UNIVERSITÁ DEGLI STUDI DI MILANO

Scuola di Dottorato in Fisica, Astrofisica e Fisica Applicata

Dipartimento di Fisica

Corso di Dottorato in Fisica, Astrofisica e Fisica Applicata

Ciclo XXX

Demonstrator System for the Phase-I Upgrade of the Trigger Readout Electronics of the ATLAS Liquid Argon Calorimeters at the LHC

Settore Scientifico Disciplinare FIS/01 - FIS/04

Supervisore: Professor Valentino Liberali

Co-supervisore: Dottor Francesco Tartarelli

Coordinatore: Professor Francesco RAGUSA

Tesi di Dottorato di:
Alessandra Camplani

Anno Accademico 2016/2017

Commission of the final examination:

External Referees:
Marco Lisboa Leite (Universidade de Sao Paulo)
Monica Alderighi (IASF - INAF, Milano)

External Members:
Mike Wirthlin (Brigham Young University of Utah)
Yuji Enari (University of Tokio)

Internal Member:
Massimo Lazzaroni (Università degli Studi di Milano)

Final examination:

January 19th, 2018

Università degli Studi di Milano, Dipartimento di Fisica, Milano, Italy

To my family.

Cover illustration:

Geneva Lake - Jet d'eau
Max Emde
March 19, 2017

MIUR subjects:

FIS/01
FIS/04

Contents

Abstract

General introduction

I started my Ph.D. at the Physics Department of the Università degli Studi di Milano in November 2014. I carried out my research activity within the ATLAS experiment at the Large Hadron Collider (LHC) at CERN, mainly focusing on the upgrade of the ATLAS Liquid Argon (LAr) electromagnetic calorimeter Phase-I trigger electronics. The main topic of my doctoral project is the implementation of VHDL firmware for the Field Programmable Gate Arrays (FPGAs) of the new calorimeter trigger electronics.

I spent the first of the three Ph.D. years working in Milano and the last two years at CERN, supported by a contract awarded from the ATLAS LAr calorimeter group. During the three years of activity, I contributed to the development and maintenance of the FPGA readout firmware for the LAr Phase-I demonstrator system, set up and installed in the ATLAS detector during summer 2014. The purpose of the system is to validate the energy reconstruction and collect real collision data using a pre-prototype of the future front-end and back-end electronics.

In addition, I joined the group working on the firmware development for the FPGAs of the new Phase-I back-end boards. I was asked to be in charge of the firmware module for decoding the Timing Trigger and Control (TTC) signals coming from the LHC central trigger processor.

Thesis introduction

This Ph.D. thesis is inserted in the context of the ATLAS experiment at the LHC. The first four chapters are mainly dedicated to give a general overview of the experiment and the system in which this project is inserted.

In the coming years, the LHC will undergo a series of upgrades leading to luminosities well beyond the nominal value, with the aim of extending and improving the physics program of its experiments. Meeting this challenge will require significant detector optimizations, changes and improvements. The ATLAS experiment at the LHC is a multi-purpose particle detector with a forward-backward symmetric cylindrical geometry, a fully azimuthal coverage and a rapidity coverage up to 4.9. The purpose of the ATLAS LAr calorimeter upgrade is to provide higher-granularity, higher-resolution and longitudinal shower information from the calorimeter to the Level-1 trigger processors to improve the trigger energy resolution and the event selection efficiency. The existing calorimeter trigger readout is based on "Trigger Towers" which sum the energy

deposited along the longitudinal layers in an area of $\Delta\eta \times \Delta\phi = 0.1 \times 0.1$[1]. The towers will evolve in a new finer scheme, called "Super Cells", which for example in the barrel "middle" sampling (where most of the energy is collected), will have a granularity of $\Delta\eta \times \Delta\phi = 0.025 \times 0.1$. In this perspective both LAr front-end and back-end electronics will be improved and extended.

The key electronic board of the upgrade is the LAr Trigger Digitizer Board (LTDB) which will receive, digitize and send to the new back-end electronics the Super Cell signals. The back-end system, called LAr Digital Processing System (LDPS), will reconstruct the transverse energy of each Super Cell and transmit the result to the Level-1 calorimeter trigger system. The LDPS consists of 32 ATCA Carrier boards, each equipped with four Advanced Mezzanine Cards (AMCs) called LATOME. The LATOME is built around a high-performance and state-of-the-art FPGA, an ARRIA-10 from Intel FPGA (formerly ALTERA).

