMS, already work with such luminosities. The fundamental difference that distinguishes LHCb, and that imposes the deployment of a different strategy in the experiment design, is that the cross-sections of the processes of interest for the experiment (b and c production) are orders of magnitude larger than the ones in ATLAS and CMS. Further, the events of interest for the physics goals of LHCb lack of easily recognizable features, such as requirements on the transverse momentum of the particle or a missing energy. It is thus crucial to reconstruct in detail each and every event to be able to select the relevant events for the physics analysis, but this task becomes more complex as the instantaneous luminosity increases.

The experiment has undergone an almost total upgrade to cope with the technical complications coming along with the higher luminosity. In this way it is expected to collect an integrated luminosity of $20 - 25\,\mathrm{fb}^{-1}$ during Run-3, targeting a total amount of $50\,\mathrm{fb}^{-1}$ by the end of Run-4.
Among the several upgrades performed by the LHCb collaboration for Run-3, an important role is occupied by the trigger and data-acquisition (TDAQ) system, that has been completely renewed compared to Run-2, being one of the largest cost items in the upgrade project. The new TDAQ system aims at reading out and at processing in real-time each and every event at the average bunch crossing rate of 30 MHz, performing event selections through a fully software trigger. This processing must occur in real time, to allow identifying the relevant physics information and operate the strong reduction of data needed to fit the capacity of permanent storage.

Track reconstruction and event selection at 30 MHz is however a very challenging task, that the collaboration decided to deal with by means of huge computing farm where *heterogeneous* computing solutions are adopted. The heterogeneous computing consists in the use of machines with different architectures in the same workflow, allowing to optimize the computing performances by assigning the most suitable tasks to each of them based on their architecture. This is often achieved by connecting traditional CPUs to more-or-less specialized co-processors, also said *hardware accelerators*, such as GPUs and FPGAs. At the LHCb the first level of the high-level trigger system is executed by a set of GPUs, reserving the usage of CPUs to the second level. FPGAs boards are employed to realize the readout system, performing pre-processing of the data directly sent by each sub-detector front-end electronics. The use of hardware accelerators in the HEP community is becoming more and more relevant to enhance the computing performances of the experiments, especially as the slowing down of Moore's law is making it more difficult to simply exploit the historical increasing trend of computing power of general-purpose architecture with time.

Many R&D projects have been established in LHCb to explore the usage of heterogeneous computing system to enhance the computing power of the experiment in foresight of the future Upgrades [19], when the instantaneous luminosity will further increase reaching, during Run-5, the peak value $\mathcal{L} = 2 \times 10^{34}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$, corresponding to a mean pile-up of ~ 55 .

1.2 Towards a new real-time reconstruction paradigm

Performing the reconstruction sequence in real-time suggests a new way of realizing, and specially of *conceiving*, the data reconstruction.

The real-time reconstruction in LHCb currently relies on the strategy of assembling the raw data within a central facility, the *Event Builder*. The assembled event, composed of raw data from each sub-detector, is analyzed in real-time to perform the trigger selection by a large computing farm composed of both GPUs and CPUs where the high-level triggers run. Much of the computing time is spent to perform low level and ripetitive tasks, such as tracking reconstruction and clusterization of particle hits on the detector.

The reconstruction sequence would be improved if such tasks would be moved to an earlier stage of the data acquisition chain, instead of being performed during the event reconstruction step. This would be indeed extremely advantageous, saving computing power and freeing the CPU farm of redundant and time-consuming tasks. This would allow to make full exploit of CPU time and computing power that can be devoted to higher level tasks in physics selection, where the flexibility of general-purpose CPUs and their associated high-level software environment is actually well employed.

The idea to perform some tasks of the reconstruction sequence at earlier and earlier stages of the data acquisition chain, even before the event-building stage, is not only relevant to enhance the computing performances of the high-level processing farm, but it also constitutes a new paradigm in HEP experiments. Moving for instance the track reconstruction at the readout level would completely change the way events are assembled at the Event Builder, which would handle already reconstructed tracks instead of raw hits on the detector. A new direction in real-time reconstruction has thus established, for which the output data from a detector would no longer be raw data, but they would be well-reconstructed primitives *e.g.* tracks or reconstructed particle hits.

The RETINA project [20] has established in this context. Its main goal is to develop on FPGAs, at the readout level of the data acquisition chain, a parallel, high throughput, low latency tracking algorithm, named *Artificial Retina*, inspired to the natural vision of mammals brain.

The RETINA project has concluded the R&D stage, producing a demonstrator that showed the feasibility of such a device. A more realistic prototype is now under implementation aiming at reconstructing the tracks generated inside the *Vertex Locator* (VELO), the silicon-pixel detector of the experiment. The first stage of the reconstruction - hits reconstruction from pixel raw data, realized by a 2D-clustering FPGA firmware implemented on the readout boards of the VELO - is now in the last commissioning steps to become the default for the data taking during Run-3.

The implementation of the 2D cluster-finder algorithm at the readout level opens up the way to new developments that were not explicitly designed for. Indeed the real-time availability of VELO clusters at a very early stage of the data acquisition system leads to the opportunity to further process them on the readout boards themselves, adding more useful features to the real-time reconstruction besides the main goal of trigger selection. Exploration of additional functions of this kind is an important task, both for their own sake, and to elucidate the actual potential gains of the transition to this new paradigm, providing further motivation to support it, and allowing to prepare for it.

In this thesis work, I tried to explore what advantages could be obtained exploiting the accessibility of particle hits data at very low level of the data-acquisition system to provide new functionalities to the experiment.

I developed a firmware - implemented on the FPGA chips of the readout boards of the VELO - aimed at accumulating cluster-based statistics. The possibility of selecting specific types of clusters, located in specific regions, offers an important advantage with respect to the possible use of raw pixel counts. In fact, by counting the number of reconstructed hits in specific regions of the sensor it is possible to monitor some quantities that are crucial for the operation of the experiment.

The quantities I have primarily addressed during my thesis work are the *instantaneous luminosity* at the LHCb interaction point and the *beamline nominal position*. The knowledge of these quantities is relevant in operating the experiment, has will be discussed in the following chapters. I exploited the real-time availability of hits data for each event also for monitoring the operational status of the VELO detector.I studied how to implement algorithms for monitoring the *detection efficiency* of the sensors and the occurrence of replicated clusters on the same region of the sensors in consecutive events, effect known as *spillover*.

2 The LHCb experiment

Chapter Contents

2.1 LHC facilities

The Large Hadron Collider (LHC) is the world's largest particle accelerator, built at CERN (European Organization for Nuclear Physics Research). It is a superconducting proton-proton and heavy-ion collider, installed in a 27 km long circular tunnel - previously housing LEP - and about 150 metres underground, beneath the Swiss-French border. A schematic of the LHC accelerator complex is shown in Figure 2.1. The LHC accelerate protons up to 6.8 TeV and lead ions up to 2.3 TeV per nucleon. Before being accelerated inside the LHC, protons and ions go through a series of acceleration stages, which consist of the older accelerators at CERN. Regarding only proton collisions, hydrogen ions are extracted from an hydrogen gas. Then the H^- ions are accelerated by LINAC4 to an energy of 160 MeV; LINAC4 injects its protons inside the Proton Synchrotron Booster, a storage ring that accelerates protons up to 1.4 GeV; here both the electrons are stripped from the ions leaving only the proton, before injecting them

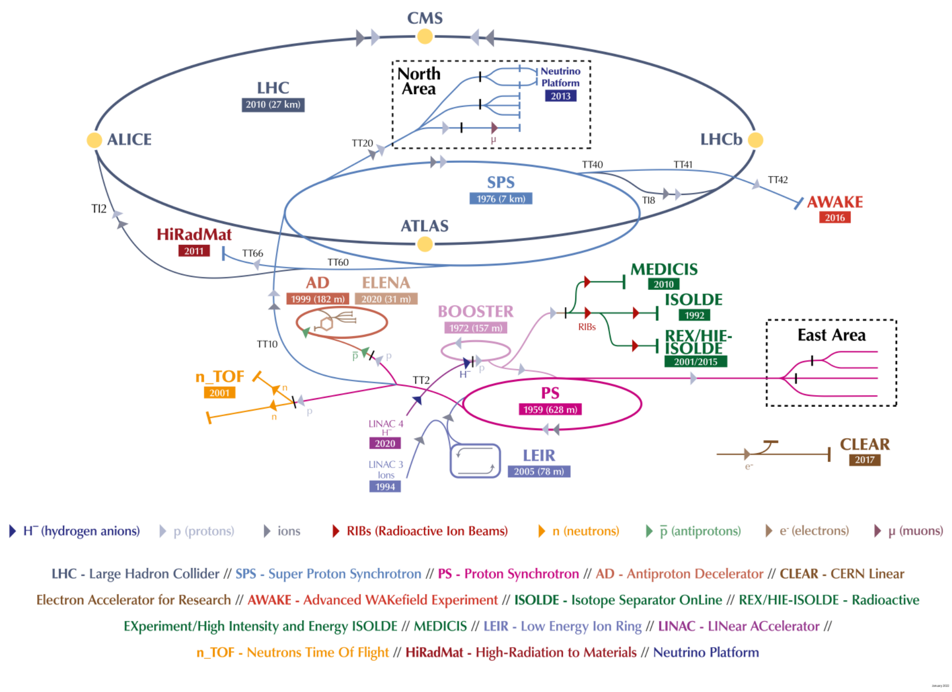

Figure 2.1: The LHC is the last ring (dark blue line) in a complex chain of particle accelerators. The smaller machines are used in a chain to help boost the particles to their final energies and provide beams to the experiments.

in the Proton Synchrotron (PS) which accelerates them at 26 GeV. Finally, the Super Proton Synchrotron accelerates the resulting beams at 450 GeV before they are at last injected (over a period of several minutes) into the main ring constituting the LHC. Here two proton or lead ion beams circulate in opposite directions, using two different beam pipes. Radio-frequencies cavities are used to further accelerate the beams up to their nominal energies. Beams are bent by 1232 superconducting dipole magnets installed all along the accelerator, while 392 quadrupole magnets are used to keep the beams focused. Superfluid-helium cooling systems are used to keep the superconducting magnets at their operating temperature of 1.9 K. Proton bunches, constituing the proton beams, are each one made up by $\sim 10^{11}$ protons, and are time-spaced for multiples of 25 ns, corresponding to a bunch-crossing frequency of 40 MHz. Beams are split in exactly 2808 bunches, so the average bunch-crossing rate is $\sim$ 30 MHz. The LHC, upgraded during the Long Shutdown 2, is designed to deliver an instantaneous luminosity of $2 \cdot 10^{34}\,\text{cm}^{-2}\text{s}^{-1}$, providing proton-proton collisions at centre-of-mass energy of $\sqrt{s} = 13.6$ GeV. Beams collide at the collision points installed in corrispondence of the LHC ring; there are four main interaction points where the main experiments at LHC are housed : ATLAS, CMS, LHCb and ALICE. Other smaller experiments are installed along the accelerator. ATLAS and CMS are general-purpose high-luminosity experiments, operating at the instantaneous luminosity directly delivered by the LHC, whose main goal is the search of new non-standard particles and the discovery of extra-dimensions and dark matter. ALICE is a heavy-ion low-luminosity experiment, running in the range $10^{29} - 10^{32}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$, aiming at investigating the quark-gluon plasma, a state of matter very similar to the condition of universe shortly after the Big Bang. The LHCb experiment is instead dedicated to heavy-quark physics, and operates in a mid-luminosity range $2 - 20 \cdot 10^{33}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$. In the following sections the LHCb-Upgrade I detector and data-acquisition system are shown.

2.2 The LHCb-Upgrade I detector

The LHCb experiment, shown in Figure 2.2, is a single-arm forward spectrometer, whose acceptance covers pseudorapidity range $2 < \eta < 5$. It is designed to study particles containing heavy-quarks, that are b and c-quarks, mainly produced at high pseudorapidity. The LHCb experiment underwent a major upgrade during Long Shutdown 2, in order to cope with the increased luminosity and to be able to process every event in real-time, at the average bunch-crossing rate of 30 MHz. The average number of pp collisions per bunch-crossing will be ~ 7.6, and the instantaneous luminosity is expected to be $\sim 2 \cdot 10^{33}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$. It is expected to collect at least $50\,\mathrm{fb}^{-1}$ of data during Run-3 and Run-4, and the analysis of this dataset will provide unprecedented precision in many measurement in the flavor sector of the Standard Model. A completely new trigger system has been developed, and many sub-detectors, as well as the readout and data-acquisition systems, have been upgraded to satisfy the new requirements. The upgraded experiment includes a new tracking system, consisting of a new silicon detector, the Vertex Locator (VELO), used to measure very precisely primary and secondary vertex positions, the silicon strips upstream tracker (UT) and the scintillating fibers (SciFi), mainly used for track fitting. A dipole magnet right after the UT and before the SciFi is present, generating a magnetic field with bending power $\sim 4\,\mathrm{T} \cdot \mathrm{m}$. Two Cherenkov light detector are used, the RICH1, before the UT, and RICH2, located after the SciFi; they are used to identify charged hadrons. The ECal and the HCal are mostly used to identify electrons, photons and hadrons, while the identification of muons is realized by means of the Muon Stations, the furthest detectors from the LHCb interaction point.

The trigger system has been completely renewed in respect to the one employed during Run-2. In previous runs there used to be a Level-0 (L0) hardware trigger, that used information from the Calorimeter system and Muon stations to reduce the bunch-crossing rate from 40 MHz to 1.1 MHz in input to the high-level software trigger, which was the maximum value at which the whole detector could be read out by design. In Run-3, the collaboration decided to remove the L0 trigger, since simulation studies showed that selection efficiency for a large number of decay channels would be quite low, and no benefit would have been gained from the increased luminosity. In order to keep a high selection efficiency for a wide set of very

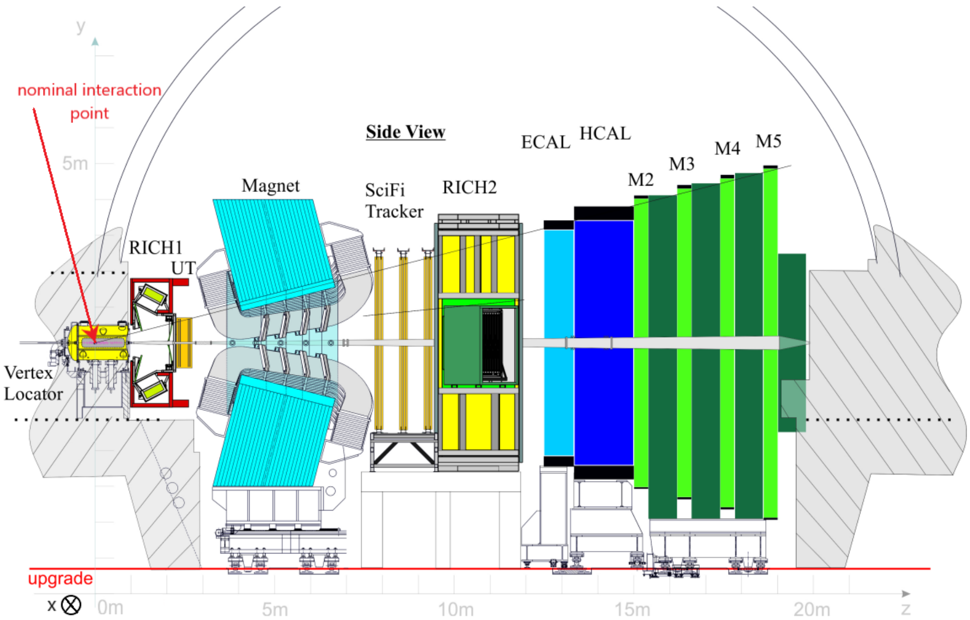

Figure 2.2: Layout of LHCb-Upgrade I detector, side-view from the center of LHC ring. The LHCb axis frame system is displayed too.

different events, the collaboration decided to upgrade the trigger system, that now is based on reconstructing in real-time the whole information provided by the detector for each and every collision occurring in the LHCb interaction point.
In the following sections, detailed information about the sub-detectors and the DAQ and trigger systems are reported.

2.2.1 The Upgrade-VELO

The Vertex Locator (VELO) [21] is the innermost sub-detector that surrounds the LHCb interaction region. The main purpose of the VELO detector is to reconstruct primary and secondary decay vertices with the spacial resolution better than the typical decay lenght of c and b-hadrons ($\mathcal{O}(100-500\mu\text{m})$). The information from the VELO is critical to identify characteristic displaced vertices of heavy flavoured particles, underpinning physics analyses within LHCb. At trigger level the VELO identifies tracks with high impact parameter and provides the first information to reduce the minimum bias rate. The offline performance of the VELO is equally important, especially high resolutions for secondary vertices are crucial to resolve heavy-mesons oscillations and to study *CP*-dependent time asymmetries. The VELO is a silicon pixel detector, designed to be radiation-hard and to be capable to operate at 40 MHz, featuring modules with integrated microchannel evaporative CO_2 cooling, technology first developed for the GigaTracker at NA62 experiment [22]. The detector is placed inside the LHC vacuum pipe, so that it can be placed as close as possible to the beamline; indeed, when

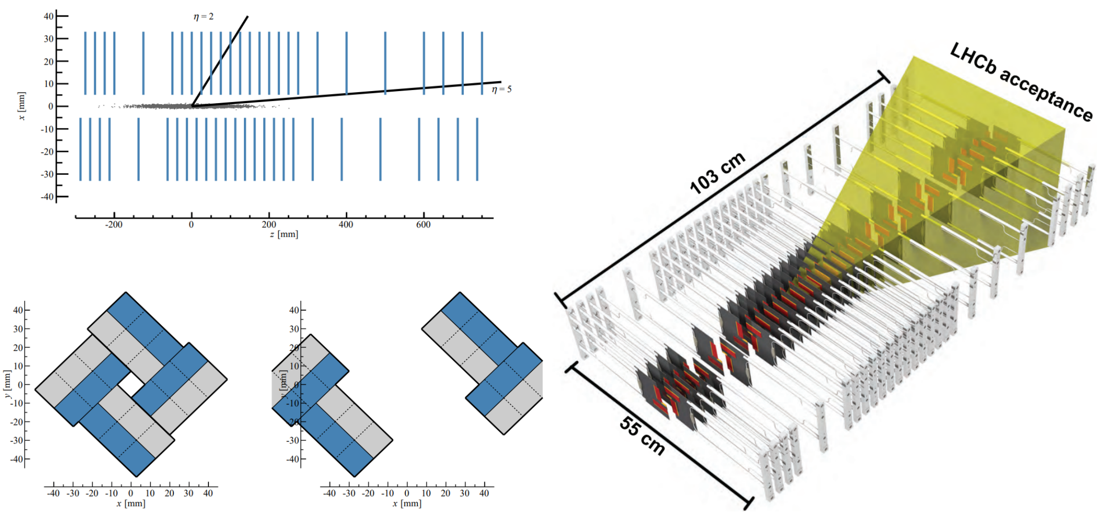

Figure 2.3: Up-Left: 2D visualization of VELO stations and interaction region seen from the xz plane; Down-Left: xy position of VELO modules in *fully-closed* and *fully-open* configuration; Right: 3D visualization of VELO detector.

full-closed, the innermost VELO sensor is just 5.1 mm away from the nominal interaction point, making it the closest detector to the beamline ever built in any HEP experiments at CERN. The VELO consists of 26 *stations*, each one made up of two halves, referred to as *modules*, placed at different z-positions surrounding the nominal interaction point, both upstream ($z > 0$) and downstream ($z < 0$), as displayed in Figure 2.3 Right. Each module is made of two *tiles*, each made up of two sensors. A VELO sensor is made up of 768×256 pixels, and is 200 μm thick *n-in-p* silicon. Sensors are mounted on both sides of the substrate: in Figure 2.3 down-left, grey sensors are mounted on the downstream side of the connector, while the blue ones are mounted on the upstream side. Blue sensors local coordinates are "reversed" compared to the grey ones. This is particularly important to consider when raw-data are processed (see section 3.1.3). Sensors are read out by means of three specific ASICs, dubbed *VeloPix*. The VeloPix ASICs features a matrix of 256×256 active silicon pixels, with cell dimensions of 55 μm × 55 μm, bump-bonded to the underlying silicon sensor. Taking into account all the sensors, there are more than 40 milion pixels to be read out. All the dimensions of the sensors have been optimized to reduce the amount of material that particles pass through, so to reduce the effects of multiple scattering and degradation of the particle energy. The raw hit resolution varies in the range 9 – 15 μm, depending on the incoming particle inclination. In order not to lose detection efficiency and acceptance for highly-inclined tracks, a 2-pixel wide overlap between sensors constituting the same tile is introduced. VELO modules are not fixed at the design positions because during the LHC ramp, beams are too instable; during this period it is thus wise to keep the sensors as far as possible (~ 3 cm in the horizontal direction) from the beamline to avoid undesired interactions with the beams. The positions of the

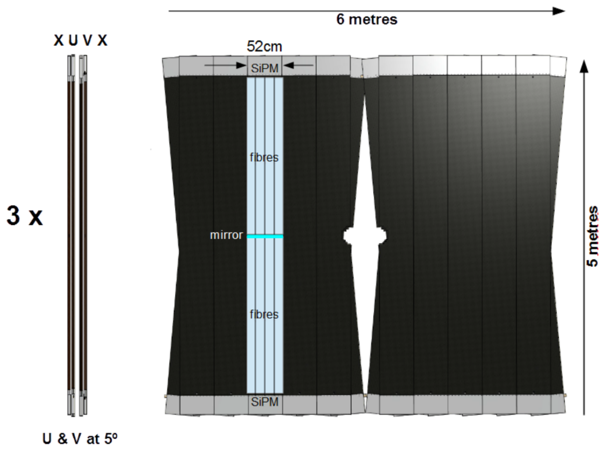

Figure 2.4: SciFi detector plane

modules in the closed (LHC stable beam operation) and open position are shown in Figure 2.3 down-left.

2.2.2 Scintillating Fibers

The Scintillating Fibers tracker (SciFi) [23] is downstream the magnet, and consists of three stations: T1, T2 and T3. They are designed to provide standalone pattern recognition with a high efficiency together with high resolution in the bending plane of the magnetic field. Each station consists of four detection planes which provide measurements of the coordinates (x, u, v, x), with strips or straws orientated at $0^\circ, +5^\circ, -5^\circ, 0^\circ$ with respect to the vertical axis: the first and the last have vertical staves and the two central silicon sensors have tilted staves. The detector modules have 2.5 m long scintillating fibres with a diameter of 250 μm. The two modules surrounding the beam pipe have a hole to accommodate it and contain six fiber layers, while the remaining modules have five fiber layers because of lower radiation exposure. Scintillating fibers are made of a polymer core with the addition of 1% in weight of fluorescent dye. The fibres are read out by Silicon PhotoMultipliers (SiPMs) contained in the so-called "Read-out Boxes" at the top and bottom of the detector. The SciFi detector is mainly used for tracking purposes, especially for downstream tracks; it provide a high precision estimate of the momentum of charged particles. This leads to a precise mass resolution for decayed particles.

2.2.3 RICH1 and RICH2

The RICH (Ring Imaging CHerenkov) system [24] of LHCb provides particle identification (PID) of charged hadrons over the momentum range 1.5 – 100 GeV. This is essential for the study of hadronic final states and central to the physics goals of the experiment, to make pre-

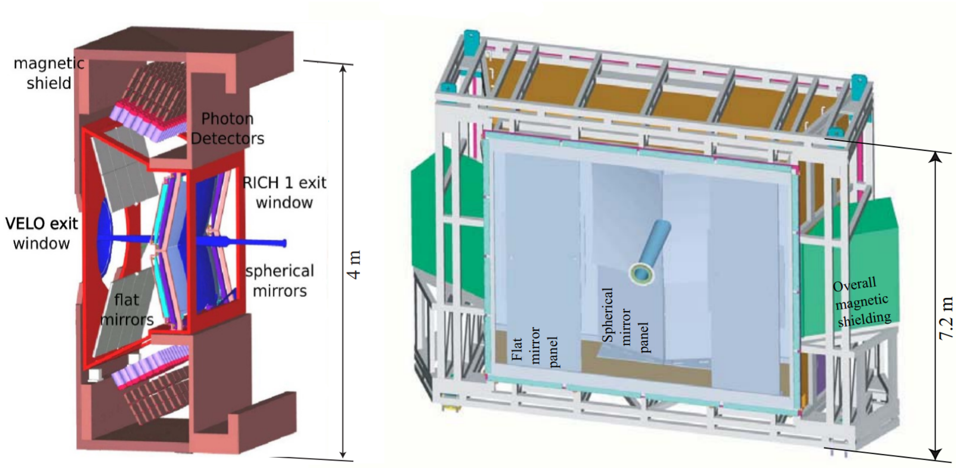

Figure 2.5: RICH1 and RICH2

cision measurements of $\mathcal{CP}$ violation and rare decays of b and c hadrons. PID at the lower end of the momentum range is required for particle-antiparticle tagging, which relies on the identification of b (and $\overline{\text{b}}$) hadrons from their decay to charged kaons via the $\text{b} \longrightarrow \text{c} \longrightarrow \text{s}$ decay chain. Also, efficient identification of protons with the RICH system for example facilitates the study of heavy-flavour baryonic states. The RICH system comprises two detectors with three Cherenkov radiators. RICH 1 comprises a C_4F_{10} gas radiator(n = 1.0014), and is located upstream of the spectrometer dipole magnet, right after the VELO. It covers an angular acceptance of 25 – 300 mrad. The RICH 2 detector has a CF_4 gas radiator (n = 1.0005), located downstream of the magnet, covering the angular acceptance 15 – 120 mrad. RICH 2 is responsible for the high momentum coverage, since such tracks are usually produced at low angles. The focusing of Cherenkov light is accomplished using spherical mirrors, tilted to bring the image out of the spectrometer acceptance. The light rays are again reflected using secondary plane mirrors to focus the ring images on the photon detectors, located above and below the beam, in a region where they can be shielded from the B-field of the spectrometer magnet by means of magnetic shields. Arrays of 'multi-anode photomultiplier tubes' (MaPMTs) are used to detect individual Cherenkov photons, covering a total area of 1.6 m^2 for RICH1 and 2.1 m^2 for RICH2.

2.2.4 Calorimeter system

The LHCb-Upgrade I Calorimeter system [24] has two main function: (1) precise measurement of $e^{\pm}$, π and γ energy; (2) contribution into particle identification algorithms. The Calorimeter system is composed of two halves: the Electromagnetic Calorimeter (ECal) and the Hadronic Calorimeter (HCal), placed downstream the dipole magnet, behind the RICH2 detector. During Run-1 and Run-2, the Calorimeter system was used to provide L0-level trigger too, es-

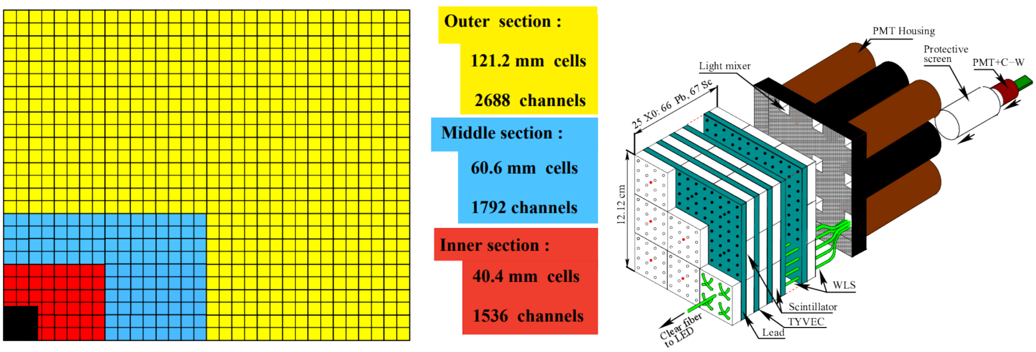

Figure 2.6: Left: quarter of ECal module displacement; Right: Shashlik modules

pecially using information from the HCal and other two sub-detector, the Scintillating Pad Detector (SPD) and PreShower (PS). The latter two have already been removed for the Upgrade I, while the HCal is expected to be removed during LS3, since its information are (were) mainly used for L0 selections.

The ECal is built in the *shashlik* calorimeter technology, which consists of a sampling structure alternating 4 mm-thick polystirene scintillating tiles and 2 mm lead sheets. The scintillation light is collected using wavelength shifting fibers and read out by individual PMTs mounted on the back of the tiles. The total thickness of the ECal is 25 radiation lenghts. The structure of the calorimeter is segmented into three zones, depending on the radial distance from the beamline, thus depending the irradiation levels. The *Inner* section consists of 40.4 mm lateral-dimension cells, the *Middle* one of 60.6 mm wide cells and the *Outer* section in made up of 121.2 mm wide cells. Depending on the type of module, the energy resolution for the ECal is $\sigma_E/E = (8.5 - 9.5)\%/\sqrt{E(\text{GeV})} \oplus 0.8\%$.

2.2.5 Muon Stations

The LHCb muon detector [24] uses the penetrating power of muons to provide a robust muon identification. The reconstruction and identification of muons are key attributes of the particle identification system of LHCb, since decay channels of b– and c–hadrons with muons in the final state represent a crucial part of the LHCb physics programme. The muon detector consists of four stations, M2 to M5, placed behind the hadronic calorimeter and interleaved with iron walls that act as muon filters. In Run-1 and Run-2 there existed M1 station too, located in front of the calorimeters, that provided p_T measurement for L0 trigger. Stations are equipped with Multi–Wire Proportional Chambers (MWPCs) and are divided into four regions, each one equipped with chambers of different granularities to cope with the large variation in particle rates passing from the central part to the detector periphery.

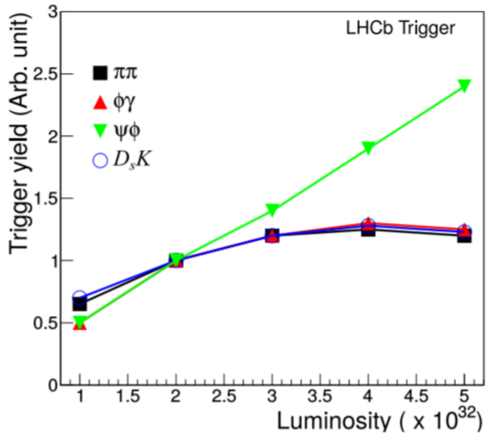

Figure 2.7: Trigger selection efficiencies for several B-meson decays of Run-2 trigger as the instantaneous luminosity increases

2.2.6 DAQ system and Real Time Analysis

The increase in the instantaneous luminosity at the LHCb interaction point and the need for more accurate measurements led the collaboration to completely renew the readout and trigger systems. The upgraded readout system is designed to operate at the average event rate of 30 MHz, and the fully-software trigger system will process every event, using information from all the sub-detectors. This decision has been driven by simulation studies [25] showing that, with the trigger sequence used during Run-1 and Run-2, there would have been loss of efficiency in hadronic B-meson decay channels along with the increase in instantaneous luminosity. The L0 trigger in previous LHCb Runs indeed based its decisions on the deposit of several GeV of transverse energy, E_T , by charged hadrons, muons, electrons or photons. While this provided high efficiencies on dimuon events, it removed half of the fully hadronic signal decays. In these hadronic decays the E_T threshold - required to reduce the rate of triggered events to an acceptable level - is already a substantial fraction of the B meson mass. Any further increase in the rate requires an increase of this threshold, which then removes a substantial fraction of signal decays. As shown in Figure 2.7, the trigger yield saturates for hadronic channels with increasing luminosity. The removal of a L0 trigger based on low-level quantities and its replacement with a trigger based on a detailed event reconstruction performed in software are crucial to allow the selectivity needed to preserve trigger efficiency at higher luminosities. The "side effect" of this new DAQ organization is its computational complexity, and the need for handling of very high input data flow $\mathcal{O}(50\,\text{TB/s})$.

Therefore, the upgrades of the readout and trigger systems have been the most important ones during the Long Shutdown 2. In this context the Real-Time Analysis project was established (Figure 2.8 Right), aiming at providing the needed reconstruction quality in real time. This includes allowing for second-level trigger (HLT2) alignment and calibration with

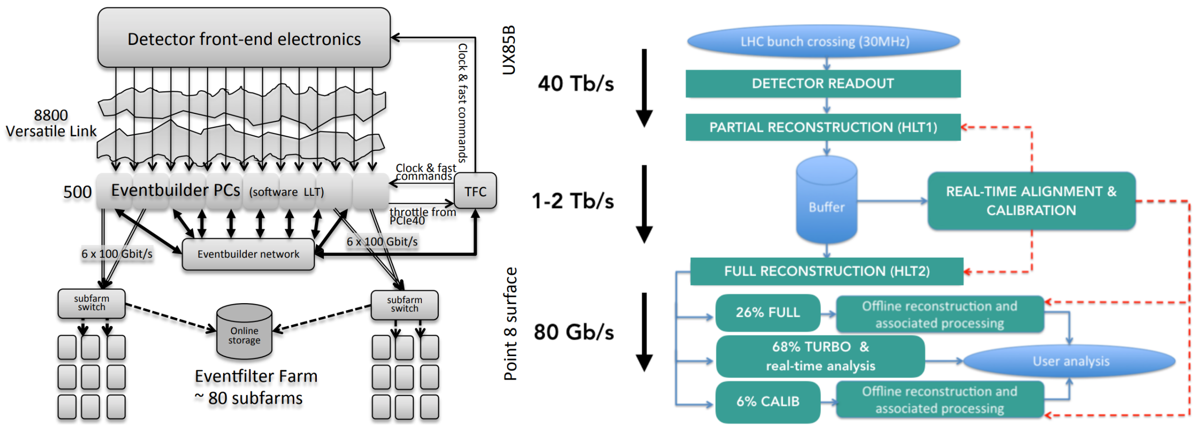

Figure 2.8: Left: DAQ system sketch; Right: High-level triggers and Real-time analysis systems.

quasi-offline precision, and a new streaming strategy (TURBO) [26, 27], that bases the physics analysis on candidates reconstructed in the second-level trigger (HLT2), discarding the raw event information - thus bypassing the traditional offline reconstruction step.

The DAQ system [28] is made up of different sub-systems: the Event Builder (EB), the Timing and Fast Control (TFC), the Experimental Control System (ECS) and lastly the Event Filter Farm (EFF), where the software trigger selections are executed. The structure of the readout system is displayed in Figure 2.8 Left.

Event Builder

Data from the detectors in the underground area are delivered to the surface by radiation-hard 300 meters-long optical fibers. The detector front-end electronics performs zero-suppression, then data from each detector are sent a given number of EB nodes. Within each EB node a custom PCIe board (called *PCIe40*) equipped with an FPGA chip is inserted, constituting the readout board for each sub-detector. They perform data-preprocessing and data-formatting tasks, and act as interface between the front-end electronics and the TFC and ECS systems. Preprocessed data are then sent to PC servers inside the EB nodes. Each event is processed by multiple nodes, reducing the amount of data that each node needs to store and process. The EB nodes exchange among them data relative to the same event, thus assembling the complete event information to be further analyzed by the EFF. A link network (Event Builder Network) is required to connect the EB nodes to each others.

Timing and Fast Control

TFC is the system responsible for the distribution of clocks, timing and trigger information and commands to the front-end and readout systems. It delivers a variety of global signals: 40 MHz clock synchronized to the LHC master clock, synchronous commands to control event processing in FE and readout, calibration commands for detectors, and distribution to FE of ECS-generated electronics configuration.

Event Filter Farm

The Event Filter Farm hosts the high level triggers hardware architectures. The HLT sequence [28] is split into two stages, conventionally referred to as HLT1 and HLT2, separated by a disk buffer which holds events between the stages while the detector is aligned and calibrated with offline precision. The main goal of the HLT1 stage is to reject events which do not contain particles of interest for any LHCb analyses, while efficiently saving those which do. The HLT1 achieves this task by executing a subset of the full offline charged particle reconstruction (tracking), and uses a small number of inclusive single or two-track selections to classify the event as interesting or not. In Run 3, the HLT1 is completely performed by an array of GPUs installed in the same servers hosting the Event Builder. The HLT1 application is implemented in C++ with CUDA extensions and is named *Allen* [29]. The Allen application performs various tasks:

- decoding of the raw input into the LHCb global coordinate system;
- clustering, *i.e.* the reconstruction of the hit, merging in a single coordinate all the measurements of a detector layer due to the passage of the same particle;
- pattern recognition, *i.e.* the finding of hits combinations associated to the same particle crossing the detector. Tracks are first searched in the VELO and in the UT, than if those tracks are compatible with the same trajectory, they are extrapolated to the SciFi;
- fitting of track candidates from the pattern recognition step using a Kalman Filter to estimate some parameters, as for example the momentum;
- reconstruction of primary and secondary vertices from the fitted tracks;
- trigger decision, based on track parameters such as impact parameter and momentum of the particle;

The majority of tracks are created near ($r \lesssim 1$ cm) the interaction point. These tracks typically have hits in the VELO, TT and forward tracking detectors and are reconstructed as *long tracks*. However, there are also neutral long-lived particles, with typical decay lengths ranging from a few to a few tens of centimeters, that are important to the physics program at LHCb. A significant fraction of such particles do not produce hits in the VELO, and therefore cannot be reconstructed as long tracks. The most common particles of this kind are s-quark final states, for example K_S^0 and Λ^0. If these particles decay after the VELO acceptance but before

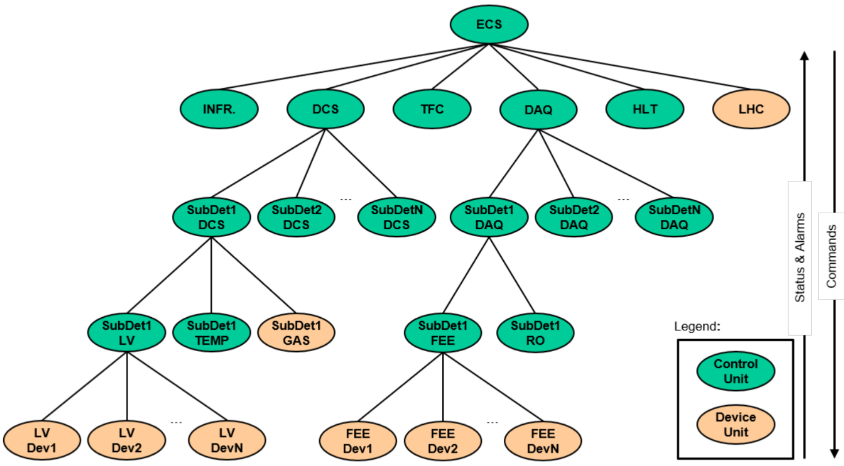

Figure 2.9: ECS schematic structure. The topmost node (the ECS itself) propagates commands to the downstream units (FE, DAQ systems, Trigger, ...) and is interfaced with the LHC system.

the TT, they can be reconstructed as *downstream tracks*. The baseline HLT1 algorithm doesn't include these tracks in the reconstruction sequence, so K_S^0 and Λ^0 particles are only available in HLT2 trigger selections. New technologies are arising to address this selection inefficiency (one of them, the *Downstream Tracker*, is discussed in Chapter 3).
Once tracks are selected by the HLT1, they are passed down to the HLT2 that performs a complete reconstruction of the event, using both the information from tracking detectors and Particle Identification ones (RICHs, calorimetes and muon chambers), and applies specialized cuts to select the physics of interest for LHCb.

Experimental Control System

The LHCb ECS system (Figure 2.9) is on the top of every subsystem componing the data-acquisition system. It incorporates [30] the control for the entire experiment, monitoring and all components from the infrastructure equipment to the data acquisition itself. It is realized as a distributed system based on a commercial SCADA system, WinCC OA, and other custom-developed components. WinCC OA [31] is a software framework used in many projects at CERN. It provides several features for the ECS such as user interfaces, data archiving, alarm-handling in a centrally managed alarm screen as well as interfaces for several different types of equipment using multiple protocols. Its open architecture allows for custom developments of the user interfaces, scripts and libraries, as well as managers and drivers for interfacing and controlling devices and readout electronics. The ECS can be modeled as a tree-like control hierarchy, based on an FSM (Finite State Machine) tree. Commands sent from the top of the hierarchy are propagated downstream to the leafs which carry on the actions, and the state of the end-nodes at the bottom of the tree is propagated up to the topmost node. At any point one can see the state of the entire LHCb experiment, as well as control it from a single place. One of the main parts of the ECS is the Run Control, which handles all the tasks that are needed for data taking. The Run Control is responsible for configuring the sub-detectors front-end electronics and readout boards, the TFC system, the sub-detectors High and Low Voltage (HV/LV) systems, and launching and configuring all the necessary DAQ processes.

2.3 The PCIe40 board

The PCIe40 board [28] is a custom-built PCI Express card designed by the LHCb collaboration with the purpose of interfacing the front-end electronics and the server memory in the EB nodes. It can have up to 48 bi-directional optical links, implementing the GBT (GigaBit Transceiver) protocol running at 4.6 Gbps and it can transmit up to 100 Gbps of data through the PCIe lanes. Its core component is a large FPGA (Arria-10 with about 1 MLE), that provides a high degree of flexibility in the use of this board, that can perform three different functions, according to the firmware loaded into the FPGA:

- **TELL40** : it acts as readout board for the LHCb Upgrade I detectors. Its function is to collect the data from the connected front-end electronics, processing them to form Multi-Event Packets (MEP). MEP data are send through the PCI Express links to the DAQ network for further processing on the online computing farm.
- **SOL40** : it is responsible for interfacing the sub-detectors front-end electronics via the GBT protocol to the TFC and the ECS. It allows the monitoring and control of the front-end electronics. It also interfaces the TFC to the front-end electronics by recovering the clock it receives from SODIN and distributing it to the electronics.
- **SODIN** (Super-ODIN): it is the readout supervisor for the experiment. It centrally synchronizes and manages the readout of events by distributing the LHC clock and generating clock-synchronous commands to dispatch the events (MEP) to the DAQ and trigger system. A SODIN board is associated with a sub-set of the readout boards (TELL40) and of the interface boards (SOL40); each board is operated as an independent entity.

The hierarchy of the PCIe40 board flavors and their connections is show in Figure 2.10

2.3.1 The Field Programmable Gate Array

The fundamental component of the PCIe40 board is the *Field Programmable Gate Array* (FPGA) chip mounted on it. The FPGAs are a type of field-programmable logic devices (PLD), integrated circuits featuring a matrix of interconnected logic gates. Unlike CPUs or GPUs, that

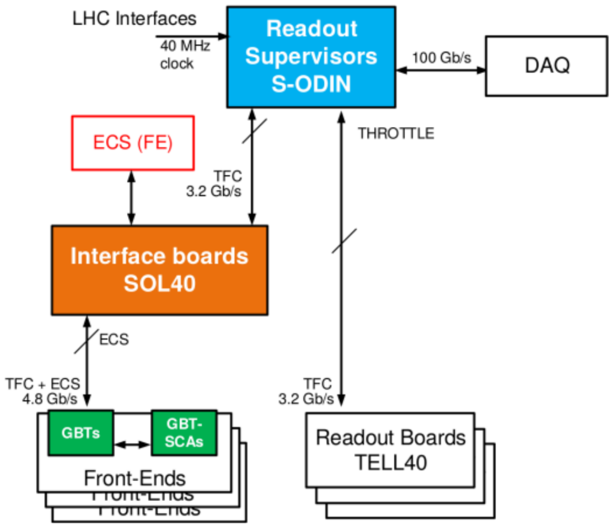

Figure 2.10: TFC and ECS system logical architecture, showing supervisor S-ODIN, SOL40 and TELL40 boards in their respective roles.

consist of well-defined and fixed architecture and execute instructions stored in the memory, the FPGAs are programmable devices whose connections between the internal logic blocks are user-configurable. Internal logic elements are connected according to the configuration setting (firmware) downloaded into a volatile SRAM. This allows to realize optimized electronics performing only the required operations to fulfill specific tasks. Although performances of FPGAs are somewhat worse in speed than ASICs (Application Specific Integrated Circuit) that employ the exact number of logic elements and connection needed for the task of interest, the possibility to freely reprogram the FPGA firmware makes them the best suited for R&D custom-electronics development. Furtherly, ASICs have the drawback that they are realized in large-scale productions, which is not the case of HEP experiment, so FPGAs are employed for the overall lower costs. The FPGA chips mounted on TELL40 boards reading out the VELO sensors are the (Intel®) ALTERA Arria-10 model 10AX115. The basic programmable logic block of the FPGA is called Adaptive Logic Module (ALM), shown in Figure 2.11. It consists of two 6-input LookUp Tables, two full adders, four registers made of D-type flip-flops and several multiplexers. When the design is compiled and the firmware is loaded into the memory, the ALM is configured so that the LUTs and the multiplexers perform the required calculations, allowing for the correct data-routing inside the logic block. The Arria-10 10AX115 FPGA features 427200 ALMs[1] , 2713 M20Ks (20 kbit memory blocks) and 20774 MLABs (640 bit memory blocks). The FPGA also implements hard Intellectual Property blocks (IP) *i.e.* pre-defined logic and memory block optimised to perform predefined tasks, such as dual-port RAMs or FIFO (First-in first-out data buffer) blocks. Every logic block of the FPGA can work in full parallel

[1]each Intel ALM corresponds to ~ 2.7 logic elements; so the Arria-10 FPGA features a total of 1.15M *equivalent* logic elements

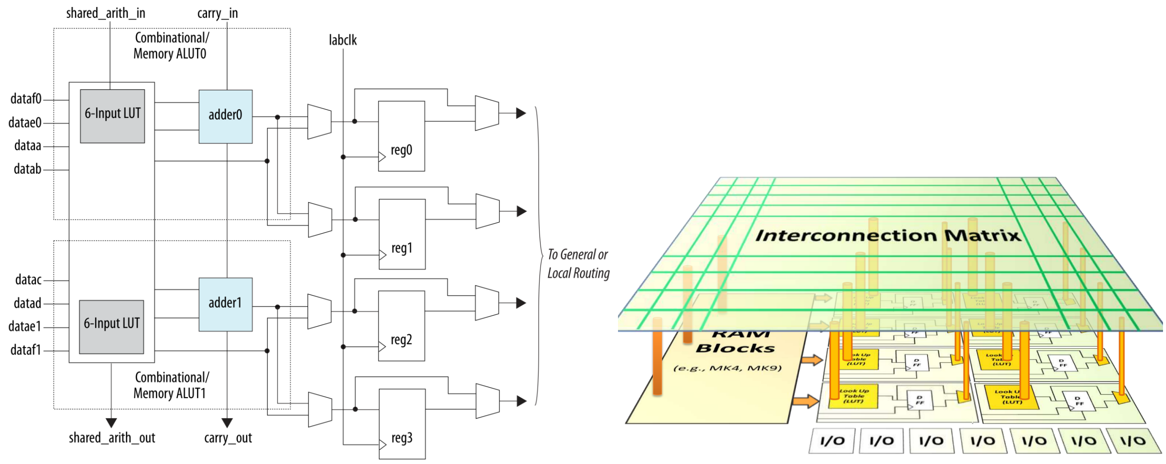

Figure 2.11: (Left) Basic programmable logic block (ALM) for Arria-10 FPGAs. (Right) Sketch of interconnection matrix between logic and memory blocks.

and indipendent way in respect to other elements. Thus, on the same chip multiple and/or different tasks can be performed at the same time, the only limitation residing in the number of available logic elements. This allows to design high-throughput and low-latency devices by exploiting the high degree of parallelism and the naturally pipeline-oriented structure offered by FPGAs. The design to be compiled into the actual firmware is described in a hardware description language (HDL) such as VHDL and Verilog.

In this thesis VHDL language has been used. VHDL is similar to software languages like C++, having its typical constructs like *if–then–else*. Nonetheless, being an hardware language, it is different from software where instructions are usually performed sequentially, while in VHDL instructions are executed exploiting full concurrency since different firmware entities work in a parallel and independent way. VHDL designs are simulated by means of specific tools [32]. During the simulation step the logical functionality of the design is verified, providing a powerful tool to eventually debug the firmware design before moving to the compilation. Compilation of FPGA designs is very different compared to the one of CPUs and GPUs programs. Compilation process converts the design file into a bitstream that can be downloaded into the device. It is composed of different tasks: 1) *synthesis*, where the HDL design file is transformed into a set of gate level macros; 2) *place* and *routing* processes, in which hardware logic elements are assigned to the design components, and the connection network between gates is optimized and built accounting for the actual FPGA resources; 3) *timing analysis*, determining if the firmware loaded into the FPGA can run at the design frequency.

2.4 Luminosity at LHCb

The LHCb experiment is aimed at studying heavy flavor physics to investigate effects of new physics in $\mathcal{CP}$ violation and in rare decays of b-quark and c-quark hadrons, relying on the capability to reconstruct very precisely vertex structures and the whole topology of final states decay chains. These tasks require high-precision tracking detectors (the VELO for instance) and a powerful TDAQ system.

The instantaneous luminosity directly delivered by the LHC is too high with respect to the target luminosity the LHCb experiment has been designed for, since of the too high event rate and irradiation dose to which the detectors would be subjected. Since of the higher detector hits multiplicity, track and vertex reconstructions would suffer of high event pile-up *i.e.* the number of *pp* collisions for each bunch crossing, that increases combinatorial background and complicates track reconstruction. To cope with these issues, the LHCb collaboration put in operation a system called *luminosity levelling* aiming at monitoring and stabilizing the instantaneous luminosity and pile-up at the interaction point of the experiment by adjusting the overlap between the two colliding beams. The instantaneous luminosity at LHCb is kept roughly constant and at a lower value compared to CMS and ATLAS experiments (see Figure 2.13). Generally, the instantaneous luminosity is not constant during an LHC fill, following an exponentially-suppressed trend due to beam intensity degradation effects. In LHCb it is kept fixed to its peak value both to maximize the collected integrated luminosity and to stabilize as much as possible the operational conditions of the DAQ and trigger systems. In fact systematic uncertainty for many physics analysis are sensitive to changes in detector and trigger operative conditions during data-taking periods, thus it is crucial to limit the operational conditions changes to reduce the systematics of such measurements.

2.4.1 Luminosity levelling and luminosity counters at LHCb

The luminosity levelling is performed by using corrector magnets at each side of the experiment, adjusting the transversal overlap between the two beams. The overlap is increased in small discrete steps as the beam intensity decreases, thus ramping the luminosity at the desired value by means of a luminosity control software (Figure 2.12) that provides information from the LHCb sub-detectors to the LHC Control Centre. The LHCb luminosity control manager - that is part of the ECS system (Section 2.2.6)- consists of a finite state machine which is driven by the LHC operational modes, monitoring the instantaneous luminosity and comparing it to the optimal target luminosity; once computed, the target luminosity and other levelling parameters are sent to the LHC Lumi Levelling driver application. On the LHC side, the application runs a “levelling algorithm” that determines when and by how much

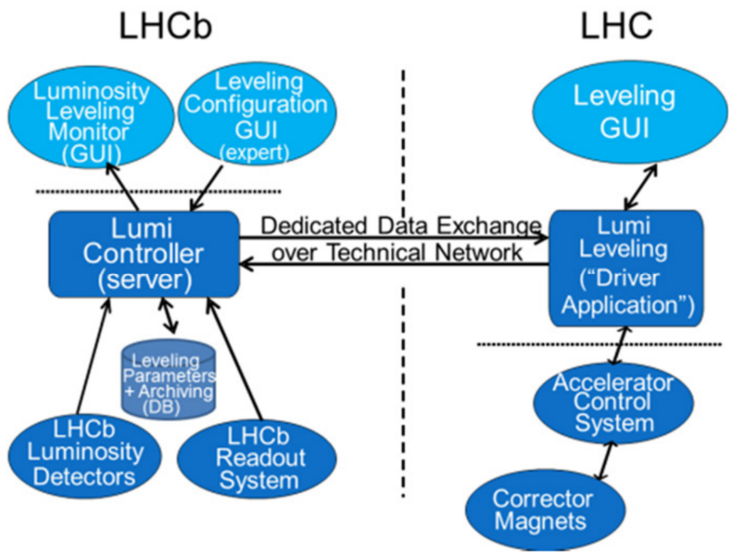

Figure 2.12: LHCb Luminosity control software diagram

the beam separation has to be modified to reach the target luminosity. The luminosity is levelled to a value that is a compromise between the physics priorities, the trigger selection efficiencies related to the reconstruction tasks, and technical limitations related to detector performance, the DAQ system and the offline data-processing. For Run-3 this optimal value is estimated to be $L = 2 \cdot 10^{33} \text{cm}^{-2}\text{s}^{-1}$ *i.e.* pile-up per bunch-crossing $\nu = 7.6$. The collaboration estimated [33] that precision of $\lesssim 5\%$ on online instantaneous luminosity would have been enough for monitoring purposes, including luminosity levelling. The online determination of the instantaneous luminosity is based on counting triggers - the so-called *luminosity counters* - satisfying very simple requirements.

During Run-1 and Run-2, online luminosity estimates were based on the occurrences of transverse energy depositions over a fixed threshold in the calorimeter system and on counts in the muon stations, both provided by the L0 hardware-trigger. These counters provided online luminosity measurement with 10% precision. An example of luminosity levelling performed using these counters during Run-2 is shown in Figure 2.13.

For Run-3, where L0 trigger has been removed and more precision is required, a dedicated *luminometer* sub-detector named PLUME (Probe for LUminosity MEasurement [34]) was built to fulfill the tasks above, aimed at providing both online and offline luminosity measurement. The PLUME detector is composed of 24 hodoscopes (*i.e.* couples of detector modules read out looking for coincidences in both modules) surrounding the beampipe and pointing at the nominal interaction point, located $\sim 1.7\,\text{m}$ upstream from the collision region and covering high-pseudorapidity range ($2.4 < \eta < 3.1$). Modules are composed of $10 \times 10 \times 5\,\text{mm}^3$ quartz crystal, coupled to photomultipliers detecting Cherenkov light generated inside the quartz material when high-speed particle (as happens at high pseudorapidity values) transverse it.

Online luminosity measurement is performed by counting the occurrences of coincidences on at least one hodoscope, identifying tracks coming from the interaction region and suppressing background activity.

Additionally to the PLUME detector, the collaboration decided to define more luminosity counters for both online monitoring and offline luminosity measurement. Having independent and several counters is indeed crucial for stability cross-checks and systematic uncertainty evaluation. Counters from each sub-detector have been proposed; they can be divided in two groups:

ECS counters Simple counters based on minimum bias conditions and operating independently of the high-level reconstruction sequence, operating at $\mathcal{O}(30\,\mathrm{MHz})$) and read out via ECS interface. These counters are mainly used for online monitoring purposes. The proposed counters are each based on:

- number of VELO hits;
- number of muon hits in muon stations (*luMUONmeter* [35]);
- Beam Control Monitor system [36]. It is a dedicated sub-detector consisting of two modules equipped with polycrystalline CVD diamond sensors, located one upstream and one downstream the interaction region .

HLT1 counters Counters based on the number of reconstructed tracks/primary vertices, to be performed within the HLT1 sequence in dedicated trigger lines, running at a reduced rate of $\sim$ 30 kHz, and aimed at providing high-precision offline luminosity measurement. The following counters are currently planned:

- VELO tracks and reconstructed primary vertices;
- SciFi tracks;
- ECal and HCal energy deposition;
- RICH1 and RICH2 hits;

During this thesis work, I developed a firmware to accumulate in real-time the number of VELO hits reconstructed at the readout level. The values of these counters are read out by the ECS system, and then processed to reconstruct the instantaneous luminosity at the LHCb interaction point.

The possibility to handle well-reconstructed hits on the VELO sensors at readout level is given by a 2D-cluster finder firmware implemented in the FPGA of readout of the detector. The VELO clustering firmware has been developed within the RETINA project, whose primarily aim is to perform track reconstruct in real-time at the readout level by exploiting the

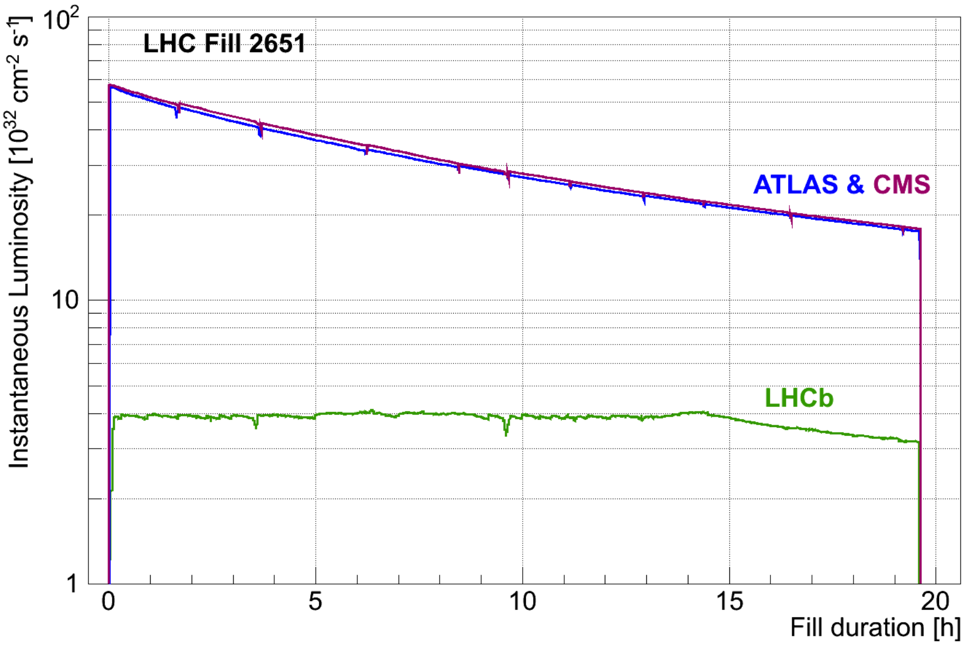

Figure 2.13: Example of instantaneous luminosity trends for CMS, ATLAS and LHCb experiments during stable beams in Run-2; the LHCb luminosity is kept roughly constant for ~ 14 h by means of the luminosity levelling system, then bunches collide head-on and luminosity decreases since of beam degradation effects. A factor ~ 5 between LHCb luminosity and CMS and ATLAS ones is due to the fact that colliding beams in LHCb are de-focused too.

computing power of the FPGAs. The next chapter describes the key features of the RETINA project and how the 2D-clustering firmware is implemented and integrated in the TELL40 boards of the VELO.

3

Leveraging the power of FPGAs: the RETINA project

Chapter Contents

As explained in the previous chapter, at LHCb the reconstruction of events with s-quark long-lived final states suffer in efficiency, because the computational load and complexity of the tracking sequence performed by HLT1 prevents the inclusion of downstream tracks in the baseline tracking algorithm. The collaboration is thus looking for new solutions, allowing to perform part of the tracking reconstruction more efficiently (*hardware accelerators*), by means of heterogeneous computing systems. In this context the RETINA project has established.

The aim of the project is to enhance the overall processing power of the experiment by implementing a real-time tracking device based on FPGAs. An important target application of the RETINA project is the "*Downstream Tracker*", a device aimed at reconstructing downstream tracks in advance of the HLT1 process. This requires a reconstruction that can be executed before the Event Builder itself, implying low latencies and high throughput.

The Downstream Tracker will receive hits from the SciFi detector in parallel to the main stream of data delivered to HLT1, reconstructing downstream track using the "Artificial Retina", a pattern-matching algorithm. This algorithm is inspired to the early stages of mammalian's brain mechanism involved in vision processing, and it was first proposed for HEP experiments in year 2000. The retina algorithm is designed to recognize specific features in experimental hits patterns by comparing them to stored reference patterns. One of the key elements of the artificial retina is that, differently from other parallel pattern-matching algorithms like the Associative Memories, is the ability to provide a non-binary (yes/no) response about the comparison with the stored patterns. but it provides a response that continuously varies depending on the distance of the track from the stored patterns. The working principle of the artificial retina can be summarized as follows:

1. track parameter space is divided in a grid of cells, each one corresponding to a *pattern-track* stored in memory: each pattern-track intersects the layers of the detector in points called *receptors*;

2. for each event, the algorithm computes the *activation level* of each of those cells, by means of a gaussian weighing function of the euclidean distance of each hit from the receptors;

3. the "Artificial Retina" identifies tracks as local maxima in the cells space, via a local cluster-finding algorithm, then it interpolates the activation level of the neighboring cells, allowing to reconstruct tracks at the native resolution;

More information about the artificial retina can be found in references [37, 38]. The retina architecture is intended to rely on a large use of bandwidth and high degree of parallelization. The *Field Programmable Gate Arrays* (FPGAs) are the best suited devices to implement such an architecture, thanks to its large internal bandwidth and high degree of parallelism. While mathematically the Artificial Retina has some similarity with other pattern-matching algorithms such as the Hough transform [39], its distinctive element is the implementation architecture.

The Artificial Retina architecture has two main components: the Engines and the Distribution Network. The former are implemented to compute the activation level for each cell of the track parameters space, as described above. Multiple engines are implemented, each computing the activation level of only one cell. In order to cope with FPGA resources limitation without increasing latency or reducing the throughput, engines are distributed over multiple chips working in a fully parallel way on the same event. Hits from the same event need to be delivered simultaneously to each FPGA device where engines are implemented. An intelligent delivery system named Switch is the basic component of the Distribution Network. It features a routing LUT containing information to deliver to each engine only those hits that are close to its receptors. This allows to achieve high throughput since each engine process a reduced number of hits. The Distribution Network and Engine can be implemented within the same FPGA chips. Modern FPGAs are equipped with high-bandwidth transceivers (XCVRs) where high-speed optical serial links between several boards can be implemented, allowing for the implementation of a Distribution Network. The boards hosting the FPGAs, where Engines and the Distribution Network modules are implemented, ("Tracking Boards") can be paired 1:1 with the readout boards of the detector, to receive and process the data before it is sent to the Event Builder.

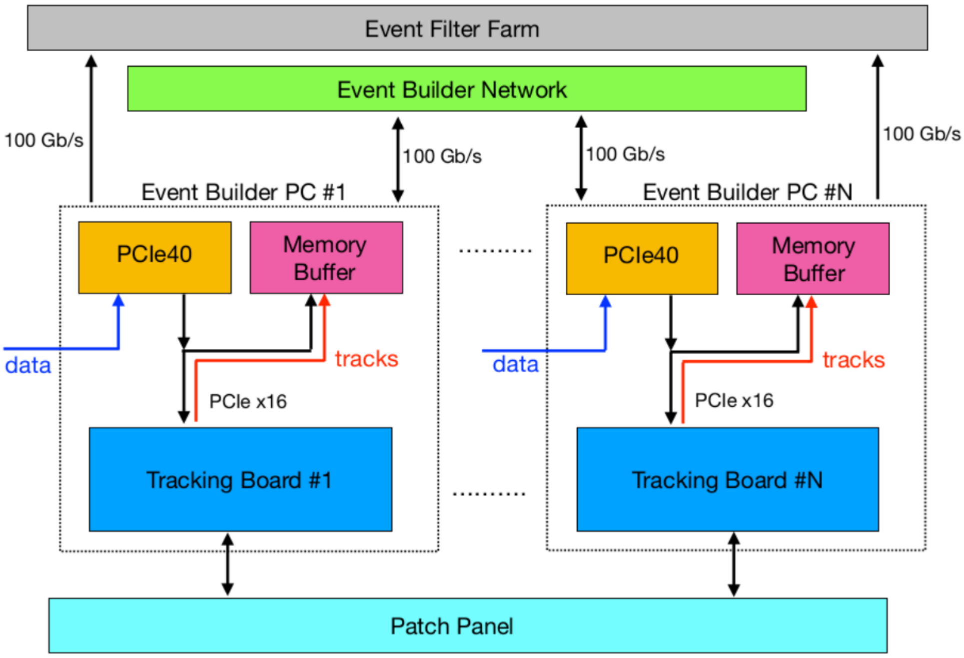

Figure 3.1: Artificial retina integration within the EB and DAQ systems

The Artificial Retina tracker discussed above works on *unbuilt* data, since hits data are directly sent from the tracking sub-detectors to the Tracking Boards. This approach provides a few advantages: only the number of boards needed to process data coming from the detector of interest are required, while already-built data are distributed on a large computing farm (see 2.2.6), so processing of built data necessarily requires a large number of cards; processing time is required to unpack data after the event-building, while no processing is needed for this task for unbuilt data. However, the integration of such a device within the LHCb DAQ system requires dedicated solutions. The best way to integrate the Artificial Retina is to install the Tracking Boards in unused slots inside the EB server nodes. Figure 3.1 shows the expected data flux inside the EB nodes including the Tracking Boards. Data from detectors are sent to the EB nodes via PCIe40 boards that send data to engines and switching modules implemented in the Tracking Boards. The Tracking Board receive a duplicate of data from the detector of interest, while the other copy of the that is sent to a memory buffer. Tracking Boards on different EB nodes are connected through an optical Patch Panel that is the backbone of the Distribution Network. When track reconstruction by the Tracking Boards is done, track data is appended to the memory buffer, allowing the event building to include pre-reconstructed data rather than just raw data.