Currently, a demonstrator of the Phase-I upgrade electronics is installed in the ATLAS detector and is collecting physics data from the calorimeter barrel. This is a pre-prototype of the future system covering 1/32 of the barrel region. The goal of the demonstrator is to gain experience on this system with real collision data and validate the LDPS algorithms, including the one for the energy reconstruction of the calorimeter signal. Two LTDB pre-prototypes are installed on a LAr front-end crate. The digitized calorimeter data are transmitted along optical links to three back-end pre-prototype boards called ABBA (ATCA test Board for Baseline Acquisition). ABBA makes use of three Stratix-IV Intel FPGAs for the data readout.

In addition, a test system for the LDPS is installed in the LAr Electronics Maintenance Facility (EMF) laboratory: the system is made of three Carrier boards and two LATOMEs. The purpose is to confirm all the back-end functionalities and ensure the stability of the system before the mass production and the installation in the detector. A new demonstrator system, with both LTDB and LDPS, will be placed in the ATLAS detector in January 2018.

LAr Phase-I demonstrator

Chapter 5 gives an overview of the work done in the framework of the demonstrator system for the Phase-I upgrade.

During the first Ph.D. year, I completed my qualification task to become an ATLAS author, implementing an I2C communication protocol to generate specific clock frequencies from crystal oscillators mounted on all ABBA boards. When installed in the detector, ABBA needed two loading steps to work properly. In the first step, the Altera embedded NIOS II processor firmware to set the reference clock frequency via I2C communication with a crystal oscillator, was loaded. Then, the main firmware was loaded in the ABBA board. The aim of this project was to integrate the frequency settings in the main firmware, having only one loading step.

The ABBA firmware was affected by data corruption during the readout of the calorimeter data. During the second year of the Ph.D., I gained a deep knowledge of the firmware, that allowed me to resolve the existing issues and improve the data quality.

[1] ATLAS uses a right-handed coordinate system with its origin at the nominal interaction point (IP) in the centre of the detector and the z-axis along the beam pipe. The x-axis points from the IP to the centre of the LHC ring, and the y-axis points upwards. Cylindrical coordinates (r, ϕ) are used in the transverse plane, ϕ being the azimuthal angle around the z-axis. The pseudorapidity is defined in terms of the polar angle θ as $\eta = -\ln\tan(\theta/2)$. Angular distance is measured in units of $\Delta R = \sqrt{(\Delta\eta)^2 + (\Delta\phi)^2}$.

The reset is a key component of the firmware. It allows to clean errors and bad behaviour, bringing back the system to a clear condition or to the initial state. The ABBA firmware has a complex structure and with the original reset connection it was difficult to reach and recover some firmware blocks. I led the effort to rethink the reset tree, to be able to reach the critical blocks independently and cure on-line instabilities.

In addition, I implemented a latency auto-correction to ensure the alignment of the calorimeter pulse inside the readout window. The misalignment was happening because of the uncertainty of the phase of the hardware transceivers locking after each reset at the beginning of the data taking. LTDBs make use of a K-comma symbol, an 8 bit pattern that can be used to find and verify the boundaries of a bitstream and for synchronization. This pattern is sent once for each bunch revolution. Knowing that also ABBA is receiving the bunch revolution information, I implemented an auto-alignment such that the pulse on a specific fiber has a constant latency.

Firmware development for LATOME

Chapter 6 is about the development and test of the VHDL code for the system-test for the Phase-I upgrade back-end electronics.

During the last Ph.D year, I worked on the firmware development for the FPGA of the LATOME board. I was asked to be in charge of the firmware module for decoding the TTC signals coming from the LHC central trigger processor. The TTC system allows the timing and trigger signals (such as the LHC clock or the hardware-based first level trigger) to be distributed to the entire experiment. TTC signals are also responsible for the detectors and sub-detectors synchronization. In the LDPS system, the TTC arrives to the Carrier board using the CERN GigaBit Transceiver (GBT) architecture and transmission protocol. On the Carrier the TTC is decoded, re-encoded and sent with a custom protocol along LVDS lines to the LATOME board.