The RETINA project has concluded the R&D phase, producing a prototype system reaching 40 MHz event throughput with sub-μs latencies when applied to a generic 6-layer tracking system. A realistic prototype is currently under implementation for the reconstruction of tracks within the VELO acceptance under Run-3 conditions. The VELO detector is read out by 52 boards, making it a relatively compact sub-detector. Given the limited number of boards required, the VELO was preferred over the SciFi - read out by ~ 140 boards - for implementing a demonstrator device. Data from the VELO detector are sent in a parasitic stream to a facility called *Coprocessor Testbed*, where innovative prototypes are being tested under realistic DAQ conditions.

3.1 VELO clustering firmware

The raw VELO-pixel data need to be processed in order to find the particle hits that are required in input to the artificial retina to perform its tracking task. While other detectors, like the SciFi, were designed with a clustering firmware embedded in their readout cards, the 2D clustering needed for the silicon pixels of the VELO sub-detector, was initially planned to be performed by the HLT1, due to its much greater complexity.

However, to make RETINA track reconstruction possible, it is imperative for this process be carried out before it, that is before HLT1 even starts. This was an important motivation for starting a project for implementing it in FPGA, in spite of its difficulty.

There is however another good motivation, based on more general grounds: the computational demands of the VELO clustering is so large that it takes away a significant fraction of HLT1 resources, detracting from the execution of higher level reconstruction tasks that are critical for the trigger functionality.

Therefore, an effort was made within the RETINA project to replace the clustering process in the HLT1 with a piece of firware running in the readout boards of the VELO detector. A new real-time 2D-cluster finding firmware was designed and implemented on VELO's back-end FPGA-based cards. The operational requirement of the detector readout make it necessary for this firmware to be capable of a throughput in excess of 30 MHz. This project, the first of its kind, has been successful completed and is now in its final commissioning steps. It will be the default system for LHCb data acquisition during Run-3.

The novelty of this realization offers an opportunity for further new developments. The aim of this thesis is to build upon the availability of these data in real time to provide further useful functionalities to the experiment. In addition to the intrinsic value of these additions, my work is motivated by gaining a better understanding of the future potential of a fully embedded event reconstruction that is the long-term goal of the RETINA project. This will be a great help in motivating and shaping the project itself.

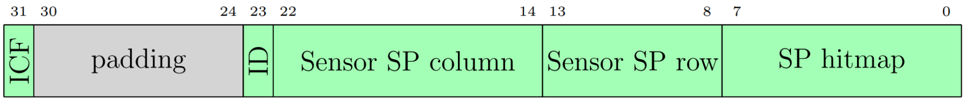

Figure 3.2: SP 32-bit encoding word

3.1.1 2D Clustering algorithm

Data from the VELO are formatted as 4×2 pixel blocks, named SuperPixels (SPs). The SP data format is a 32-bit word, displayed in Figure 3.2. 1 bit refers to the Sensor Tile identifier on the sensor pair, 9 bits and 6 bits refer to the SP column ID and row ID respectively, and 8 bits constitute the SuperPixel hitmap *i.e.* the encoding map of active and non-active pixels within the SP. SPs are transmitted in 256-bits packets called *frames*, containing 8 SPs each.

Along with the SP data packets, other SuperPixels-related data comes out from the VELO pre-processing firmware implemented on the FPGA Arria-10 chip mounted on TELL40 boards. *Start Of Package* (`SOP`) and *End Of Package* (`EOP`) are 1-bit wide signals identifying frames coming from different bunch-crossings; the `SOP` signal is sent together with the first word of each event, while the `EOP` signal is received with the last input word. If the event is empty, or if the event consists of only one frame, the `SOP` and `EOP` signals are generated simultaneously. The `Data_Valid` signal is sent together with the SPs data.

SP words in input to the clustering block are flagged as *isolate* or *non-isolate* with an *Isolation Cluster Flagging* (`ICF`) bit : a SP is flagged as isolated if none of its eight neighboring SPs has active pixels. This information is relevant in optimizing the clustering performances, since the cluster reconstruction for isolated SPs can be realized very easily. Isolated SPs are resolved into clusters using a look-up table (LUT), that includes each of the $2^8 = 216$ possible pixel configuration within a SP. Reconstruction of isolated SPs in such a way allows extremely-fast cluster finding with a minimal usage of logic resources, relevant for high-throughput performances, being the isolated SPs about 50% of the total amount of non-empty SPs.

Clustering for non-isolated SPs requires a more complex algorithm, involving the parallel processing of the ensemble of neighbouring SPs. A matrix-filling mechanism, showed in Figure 3.3 is performed to put together 9 (3 rows $\times$ 3 columns) adjacent SPs in a fully parallel way. All non-isolated SPs fill those matrices, that are created at the beginning as empty entities: the first incoming SP fills the center of a blank matrix, determining the coordinates of the SPs that can fill the neighbouring cells of the matrix. Subsequent SP fills this matrix if those coordinates are matched, otherwise it moves forward looking for the matrix it belongs to until no more non-empty matrix are defined, finally filling the center of a blank one. When an `EOP` signal is received, signaling the end of an event, every pixel in each matrix checks in a fully parallel way if it belongs to specific pattern of pixels. L-shaped sequences of inactive pixels are used to fulfill this task. When one of these patterns is matched, the cluster candidate is

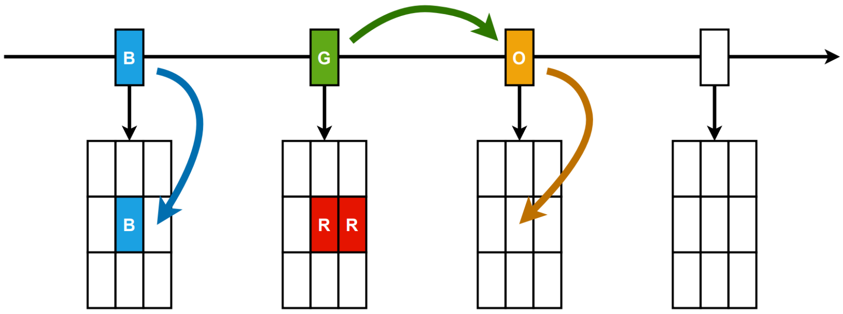

Figure 3.3: Matrix-filling mechanism for non-isolated SPs

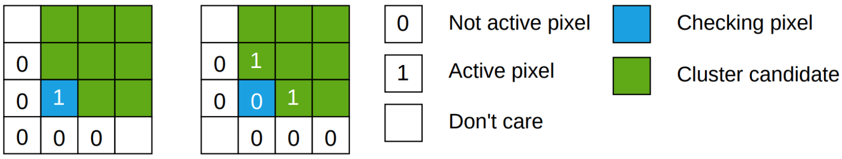

Figure 3.4: Examples of patterns that generate a cluster by the pattern-matching algorithm for non-isolated SPs

recognized in a 3×3 grid, as shown in Figure 3.4. The coordinates of the reconstructed cluster center-of-mass position (centroid) are determined using a LUT of $2^9 = 512$ entries, and the absolute position is computed, based on the position of the matrix inside the VELO detector and the position of the cluster candidate within the matrix.

A number of constraints, coming from the amount of available FPGA resources and intrinsic hardware limitations forced the adoption of same approximations, that might in principle affect the reconstruction performance. They are mainly related to the pattern-matching mechanism, the cluster candidate size (3×3 pixels) and the matrix filling scheme. These features can lead to a partial cluster reconstruction, or to the splitting of large ensembles of active pixels, thus worsening the reconstruction quality, and potentially lead to inefficiencies. Extensive studies on physics performances, carried out using detailed simulations, have been conducted to investigate these effects. The studies showed that the inefficiencies introduced by the FPGA-based implementation are completely negligible with respect to the required precisions. Example of one of such studies is shown in Figure 3.5, showing that tracking efficiency is negligibly affected by the FPGA-based clustering.

3.1.2 Firmware implementation

The basic components of the clustering firmware and their internal connection are displayed in Figure 3.6. A decoding stage splits input SPs into 8 separate streams, encoding SOP and EOP signal into one unique flag that identifies the end of each event. Then SPs are sent in parallel to two 4-to-4 switch blocks under the control of finite-state machines. Switches assure that

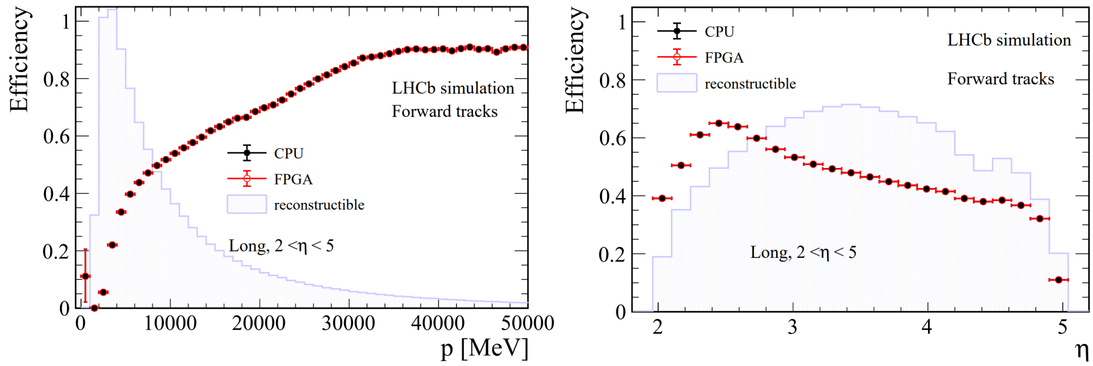

Figure 3.5: Efficiency for reconstruction of all reconstructible long tracks using FPGA clusters and CPU ones, as a function of the momentum (left) and pseudorapidity (right) of the particle.

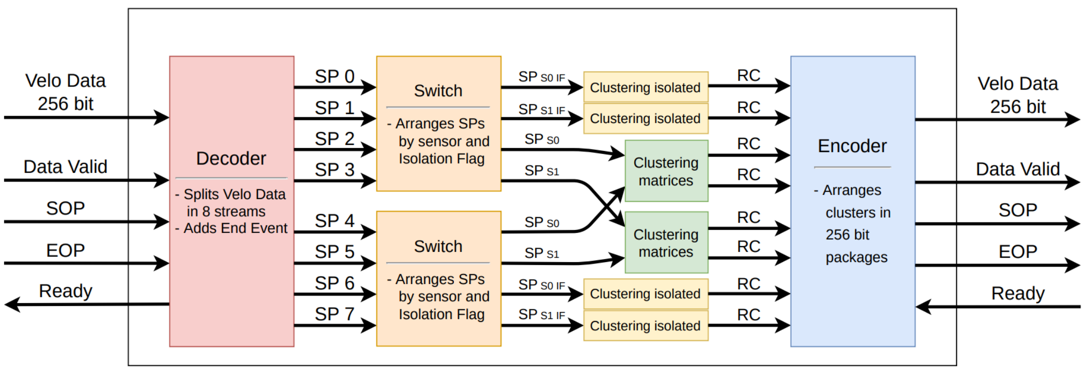

Figure 3.6: Cluster-finding firmware basic structure. RC stands for *RetinaCluster*

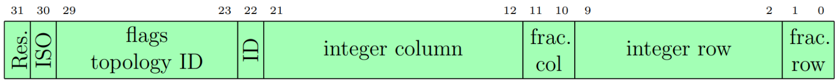

Figure 3.7: 32-bit word encoding of the Retina Cluster

every input data can go to any of the four output streams, allowing to send both isolated and non-isolated SPs to the respective clustering stage. Clustering for isolated SPs is performed by means of a LUT, being the most effective solution in terms of throughput and resource usage.

Non-isolated cluster are resolved by a chain of 20 matrices, performing the matrix-filling mechanism and pattern-matching algorithm presented in Section 3.1.1. Each matrix receives data from two independent input lines, increasing the data processing rate with respect to a single line. If, due to the matrix-filling process, all the 20 matrices are already busy, the input SP is resolved by the LUT implemented for the isolated SPs. Lastly, an encoding stage reconstructs the cluster word in the appropriate output format, shown in Figure 3.7, and the other cluster-related signals.

Clusters are transmitted in 256-bit wide words, each holding 8 clusters. Each cluster is encoded in a 32-bit word. Of those, 22 bits are reserved for the position of the cluster centroid: 8 bits for the integer part of the row and 2 for its fractional part, while 10 bits are reserved for the integer part of the column and 2 more for its fraction. Each cluster centroid position is thus encoded with a resolution corresponding to 1/4 of the size of a pixel.

One bit encodes the sensor ID, and another one identifies whether the cluster is reconstructed from an isolated or non-isolated SP. Further 8 bits are used to encode the topology of the cluster and additional reconstruction quality flags. Cluster topology encodes, together with the row and column fractional parts, the full shape of the cluster. The quality flags are useful to identify partially reconstructed clusters or clusters split between different matrices. They encode information about the typology of the isolated cluster *i.e.* if actually isolated or coming from non-isolated SPs overflowing the maximum number of instantiated matrices, thus treated as isolated. They carry also information about the position within the matrices ensemble where non-isolated-SPs clusters reside *i.e.* self-contained inside the matrices ensemble and/or on its edge.

A back-pressure mechanism is implemented throughout the whole firmware. Each processing block sends a ready signal to the preceding unit when it is capable of receiving data.

The clustering firmware runs on an external clocks, operated at 350 MHz. This clock drives the components used for the actual cluster reconstruction. This processing rate has been optimized to ensure clustering throughput greater then the nominal bunch-crossing rate of 30 MHz.

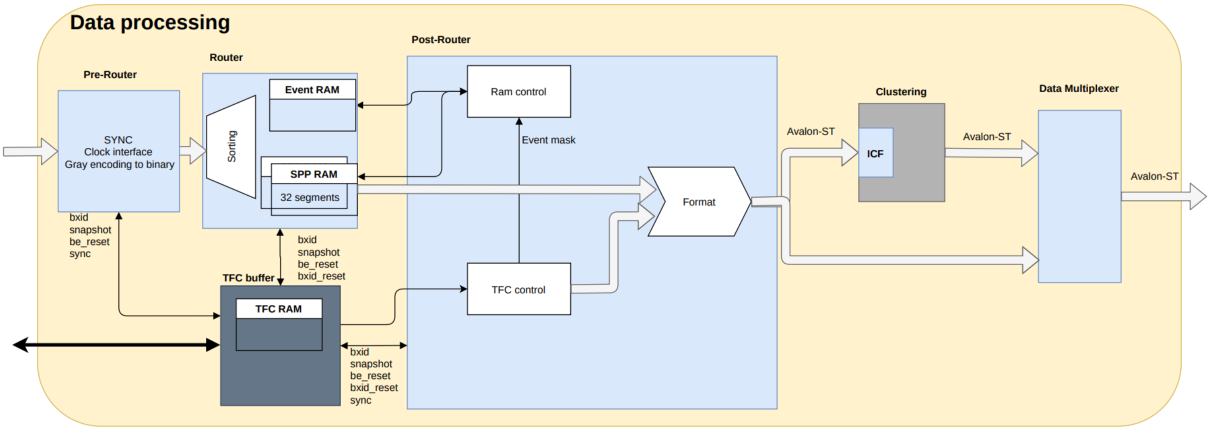

Figure 3.8: Data processing within VELO firmware implemented on TELL40 readout boards. At the end of the processing architecture a multiplexer manage the output data, sending only RetinaClusters or both RCs and SPs. This feature is mainly required for commissioning purposes.

3.1.3 Integration within the VELO firmware

The cluster-finder firmware has been integrated inside the TELL40 readout boards of each of the 52 VELO modules, where other pre-processing components are implemented. Each TELL40 board receives data from 12 VeloPix ASICs *i.e.* two sensor tiles, processing them in two parallel data flows, named DATA_FLOW_0 and DATA_FLOW_1. The structure of the data processing firmware for each parallel data stream is displayed in Figure 3.8. The other elements making up the firmware (Pre-Router, Router, Post-Router, TFC buffer) are essentially required to time-rearrange SPs[1] and to include TFC time and bunch-crossing synchronization.

The clustering firmware - developed and tested on a Stratix-V FPGA chip during the very early stage of the project - needs to be modified to be interfaced with the Arria-10 FPGA and to meet the specific requirements of the VELO detector. A modification is introduced to the clustering algorithm to account for the reversed local coordinates of sensors mounted on the upstream side of the substrate (see Section 2.2.1). In fact, using the L-shaped pattern with checking pixel at the bottom-left corner of the 3×3 matrix for these sensors, the clustering algorithm would only partially reconstruct the (small) fraction of large clusters released by particles passing nearly parallel to the sensors [40]. The best way to deal with this effect is to ”reverse” the L-shaped pattern and to place the checking pixel at the bottom-right corner of the pattern-matching matrix. This improves the robustness of the algorithm in presence of large clusters.

Other major modifications are made to the input and output interfaces. The Clustering block receives in input and sends to the output a set of signals, shown in Figure 3.9:

[1] SPs coming from the detector front-end are time-shuffled; any data processing require a processing step where SPs coming from the same bunch-crossing are grouped together.

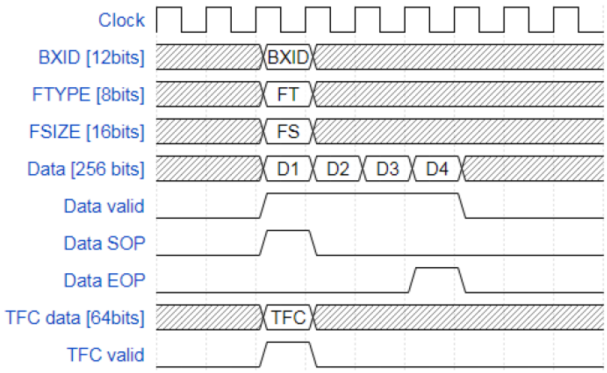

Figure 3.9: Both input and output data and metadata of the clustering firmware. The clock operates at 250 MHz rate.

- `BXID`: 12 bit wide signal, carrying the bunch crossing number corresponding to data transmitted simultaneously to it;
- `FTYPE`: eight bit wide, it encodes the transmitted data type, distinguishing between SuperPixels and RetinaClusters;
- `FSIZE`: 16 bit wide, it counts the number of SPs(input) or RCs(output) sent within a specific `BXID`.
- `DATA`, `DATA VALID`, `SOP`, `EOP` : the same meaning as explained in previous section;
- `TFC` data: 64 bit wide signal, it carries commands and the `BXID` of the event to which the commands have to be applied, along to information relative to the LHC synchronization. It carries a 2-bit signal encoding the type of collision (`BXTYPE`) corresponding to the bunch crossing transmitted in parallel to it *i.e.* if the colliding beams are empty or not. Four types of bunch crossing exist: `BB, BE, EB, EE`, where `B` is for ”bunch” and `E` is for ”empty”.

At the output of the clustering block, the data - previously processed with the 350 MHz clock - cross to the 250 MHz domain, being driven by a 250 MHz clock. This is the processing clock of the PCIe40 board, to which every component must aligned at the output and input interfaces.

Clusters produced by this firmware are eventually sent to the host server, that sends them to the EB. However, the structure and organization of the above described firmware allows performing some additional opportunistic processing over the entire set of cluster data, by tapping into the stream of cluster data flowing through the device. This offers a great opportunity to implement *ad-hoc* components (albeit within tight resource limitations) without disturbing or slowing down the main output stream.

3.2 This thesis: real-time monitoring using FPGAs

The availability of all reconstructed hits on a complex detector - such as the VELO - at readout level leads to the opportunity to further study and to explore the potential of real-time reconstruction and analysis using FPGAs. VELO hits - reconstructed by the clustering firmware - can be used in a parallel stream in respect to the main data stream without affecting the performances of the data processing firmware. The internal pipelined structure of the data stream allows to perform "on the fly" further processing of the data, without introducing any buffering stages while running at full speed *i.e.* at the bunch-crossing rate of 40 MHz, and using a minimal amount of both logical and memory resources.

In this thesis work I developed a piece of firmware to further process data coming from the data processing block, with the aim to perform :

- real-time monitoring of the instantaneous luminosity at the interaction point of the LHCb experiment. This task is related to the online luminosity monitoring system to drive the luminosity control system as discussed in Section 2.4.1.

- monitoring of the luminous region nominal position. Knowledge of the beamline parameters such as its nominal spatial position is relevant to correct the luminosity estimate, thus defining a luminosity estimator as robust as possible against spatial displacements of the luminous region. However, there is an other relevant point related to the real-time monitoring of the luminous region position. This information may be potentially used as a *seed* for the track and primary vertices finding tasks performed by the HLT1 system, speeding up its reconstruction sequences and thus increasing the throughput of the experiment.

- real-time monitoring of the VELO sensors "health" status, in particular studying spillover effects[2] and its detection inefficiencies. These effects are significant to account for in the measurement of the instantaneous luminosity, since they produce respectively an excess or a decrease in the number of hits on the sensors, affecting the luminosity and beamline parameters estimates.
  These tasks are particularly relevant for operating the experiment too: sensor detection efficiency is deeply studied during characterization phase and dedicated data-taking periods, but there is no actual knowledge about it during physics data-taking. The same is valid for the spillover effects, since no study in realistic data-taking conditions has ever been conducted. The direct access to reconstructed hits at readout level offers the

[2]when a particle crosses the sensor, an analog signal proportional to the deposited charge is generated by the ASIC. If the signal is over threshold at the subsequent bunch crossing, the pixel is flagged as active even if no particle crossed it in that event. This effect is the so-called *spillover*.

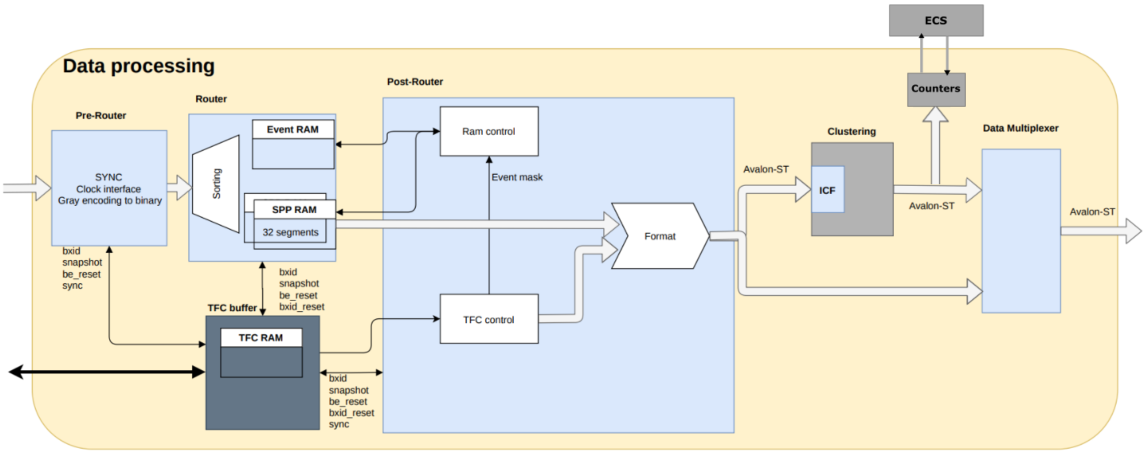

Figure 3.10: New version of Data Processing firmware including monitoring counters, right downstream of the clustering block.

opportunity to easily account for these effects. leading to the realization a constantly-operative online monitoring system for the detector, useful both to monitor the ageing effects and to detect any unexpected behaviors of the sensors.

The firmware I implemented is completely written in VHDL language to optimize its performances in order to process cluster data at the nominal average rate of 30 MHz. It receives data from the clustering block and processes them independently of the remaining firmware components. This component is then interfaced to the ECS to expose the accumulated statistics to the online monitoring system. The resulting new version of Data Processing firmware is shown in figure 3.10.

The structure of the remaining part of this thesis is as follows:

- in Chapter 4 the firmware implementation of luminosity counters is presented.
- in Chapter 5 reconstruction algorithms to compute the instantaneous luminosity and the beamline position are discussed. These studies rely on the analysis of LHCb official simulations .
- in Chapter 6 firmware prototypes implemented for detector monitoring are described.
- in Chapter 7 the integration of the luminosity counters within the LHCb online monitoring system is described, together with the commissioning steps required to validate them .

4 Implementation of real-time counters

Chapter Contents

The basic idea that drives my thesis is to accumulate count statistics of hits on the VELO detector to realize online monitoring of luminosity and of sensor-related parameters. In my work, I have tried to explore what advantages could be obtained by the newly achieved real-time reconstruction of VELO clusters at the readout level. We will see in the following, that even the simple counting of clusters, when performed in real-time at the earliest DAQ level, can provide surprisingly useful benefits.

In this chapter the FPGA implementation of luminosity counters of VELO clusters is described, while the optimization studies realized using detailed simulations and luminosity reconstruction algorithm are reported in Chapter 5.

4.1 Luminosity counters implementation

The structure of the luminosity counters firmware I implemented is shown in Figure 4.1. This design entity, called *Luminosity register* runs on an external clock, that is operated at 250 MHz. This is the same clock driving the output of the clustering block, ensuring that the processing of cluster data inside my piece of firmware is aligned with the data itself. Required input data are delivered by the clustering firmware using the parasitic stream as shown in Figure 3.10. The availability of well-reconstructed clusters data offers some advantages with respect to the usage of raw pixel data. Indeed the possibility to select reconstructed hits on the VELO sensor, having access to quality flags too, opens the opportunity to better control background contribution and to perform more sophisticated selections, while having only raw pixel data it wouldn't be possible to perform such tasks.

I designed this unit to make it possible to select clusters depending on their position within the sensor and their topology; selected clusters are then used to update a number of accumulator registers.

An ECS-driven mechanism is realized to send only the required counter content to the output. The output is then interfaced to the LHCb online monitoring system, centrally managing the read requests to be sent to the luminosity register firmware. A monitoring script is implemented in the VELO WinCC project (being part of the ECS system), where counters from each VELO module are combined with the reconstruction algorithm I developed, discussed in Section 5.6. Finally the luminosity estimate computed starting from the clusters counters I implemented is shown in monitoring panels in the LHCb control room. The integration of the luminosity register within the LHCb online system is discussed in details in Chapter 7.

Several counters are defined:

- Counters of total selected clusters and of occurrencies of the number of selected clusters for each event, divided according to the type of bunch-crossing, named *average counters*. These counters allow to monitor the instantaneous luminosity averaged over a time window, and to distinguish the contributions of background interaction with the beams (or with the background itself) from the beam-beam interactions (see Eq. 5.23).

- A counter register is defined for each bunch-crossing ID occurring at the LHCb interaction point. The number of different bunch crossings for each revolution of the LHC is 3564, thus so many counters are defined. They are stored in a 3564 addresses RAM (*RAM counters*). Having cluster counters divided according to the ID of each event is relevant to compute the instantaneous luminosity of each bunch crossing.

The firmware I implemented is composed of different entities that work in pipeline, allowing thus to process input data at the full rate of 30 MHz without introducing any buffering (e.g. FIFOs). Pipeline-delay stages are instead required to avoid losing synchronization between different parallel signals. In fact, the processing stages I implemented to perform cluster selections and the various accumulating steps require some clock cycles each to complete their tasks, the exact number depending on the specific component. Since different signals follow different processing paths before being sent together in input to any processing component, signals need to be properly delayed to guarantee time alignment among them.

In the following discussion these delays are omitted to ease the description of the firmware structure.

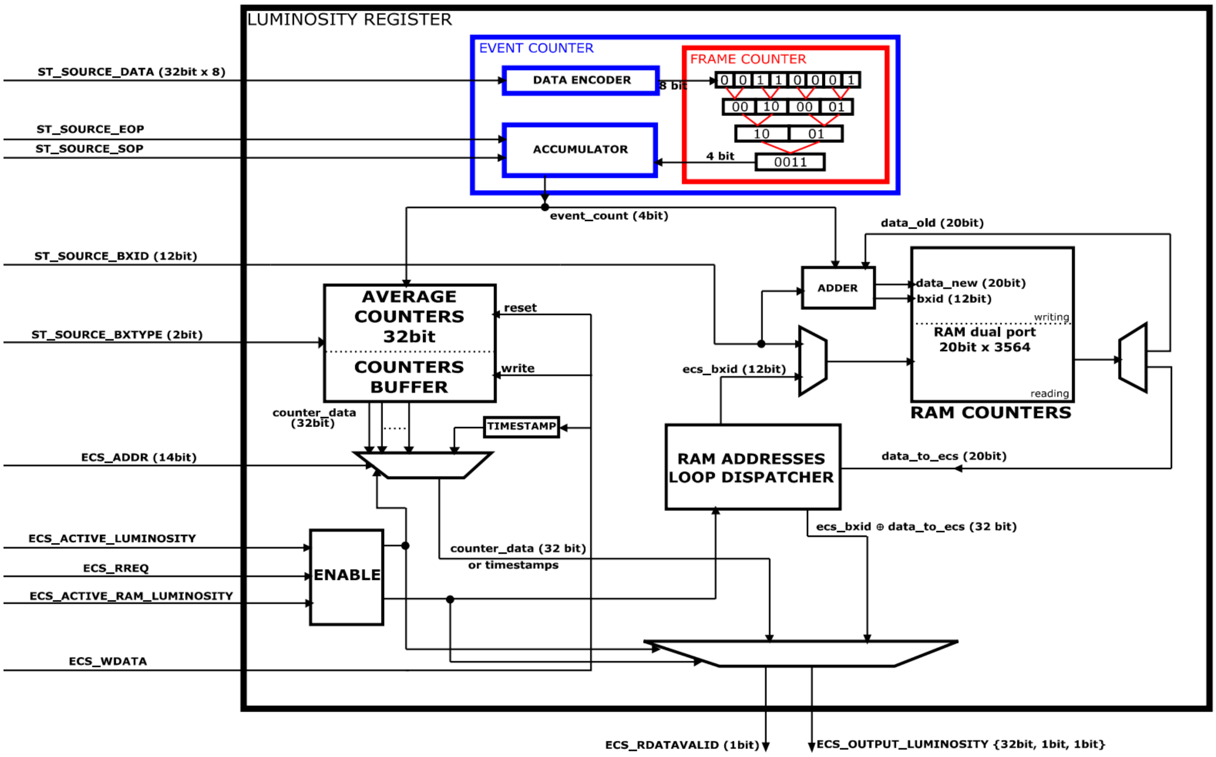

Figure 4.1: Luminosity register firmware structure. Only basic components are shown; pipeline delay of each signal, EVT_READY and RAM_READY signals are hidden.

4.1.1 Event and frame counter

An entity called *Event Counter* (blue component in Figure 4.1) performs cluster selection and accumulates the number of selected clusters for each event. Clusters data are encoded in 256-bit packages (ST_SOURCE_DATA in Figure 4.1); they are split in parallel into 8 data streams of 32-bit words each encoding different RetinaClusters. Cluster selection is performed in parallel for each data stream by means of 8 comparators (DATA ENCODER component), establishing whether valid clusters reside within specific regions of the sensor and if specific conditions on their topology (*i.e.* the number of pixels belonging to the cluster) are met. A flag is raised for each data stream, making up an 8-bit wide signal with '1' in correspondence of selected clusters and '0' otherwise.

The 8-bit signal is sent to the FRAME COUNTER (red component in Figure 4.1) that counts in three pipelined steps (one clock cycle each) the number of '1's present in the input signal *i.e.* the number of clusters that are selected by the Data Encoder for each data frame. Three is the minimum number of steps required to add up 8 one-bit wide signals using 2-input adders as basic entities. A 4-bit cluster count is then sent back to the event counter. Depending on SOP and EOP signal, it accumulates through a 4-bit adder frame counts belonging to the same event, allowing to sum up to $2^4 - 1 = 15$ clusters for bunch crossing. If the number of clusters is larger the value of the accumulator is set to its maximum; this is a rather rare case, since the average occupancy of the (small) regions of our interest (see Figure 5.6 left) is below $\sim 1.2/4 = 0.3$ clusters/evt up to the highest value of luminosity that will be reached, even at the end of Run-4.

When a `SOP` signal is received, the value of the accumulator is set to the value coming from the frame counter, thus discarding the old event counts; when a `EOP` signal is received, `event_count` signal is sent to the average counters and to RAM counters to perform accumulating tasks depending on `BXTYPE` and `BXID` respectively. A `EVT_READY` signal is sent to the subsequent components when the accumulation task over each event is done and event counts are ready to be processed.

4.1.2 Average counters

The average counters, as said above, are defined to provide a time averaged estimate of the instantaneous luminosity, counting the number of selected clusters within a time integration window and dividing the contribution to the accumulated clusters depending on the bunch crossing typology. It is also useful to define some *histogram-like* counters, each storing the occurrences of events with a given number selected clusters.

The average counters are used to perform the measurement of luminosity by means of the *linear method* (Eq. 5.7). The "zero-count" histogram-like accumulator is instead used to exploit the *log-0 method* (Eq. 5.8) for measuring the instantaneous luminosity. The other histogram-like counters are useful to sample the cluster number distribution per each event, providing a tool to check if the cluster number distribution follows the expected Poisson distribution or if unexpected distributions arise.

The `event_count` signal is sent to a multiplexer, that further drives it to a set of 32-bit wide accumulators, each corresponding to the `BXTYPE` of the event they refer to. Each accumulator value is stored in the flip-flops registers implemented within the ALM elements.

For each type of bunch-crossing, 6 independent 32-bit wide accumulators are defined: one simple accumulator (average counter) of the total number of clusters and 5 accumulators (histogram-like counters) of the occurrences of the `event_count` signal *Most Significant Bit* (MSB). This is done by raising a MSB flag, by means of opportune chains of XOR gates. Calling the `event_count` signal C whose indices range form 0 to 3, being C[3] the most significant bit, then the MSB flags are defined as:

- MSB-4 flag : C[3]; correponds to #clusters≥ 8
- MSB-3 flag : (C[2] **XOR** C[3]) **AND** C[2]; correponds to $4 \leq$ #clusters ≤ 7

- MSB-2 flag : (C[1] **XOR** C[2]) **AND** (C[1] **XOR** C[3]) **AND** C[1]; correponds to $2 \leq \#\text{clusters} \leq 3$
- MSB-1 flag : (C[0] **XOR** C[1]) **AND** (C[0] **XOR** C[2]) **AND** (C[0] **XOR** C[3]) **AND** C[0]; correponds to #clusters = 1
- MSB-0 flag : empty events flag when every bit of C is zero, corresponding to #clusters = 0;

Depending on the MSB flag, the corresponding counter is selected using a multiplexer and it is updated increasing its value by one.

Depending on `ECS_WDATA` input signal (see Section 4.1.4), counters contents are copied into the COUNTERS BUFFER, storing the values of counters that are actually sent in output. Once copied, the AVERAGE COUNTERS values are reset. The counter buffer is updated at a much lower frequency ($\mathcal{O}$(Hz)) than the previous counters, assuring that every counter data sent to the output refer to the same time integration. The time window is computed through a 64-bit timestamp counter (splitted in two 32-bit registers), counting the number of 250 MHz clock cycles from the reset. Its value is stored in a memory register and exposed to the output when requested. Once sent in output, its value is reset like the other counters.

4.1.3 RAM counters

It is very useful to store events counts divided per bunch-crossing ID. In this way it is possible to characterize the luminosity properties of each different bunch. Instead of accumulating the total number of selected clusters, the occurrences of EVENT_COUNT value greater then a fixed threshold (for example #clusters > 0). This is equivalent to store the number of events with zero cluster counts, thus the RAM counters exploit the log-0 method to perform measurements of per-bunch-crossing luminosity.

Since there are 3564 different bunch crossing, the most practical way to implement such counters is through a 3564 addresses RAM, defined as an hard IP generated with the Intel Quartus Prime software. The RAM addresses are 12 bit wide. The width of the data is set to 20 bit, minimizing the number of M20K memory blocks used to implement the RAM [41]. 20 bits counters are wide enough to prevent overflow in typical ECS readout times ~ 3 s (overflow occurs in the worst case scenario in $2^{20}/\nu_{\text{LHC}} \simeq 2^{20}/11\,\text{kHz} \approx 95\,\text{s}$).

EVENT_COUNT signals from the Event Counter entity are sent in input to the RAM along to the `ST_SOURCE_BXID` of the event. In order to accumulate statistics, a read and a subsequent write operations are needed to update the value of counters referring to the input `BXID` data. The write and read operation cannot be performed at the same time because of the IP design. The read operation takes 2 clock cycles, the count-updating process needs one more cycle as the write operation does. Using only one input line for the RAM (One-port), these delays would make necessary the implementation of an asynchronous buffering stage (FiFo) in order to avoid data desynchronization.

A more practical way to overcome this issue is to implement a Dual-Port Ram, reserving one data stream for reading-only purposes and the other one for writing-only operations. `ST_SOURCE_BXID` is used as input address to read the value of the corresponding counter stored into the RAM, when a `READY` signal is received. The (old) value of the selected register is sent to a 20-bit wide adder that increases its value by one if the EVENT_COUNT is greater then the threshold. The (new) value for the register content is written into the RAM at the corresponding `BXID` address.

RAM counters are updated until a fixed number of LHC revolutions are integrated, rather then for a fixed/user-controlled time window as in the case of the average counters. This is done to assure uniformity between different counters, being each accumulated over the same number of bunch crossings.

A `RAM_READY` flag is raised when the integration over (for instance) 10^4 LHC revolutions (*i.e.* once every $\sim$ 0.89 s) is reached, then no longer accumulation is performed until RAM counters are dumped to the output following a read request.

4.1.4 ECS output interface

The output mechanism is controlled by the ECS system, driven by a 50 MHz clock (ECS clock). The controlling input signals are given below:

- `ECS_RREQ` : 1-bit signal asking for data on the output;
- `ECS_ACTIVE_LUMINOSITY` : 1-bit signal asking for the output for the average counters;
- `ECS_ACTIVE_RAM_LUMINOSITY` : 1-bit signal requesting RAM counters;
- `ECS_ADDR` : 14-bit word encoding which of the average counters stored in the buffer is requested to the output;
- `ECS_WDATA` (and `ECS_WREQ`) : 32-bit signal (1-bit) used to command the write operation of counter registers into the counters buffer before resetting them. It drives the timestamp counter too.

The output signals are instead:

- `ECS_RDATAVALID` : signal acknowledging the validity of data sent at the output.

- ECS_OUTPUT_LUMINOSITY : 32-bit ⊕ 1-bit ⊕ 1-bit signals carrying respectively the content of the required luminosity counter (or timestamp), the read and the write acknowledges. The latter two are not used by this firmware and are set to 1, being implemented just to match the expected ECS format.

An ENABLE REGISTER separately activates the output mechanism for the average counters or for the RAM counters, depending on ECS_RREQ signal and which of the ECS_ACTIVE flags is raised.

If ECS_RREQ **AND** ECS_ACTIVE_LUMINOSITY = 1, counters buffer is accessed. A multiplexer selects one luminosity counter (or timestamp counter) depending on the address of ECS_ADDR, sending it to the output. ECS_RDATAVALID flag is set HIGH.

If ECS_RREQ **AND** ECS_ACTIVE_RAM_LUMINOSITY = 1 then RAM contents are accessed. In this case a different strategy needs to be employed, since of the higher number of counters defined. Instead of sending to the output each counter separately from the other ones when an ECS read request is received, a full-serial output strategy has been adopted. A *Loop Dispatcher* is implemented to control the serial output process: once enabled, it sends a 12-bit wide signal in input to the reading-only data stream of the RAM. If RAM_READY flag is not raised the address from the dispatcher is not accepted and null-data are sent back to it. ECS_RDATAVALID flag is then set LOW. When instead the flag is raised, the address is accepted and the corresponding counter value is sent back to it; correspondingly the ECS_RDATAVALID flag is set HIGH. The selected counter content is appended to the corresponding address building up a 32-bit signal that is sent to the output. The address from the loop dispatcher is then incremented by one every ECS clock cycle. In 3564 ECS clock cycles ($\sim 71.3\,\mu s$) every content of the RAM is sent to the output. Finally, the address from the loop dispatcher is reset to zero, the RAM contents are reset and the accumulation starts again.

4.1.5 Compilation results

The Luminosity register firmware is embedded within the data processing firmware, sharing the available logic and memory resources of the Arria-10 chip mounted on TELL40 boards. This additional component has been compiled within the complete VELO firmware, showing no related timing violations and a very modest resource usage, employing 989 ALMs (2.3‰) and 4 M20k blocks (1.4‰) per data flow. In Figure 4.2 the results of placing and routing operation from the compiler is shown.

Given the positive results of the compilation, the firmware I developed can proceed into the next steps required to validate its functionality and feasibility for actual data-taking. Integration of the luminosity counters into the LHCb online system is described in Chapter 7, as well as its commissioning.

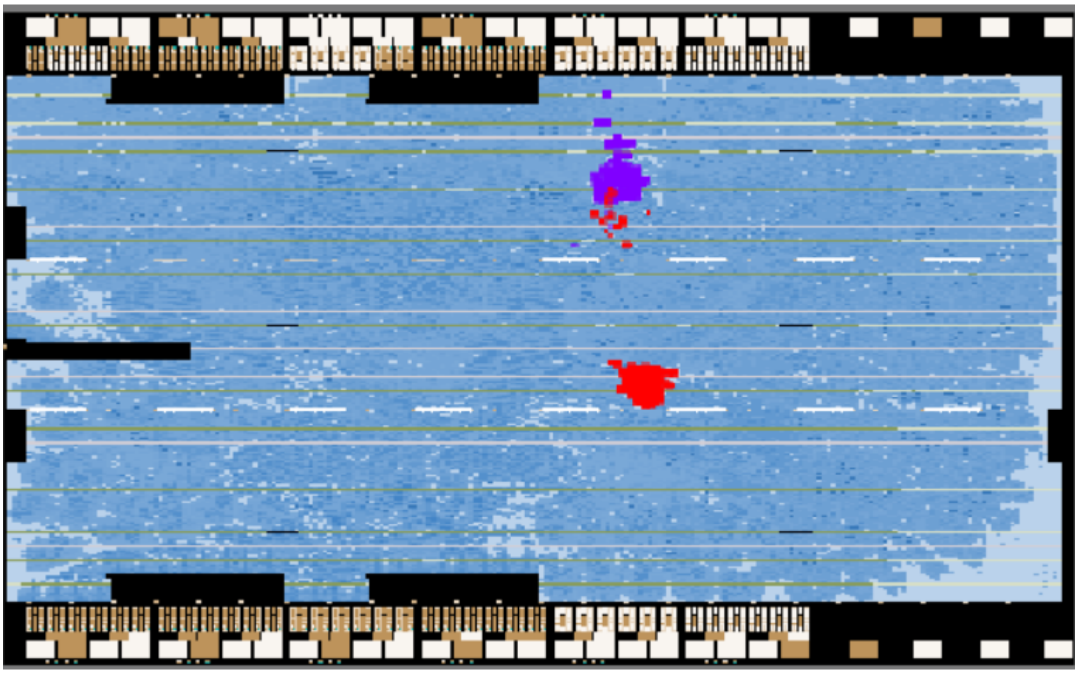

Figure 4.2: Placement of luminosity counters components inside the actual Arria-10 chip. Components required for the DATA FLOW 0 are displayed in red, while in purple those ones for the DATA FLOW 1. Dark-blue regions are the components placed for other tasks of VELO firmware, while the light-blue ones are the non-used gates instated on the chip.

5

Measuring the LHCb luminosity

Chapter Contents

In the previous chapter I have described the system I set up to measure in real-time the rates of VELO clusters, under a variety of programmable conditions. With the successful implementation of this system, it becomes possible to monitor various quantities of interest at a very high rate. In this chapter, I describe the studies I performed to develop the methodology for turning these raw rates into measurements of physics interest – most notably the instantaneous luminosity at the LHCb interaction point, and the beamline parameters.

I performed extensive studies to validate the feasibility of luminosity reconstruction in such a way using detailed simulations taking into account the whole Upgrade I detector and the its reconstruction sequence.

5.1 Methods for measuring the instantaneous luminosity

The luminosity is amongst the key parameters of any particle experiment at a collider. Given a process with cross section σ and n total observed events during the experiment data-taking lifetime ΔT, their relationship is given by

$$n = \mathcal{L} \cdot \sigma = \sigma \int_{\Delta T} \mathcal{L}_{\text{inst}}(t) \text{d}t \tag{5.1}$$

where $\mathcal{L}_{\text{inst}}(t)$ is the instantaneous luminosity at time t and $\mathcal{L}$ is the integrated luminosity. The instantaneous luminosity can be expressed in terms of the beams geometries and parameters.

In the case of LHC two distinct bunches collide with frequency f_{rev}; the luminosity per bunch pair is given by [33, 42]

$$\mathcal{L}_{\text{bunch}} = f_{\text{rev}} N_1 N_2 \sqrt{\frac{(\overrightarrow{v_1} - \overrightarrow{v_2})^2 - (\overrightarrow{v_1} \times \overrightarrow{v_2})^2}{c^2}} \iiiint \rho_1(x, y, z, t)\rho_2(x, y, z, t)\text{dxdydzdt} \tag{5.2}$$

where N_i, $\overrightarrow{v_i}$ and $\rho_i(x, y, z, t)$ are the bunch popoulation (# protons), velocity and density of beam i. The term in square root accounts for the relative velocities of the two bunches and their relative angles, while the integral is the *overlap integral*, accounting for the spatial densities of the bunches and their time evolution while overlapping in the interaction point. The integral $F(x, y, z) = \int \rho_1(x, y, z, t)\rho_2(x, y, z, t)\text{dt}$ is the time-integrated beam overlap distribution, also said *luminous region*. Under the assumption that both colliding beams are gaussian-shaped, the luminous region is itself a three-dimensional gaussian distribution [42, 43] [1]:

$$F(x, y, z) \propto \exp\left(-\frac{1}{2}\frac{(x - \overline{x})}{\Sigma_x^2} - \frac{1}{2}\frac{(y - \overline{y})}{\Sigma_y^2} - \frac{1}{2}\frac{(z - \overline{z})}{\Sigma_z^2}\right) \tag{5.3}$$

where $\overline{x}, \overline{y}, \overline{z}$ are the *nominal position* of the center of the luminous region, and $\Sigma_i^2 = \sigma_{i1}^2 + \sigma_{i2}^2$ are the convoluted transverse width of the bunches.

Going back to the luminosity per bunch, the actual instantaneous luminosity for a train of colliding bunches (as in LHC) is computed by adding up bunch luminosities for every bunch-bunch pair occurring at the crossing point during the period of an accelerator revolution.

Expression 5.2 can be greatly simplified when the half-crossing angles α (half of the angle formed between the beams directions) is small and the bunches are equal in size with symmetric Gaussian-shaped transverse distributions with standard deviation σ, giving [43, 44]:

$$\mathcal{L}_{\text{bunch}} \cong 2cf_{\text{rev}}\frac{N_1 N_2 \cos^2\alpha}{4\pi\sigma^2} \tag{5.4}$$

It is easy to see that, for fixed N_i and f_{rev}, $\mathcal{L}_{\text{bunch}}$ essentially depends on the crossing angles between beams and the bunches transversal dimension, and to increase the luminosity one needs to reduce the α and to increase beam *focusing i.e.* reduce the transverse spread of the bunch. Changing this values will also modify the shape of the luminous region (Figure 5.1), thus the spatial region where primary vertices are produced. Detectors are designed to work optimally with certain values of both α and σ.

The luminosity measurement is relevant to perform cross-section measurements. Indeed equation 5.1 can be inverted to achieve an expression for the cross section: $\sigma = n/\mathcal{L}$. Mea-

[1]this is strictly true for head-on collisions *i.e.* when beams collide with no crossing-angle α. When $\alpha \neq 0$ then the distribution $F(x, y, z)$ is not exactly factorizable; nevertheless it is still possible to define the nominal positions of the luminous region and their width, although $\overline{x}, \overline{z}, \Sigma_x$ and Σ_z being correlated

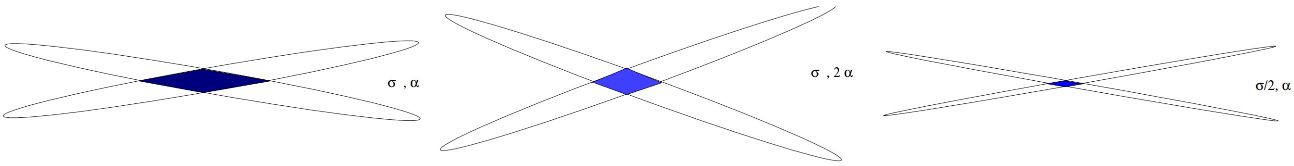

Figure 5.1: Schematic illustration of the luminous region (blue area) assuming equal shaped beams. The half-crossing angle α is half of the angle between the two bunches.

surements of the instantaneous luminosity are thus required to compute the integrated luminosity that is needed for the measurement of the cross-section of any process. The systematic uncertainty on the integrated luminosity is usually the most contributing factor to cross-section systematics [33]. During Run-2, the systematic uncertainty on the integrated luminosity was of 3.9%, by far the principal source of systematics in most of the measurements of cross-sections (see for instance [45]). The aim of the collaboration for Run-3 is to increase the precision in the integrated luminosity measurement, aiming at reaching an accuracy of 1 − 2% [33]. For this reason the collaboration decided to implement for Run-3 as many detector and systems as possible (see 2.4.1) to monitor the instantaneous luminosity at the interaction point during the data-taking period.

Multiple methods exist for luminosity measurement: one can adopt an accelerator-independent approach by counting reactions whose cross-sections and the corresponding detector efficiencies are well known. In another approach, one measures the differential proton-proton elastic cross-section to infer the luminosity by means of the optical theorem [46].

A third approach relies in measuring precisely - but on an arbitrary scale - the event rate dn/dt in Equation 5.1 for a reference process. This method is named *relative-luminosity measurement* , and is amongst the most used due to its simplicity. Equation 5.1 can be written considering pp inelastic process[2,3] σ_{inel} and small time-integration window, the expectation value of the the instantaneous luminosity is then given by:

$$\langle \mathcal{L}_{bunch} \rangle = \frac{\left\langle \frac{dn_{\text{inel}}}{dt} \right\rangle}{\sigma_{\text{inel}}} = \langle \mu_{\text{inel}} \rangle \frac{f_{\text{rev}}}{\sigma_{\text{inel}}} \tag{5.5}$$

where $\langle \mu_{\text{inel}} \rangle$ is the mean number of inelastic pp collisions (pile-up).

The mean pile-up per event can be measured employing different methods. One can for instance directly count the number of reconstructed primary vertices per each event, being this proportional to μ_{inel} up to an acceptance×efficiency factor ϵ. One can also measure the a *count rate* R_{vis}, such as the number of tracks per bunch crossing, the number of hits on a

[2] elastic pp collisions are not detected in LHCb since protons are scattered inside the beam pipe.
[3] One may actually consider any process, being the method very general.

detector or the number of reconstructed particle. In this case an indirect measurement of the mean number of pp collisions is performed. Neglecting detector-related effects, the measured count rate is proportional to the pile-up up to a scale factor A. The different methods used to perform such count rates are usually named *luminosity counters*.

In both cases one measures a quantity that is proportional to the pile-up. The expression for the luminosity can be finally written as:

$$\langle \mathcal{L}_{bunch} \rangle = \langle \mu_{\text{vis}} \rangle \frac{f_{\text{rev}}}{\sigma_{\text{vis}}} \tag{5.6}$$

where $\langle \mu_{\text{vis}} \rangle$ is usually referred to as *visible pile-up* and can be measured both directly by counting the number of reconstructed primary vertices ($\langle \mu_{\text{vis}} \rangle = \varepsilon \langle \mu_{\text{inel}} \rangle$) or by measuring a count rate ($\langle \mu_{\text{inel}} \rangle = A \cdot R_{\text{vis}}$). The quantity σ_{vis} is referred to as *visible cross-section* and accounts for the ε factor previously said ($\sigma_{\text{vis}} = \varepsilon \sigma_{\text{inel}}$).

The value of $\langle \mu_{\text{vis}} \rangle$ is thus in principle proportional to the luminosity up to a constant scale factor, being used to perform a relative measurement of luminosity. Depending on the characteristics of the sensor adopted as input for the luminosity counter, non-linearities may arise especially for high-luminosity values. The absolute measurement of luminosity is performed once the visible cross-section is known; calibration procedures such as the *Van der Meer scan* [47, 48] are employed to measure σ_{vis} thus making it possible to measure the luminosity absolute value starting from luminosity counters.

In the contest of this thesis, $\langle \mu_{\text{vis}} \rangle$ is measured by means of the VELO cluster counters described in Section 4. The number of clusters selected in a region of the sensor can be related to the visible pile-up by exploiting its expected linearity properties and its statistical distribution. In particular, two methods are mainly accustomed:

Linearity Counts N_i of selected clusters for each bunch crossing are accumulated over a fixed time-window. Saying n the total number of event, then assuming linearity

$$\langle \mu_{\text{vis}} \rangle \propto \frac{\sum_i N_i}{n} \tag{5.7}$$

While extremely simple, this method requires linearity to hold in full range of detector counts - that is not necessarily true especially at high particle multiplicities. For example, the coincidence of two particle hits on contiguous pixels of the VELO leads to the reconstruction of a single cluster, defeating linearity.

Log-0 Method One can define the fraction of *empty* events $P(0)$ where counts are zero *i.e.* $N_i = 0$. Assuming Poisson distribution for the number of counts per event, then (θ_H is the Heaviside step function):

$$\langle \mu_{\text{vis}} \rangle \propto -\log P(0) \propto -\log \frac{\sum_i \theta_H(N_i = 0)}{n} \tag{5.8}$$

This methods is insensible to high track multiplicity events, that are all simply accounted for as non-empty events and thus not considered in expression 5.8. Instead the computation of μ_{vis} may be impossible for high values of pile-up, since the probability of having empty counts events is exponentially suppressed. Increasing the value of the threshold *i.e.* requiring $N_i \leq N_{thr}$ increases in turn the value of $P(0)$, nonetheless systematic biases will arise since the equation 5.8 will not be valid anymore.

I studied both methods to reconstruct the average rate of my cluster counters, showing that proportionality between them and the luminosity holds.

5.2 The LHCb Upgrade I simulation

Feasibility studies to determine the actual relationship between the measured rates and the accelerator luminosity are performed by means of official LHCb simulations, produced changing the settings of the luminosity and varying the beamline parameters.

The LHCb simulations are carried of by several different software applications each executing specific tasks. Generation of primary particles from the *pp* collision is provided by the Pythia package [49], a general purpose event generator. Time evolution and particle decay is modelled by the EvtGen framework [50], while particle transportation inside the detector is realized by the Geant4 toolkit [51], where the whole and detailed geometrical description of each component of the detector is implemented. These tasks are embedded within the Gauss application [52], thus responsible of the particle event simulation. It manages luminosity-related parameters too, such as pile-up, luminous region xyz position and its width, the beam crossing-angle and beamline inclination.

Subdetectors responses when interacting with the simulated particles are provided by the Boole application, responsible of signals digitization in the same format as the one from the actual experiment DAQ system. Simulation data are then sent through the same path that real data follow, thus being processed by the same trigger, reconstruction and analysis software applications. Monte Carlo data after digitization step and real data coming from the DAQ system are selected by the trigger. The application providing the high-level trigger is named Moore [53], where trigger algorithms are implemented both for HLT1 and HLT2 systems.VELO clustering is embedded in this framework, where a software version is implemented. In order to study the properties of the FPGA-based counters I implemented, I modified the MOORE application implementing an algorithm that emulates the functionality of the luminosity counters firmware loaded into the VELO TELL40 readout boards.

Several simulations have been carried out, each with different settings of the instantaneous luminosity and beamline parameters. They are all Minimum Bias samples (*i.e.* generic pp inelastic collisions) produced with the LHCb Upgrade I detector simulation, with center-of-mass energy of the colliding protons of $\sqrt{s} = 13.6\,\text{TeV}$ and with bunch spacing of 25 ns. For each simulation, $N_{evt} = 100\text{k}$ events are analyzed, if not explicitly specified. Four different types of simulations have been produced:

- a scan of event pile-up ν[4] in the range $\nu \in [3.8; 35.8]$, corresponding to $\mathcal{L}_{\text{inst}}$ from $1 \cdot 10^{33}$ to $9.4 \cdot 10^{33}\,\text{cm}^{-2}\text{s}^{-1}$. The mean position of the interaction region is set to $(\overline{x}, \overline{y}, \overline{z}) = (0.0, 0.0, 0.0)\,\text{mm}$ (see Figure 2.2 for axis reference frame). Simulations are produced both in MagDown and in MagUp configurations[5]. The instantaneous luminosity is explored in a wider range than its expected value for Run-3, both to better study the lumi-counters properties and to show their robustness for higher luminosities planned for future Runs.

Each simulation is analyzed to study the luminosity properties of the VELO clusters counters. Once any lumi-counter system is calibrated for data taking, its calibration constants depend on the parameters of the beamline during the calibration phase. If the luminous region suffers of spatial displacements over time, then the luminosity estimate would naturally be biased. It is thus crucial to prove that using the VELO cluster counters it is possible to define an estimator as robust as possible to beamline displacements, or alternatively to show that is possible to correct the estimate accounting for a real-time position measurement of the luminosity region. The following set of simulations are produced to allow to study the dependence of cluster counters to the luminous region nominal position:

- a scan of luminous region mean z-position, from $-120\,\text{mm}$ to $120\,\text{mm}$ in respect to the LHCb reference frame, with $\nu = 7.6$ and $\overline{x} = \overline{y} = 0.0\,\text{mm}$, in MagDown Configuration.

- a scan of luminous region mean x-position from $-1.5\,\text{mm}$ to $1.5\,\text{mm}$, with $\nu = 7.6$, $\overline{z} = 0.0\,\text{mm}$ and $\overline{y} = 0.0\,\text{mm}$, in MagDown Configuration.

- a scan of luminous region mean y-position from $-1.5\,\text{mm}$ to $1.5\,\text{mm}$, with $\nu = 7.6$, $\overline{z} = 0.0\,\text{mm}$ and $\overline{y} = 0.0\,\text{mm}$, always in MagDown Configuration.

The results of the simulations analysis are reported in the following sections.

[4]from now on the pile-up is indicated with the symbol ν.

[5]The dipole magnet polarity can be switched to be aligned along the positive (negative) y axis for the up (down) polarity configurations. During physics data-taking polarity reversal is carried out to control detection asymmetries that could introduce systematic biases in $\mathcal{CP}$ asymmetry estimates [54].

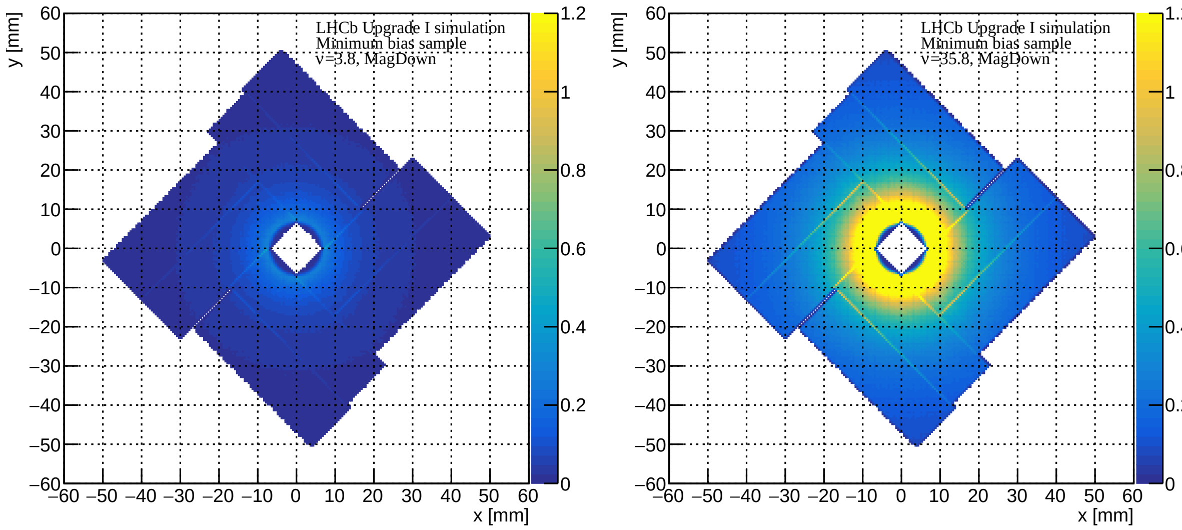

Figure 5.2: Cluster occupancy per event in $0.5 \times 0.5\,\mathrm{mm}^2$ sensor areas for $\nu = 3.8$ (left) and $\nu = 35.8$ (right). Overlap regions contain more clusters because an incoming particle produces hits on both sensors, thus two clusters are reconstructed.

5.3 From cluster counts to luminosity

Counters linearity with luminosity is studied with different simulations, with $\nu = 3.8$, 7.6, 14.3, 21.5, 28.7, 35.8. Figure 5.2 shows the cluster density on the VELO detector for the lowest and highest luminosity considered.

In order to study the counter properties and to optimize the selection region to provide the best unbiased estimate of luminosity, 1 mm-wide circular crowns selection regions, ranging from $r = 7\,\mathrm{mm}$ to $r = 31\,\mathrm{mm}$, were defined. Linear fits are performed to the mean number of clusters per event as the pile-up changes. The intercepts and pvalues resulting from the fits are reported in Figures 5.3-5.4, displaying only the innermost and outermost selection regions. The results of the fits for each radial selection region are reported from Fig A.1 to A.4. The inner regions (closest to the beamline) have worse linear properties than the outer ones. Indeed for the inner regions the extrapolated intercept at $\nu = 0$ is positive (left plot in Figure 5.3) , implying that cluster counts for high values of pile-up are understimated. This is presumably an effect of the overlap of different particle hits falling on the same pixel.

The regions farther from the beamline are best suited for the linearity of counts with the luminosity. Nonetheless, the inner regions of the sensors are more sensitive to displacements of the beamline due to their higher occupancy, and are therefore more useful in monitoring its position.