The TTC information is sent along two channels: channel A and channel B. Channel A carries only a one bit information, the hardware-based first level trigger, called Level-1 Accept (L1A). The data stream coming along the channel B can be of two types: broadcast commands or individually-addressed commands/data. Broadcast commands are used to deliver messages to all TTC destinations in the system while individually-addressed commands/data are used to transmit user-defined data and commands over the network to specific addresses and sub-addresses.

Along the short broadcast command, two important values are sent: the Bunch Counter Reset (BCR) and the Event Counter Reset (ECR). These two value are used to reset/increase the counters (BCID and EVID) labelling the data accepted by the trigger. Along the long address command, another important information is sent: the Trigger Type (TType). The TType indicates which sub-detector or sub-system fired the trigger.

I developed a dedicated module for the TTC with a stable signal decoding and wrote custom interfaces towards other firmware modules. The Input Stage (IStage) firmware module of the LATOME is in charge of aligning the data coming from the LTDB. This is done comparing the BCID computed by my block with the BCID information coming from the LTDB. I implemented all the BCR-related signals on the requested clock domain along a custom interface. At the same time I took care of decoding and providing L1A, ECR and TType signals to the TDAQ and monitoring module, on custom interfaces. This module, together with the ATLAS software, will monitor the data quality and integrity during the data taking.

Furthermore, I am actively taking part in the integration campaigns that are ongoing at EMF for testing the system. A LATOME firmware version comprising the TTC block

was fully tested, monitoring the TTC status through specific TTC registers, to assure that the system complies with the requirements. A prototype of the LDPS system, one ATCA Carrier board and two LATOMEs, running the firmware to which I am contributing, will be installed in ATLAS at the beginning of January 2018.

CHAPTER 1

Cern and LHC

1.1 CERN

CERN, from the French acronym "Conseil Européen pour la Recherche Nucléaire", or European Council for Nuclear Research, was founded in 1952 with the idea of establishing a fundamental physics research organization in Europe. At that time, pure physics research was concentrated on the understanding the atoms and their constituents, explaining why its name contains the word "nuclear".

Nowadays, the understanding of matter goes much deeper than the nucleus and CERN main area of research is particle physics – the study of the fundamental constituents of matter and of the forces acting between them.

At CERN, physicists and engineers are probing the fundamental structure of the universe and they use the world's largest and most complex scientific instruments to study the basic constituents of matter.

1.1.1 The beginning

After the World War II, a first general idea of a re-unified Europe came form Winston Churchill during a speech given at the University of Zurich on the 19th September 1946:

> This noble continent, comprising on the whole the fairest and the most cultivated regions of the earth, enjoying a temperate and equable climate, is the home of all the great parent races of the western world.
> [...] It is the origin of most of the culture, the arts, philosophy and science both of ancient and modern time. If Europe were once united in the sharing of its common inheritance, there would be no limit to the happiness, to the prosperity and the glory which its three or four hundred million people would enjoy.
> It is to re-create the European Family, or as much of it as we can, and to provide it with a structure under which it can dwell in peace, in safety and in freedom. We must build a kind of United States of Europe. [...] If we are to form the United States of Europe, or whatever name it may take, we must begin now.
>
> (EU archives [1])

Also from the scientific point of view, Europe was no longer leading the field.

Following the example of other international organizations, a group of scientists had the idea to create a European atomic physics laboratory. Raoul Dautry, Pierre Auger

and Lew Kowarski in France, Edoardo Amaldi in Italy and Niels Bohr in Denmark were among these pioneers. Such a laboratory would not only unite European scientists but would also allow them to share the increasing costs of nuclear physics facilities. The French physicist Louis de Broglie put the first official proposal for the creation of a European laboratory at the European Cultural Conference, in Lausanne on 9 December 1949. A further push came at the fifth UNESCO General Conference, held in Florence in June 1950, where it was emphasized to "assist and encourage the formation of regional research laboratories in order to increase international scientific collaboration...". Finally, during an intergovernmental meeting of UNESCO in Paris in December 1951, the first resolution concerning the establishment of a European Council for Nuclear Research was adopted. Two months later, 11 countries signed an agreement establishing the provisional council. CERN was born.

1.2 The Large Hadron Collider (LHC)

LHC [2] is a two-ring superconducting collider placed at 100 m underground with a circumference of 27.6 km able to accelerate counter-rotating proton beams to a center of mass energy $\sqrt{s} = 13$ TeV [1] and lead ions (Pb) to 2.8 TeV per nucleon. The two beams collide in four points, as shown in Figure 1.1. Four experiments are placed around these points: ATLAS (A Toroidal LHC ApparatuS) [3], CMS (Compact Muon Solenoid) [4], LHCb (LHC-beauty) [5] and ALICE (A Large Ion Collider Experiment) [6].