Taking into account firmware design contraints (ease of defining selection regions) to-

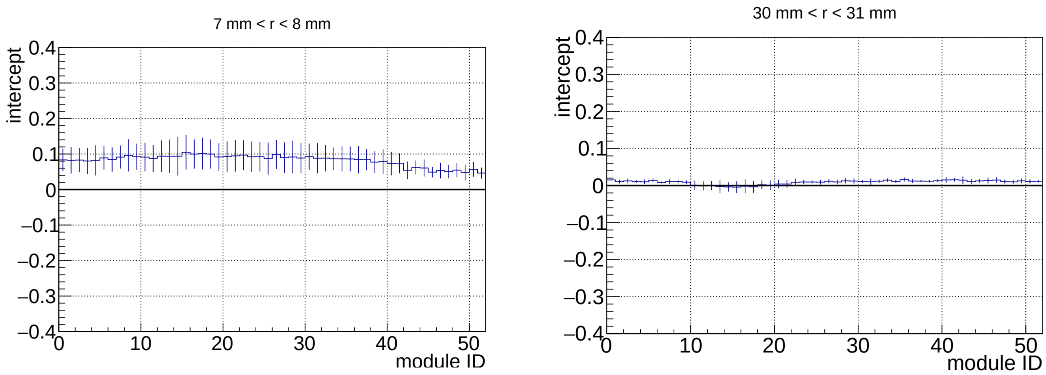

Figure 5.3: Value of the intercept of the linear fit to the mean number of clusters, in function of the number of the VELO module. The selection regions displayed are 7 mm $< r <$ 8 mm (left) and 30 mm $< r <$ 31 mm(right).

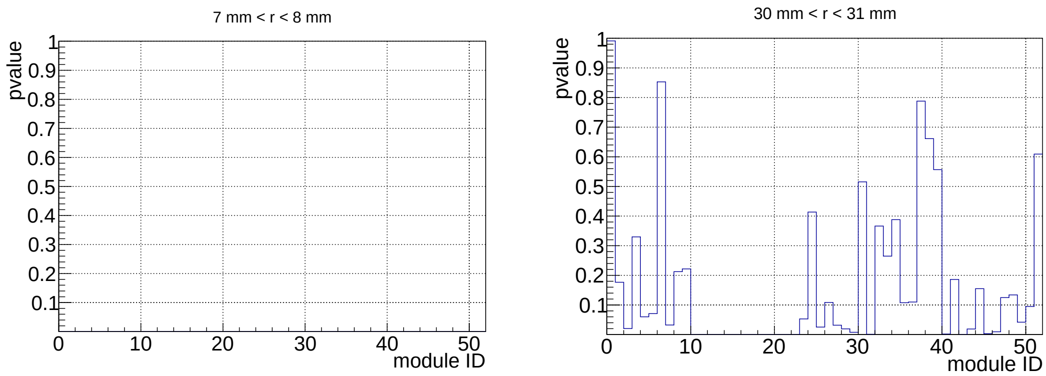

Figure 5.4: Value of the pvalue of the linear fit to the mean number of clusters, in function of the number of the VELO module. The selection regions displayed are 7 mm $< r <$ 8 mm (left) and 30 mm $< r <$ 31 mm(right).

gether with the above considerations, I choose to implement one selection region per sensor, leading to four *inner counter*s and four *outer counter*s per VELO station (see Figure 5.5).

Inner counters are set at $r \sim 14.3\,\mathrm{mm}$ from the nominal interaction point, while the outer counters are placed at $r \sim 25.6\,\mathrm{mm}$. All have rectangular shapes, 109×20 pixels wide.

The linearity of the rates measured as a function of luminosity is shown in Figure 5.6. The rate was evaluated both as $\sum_i \frac{N_i}{N_{evt}}$ ("average method"), and as $-\sum_i \log\left(\frac{N(0)_i}{N_{evt}}\right)$ ("log0 method"), with N_i being the number of clusters in the selection region of sensor i. In the assumption of Poisson-distributed counts and that linearity holds, both estimators are expected to yield a linear response.

The possibility to compute the sums in the above expressions is not obvious considering the firmware implementation of the luminosity counters. Indeed, the 52 TELL40 boards read out each one a different VELO modules, and process only data coming from the VELO module to which they are connected. Since there is no communication network among them, the cluster count data from different modules are separated and cannot be handled and combined together at readout level. This task can be instead performed at an higher level than the readout one, namely via the ECS. It in fact can access to every TELL40 boards simultaneously, thus the combination of cluster counts (and any further processing that is described in the following sections) can be centrally managed via ECS. Although the ECS system is not suited to perform high-rate computations, this does not represent a limitation. In fact the high-rate ($\mathcal{O}(30\,\mathrm{MHz})$) computations - selection and accumulation of clusters - are performed directly by the luminosity register firmware, while the ECS is only responsible of retrieving the data from each TELL40 board and to combine them at a much slower speed ($\mathcal{O}(\mathrm{Hz})$).

Coming back to the rates defined to exploit the average and log-0 methods, as said above, linear fits have been performed in both cases to check proportionality with the luminosity. Both methods show excellent properties (Figure 5.6), as can be seen from the high *pvalues* of the linear fits. It can be then concluded that luminosity can be consistently measured by using counters of hits on VELO sensors.

A value of luminosity can be evaluted separately for each VELO station:

$$\hat{l}_i = \frac{x_i - \mathrm{intercept}_i}{\mathrm{slope}_i} \qquad i = 1, \dots, 26 \tag{5.9}$$

where i is the number of the station, x_i is a count rate, and $\mathrm{intercept}_i$, slope_i are the parameters in Figure 5.6 for linear method and log0 method respectively. Since the linear method is easier to handle and provides an excellent description of the properties of the luminosity counters, in the following analysis it will be the method of choice to reconstruct the luminosity. Anyway, any discussion that will be reported can be similarly applied to the log0 method.

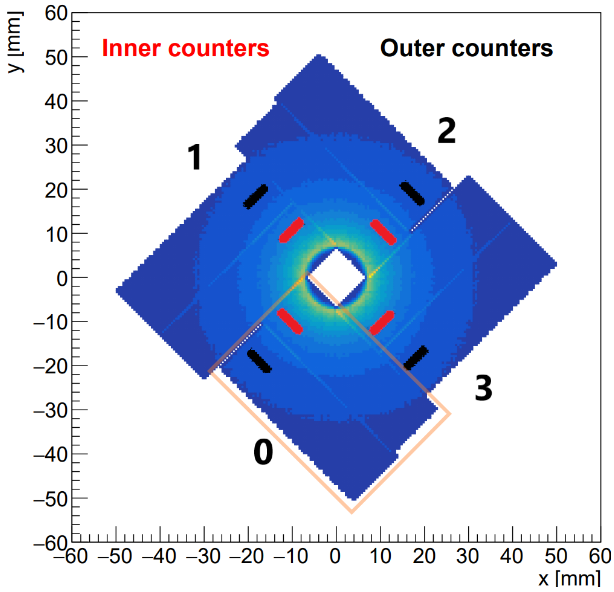

Figure 5.5: Position of outer counters (black regions) and inner counters (red) within a VELO station. Each couple of sensors (orange rectangle) is identified by an index $j = 0, 1, 2, 3$.

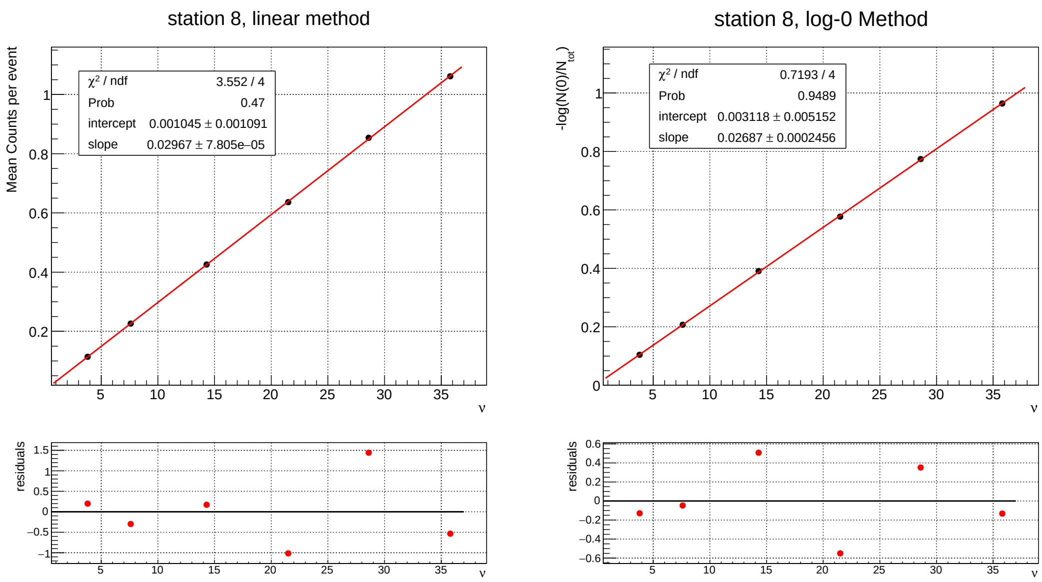

Figure 5.6: (Left) Linearity method and (right) log0 method for the outer counters on station=8 (the full set can be found in Appendix A). The pulls are shown in the bottom plots. Intercept at $\nu = 0$ are consistent with 0-value.]

The proportionality constants obtained from the above fits are calibrated on simulations in which the center of the luminosity region was fixed at the nominal interaction point. However, this is not necessarily the case in real data-taking conditions. Any deviation of the luminous region from the nominal position may affect the cluster rate measured in any selection region, thus introducing a bias in the luminosity estimate. The following sections are devoted to the study of this effect.

5.4 Effect of z displacements

The dependence of cluster rates to z displacements of the luminous region was evaluated for every VELO station. The effects of the longitudinal displacement are studied by means of simulations in which the nominal interaction point changes. In particular, 12 different simulations have been studied, each with $z_{lumi} = \pm 20, \pm 40, \pm 60, \pm 80, \pm 100, \pm 120\,\text{mm}$.

In order to understand how the count rates are affected by the displacements of the beamline, the distribution of counts for each simulation are compared to each other. The *figure of merit* of this study is the relative standard deviation of the station counts as the luminous region changes. Let x_{il} be the number of counts at the station i and relative to simulations where the luminous region is at $z_{lumi} = l \cdot (20\,\text{mm})$, where $|l| = 1, 2, 3, 4, 5, 6$. Then I define

$$\hat{\mu}_i = \frac{1}{12}\sum_l x_{il} \qquad \sigma_i = \sqrt{\underset{l}{\text{VAR}}(x_{il})} \tag{5.10}$$

The relative standard deviation of counts for each station is given by $\sigma/\hat{\mu}$. Figure 5.7 shows a 2D plot of the relative standard deviation above, as a function of the distance r from the center. It can be seen that the innermost regions are the more stable ($\sigma/\mu \sim \mathcal{O}(2\%)$) against z displacements of the beamline, while the outer ones are less stable, showing peaks in the relative standard deviation ($\mathcal{O}(12\%)$), whereas especially for the furthest modules from the nominal interaction regions (#module $>$ 35) the $\sigma/\hat{\mu}$ remains low. This suggests to choose anyway the outer regions for defining the selection regions since of their better linear properties, but combining the counts from different stations so that those ones with the lower values of $\sigma/\hat{\mu}$ are favored.

The distribution of mean outer counts per each station is shown in Figure 5.8, as a function of longitudinal position of each station. Each color represent a different simulation with a fixed value of the nominal z position of the luminous region.

The goal is now to define an estimator as robust as possible to z displacements of the beamline by properly combining the selected cluster counts from the several VELO stations.

One can see that the number of cluster in each selection region on a station is weakly correlated with the counts on other stations. Indeed the correlation matrix - Figure 5.9 - is

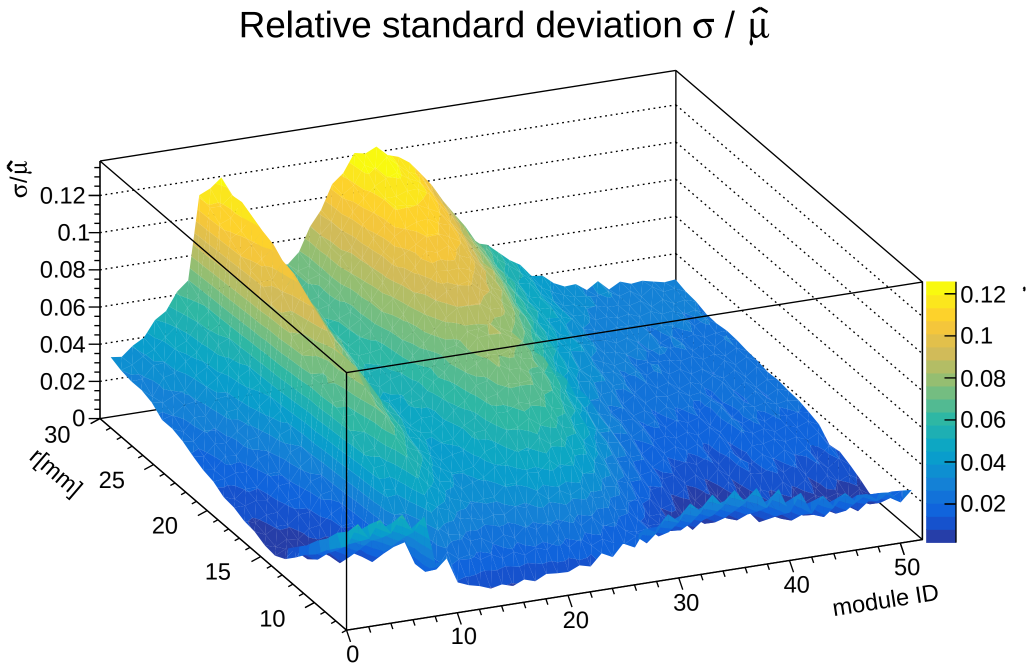

Figure 5.7: Relative standard deviation $\sigma/\hat{\mu}$ of the counts in circular crown regions as the z–position of the luminous region changes from -120 mm to 120 mm. The figure displaying the distributions of the counts per module in function of the z position is reported in Figure ??.

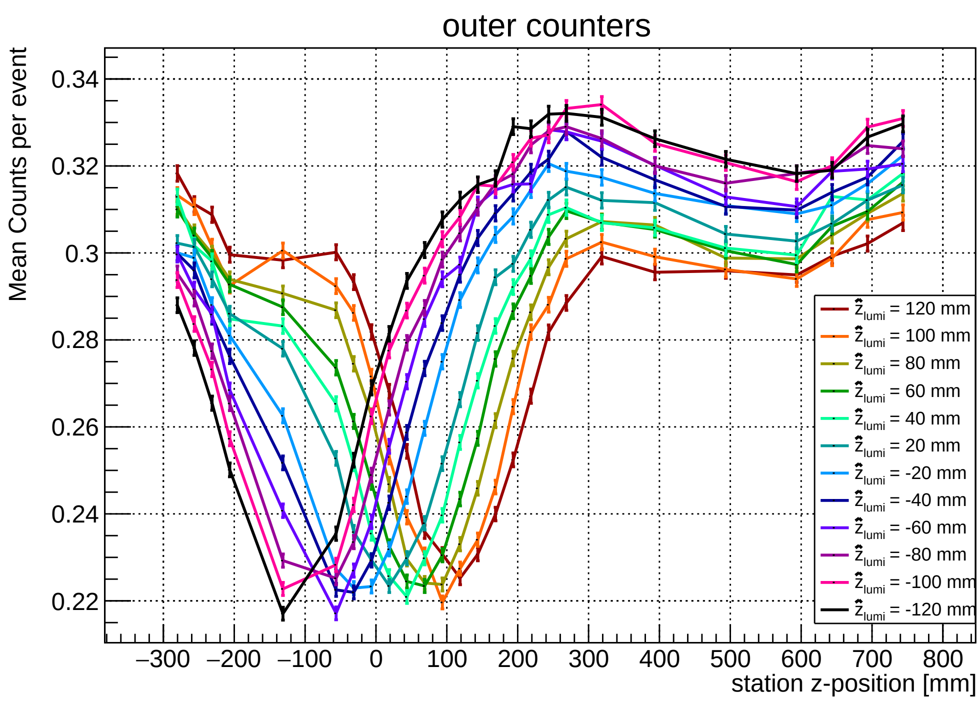

Figure 5.8: Mean count distribution for outer counters as a function of the longitudinal position of the station and varying the nominal position $\hat{z}_{\text{lumi}}$ of the luminous region.

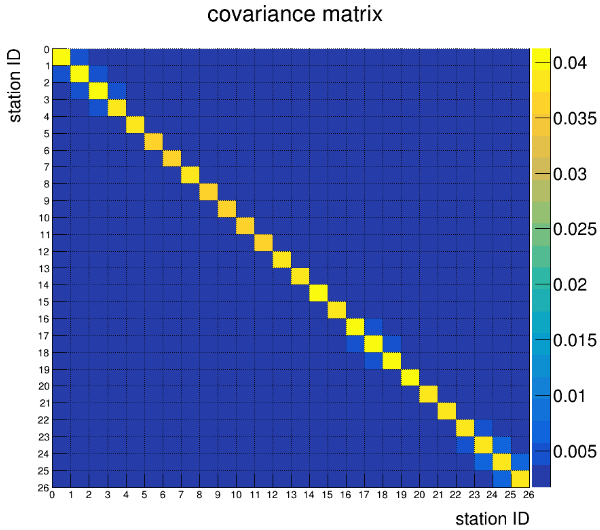

Figure 5.9: Correlation matrix of the number of selected clusters in each station.

(quasi-) perfectly diagonal, meaning that the counts on different stations are (quasi-) uncorrelated. Off-diagonals elements of the correlation matrix in this discussion are thus neglected, since contributing of a small factor in the computation of the uncertainty associated to the luminosity estimate. This ensures that stations can be combined without accounting for correlation terms in the evaluation of the uncertainty associated with the estimate of luminosity.

Following a similar approach as discussed in reference [55], the *best linear unbiased estimator* for non-correlated counters is defined as

$$\hat{L} = \sum_{i=1}^{26} \frac{N}{(\sigma_i/\hat{\mu}_i)^2} \hat{l}_i \tag{5.11}$$

where $N = \sum_j (\sigma_j/\hat{\mu}_j)^2$ is a normalization factor and $\hat{l}_i$ is the luminosity estimate from each station (Eq. 5.9).

Although the estimator defined in 5.11 reduces the dependence of the estimate on longitudinal displacements of the luminous region, the reconstructed luminosity will be biased accounting for it. Indeed counts in stations far from the interaction point (station z-position > 300 mm in Figure 5.8) still depends on the nominal position $\hat{z}_{lumi}$ in a nearly linear way. It is thus expected that the luminosity estimator will have an almost linear bias with $\hat{z}_{lumi}$.

In order to deal with the bias introduced by the luminous region displacements, counts distribution on different stations are used to infer the $\hat{z}_{lumi}$ position. In particular, the shape of the counts distribution in Figure 5.8 around the interaction point is approximately parabolic, thus performing a quadratic fit in proximity of the local minimum one can derive an estimate

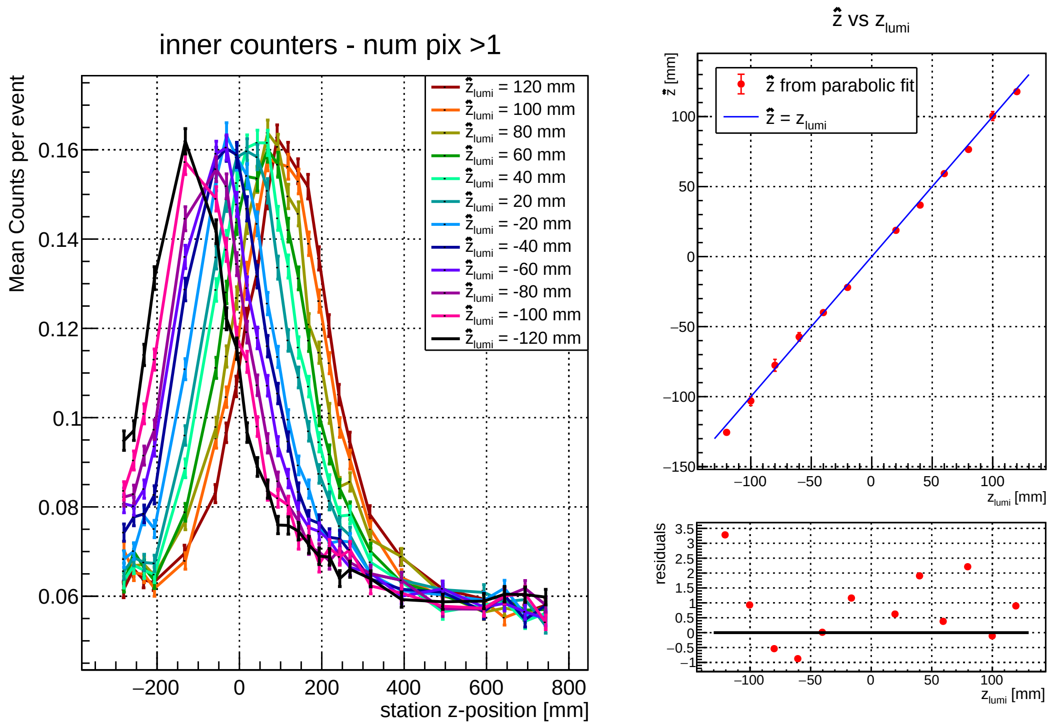

Figure 5.10: (Left) Cluster counts distribution per station for inner selection regions, with the requirement #pixels > 1. (Right) Reconstructed positions $\hat{z}$ of the luminous region against the true value z_{lumi}. The blue line is not a fit to the reconstructed $\hat{z}$, being instead the bisector $\hat{z} = z_{lumi}$. In these plots 20k events are analyzed.

of the position of the interaction region.

My firmware design allows however to select clusters also based on their topologyID as provided by the clustering module (Figure 3.7). This allows the flexibility to choose the most useful clusters for this rate measurement. I found out that the most sensitive measurement of $\hat{z}_{lumi}$ posizion is obtained from the inner selection regions, when requiring the selected cluster to include at least 2 active pixels. The distribution obtained with the above selection criteria is shown in Figure 5.10, together with the reconstructed z−position of the beamline. In this case, parabolic fits are performed around the local maximum of counts distribution. The shape is now inverted with respect to Figure 5.8. This happens because the requirement that each cluster must be bigger than one pixel selects those particles crossing the selection region with high inclination *i.e.* with low pseudorapidity. The majority of these tracks are in the proximity of the interaction region, while far from it they are outside the sensor acceptance. The resolution of these estimates is $\mathcal{O}(1\,\text{cm})$, and they can effectively be used to correct the luminosity estimate, providing a more reliable and accurate result.

The correction factor is estimated performing a linear fit to the luminosity estimates com-

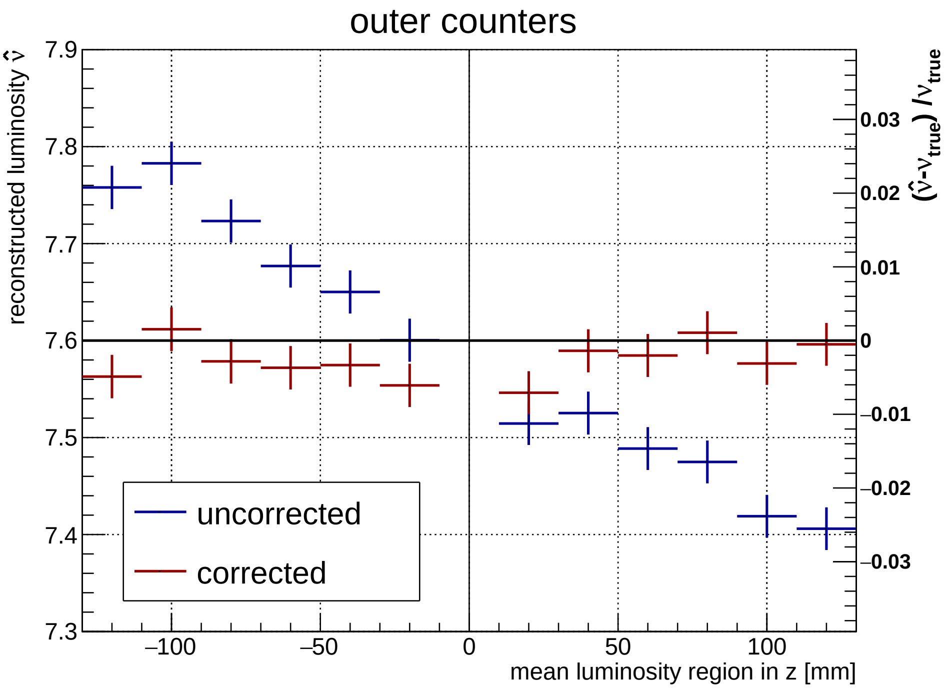

Figure 5.11: Uncorrected and corrected luminosity estimates, as a function of the luminous regions displacement in z. The true value of the pile-up is $\nu = 7.6$. The corrected estimates are obtained using expression 5.12. The horizontal bars do not represent any uncertainty.

puted with the expression 5.11. I define the *corrected estimate* of the luminosity as:

$$\hat{L}_{corr} = \hat{L} - \text{slope}_z \cdot \hat{z} \tag{5.12}$$

where slope_z is the slope of the linear fit performed to the uncorrected luminosity estimates, and $\hat{z}$ is the estimate of the position of the luminous region. Since the cluster counts have good linearity properties, it is expected that the correction term, when expressed in relative units, is independent of the scale of the luminosity. It can be thus used as an overall correction term to the luminosity estimator. The uncorrected estimates and the corrected ones are shown in Figure 5.11 . As can be seen, the luminosity estimator is efficiently corrected accounting for the reconstructed longitudinal position of the luminous region. The punctual value of the reconstructed luminosity is affected by a modest bias ($\mathcal{O}(5\ ‰)$), that is determined by the calibration procedure.

5.5 Dependence on $x - y$ displacements

As for the longitudinal displacements of the beamline (Section 5.4) the position of the luminous region in the xy plane is also expected to affect the luminosity estimate.

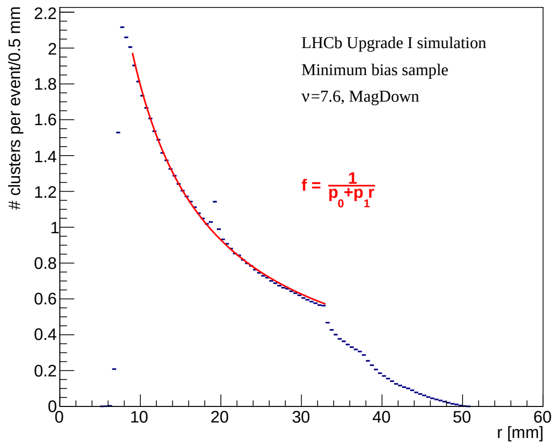

Figure 5.12: Radial distribution of reconstructed clusters on a VELO station. Non-smooth distribution at high-r is expected since of the finite size of the sensors. The peak structure at $r \approx 19$ mm is due to the presence of overlap regions, the same effect shown in Figure 5.2. In this plot the width of the circular crown is $\delta = 1$ mm.

In order to better understand the effects of a spatial displacement in the xy plane, and how to measure it, it is useful to start from an analytical argument.

The radial density of reconstructed hits, integrated in $r^* < r < r^*+\delta$ is shown in Figure 5.12. The distribution function f' in the region of interest $10\,\text{mm} < r < 30\,\text{mm}$ can be approximated by:

$$f'(r) = \frac{1}{p_0 + p_1 r} \tag{5.13}$$

The corresponding distribution of clusters is given by

$$f(r) = \frac{1}{p_0 + p_1 r} \frac{1}{\pi\,(r - k_\delta)} \tag{5.14}$$

where k_δ depends on the size of δ. The cluster rate in a selection region $\Omega \equiv \{x \in [x^* - \delta_x, x^* + \delta_x]$, $y \in [y^* - \delta_y, y^* + \delta_y]\}$ is then given by

$$N_\Omega = \int_{x^*-\delta_x}^{x^*+\delta_x} \int_{y^*-\delta_y}^{y^*\delta_y} f(r)dxdy \underset{\delta_x,\delta_x \to 0}{\longrightarrow} f(r^*) \cdot \delta_x \delta_y \tag{5.15}$$

If the beamline moves from $(0,0) \longrightarrow (\overline{x}, \overline{y})$, then the distribution function is shifted:

$$\begin{aligned} f(r) \longrightarrow f(r, \overline{x}, \overline{y}) &= \frac{1}{p_0 \cdot \sqrt{(x-\overline{x})^2 + (y-\overline{y})^2} + p_1} \frac{1}{\pi\left(\sqrt{(x-\overline{x})^2 + (y-\overline{y})^2} - k_\delta\right)} \\ &= f(r) + g(r)(x\overline{x} + y\overline{y}) + \mathcal{O}(\overline{x}^2, \overline{y}^2) \end{aligned} \tag{5.16}$$

where $g(r) = f(r)^2 \frac{\pi(2p_0 r - p_0 k_\delta + p_1)}{r}$. From Eq. 5.15 and 5.16 we can clearly see that the rate of clusters in a region Ω will depend on the position of the beamline. Taking the counters symmetrically around the nominal interaction point *i.e.* centered at the coordinates $(x^*, y^*) = (0, \pm r^*), (\pm r^*, 0)$, then the sum N of the counters is independent - up to the second order - of the nominal position of the beamline in the xy plane:

$$N = \sum_i \frac{N_i}{N_{evt}} = \frac{4f(r^*)}{N_{evt}} \delta_x \delta_y + \mathcal{O}(\overline{x}^2, \overline{y}^2) \tag{5.17}$$

This gives the reason why the luminosity estimators in Figure 5.6 are both obtained as the sum of all (outer) counters, in order to make the luminosity estimator as robust as possible itself to xy luminous region displacements. However a quadratic term is still present in Eq 5.17, and if the displacement is sufficiently large, a further correction to the luminosity estimator needs to be considered. By properly combining the N_i for the different sensors one can - at least conceptually - compute the nominal $\overline{x}$ and $\overline{y}$ positions of the luminous region in a fully closed way, with no need for an external calibration.

Considering the first-order approximation in Eq. 5.16 and a reference system $(\overline{x}', \overline{y}')$ rotated anticlockwise of 45°, one sees with few algebraic computations that the asymmetry for between two counters (for instance between those in sensor 0 and 2, see Figure 5.5) at distance r from the nominal IP is

$$A_{20} = \frac{N_2 - N_0}{N_2 + N_0} = \frac{2r - k_\delta + c}{(r - k_\delta)(r + c)} \cdot \overline{x}' \tag{5.18}$$

where $c = p_0/p_1$ is the ratio of the parameters characterizing the distribution function 5.13. An similar expression holds for A_{13}, with $\overline{y}'$ replacing $\overline{x}'$. The unknown parameters in the expression of the counts asymmetries are $\overline{x}, \overline{y}$ and c, which is not a fixed parameter since it changes with luminosity and depends on the z-position of the luminous region. The best way to obtain estimates for $\overline{x}'$ and $\overline{y}'$ is to solve a system of three independent asymmetry equations. For instance, using both inner and outer counters at distance from the center r_i and r_o respectively, one can solve the system (i and o refer to inner and outer selection region respectively), obtaining:

$$\begin{cases} A^i_{20} = \dfrac{N^i_2 - N^i_0}{N^i_2 + N^i_0} = \dfrac{2r_i - k_\delta + c}{(r_i - k)(r_i + c)}\overline{x}' \\ A^i_{13} = \dfrac{N^i_1 - N^i_3}{N^i_1 + N^i_3} = \dfrac{2r_i - k_\delta + c}{(r_i - k_\delta)(r_i + c)}\overline{y}' \\ A^o_{12,03} = \dfrac{N^o_1 + N^o_2 - N^o_0 - N^o_3}{N^o_1 + N^o_2 + N^o_0 + N^o_3} = \dfrac{2r_o - k_\delta + c}{2(r_o - k_\delta)(r_o + c)}(\overline{x}' + \overline{y}') \end{cases} \tag{5.19}$$

Although the problem is well defined and mathematically closed, this method can fail at reconstructing the correct values of $\overline{x}'$ and $\overline{y}'$. This occurs because each expression for the symmetries is derived from the approximations made for the number of clusters in a selection region (Eq. 5.15 and 5.16) that are accurate at the % level. When combining different N_i - as it is done to define count asymmetries - significant biases can arise, thus making the system unsolvable or returing uncorrect values for the parameters. One could include further expansions terms in both equations 5.15 and 5.16 to reduce the biases in the expression of the asymmetries, but this yields a non-linear system of equations that requires more complex approaches (such as Newton method [56]).

In order to avoid systematic biases induced by the above approximations and to avoid getting stuck in complex algorithms, the values of the nominal positions of the luminous regions (in the non-rotated axis system) are computed combining two asymmetries:

$$\hat{x} = k\frac{1}{\sqrt{2}}(A^i_{20} - A^i_{13}) \tag{5.20}$$

$$\hat{y} = k\frac{1}{\sqrt{2}}(A^i_{20} + A^i_{13}) \tag{5.21}$$

where the parameter k is a calibration constant optimized to make the proportionality constant as close as possible to the unity in both cases. This is done by minimizing the quantity $t = (s_x - 1)^2 + (s_y - 1)^2$, where $s_{x,y}$ are the slopes of linear fits to the functions $\hat{x}$ and $\hat{y}$ as the beamline positions $(\overline{x}, \overline{y})$ change. The optimal value for k is found to be $k \cong 7.32$. The reconstructed values of $\hat{x}$ and $\hat{y}$ are shown in Figure 5.13. In particular, Eq. 5.20-5.21 hold for each station of the VELO. The estimator for the nominal positions of the beamlines are defined as the truncated mean[6] of the estimates from each station, excluding from the computation the lower 6 and the upper 6 xy estimates :

$$\hat{x} \equiv \frac{1}{14}\sum_{l=6}^{20}\hat{x}_l \qquad \hat{y} \equiv \frac{1}{14}\sum_{l=6}^{20}\hat{y}_l$$

where the $\hat{y}_l, \hat{x}_l$ are sorted by magnitude.

A non-vanishing intercept for the estimate of $\hat{x}$ is expected. Firstly because the beams collide with a non-zero crossing angle (implemented in the simulation too), producing an imbalance of particles in the half of the detector at negative x *i.e.* on the C-side of the VELO. This can be seen in Figure 5.14, showing a measurement of the beams position realized using the *Beam-Gas Imaging* method [44].

Secondly, because of the residual magnetic field in the VELO generated by the dipole magnet. The effect is enhanced with the magnet in MagDown configuration, decreased in the MagUp one. This can be seen in Figure 5.15, where the cluster count distribution per module are reported. C-side modules of the VELO (located at negative x, *i.e.* sensors 0 and 1

[6]truncated mean is introduced to regularize the estimates.

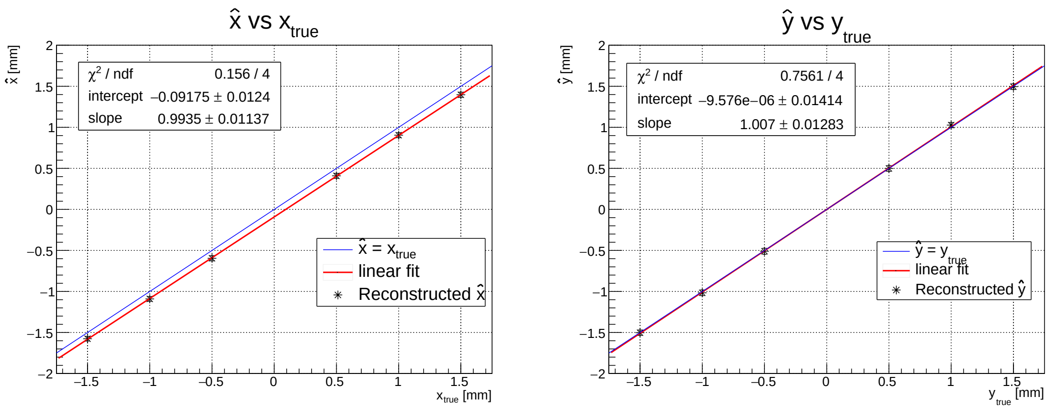

Figure 5.13: Reconstructed $\hat{x}$ (left) and $\hat{y}$(right) as a function of the true xy displacement of the luminous region.

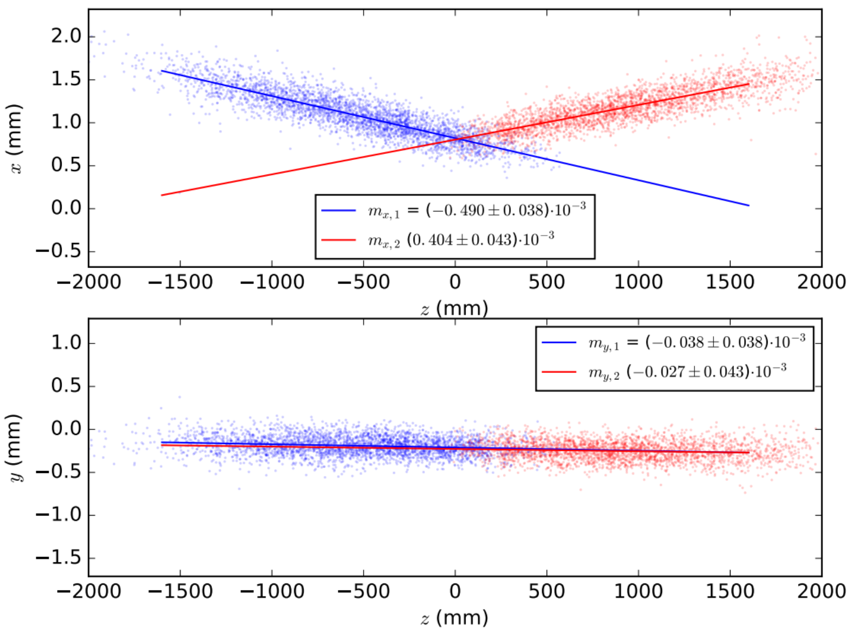

Figure 5.14: Example of bunches position and angle when colliding. The measurement is performed using the BGI method [44] during the run 174625 of fill 4937 in Run-1. The m_{ij} labels are the slopes of the bunches in the laboratory frame: beams have a non-zero crossing angle in the xz plane, producing an inbalance of secondary particles for negative values of x. In the yz plane instead the beams collide with no crossing angle.

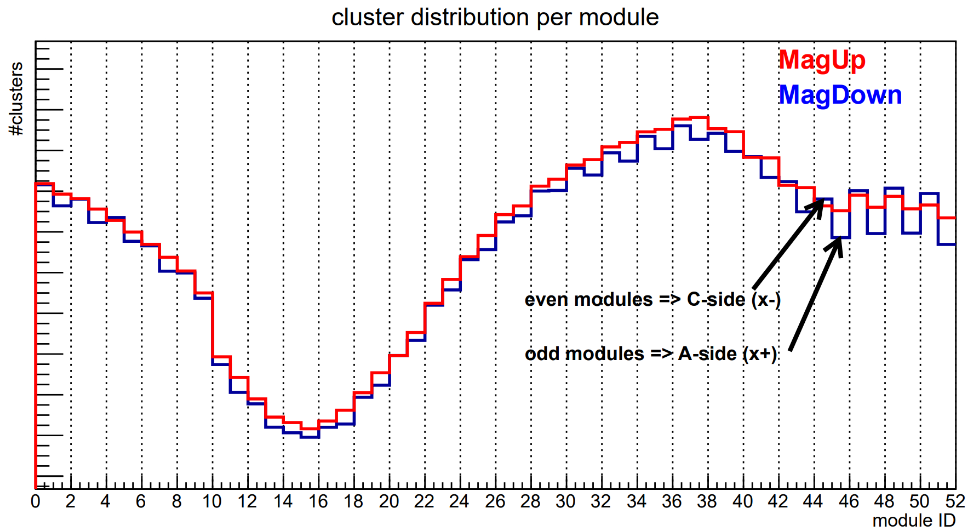

Figure 5.15: Cluster count distribution per module in MagUp (red line) and MagDown (blu line) configuration. The vertical axis is in arbitrary scale. The simulations providing these distributions are produced with the same configurations settings - apart of the magnet polarity - and both at $\nu = 7.6$ and $(\overline{x}, \overline{y}, \overline{z}) = (0, 0, 0)$.

in Figure 5.5) have more counts then the A-side modules (positive x). This occurs because of the non-zero crossing angle in the xz plane as displayed in Figure 5.14. The count asymmetry between A and C-side modules is enhanced in MagDown configuration, while in MagUp configuration the asymmetry is almost balanced. This effect can be understood considering that pp collisions produce an ensemble of particles with an overall charge *i.e.* $Q = +2$. Changing the magnetic field polarity will affect how tracks are in average deflected in the xz plane.

The resolution on the luminous region transverse position attained in this way depends only on the statistical uncertainty associated to the asymmetry value: $\mathrm{Var}(A_{ij}) = \dfrac{4N_iN_j}{(N_i + N_j)^3}$ [57]. Considering 100k collision events as I have done during the analysis - corresponding to 3.3 μs in actual data taking conditions - the attained resolution is $\mathcal{O}(10\,\mu m)$. With increasing the statistics *i.e.* integrating the counts in a wider time window, the uncertainty decreases following the asintotic behavior $\sigma_{x,y} \sim N_{evt}^{-1/2}$.

The information of the xy position of the beamline is useful in correcting the luminosity estimate. As only a quadratic deviation term is expected when the beamline is displaced from $(\overline{x}, \overline{y}) = (0, 0)$ (Eq. 5.17), a quadratic fit to the estimated luminosity at different values of $\overline{x}$ and $\overline{y}$ allows to compute the correction to the luminosity estimate. Quadratic fits $f_x = \mathrm{slope}_x \cdot x^2 + b_x$ and $f_y = \mathrm{slope}_y \cdot y^2 + b_y$ are performed to the uncorrected data in the same way as in Equation 5.12, and the luminosity estimator is corrected accounting for the value of the parameters slope_x and slope_y :

$$\hat{L}_{corr} = \hat{L} - \mathrm{slope}_x \cdot \hat{x}^2 - \mathrm{slope}_y \cdot \hat{y}^2 \quad (5.22)$$

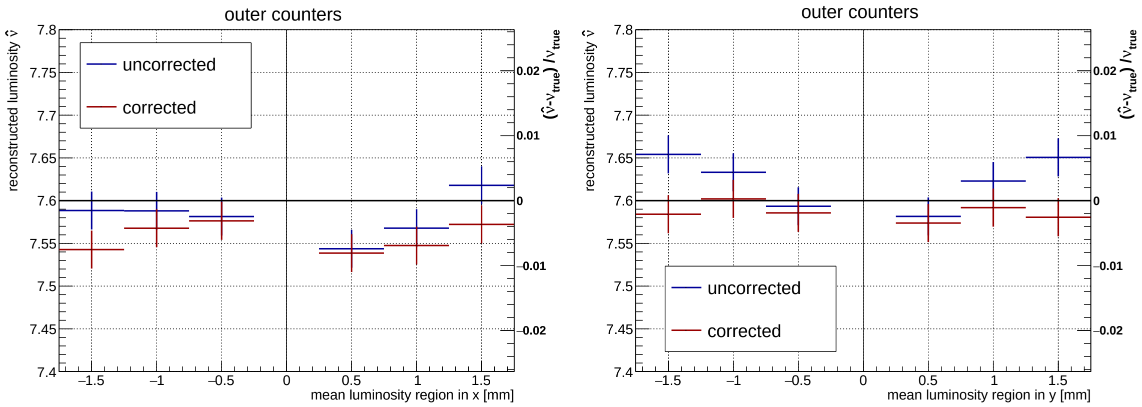

Figure 5.16: Uncorrected and corrected luminosity estimates in function of the luminous region (left) x and (right) y displacements. The true value of the luminosity is $\nu = 7.6$. The horizontal bars do not represent any uncertainty.

The resulting corrected luminosity estimates are shown in Figure 5.16 . While for the right plot (y–displacements) the uncorrected luminosity estimates are clearly parabolic in shape, for the left plot (x–displacements) the parabolic shape seems to be distorted. The effect may be attributed to the fact that the actual hits distribution does not have cylindrical symmetry because of non-zero beams crossing-angle in the xz plane. Anyhow, the effect is small $\mathcal{O}(5‰)$ and comparable with the bias induced by calibration procedure.

5.6 Reconstruction algorithm

The previous analysis results have shown that a robust, low-bias and high-precision estimator can be defined, based purely on the counters of VELO clusters implemented in FPGA firmware. To summarize, the recipe is given by the following steps:

- Define *outer* selection regions on the each VELO sensor, far from the interaction point: this enhances the linearity properties of the counters. Use these counters to measure the cluster rate.
- Sum up the values of the four outer counters for each VELO station (Eq. 5.9).
- Evaluate a weighted sum according to (Eq. 5.11).
- Define *inner* selection region, more sensitive to luminous region spatial displacements. Use these counters to measure the nominal position of the beamline and to correct the luminosity estimates.
- Correct the estimates 5.11 accounting for the reconstructed values of the nominal longitudinal position of the beamline, computed from the inner counters (Eq. 5.12).

- Further correct the estimates accounting for the reconstructed estimates of the nominal position of the luminous region in the xy plane, computed from the rate asymmetries of the inner counters (Eq. 5.22).

The luminosity estimator defined in this way is very robust against spatial displacements of the beamline and has very good linearity.

When analyzing 100k events, corresponding to a time integration of ~ 3.3 μs, the luminosity estimate is measured with a relative uncertainty of $\mathcal{O}(5‰)$, as can be seen from each point in Figures 5.11-5.16. This uncertainty is dominated by the statistic uncertainty. The calibration procedure instead introduces a small systematic bias, of the order of $\mathcal{O}(5‰)$. The statistical uncertainty becomes negligible when the standard integration time of 3 s is used. It is therefore expected that the actual uncertainty will be systematics-dominated.

The actual magnitude of systematic uncertainty can only be measured during dedicated data-taking periods, where actual performances of the detector and of the reconstruction algorithm are evaluated.

The above discussion ignores the possible presence of background. In the actual environment of the LHC, it is expected that there will be signals in the VELO also in those beam crossings where only one bunch is actually present in that position of the accelerator orbit (”bunch-empty”, or ”empty-bunch”), or no bunches at all (”empty-empty”). Background signals may be produced by detector noise, or by actual particles produced by interaction of the beam with residual gas atoms still present in the accelerator vacuum. The first two typologies of collisions generate particle in the forward and backward direction respectively, while the latter one produces background not related to the actual beams, isotropically distributed in the detector acceptance. These additional detector signals are not related to physics processes of interest, and they introduce a bias in the measurement of the luminosity, and consequently to any cross-section measurement. An estimate of the ”luminosities” related to these collisions can be performed in the same way as seen in the previous sections. Calling the luminosity associated to the bunch-empty (*be*), empty-bunch (*eb*) and empty-empty (*ee*) collisions $\hat{L}_{be}$, $\hat{L}_{eb}$ and $\hat{L}_{ee}$ respectively, and $\hat{L}_{bb}$ the luminosity estimate associated to the beam-beam events, then the actual luminosity $\hat{L}$ is given by:

$$\hat{L} = \hat{L}_{bb} - \hat{L}_{be} - \hat{L}_{eb} + \hat{L}_{ee} \tag{5.23}$$

where it is assumed that the background produced by one beam does not depend on the other, and that the background not related to beams (*ee*) does not depend on the presence of any beams.

The firmware I implemented is suitable to account also for the corrections to the luminosity provided by these different sources. I defined the average counters (Section 4.1.2) explicitly divided depending on the typology of bunch-crossing to make it possible to compute the corrected luminosity 5.23. Also using the RAM counters (Section 4.1.3) it is possible to compute the background-subtracted estimate of luminosity. Indeed the bunch-crossing typology is associated to each bunch-crossing, being this information directly given by the LHC when the fill-scheme is prepared.

In conclusion, I have shown that it is possible to perform a low-bias luminosity measurement based purely on the VELO clusters counts registered by the firmware described in the previous chapter.

6

Monitoring and correcting for detector effects

Chapter Contents

The method for luminosity measurement described in the previous chapter is based on the assumption of an ideal detector response. The efficiency for instance, if not perfect, was at least assumed to be constant in time and uniform over the entire active region of the detector - an assumption that does not hold with very good precision for most HEP detectors, and this carries over directly to the uncertainty in the luminosity determination.

Another tacit assumption has been that signals from different crossings are independent. Considering that the time interval between LHC crossings is a mere 25 ns, one might wonder about the correctness of that assumption as well. In fact, the amplitude of the analog signals produced by the front-end ASIC when a particle crosses the sensor is known to be still somewhat above zero at the time the next crossing comes. Obviously, this increases the probability that a new arriving signal will pass the detection threshold, and might even be sufficient by itself in setting a pixel to 'on'. This effect is usually referred to as *spillover*. Obviously, the presence of *spillover* causes a generalized increase in the number of reconstructed clusters, and a breaking of the assumption of Poisson distribution and independence underlying the subtraction of the backgrounds, thereby adversely affecting the accuracy of the luminosity determination.

What makes things particularly difficult is that these type of effects can significantly vary between different points of the same detector, and with time, operating conditions, ageing of the detector, and possibly other obscure factors. For a detector with a huge number of channels like the VELO, this means that the amount of data that needs to managed and stored to measure, monitor, and correct those effects is prohibitive. In fact, this type of correction is generally only performed in a much simplified way, or quite often not at all. The associated uncertainties become in this way an irreducible part of the systematic uncertainty. Looking at the specific LHCb situation, it is worth noting that the majority of Run-3 data will be saved in the so-called *turbo mode* (Figure 2.8 right), where only relevant summary data for physics analysis are stored, discarding the majority of raw data. This means that this type of detector-related effects could only be studied via a very limited number of dedicated trigger lines or special data-taking periods.

However, the accessibility of cluster data at readout level at a full rate, together with the flexibility and speed of the FPGA platform, offers a unique opportunity to monitor in real-time such detector-related effects, and possibly process the results in-situ. It is worth adding that such monitoring tasks are relevant beyond the goal to produce a very accurate luminosity estimate - they are also very important to the detector experts to diagnose the detector status and overall health during data-taking.

Considering that the aim of this thesis is precisely that of exploring possible new opportunities offered by this novel technological approach, there is a very good motivation to do so also for this topic. In the rest of this chapter, I will discuss how spillover and detection efficiency effects can be addressed within my firmware project.

6.1 Spillover monitoring

The occurrence of spillover depends on the very front-end electronics that produce a finite-duration analog signal being processed later to discriminate signal from noise. The VeloPix ASICs front-end electronics [58] is divided in an analog and a digital part. The former consists of a preamplifier and a discriminator; if the signal amplitude from the preamplifier is over the discriminator threshold, a *Time-over-Threshold* (ToT) is used to discard those hits whose ToT is not equal to or larger than the ToT threshold[1] . Depending on the discriminator and the TOT thresholds and the charge deposited by the incoming particle, spillover can occur in subsequent bunches. At the moment no systematic study in realistic data-taking conditions has been conducted to investigate the occurrence of spillover in VELO sensors; nonetheless, during the commissioning of the VELO detector, effects similar to spillover have been observed.

The data processing architecture implemented in the readout FPGA offers the opportunity to further process cluster data looking for time-dependent occupancy correlations, processing data at full rate in a strictly similar way to how it was done for the luminosity counters. The basic idea is still to look at a small region of the VELO sensors, only this time the correlation of the occupancies in such region between different events is also considered..

When counting the number of clusters falling in the selection region, the correlation co-

[1] ToT threshold is different from the discriminator threshold. The former time-based (a linear feedback shift register is used to measure the TOT), the latter is based on the voltage amplitude of the preamplified signal.

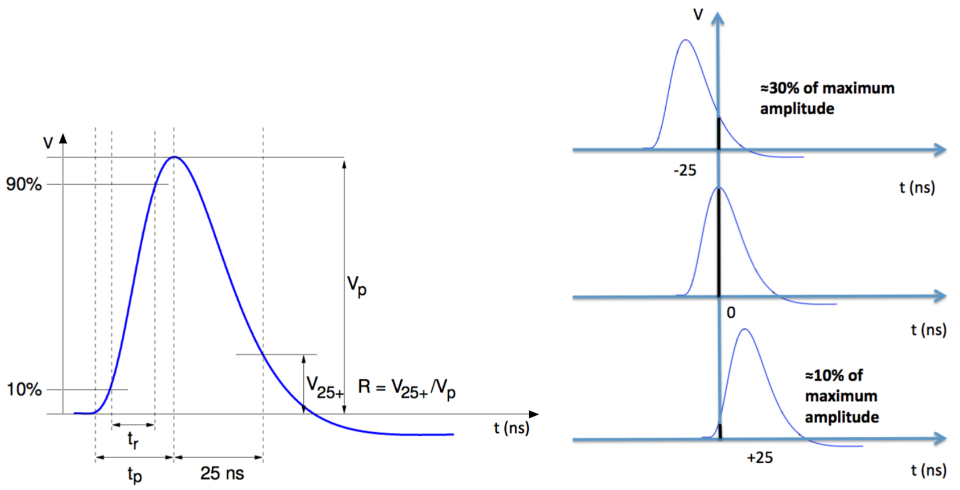

Figure 6.1: Sketch of spillover effect [59]. (Left) Typical shape of the analog signal generated by readout ASICs following a charge deposition. (Right) Shape of the signal from previous, current and next collision (25 ns time-spaced). Depending on the threshold value and signal shape, a spillover hit can be digitized both in previous and next collision cases.

efficient is given by :

$$\rho_{XY} = \frac{s_{X,Y}}{\sigma_X \sigma_Y} = \frac{\frac{\sum_{i=1}^{n} x_i y_i}{n-1} - \frac{\sum_{i=1}^{n} x_i}{n-1} \frac{\sum_{i=1}^{n} y_i}{n}}{\sigma_X \sigma_Y} \tag{6.1}$$

where x_i and y_i are the counts of selected clusters for the event and the next-event respectively, n the number of bunch crossings and $\sigma_{X(Y)}$ is the std deviation of the (next-)event cluster number population.

The correlation coefficient can be simplified by considering only a binary value for the occupancy of the selection region: if at least a cluster falls in the selection region then the event is flagged as ”cluster” (C), otherwise it is flagged as ”no cluster” (NC). The distribution of selected cluster number per event is Poisson-distributed; when its mean value is $\mu \ll 1$, then counting the occurrences of the “cluster - cluster”(CC), “cluster - no cluster” (CN) and “no cluster - cluster” (NC) for any consecutive events is nearly equivalent to perform the sums in numerator of Formula 6.1. This approximation is well satisfied given the actual VELO sensor occupancy. With a bit of algebra it easy to show that the correlation coefficient can be written as:

$$\rho \approx \frac{\nu_{CC} - \nu_{CN}\nu_{CC} - \nu_{CN}^2}{\sqrt{\nu_{CN}\left(\nu_{CN} + \nu_{CC}\right)}} \tag{6.2}$$

where ν_{IJ} is the fraction of the IJ occurrences. ν_{NC} does not appear in the above expression since it is not independent of the other two fractions, being valid the relation $\nu_{CC} + \nu_{NC} + \nu_{CN} = 1$.

6.1.1 Firmware implementation

The monitoring of spillover would be practically impossible by means of the HLT1, since during physics data-taking operations data are saved in the turbo mode, not storing most of the raw data. Raw data of consecutive events on the same region of the sensor are thus not directly accessible - if not accidentally - so monitoring the spillover effect with the baseline sequence of the HLT1 is not feasible. This is instead possible at the readout level. In fact the accessibility of all cluster data of each event at the readout level can be fully exploit to perform the spillover monitoring task directly on the FPGAs of the readout boards of the VELO in a simple, transparent and continuous way.

The formula 6.2 is better suited for implementation on FPGA, given the absence of any multiplication term (no DSP is needed), thus saving resource usage, and the accumulation task for the desired occurrences is rather simple. In Figure 6.2 the implementation of the spillover counters is shown. As in the case of the luminosity register (Chapter 4), the firmware runs on an external clock operated at 250 MHz. Each component operates in pipeline, and just as in the luminosity register firmware pipelined delays are employed to assure data synchronization at each step of the data processing.

Cluster data coming in 256-bit packages (8×32-bit cluster words, `ST_SOURCE_DATA` in Figure 6.2) are selected by a set of comparators. They check in 8 parallel streams whether input clusters reside within an online-defined selection region on the sensor. The selection region is defined via ECS: a 32-bit signal (`ECS_WDATA`) is sent in input to the spillover register, reserving 19 bits to encode the selection region. A decoder splits up the relevant bits, generating a sensor ID flag, an 8-bit signal encoding the sensor row and a 10-bit signal for the sensor column number. Clusters are selected if their centroids reside in a 5-pix × 5-pix region whose bottom-left corner is given by the ECS input data.

Once clusters are selected, an 8-bit signal is sent to a multiple-input OR gate, enabled when incoming clusters are valid and reset at the beginning of each different bunch-crossing. The aim of this component is to establish the presence of a cluster within the selection region for each event. An OR gate is needed both to check this for clusters coming from the same frame package and from different frames but belonging to the same event. The latter task is fulfilled sending the output of the gate back to it in input, allowing to check the occupancy of the selection region over the whole event.

At the end of each event the OR gate output is sent to a 2-bit *serial in - parallel out* shift register, whose clock is driven by the `EOP` signal. Every time an end-of-package signal is received, the register first bit is shifted in place of the second one, discarding its old value. The value of the input signal is then registered in the first bit location. In this way, the occupancies of the selection region in consecutive events are coupled together and can be used to accumu-

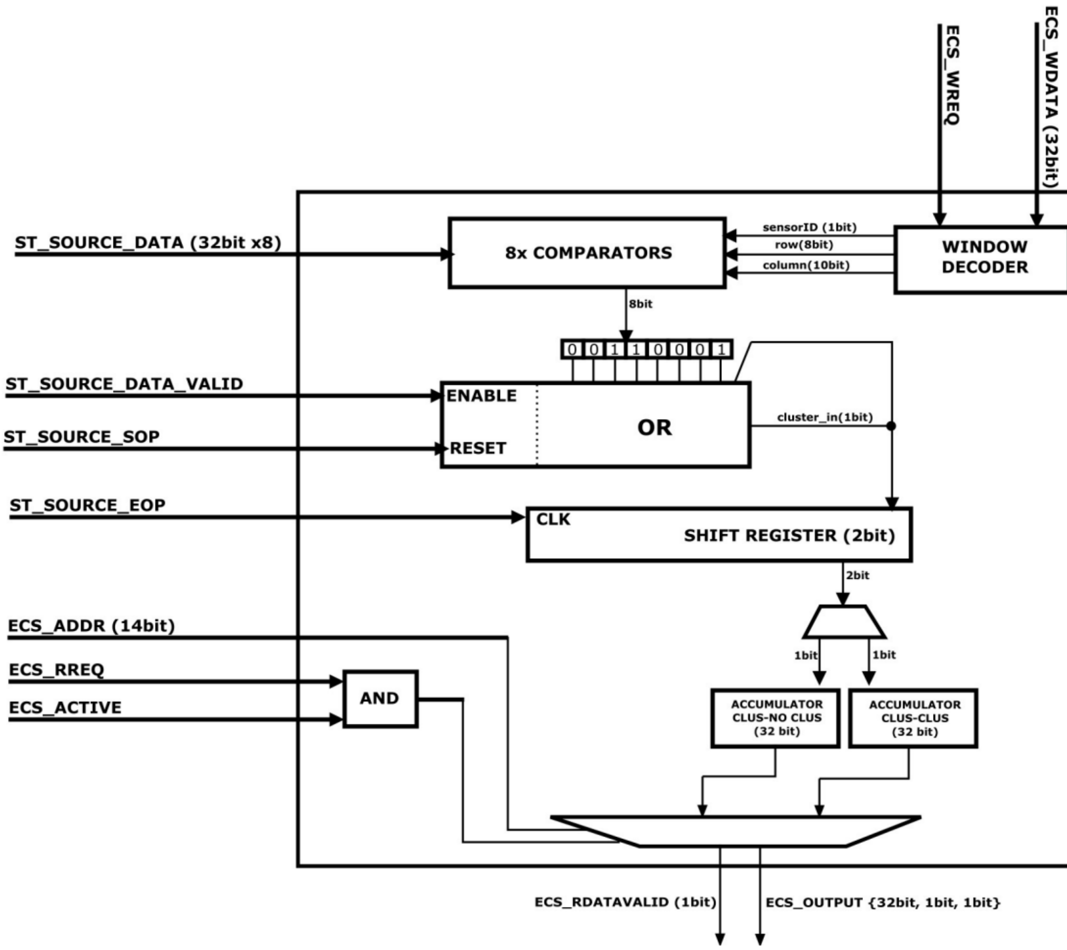

Figure 6.2: Structure of the spillover register firmware.

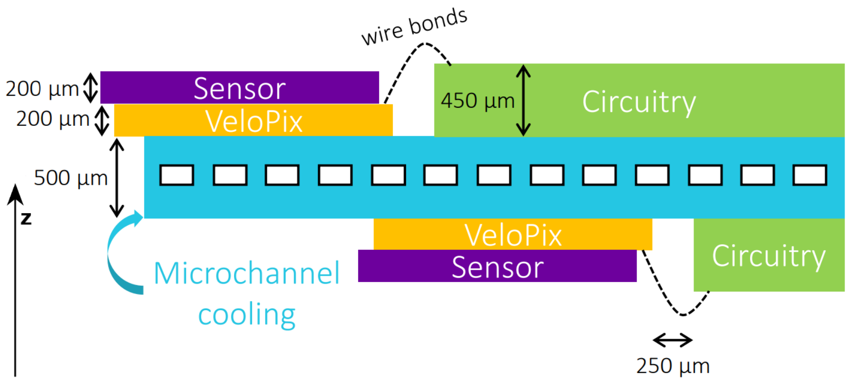

Figure 6.3: VELO sensor placement on the substrate. The two sensors forming a VELO tile are overlapped by two pixels.

late correlation statistics. Depending on the value of the shift register content, accumulator of `clus- no clus` (register content = ’10’) and accumulator of `clus-clus` (content = ’11’) are increased by one. The accumulation occurs for a fixed number of event. Then, once the required number of events is reached, the counting process is disabled.

The value of the defined accumulators is sent to the output depending on ECS requests, with a mechanism similar to the one implemented for the luminosity register reported in Section 4.1.4. A multiplexing stage selects which of the counters to output depending on the `ECS_ADDR` signal, when the read-request and active signal are both HIGH. A valid signal is raised when the accumulation process is disabled, otherwise it remain in LOW state.

6.2 Efficiency monitor

Using the same approach of the spillover monitoring system, one can - at least in principle - implement a firmware to monitor the detection efficiency of the sensors. In fact, one may exploit the overlap regions of the sensors in the same tile to infere the efficiency by looking for the coincidences of clusters in the same physical region of the tile in the same event, but on the two different sensors.

The overlap between the sensors is shown in Figure 6.3. Sensors are $900\,\mu m$ distant in the z coordinate, while the overlap consists of 2 pixels. Depending on the inclination of the particle crossing the sensors, there can be hits on both front and rear sensors (lower and upper sensors respectively in Figure 6.3, considering stations downstream of the interaction point). The coincidence of hits on the two sensors depends on the geometrical acceptance $\mathcal{A}$ of the overlap region. Defining a small selection region on the front sensor (for instance 2×2 pixels wide on the edge of the overlap region) and a larger one onto the rear sensor (*e.g.* 5×5 pixels wide region always on the edge of the overlap region)[2], the detection efficiency of the rear sensor is given by

$$\varepsilon_{rear} = \frac{N_{\text{double}}}{N_{\text{front}}}\mathcal{A} \tag{6.3}$$

where N_{double} is the number of coincident hits on the rear and front selection region and N_{front} is the total number of hits on the front sensors. A similar argument is valid to infere the detection efficiency of the front sensor, but inverting the dimensions of the selection regions.

The acceptance $\mathcal{A}$ depends on how the selection regions on both sensors are defined and can only be evaluated recurring to detailed simulations, depending necessarily on the nominal spatial position of the luminous region. These factors are complex to account for during the realistic data-taking. If one assumes the acceptance as a fixed (but unknown) quantity, Eq. 6.3 can still be used to infere the *relative* detection efficiency of the sensors, looking for any instability over time. The accumulation of cluster counts within each selection region on both sensor, and of their coincidence, can be performed on FPGAs using the same working principle of the spillover monitoring firmware.

The firmware I implemented to perform the efficiency monitoring is displayed in Figure 6.4. It is basically the same structure of Figure 6.2 with few modifications:

- two blocks of comparators are used to verify the presence of a cluster within the selection region in the rear sensor (SENSOR 0) and in the front sensor (SENSOR 1);
- the serial in - parallel out shift register is replaced by a *parallel in - parallel out* shift register, since the occurrence of coincidences has to be checked during the same event but in two distinct selection regions, instead of the spillover case where coincidences are looked for in subsequent events but on the same region of the sensor.

The accumulated statistics are then used to compute Eq. 6.3, where N_{double} is the number of accumulated CLUS-CLUS coincidences, while N_{front} is given by the sum of the accumulator of CLUS-NO CLUS events and CLUS-CLUS ones: $N_{\text{front}} = N_{\text{CC}} + N_{\text{CN}}$.

The firmware structures showed above are still in phase of development, being only tested in behavioral simulations. Nevertheless it is possible to conclude that their implementations on FPGA are certainly feasible. The actual possibility to measure the spillover and detection efficiency anyway needs to be evaluated by means of detailed simulations featuring the complete response of the detector and/or during data taking periods.

[2]an optimization of the selection region width has not been performed.

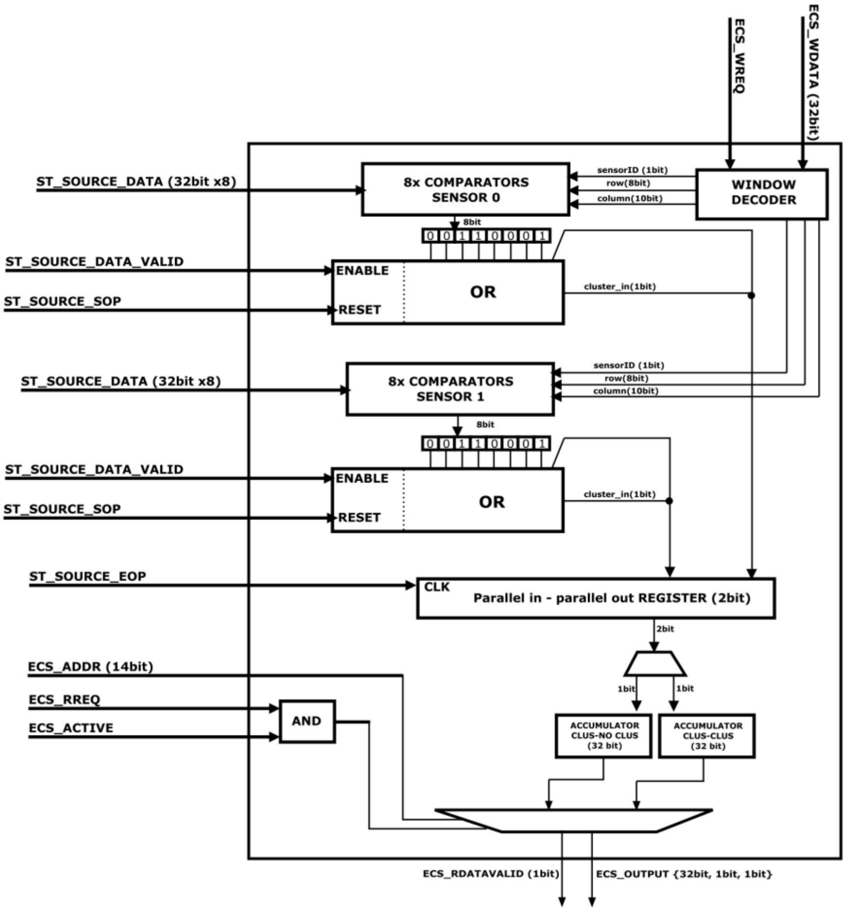

Figure 6.4: Structure of the efficiency register firmware. For the sake of simplicity, only half of the firmware is reported, showing the required components to monitor the efficiency of only one sensor.

7

Integration and commissioning in LHCb

The luminosity counters firmware described in Chapter 4 accumulates cluster-counts statistics. These high-rate statistics are used to reconstruct the instantaneous luminosity and the nominal position of the luminous region as discussed in Chapter 5.

The task of combining the accumulated statistics is performed at a higher level of the DAQ system. In fact, only hits on the same VELO module are directly accessible from one TELL40 board: there is no communication network between different VELO readout boards. Counters data can be instead combined and managed at ECS level, which has access to all TELL40 boards simultaneously. The combination of the counters data is thus performed by an ECS script, performing the reconstruction algorithm described in Section 5.6.

7.1 Integration into the LHCb online system

Luminosity counters data from each TELL40 need to be exposed to the ECS system. The ECS output interface of the luminosity register firmware is described in Section 4.1.4. It allows to output the value of a specific register, according to the ECS requests. Referring to counters registers only[1], the `ECS_ADDR` signal selects which of the counters to send in output. Each counter has a unique address defined in a *memory map* configuration file, that during the firmware compilation loads these addresses into the FPGA's memory.

The luminosity register firmware features a total of 26 average counters. The addresses associated to each counter are displayed in Table 7.1.

The LUMI_COUNTER_IJ_AVG and LUMI_COUNTER_IJ_HIST_K (where I,J=B,E and K={0,1, 2,3,4}) counters are based on the outer selection region. They are used to reconstruct the luminosity. The former are the total number of clusters in the outer selection region when the beam crossing is of the type IJ, used to employ the linearity method (Eq. 5.7). The latter (with K=0) are the number of events of IJ type with null counts. They are used to execute the log-0 method (Eq. 5.8).

LUMI_COUNTER_BB_AVG_INNER and LUMI_COUNTER_BB_AVG_INNER_PIX counters are based on the inner selection region and, in the latter one, with the additional requirement that the selected clusters must be made of more than one pixel. They are used to compute

[1]RAM counters have not yet been interfaced to the LHCb online system.

Counter name	Address	Counter name	Address
LUMI_COUNTER_BB_AVG	x500	LUMI_COUNTER_BB_HIST_2	x54C
LUMI_COUNTER_BB_AVG_INNER	x504	LUMI_COUNTER_BE_HIST_2	x550
LUMI_COUNTER_BB_AVG_INNER_PIX	x590	LUMI_COUNTER_EB_HIST_2	x55C
LUMI_COUNTER_BE_AVG	x508	LUMI_COUNTER_EE_HIST_2	x560
LUMI_COUNTER_EB_AVG	x50C	LUMI_COUNTER_BB_HIST_3	x564
LUMI_COUNTER_EE_AVG	x510	LUMI_COUNTER_BE_HIST_3	x568
LUMI_COUNTER_BB_HIST_0	x514	LUMI_COUNTER_EB_HIST_3	x56C
LUMI_COUNTER_BE_HIST_0	x518	LUMI_COUNTER_EE_HIST_3	x570
LUMI_COUNTER_EB_HIST_0	x51C	LUMI_COUNTER_BB_HIST_4	x574
LUMI_COUNTER_EE_HIST_0	x520	LUMI_COUNTER_BE_HIST_4	x578
LUMI_COUNTER_BB_HIST_1	x52C	LUMI_COUNTER_EB_HIST_4	x57C
LUMI_COUNTER_BE_HIST_1	x530	LUMI_COUNTER_EE_HIST_4	x580
LUMI_COUNTER_EB_HIST_1	x53C	LUMI_COUNTER_TIMESTAMP0	x588
LUMI_COUNTER_EE_HIST_1	x540	LUMI_COUNTER_TIMESTAMP1	x58c

Table 7.1: Map of the register addresses associated to the luminosity counters. (The leading **x** indicates hexadecimal format)

the nominal spatial position of the luminous region.

The 64-bit time integration counter value is stored in the LUMI_COUNTER_TIMESTAMP0 and LUMI_COUNTER_TIMESTAMP1 registers, each containing respectively the former 32 bits and latter 32 bits of the timestamp counter. Their combined value is used to compute the count rate of the average counters described above.

Upon assigning memory addresses to each counter, the ECS system becomes capable of sending a read request to the luminosity firmware and to receive back the value of the requested counter. However, in order to make this process user-independent and performed automatically and with regularity by the ECS system, a WinCC control-script is needed.

The counters addresses are added as *datapoints* to the VELO WinCC project. A WinCC script performs the polling of each of these datapoints once every 3 seconds. Once all the counters are read, a signal is sent (`ECS_WDATA` in Figure 4.1) to reset the counters value. Counters data are archived and the LHCb online monitoring system application *Monet* [60] displays them realizing trending plots accessible from any browser[61].

7.2 Commissioning

The luminosity firmware is now being commissioned in order to verify its correctness and functional properties. It has been tested on the *MiniDAQ* system [30], consisting of one VELO module read out by a TELL40 board, connected to a SOL40 board and a SODIN supervisor.

These constitute the minimal amount of hardware and readout components required to test in a realistic DAQ system new electronics systems and new firmware versions of the PCIe40 boards. The luminosity counters are tested by sending digital *testpulses* to the pixels constituting the selection regions. Testpulses commands are sent via SOL40 board to the VeloPix ASIC. These commands disable the analog comparator of the ASIC, forcing its digital electronics to set HIGH a specified pixel, that is read out as an active pixel. In this way it is possible to simulate hits on the sensor without having to use a physical particle source. The tests have shown that the accumulated counts are correct, thus validating the logic structure of the firmware. The collaboration is now pushing to have as many luminosity counters as possible by the end of October 2022. In this way it would be possible to perform dedicated data taking sessions to verify the linearity properties and cross-consistency checks of the several systems measuring the luminosity, and lastly to calibrate them so to be ready for the Run-3 physics data-taking periods. Efforts are ongoing to perform as soon as possible tests of the functionality of the luminosity counters firmware loaded and executed in the actual DAQ chain of the experiment. The next section describes the first results on real data.

7.3 Preliminary results on real data

An example of trending plot implemented in Monet is shown in Figure 7.1, displaying the timestamp counter value read out by the ECS as a function of the daytime. The timestamp value is expected to be nearly constant, with a mean value of $\sim 7.5 \times 10^8$ counts, corresponding to ~ 3 s.

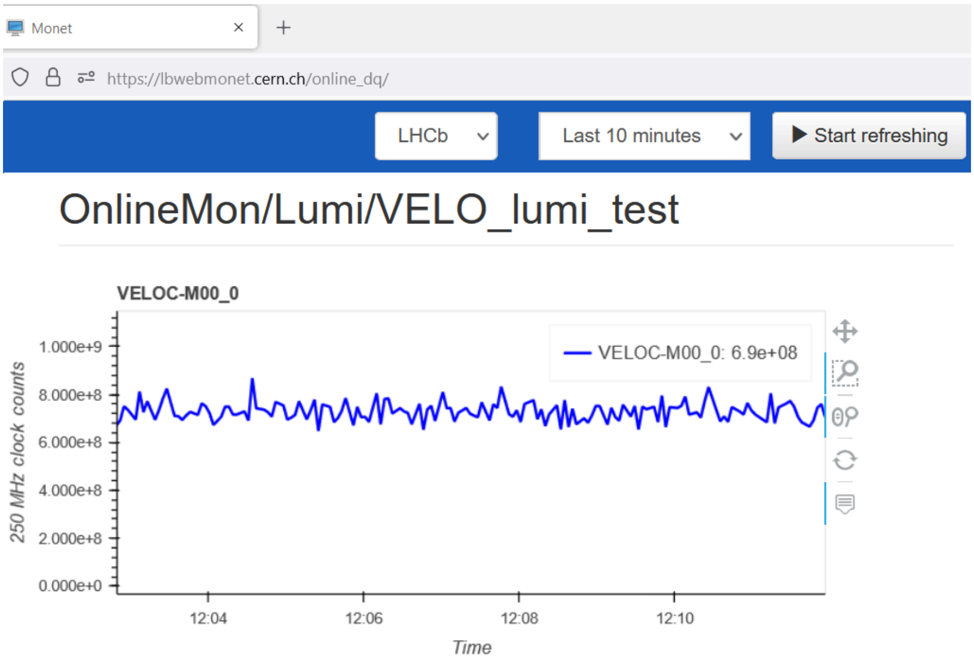

Figure 7.1: Trending plot of the timestamp counter of the luminosity register firmware loaded in the module 0 (M00_0), on the C-side of the VELO (VELOC). Data taken during the LHC stable beam operation of 02/10/2022.

Figures 7.2-7.3 show the count distributions per station of bb and ee counters from outer regions, implemented in sensors 0 and 1 of the VELO. These data have been taken during a *stable beams* LHC fill at 17:18:20 of 01/10/2022, with the VELO detector in *open* configuration. The shape for bb count distributions is remarkably similar to what expected from simulations (Figures 5.8 and 5.15), giving support to the correctness of the luminosity counters implementation. The background counts distribution (ee counters) is almost flat, indicating that it is independent of the beams, as expected. Its lower level is a good indication that the pattern seen in the bb case is not dominated by simple noise (possibly from the detector itself), and at the same time confirms that the firmware is correctly distinguishing between different types of crossings.

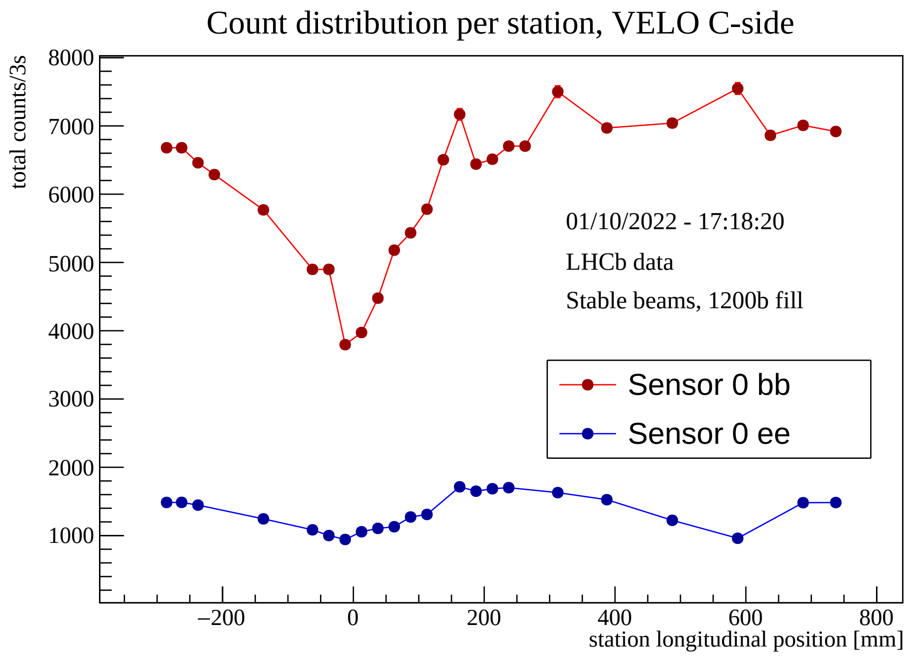

Figure 7.2: Outer *bb* and *ee* counters values per station using real data taken at 17:18:20 of 01/10/2022, during a stable beams LHC run with 1200 bunches fill. The outer counters here shown are implemented in sensors 0 of VELO C-side modules.

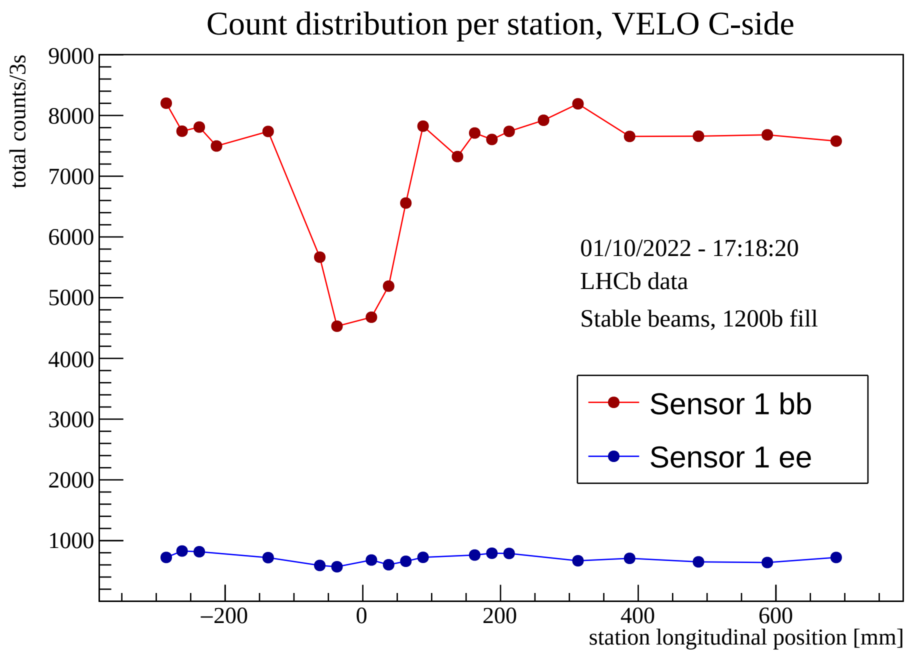

Figure 7.3: Outer *bb* and *ee* counters values per station using real data taken at 17:18:20 of 01/10/2022, during a stable beams LHC run with 1200 bunches fill. The outer counters here shown are implemented in sensors 1 pf VELO C-side modules.

These results are a first evidence that the firmware I designed is capable of operating within the actual LHCb data acquisition system, and that the simulation studies I carried out in Chapter 5 are consistent with the observed data. There are thus strong hopes that the luminosity reconstruction algorithm I implemented will actually work in data taking conditions, accounting also for the correction terms provided by the knowledge of the position of the luminous region. Ongoing studies about the actual possibility to perform a detector monitoring algorithm at the readout level will indeed show whether it is feasible to further correct the luminosity estimate accounting for detector-related effects.

Further efforts are being driven to fully integrate the system I implemented within the LHCb online system. Tests in real data-taking conditions will provide the ultimate probe to assess the feasibility of this innovative approach to measure the instantaneous luminosity.

These studies will eventually show the benefits of the FPGA-embedded real-time reconstruction performed at the earliest stages of the data-acquisition system, providing strong motivations to support it and paving the way to future developments exploiting this new paradigm.

A
Appendix

The properties of the luminosity counters object of this thesis are evaluated using detailed simulations. The extended results of the studies I performed to evaluate their linearity properties and sensitivity to luminous region displacements are reported in this appendix.

In order to optimize the choice of the selection region over the VELO sensors, I defined circular crown selection regions, ranging from $r = 7\,\text{mm}$ to $r = 31\,\text{mm}$. Linear fits are performed to the average number of clusters in function of the luminosity for each of these selection regions. Figure A.1 and A.2 show the *pvalues* of the fits, while Figure A.3 and A.4 show the value of the intercept at $\nu = 0$. These figures show that innermost regions, closest to the beampipe, have worse linear properties than the outer regions of the sensor.

Luminosity counters are defined on the outer regions of each sensor, namely at $r \sim 25.6\,\text{mm}$ from the interaction point. For each VELO station four (outer) selection regions are defined. Figure A.5 and A.6 show the proportionality properties of these counters.

For the former, linear fits are performed to the quantity $\sum_i \frac{N_i}{N_{evt}}$, where N_i is the number of count is selection region i and N_{evt} is the total number of events. This exploits the linear method to reconstruct the luminosity.

For the latter figure, the quantity $\sum_i \log\left(\frac{N(0)_i}{N_{evt}}\right)$ is used to evaluate the proportionality of the log-0 method.

I have carried out also studies on the effects of luminous region displacement on the luminosity estimator (Section 5.4). Figures A.7 and A.8 show how the count rate changes as the luminous region longitudinal position changes.

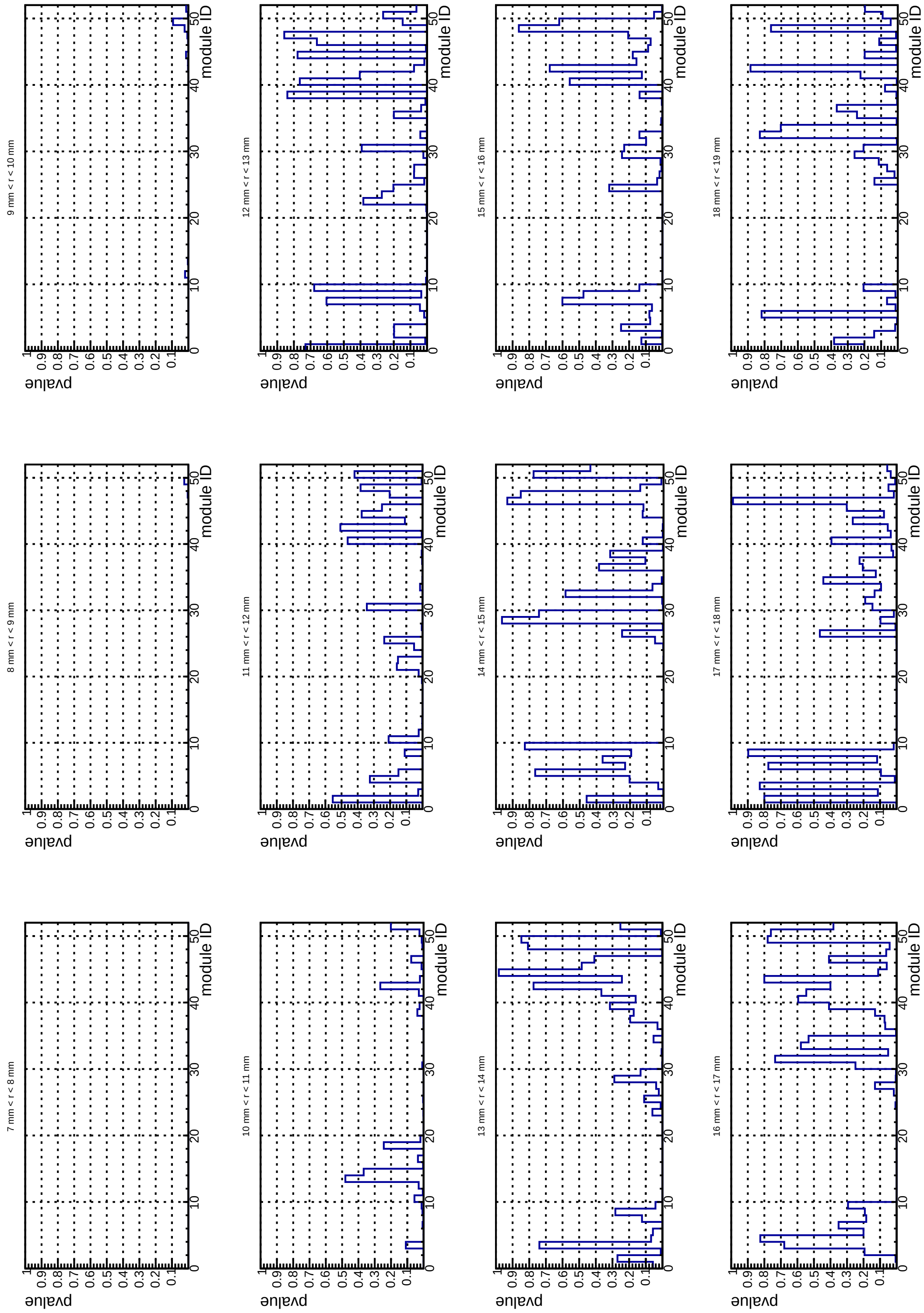

Figure A.1: Value of the *pvalue* of the linear fit to the mean number of clusters, in function of the number of the VELO module. The selection regions displayed range from 7 mm to 19 mm.

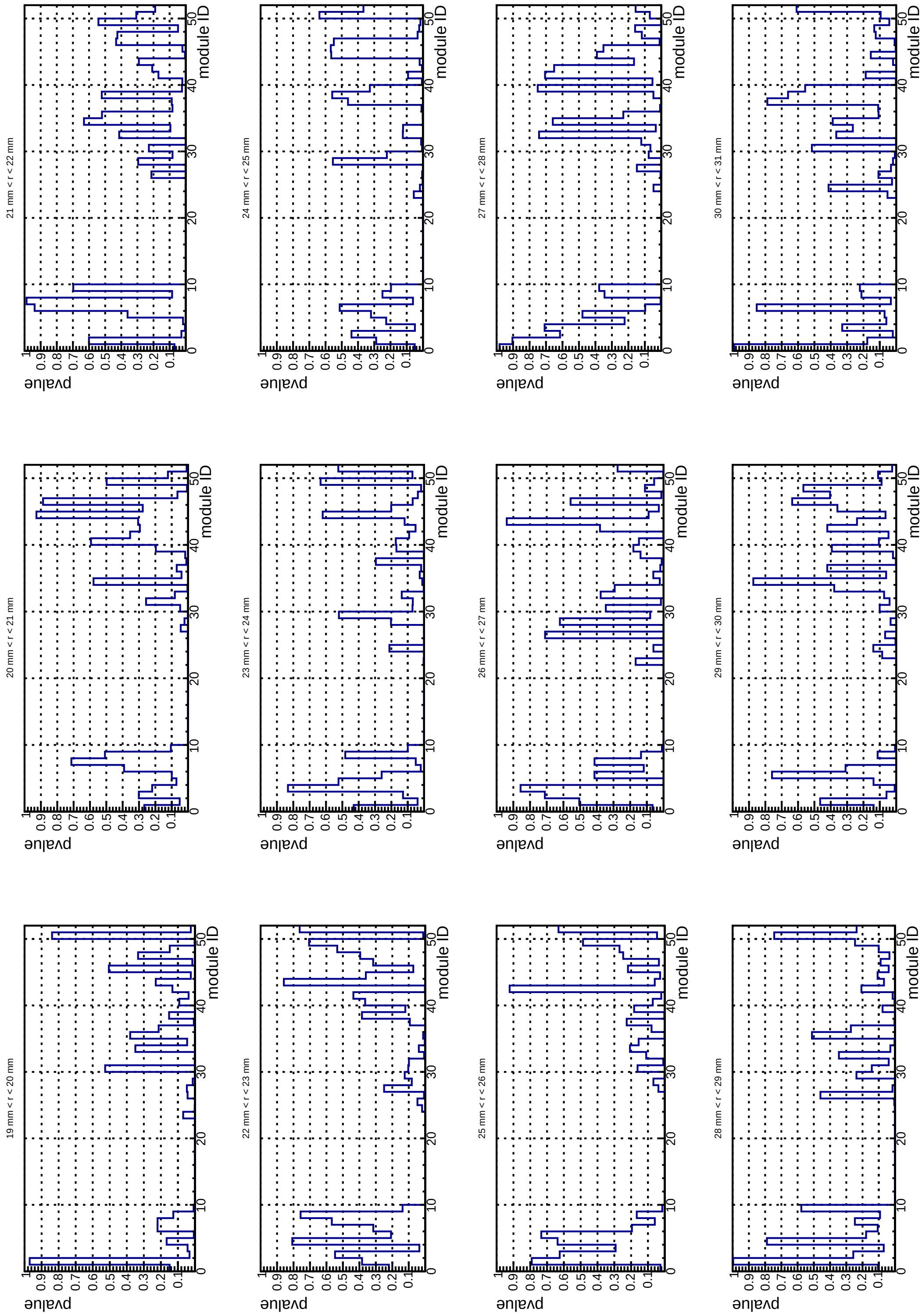

Figure A.2: Value of the *pvalue* of the linear fit to the mean number of clusters, in function of the number of the VELO module. The selection regions displayed range from 19 mm to 31 mm.

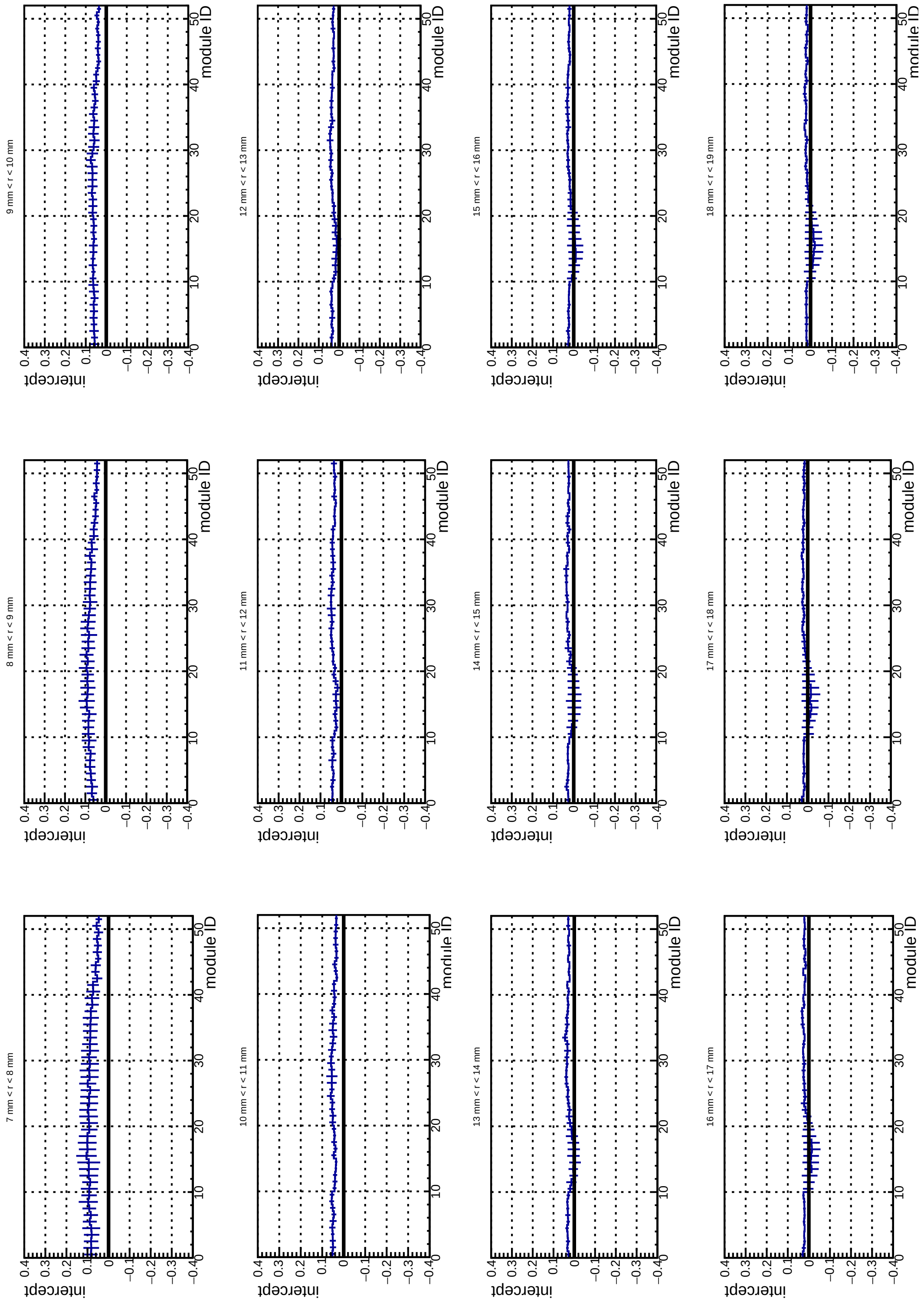

Figure A.3: Value of the intercept of the linear fit to the average number of clusters, in function of the number of the VELO module. The selection regions displayed range from 7 mm to 19 mm.

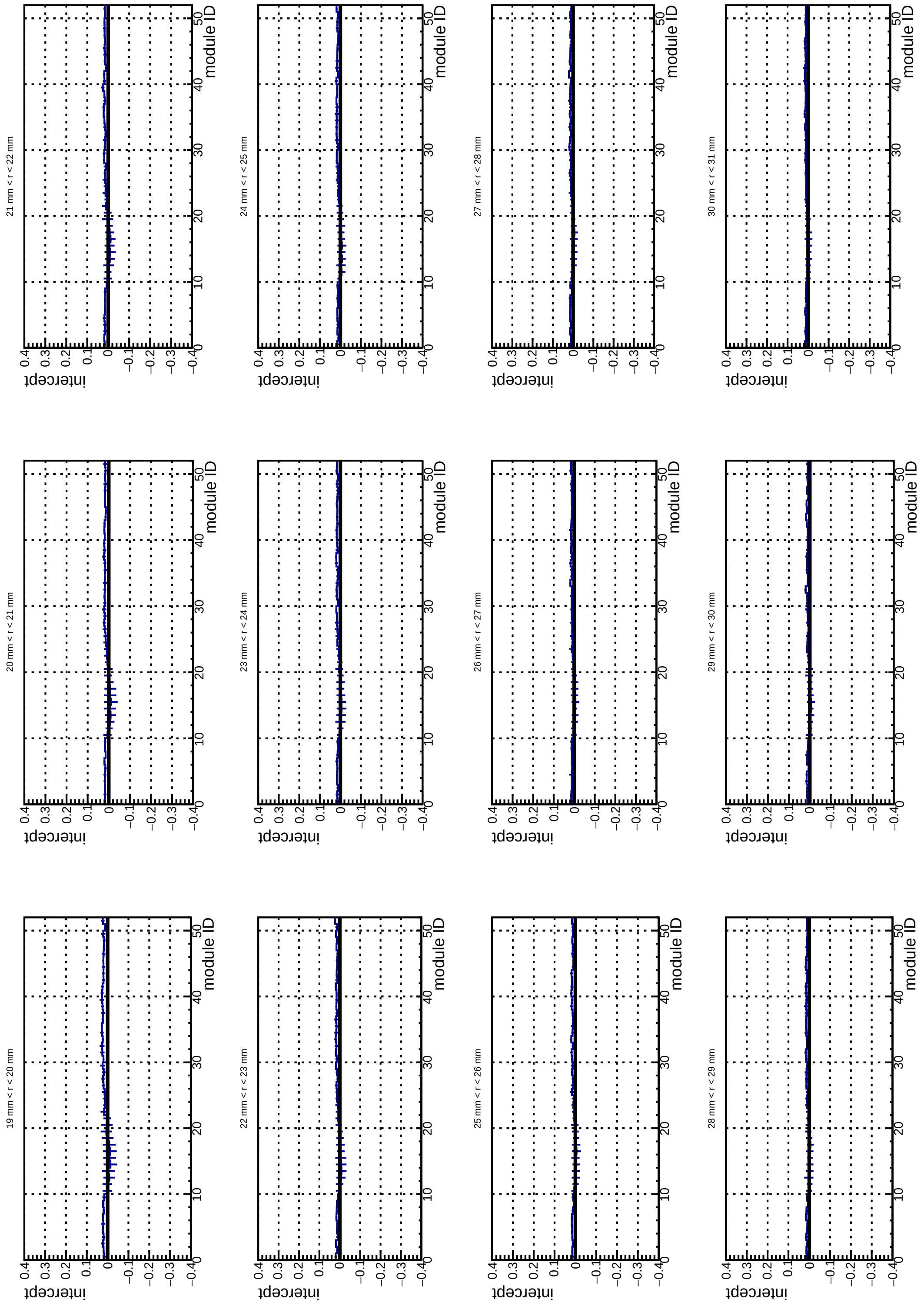

Figure A.4: Value of the intercept of the linear fit to the average number of clusters, in function of the number of the VELO module. The selection regions displayed range from 19 mm to 31 mm.

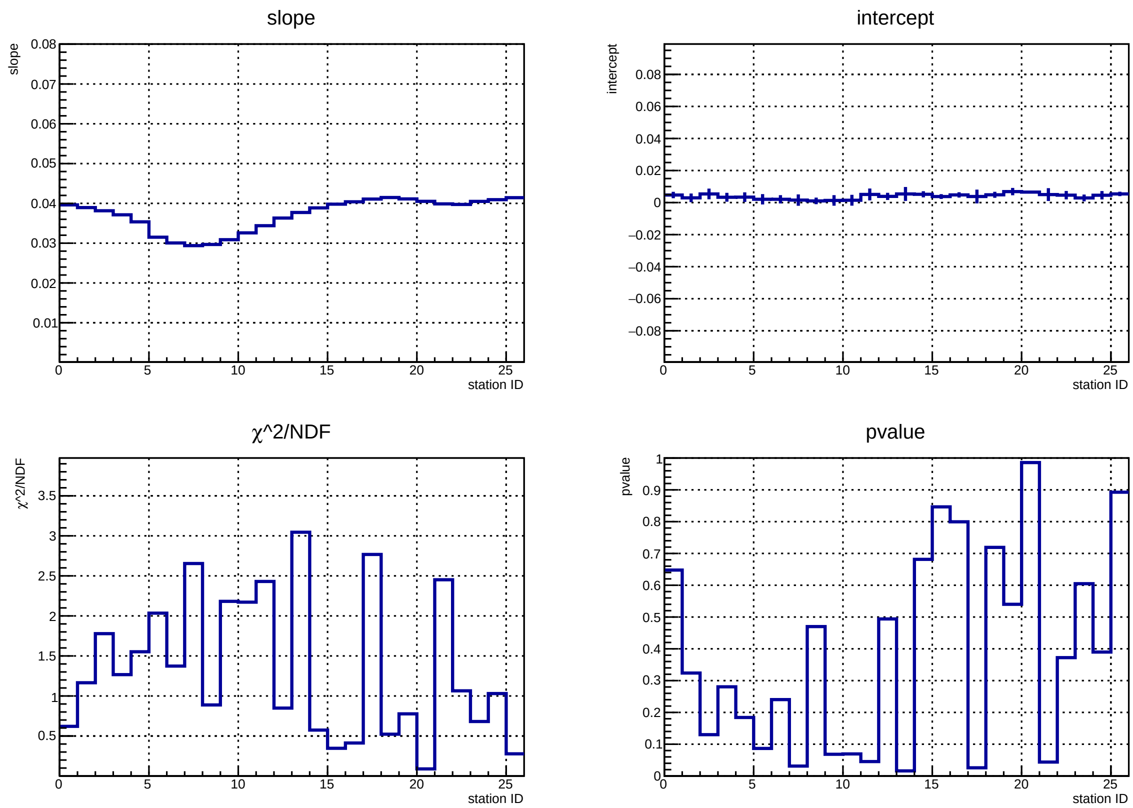

Figure A.5: Slope, intercept, χ^2 and *pvalues* for the linear fit to $\sum_i \frac{N_i}{N_{evt}}$ (linear method)

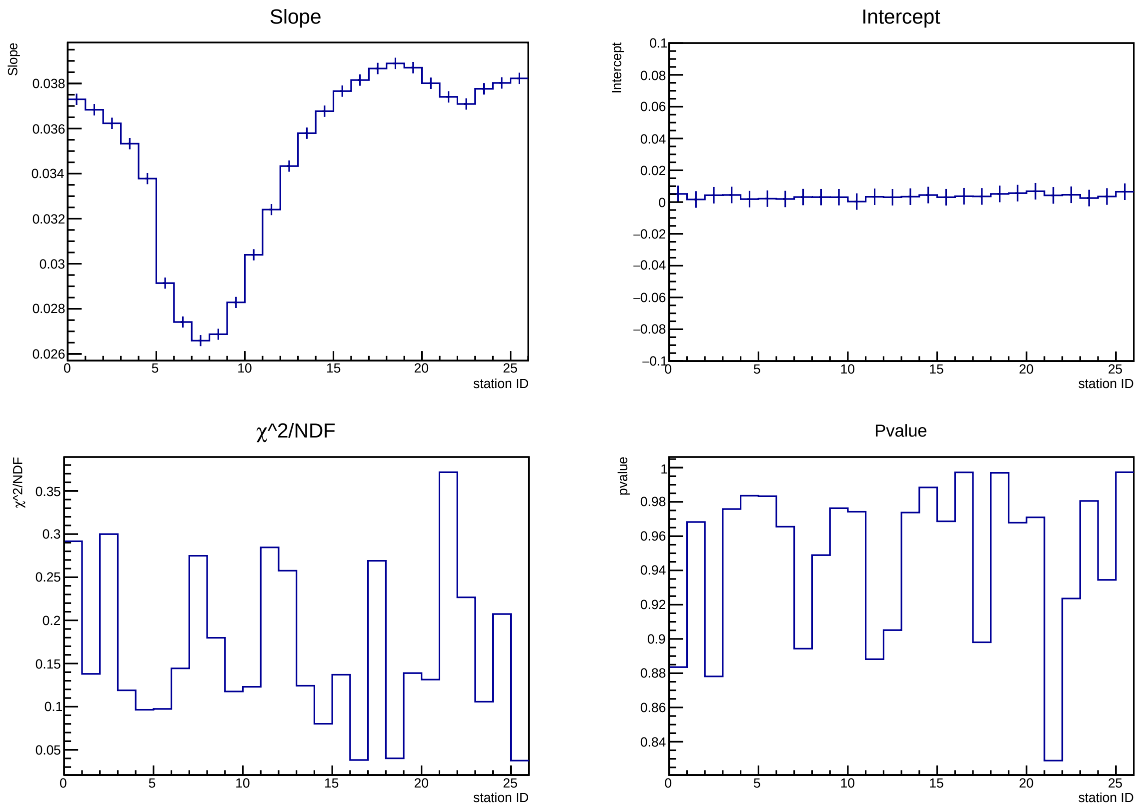

Figure A.6: Slope, intercept, χ^2 and *pvalues* for the linear fit to $\sum_i \log\left(\frac{N(0)_i}{N_{evt}}\right)$ (log0 method)

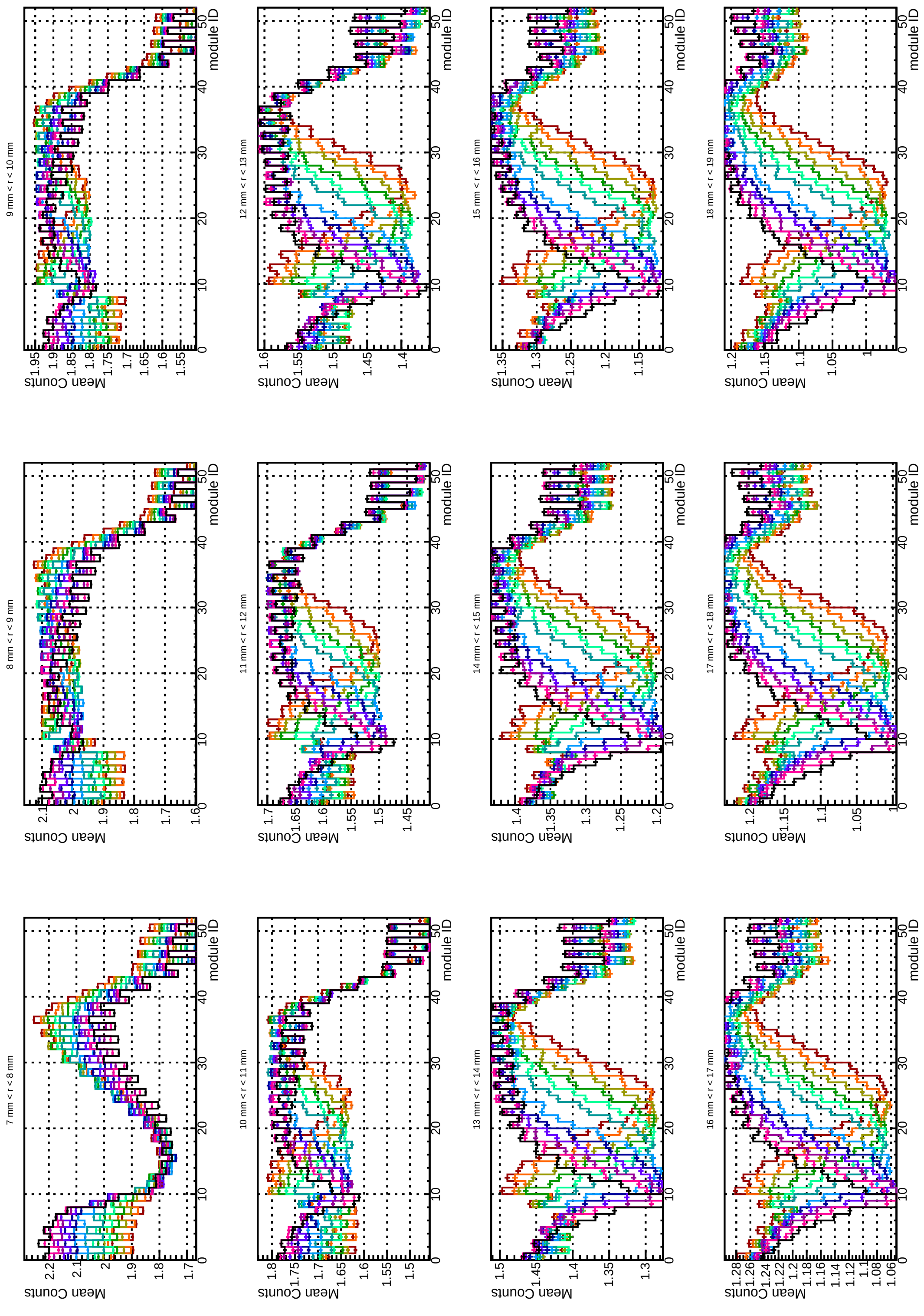

Figure A.7: Distribution of the cluster counts in circular crown regions as a function of the radial distance, ranging in $7\,\text{mm} < r < 19\,\text{mm}$. Each color represents the counts distribution for different value of the nominal z-position of the luminous region.

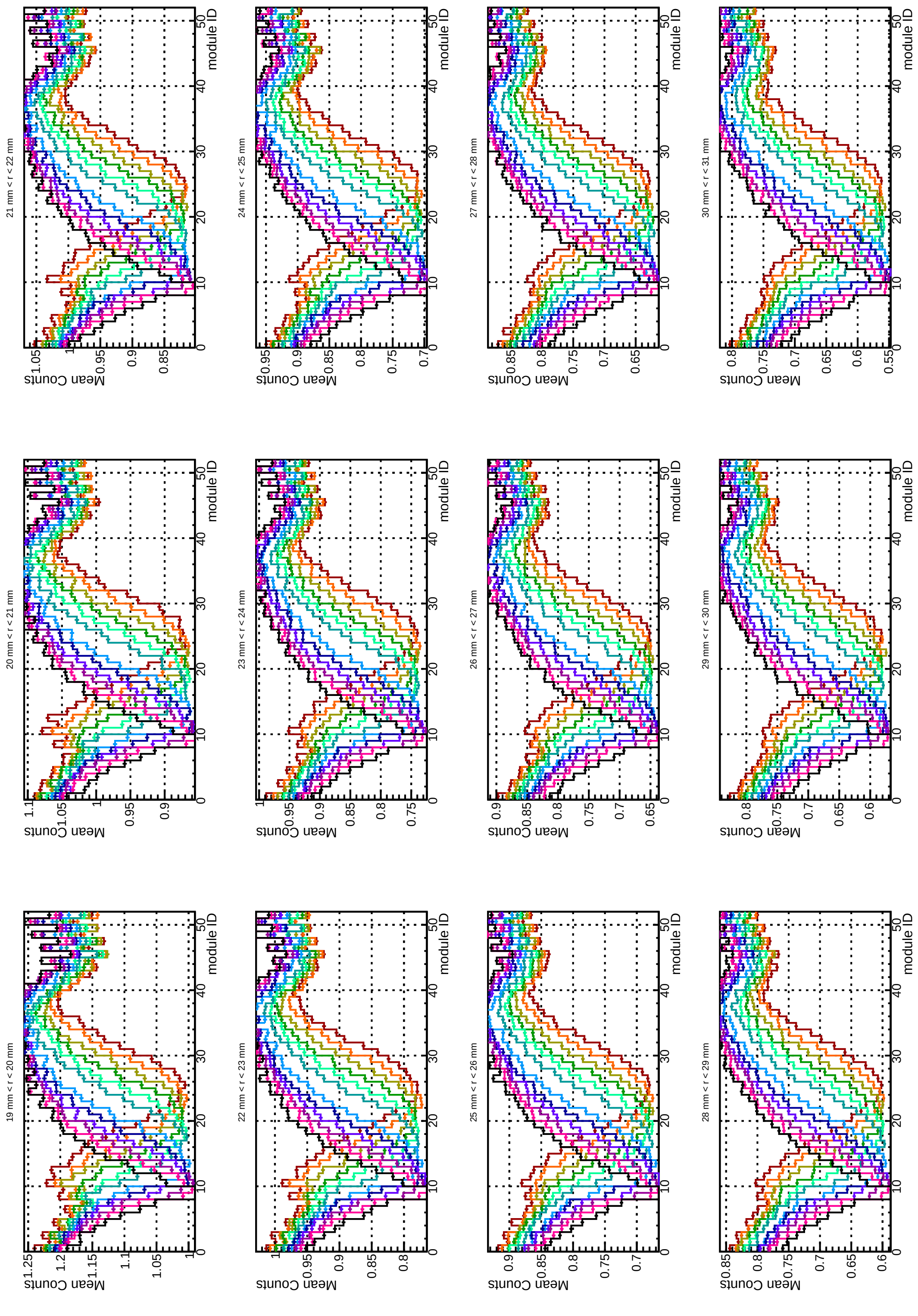

Figure A.8: Distribution of the cluster counts in circular crown regions as a function of the radial distance, ranging in 19 mm < r < 31 mm. Each color represents the counts distribution for different value of the nominal z-position of the luminous region.

Bibliography

[1] G. Aad, et al., "Measurements of Higgs boson production and couplings in diboson final states with the ATLAS detector at the LHC", *Physics Letters B* **2013**, *726*, 88–119, DOI https://doi.org/10.1016/j.physletb.2013.08.010, https://www.sciencedirect.com/science/article/pii/S0370269313006369.

[2] P. C. Bhat, "Observation of a Higgs-like Boson in CMS at the LHC", *Nuclear Physics B - Proceedings Supplements* **2013**, *234*, 16th International Conference in Quantum ChromoDynamics, 7–14, DOI https://doi.org/10.1016/j.nuclphysbps.2012.11.003, https://www.sciencedirect.com/science/article/pii/S092056321200494X.

[3] G. Isidori, Y. Nir, G. Perez, "Flavor Physics Constraints for Physics Beyond the Standard Model", *Annual Review of Nuclear and Particle Science* **2010**, *60*, 355–380, DOI 10.1146/annurev.nucl.012809.104534, https://doi.org/10.1146%2Fannurev.nucl.012809.104534.

[4] A. Sakharov, "Violation of CP invariance, C asymmetry, and barion asymmetry of the Universe", *Zhurnal Eksperimental'noi i Teoreticheskoi Fiziki* **1967**, *5*, 32–35.

[5] Y. G. Kolomensky, "Firs Measurement of the CP-violating asymmetries with BABAR", *International Journal of Modern Physics A* **2001**, *16*, 611–615, DOI 10.1142/s0217751x01007601, https://doi.org/10.1142%2Fs0217751x01007601.

[6] Belle Collaboration, H. Tajima, B CP-Violation, Mixing and Lifetime Results from Belle, **2001**, DOI 10.48550/ARXIV.HEP-EX/0105024, https://arxiv.org/abs/hep-ex/0105024.

[7] H.-Y. Cheng, CP Violation in $B^{\pm} \longrightarrow \rho^0\pi^{\pm}$ and $B^{\pm} \longrightarrow \sigma\pi^{\pm}$ Decays, **2020**, DOI 10.48550/ARXIV.2005.06080, https://arxiv.org/abs/2005.06080.

[8] R. Aaij, et al., ""Measurement of the $B_S^0 \longrightarrow \mu^+\mu^-$ branching fraction and search for $B^0 \longrightarrow \mu^+\mu^-$ decays at the LHCb experiment"", *Physical Review Letters* **2013**, *111*, DOI 10.1103/physrevlett.111.101805, https://doi.org/10.1103%5C%2Fphysrevlett.111.101805.

[9] Martha Hilton, "CP violation in charm with LHCb", *Proceedings of 7th Annual Conference on Large Hadron Collider Physics — PoS(LHCP2019)* **2019**.

[10] R. Aaij, et al., "Measurement of the CKM angle γ from a combination of LHCb results", *Journal of High Energy Physics* **2016**, *2016*, DOI 10.1007/jhep12(2016)087, https://doi.org/10.1007%5C%2Fjhep12%5C%282016%5C%29087.

[11] R. Aaij, et al., "Measurement of the CP-violating phase ϕ_s in $\overline{B}_s^0 \longrightarrow J/\psi\pi^+\pi^- decays$", *Physics Letters B* **2014**, *736*, 186–195, DOI 10.1016/j.physletb.2014.06.079, https://doi.org/10.1016%5C%2Fj.physletb.2014.06.079.

[12] A. Arbey, T. Hurth, F. Mahmoudi, D. M. Santos, S. Neshatpour, "Update on the $b \longrightarrow s$ anomalies", *Phys. Rev. D* **2019**, *100*, 7 pages, 3 figures, 3 tables. Addendum to Phys. Rev. D96 (2017) 095034 and Phys. Rev. D98 (2018) 095027, 015045. 6 p, DOI 10.1103/PhysRevD.100.015045, https://cds.cern.ch/record/2672224.

[13] B. Fang, "Recent LHCb results on heavy hadron spectroscopy", **2022**, https://cds.cern.ch/record/2826803.

[14] R. Aaij, et al., "Observation of $J/\psi p$ resonances consistent with pentaquark states in $\Lambda^0 b \longrightarrow J/\psi K^- p$0b→J/Kp decays", *Physical Review Letters* **2015**, *115*, DOI 10.1103/physrevlett.115.072001, https://doi.org/10.1103%2Fphysrevlett.115.072001.

[15] LHCb Collaboration, "Measurement of the Z boson production cross-section in proton-lead collisions at $\sqrt{s_{\rm NN}} = 8.16$ TeV", **2022**, https://cds.cern.ch/record/2810073.

[16] N. V. Raab, "Exclusive production of J/ψ and ϕ mesons at the LHCb experiment", **2022**, Presented 04 Apr 2022, https://cds.cern.ch/record/2815873.

[17] W. Duan, "Quarkonia production in ultraperipheral PbPb collisions at LHCb", **2022**, https://cds.cern.ch/record/2826531.

[18] LHCb Collaboration, Framework TDR for the LHCb Upgrade II, tech. rep., CERN, Geneva, **2021**, https://cds.cern.ch/record/2776420.

[19] LHCb Collaboration, Expression of Interest for a Phase-II LHCb Upgrade, tech. rep., CERN, Geneva, **2017**, https://cds.cern.ch/record/2244311.

[20] "https://web.infn.it/RETINA/index.php/en/".

[21] LHCb Collaboration, LHCb VELO Upgrade Technical Design Report, tech. rep., **2013**, https://cds.cern.ch/record/1624070.

[22] G. Aglieri Rinella, et al., "The NA62 GigaTracker", *Nuclear Instruments and Methods in Physics Research Section A* **2017**, *845*, Proceedings of the Vienna Conference on Instrumentation 2016, 147–149, DOI https://doi.org/10.1016/j.nima.2016.06.045, https://www.sciencedirect.com/science/article/pii/S0168900216305885.

[23] LHCb Collaboration, LHCb Tracker Upgrade Technical Design Report, tech. rep., **2014**, https://cds.cern.ch/record/1647400.

[24] LHCb Collaboration, LHCb PID Upgrade Technical Design Report, tech. rep., **2013**, https://cds.cern.ch/record/1624074.

[25] LHCb Collaboration, Letter of Intent for the LHCb Upgrade, tech. rep., CERN, Geneva, **2011**, http://cds.cern.ch/record/1333091.

[26] G. Gazzoni, "Real-time analysis from the trigger candidates and novel calibration strategy at the LHCb experiment", *PoS* **2018**, *EPS-HEP2017*, 495. 7 p, DOI 10.22323/1.314.0495, https://cds.cern.ch/record/2670650.

[27] LHCb Collaboration, Computing Model of the Upgrade LHCb experiment, tech. rep., CERN, Geneva, **2018**, DOI 10.17181/CERN.Q0P4.570N, https://cds.cern.ch/record/2319756.

[28] LHCb Collaboration, LHCb Trigger and Online Upgrade Technical Design Report, tech. rep., **2014**, https://cds.cern.ch/record/1701361.

[29] LHCb Collaboration, LHCb Upgrade GPU High Level Trigger Technical Design Report, tech. rep., CERN, Geneva, **2020**, DOI 10.17181/CERN.QDVA.5PIR, https://cds.cern.ch/record/2717938.

[30] L. Cardoso, et al., "LHCb MiniDAQ control system", *EPJ Web of Conferences* **2019**, *214*, 01005, DOI 10.1051/epjconf/201921401005.

[31] https://winccoa.com/.

[32] https://eda.sw.siemens.com/en-US/ic/modelsim/.

[33] The LHCb Collaboration, "Precision luminosity measurements at LHCb", *Journal of Instrumentation* **2014**, *9*, P12005–P12005, DOI 10.1088/1748-0221/9/12/p12005, https://doi.org/10.1088%5C%2F1748-0221%5C%2F9%5C%2F12%5C%2Fp12005.

[34] LHCb Collaboration, LHCb PLUME: Probe for LUminosity MEasurement, tech. rep., CERN, Geneva, **2021**, DOI 10.17181/CERN.WLU0.M37F, https://cds.cern.ch/record/2750034.

[35] P. Albicocco, et al., "A Method Based on Muon System to Monitor LHCb Luminosity", *Symmetry* **2022**, *14*, 860. 13 p, DOI 10.3390/sym14050860, https://cds.cern.ch/record/2812716.

[36] C. Ilgner, et al., The Beam Conditions Monitor of the LHCb Experiment, tech. rep., **2010**, https://cds.cern.ch/record/1233669.

[37] L. Ristori, "An artificial retina for fast track finding", *Nuclear Instruments & Methods in Physics Research Section A-accelerators Spectrometers Detectors and Associated Equipment* **2000**, *453*, 425–429.

[38] F. Lazzari, et al., "FPGA-based real-time data processing for accelerating reconstruction at LHCb", **2022**, 8 p, DOI 10.1088/1748-0221/17/04/C04011, https://cds.cern.ch/record/2801062.

[39] P. V. Hough, "METHOD AND MEANS FOR RECOGNIZING COMPLEX PATTERNS", **1962**.

[40] L. Giambastiani, "A 2D FPGA-based clustering algorithm for the LHCb silicon pixel detector running at 30 MHz", **2020**, https://cds.cern.ch/record/2725831.

[41] https://www.intel.com/content/www/us/en/products/details/fpga/development-tools/quartus-prime.html.

[42] P. Grafström, W. Kozanecki, "Luminosity determination at proton colliders", *Prog. Part. Nucl. Phys.* **2015**, *81*, 97–148. 52 p, DOI 10.1016/j.ppnp.2014.11.002, https://cds.cern.ch/record/2153734.

[43] M. Ferro-Luzzi, Luminosity and luminous region shape for pure Gaussian bunches, tech. rep., CERN, Geneva, **2012**, https://cds.cern.ch/record/1481670.

[44] G. Coombs, "Beam-Gas Imaging at the LHCb Experiment", **2021**, Presented 10 Mar 2021, https://cds.cern.ch/record/2767576.

[45] R. Aaij, et al, "Measurement of forward J/ψ production cross-sections in pp collisions at $\sqrt{s}$ =13", *Journal of High Energy Physics* **2015**, *2015*, DOI 10.1007/jhep10(2015)172, https://doi.org/10.1007%5C%2Fjhep10%282015%29172.

[46] W. Herr, B. Muratori, "Concept of luminosity", **2006**, DOI 10.5170/CERN-2006-002.361, https://cds.cern.ch/record/941318.

[47] V. Balagura, "Van der Meer scan luminosity measurement and beam–beam correction", *Eur. Phys. J. C* **2021**, *81*, 26, https://doi.org/10.1140/epjc/s10052-021-08837-y.

[48] S. van der Meer, Calibration of the effective beam height in the ISR, tech. rep., CERN, Geneva, **1968**, https://cds.cern.ch/record/296752.

[49] T. Sjöstrand, S. Mrenna, P. Skands, "PYTHIA 6.4 physics and manual", *Journal of High Energy Physics* **2006**, *2006*, 026–026, DOI 10.1088/1126-6708/2006/05/026.

[50] D. J. Lange, "The EvtGen particle decay simulation package", *Nuclear Instruments and Methods in Physics Research Section A* **2001**, *462*, BEAUTY2000, Proceedings of the 7th Int. Conf. on B-Physics at Hadron Machines, 152–155, DOI https://doi.org/10.1016/S0168-9002(01)00089-4.

[51] S. Agostinelli, et al., "Geant4—a simulation toolkit", *Nuclear Instruments and Methods in Physics Research Section A* **2003**, *506*, 250–303, DOI https://doi.org/10.1016/S0168-9002(03)01368-8.

[52] M. Clemencic, et al., "The LHCb Simulation Application, Gauss: Design, Evolution and Experience", *Journal of Physics: Conference Series* **2011**, *331*, 032023, DOI 10.1088/1742-6596/331/3/032023.

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

[54] M. Vesterinen, Considerations on the LHCb dipole magnet polarity reversal, tech. rep., On behalf of the LHCb collaboration, CERN, Geneva, **2014**, https://cds.cern.ch/record/1642153.

[55] L. Lyons, D. Gibaut, P. Clifford, "How to combine correlated estimates of a single physical quantity", *Nuclear Instruments and Methods in Physics Research Section A* **1988**, *270*, 110–117, DOI https://doi.org/10.1016/0168-9002(88)90018-6.

[56] R. H. Byrd, M. E. Hribar, J. Nocedal, "An Interior Point Algorithm for Large-Scale Nonlinear Programming", *SIAM J. Optim.* **1999**, *9*, 877–900.

[57] F. James, "Statistical Methods in Experimental Physics: 2nd Edition", **2006**, DOI 10.1142/6096.

[58] T. Poikela, et al., "VeloPix: the pixel ASIC for the LHCb upgrade", *JINST* **2015**, *10*, C01057, DOI 10.1088/1748-0221/10/01/C01057, https://cds.cern.ch/record/2189759.

[59] V. Battista, et al., A study of spillover clusters and ghost tracks in the Silicon Tracker with 25 ns bunch spacing, tech. rep., CERN, Geneva, **2016**, https://cds.cern.ch/record/2130806.

[60] M. Adinolfi, F. Archilli, W. Baldini, A. Baranov, D. Derkach, A. Panin, A. Pearce, A. Ustyuzhanin, "LHCb data quality monitoring", *J. Phys.: Conf. Ser.* **2017**, *898*, 092027. 5 p, DOI 10.1088/1742-6596/898/9/092027, https://cds.cern.ch/record/2298467.

[61] https://lbwebmonet.cern.ch/.