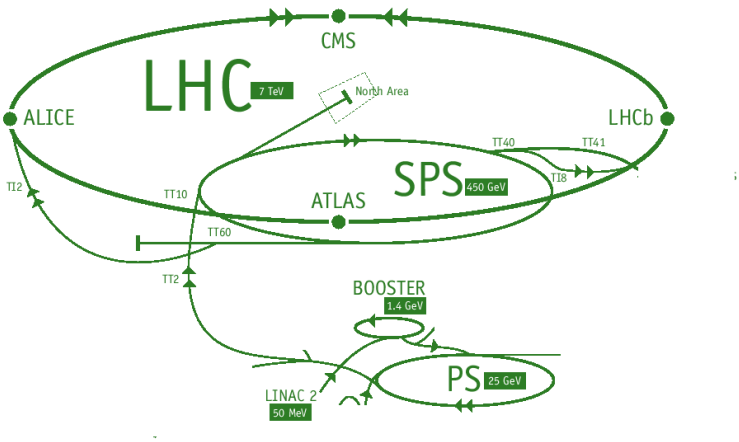

Figure 1.1: The CERN accelerator system [7].

The LHC project started in the 1980s, when the scientific community began to design a high energy physics collider able to deliver a center of mass energy higher than the other already existing colliders (LEP and Tevatron). The main purpose of this new machine was the investigation of the nature of electroweak symmetry breaking and the

[1] LHC was designed to run at a maximum collision energy of 14 TeV. However some dipole magnets have problems to reach their nominal field and would need to be retrained, demanding a larger number of quenches. It has been decided so far to operate at 13 TeV and to postpone this time-consuming operation to a later date.

search for physics beyond Standard Model at the TeV scale: this included the search for the Higgs boson [8, 9].

The approval for the LHC construction arrived in December 1994 and between 1996 and 1998, the four experiments received official approvals and started the construction. LEP was dismantled in 2000 and the tunnel hosting it is now used by LHC. LEP had eight crossing points while for LHC four interaction points were chosen. Two of them were equipped with new underground caverns.

The center of mass energies reachable at the LHC is much higher respect to ones achieved by the previous experiments. As shown in Figure 1.1, several steps are needed to reach the center of mass energy of $\sqrt{s} = 13$ TeV:

- in a hydrogen container, electrons are stripped from the atoms to get protons
- inside Linac2 protons energy is raised to 50 MeV
- the circular Booster (PSB) accelerates them to 1.4 GeV
- in the Proton Synchrotron (PS) protons reach 25 GeV
- then in the Super Proton Synchrotron (SPS) particles are accelerated to 450 GeV.

After these steps, protons are finally transferred to the LHC where each beam is accelerated for 20 minutes to 6.5 TeV. The beams are injected in bunches ~ 5.1 cm long and nominally spaced by 25 ns. Bunch crossings occur at a frequency of 40 MHz. Proton beams can circulate for many hours inside the LHC under normal operating conditions [7, 10].

1.2.1 ATLAS and CMS

ATLAS and CMS are two general-purpose detectors used to investigate the largest range of physics possible. They were built to take advantage of the full discovery potential of the accelerator. Several physics areas are under study: the search for supersymmetric and exotic particles, the accurate measurement of the top and Higgs properties and Standard Model [11] precision measurements.

Although the scientific goals are the same, different technical solutions are used. The independent design is very important in order to have the possibility to reduce systematic errors and cross-confirm any new discovery. An example of different detector solutions is the magnet system. ATLAS is characterized by a solenoid magnet producing a 2 T magnet field along the beam and a toroidal magnet system. The outer toroidal magnetic field is produced by eight very large air-core superconducting barrel loops and two end-caps air toroidal magnets, all situated outside the calorimeters and within the muon system. The CMS detector is built around a solenoid magnet, a cylindrical coil of superconducting cable that generates a field of 4 T. The field is confined by a steel yoke that forms the bulk of the detector 14000-tonne weight.

1.2.2 ALICE and LHCb

ALICE and LHCb are detectors optimized for the study of specific phenomena.

ALICE studies heavy ions collisions to understand the behaviour of high energy nuclei interaction. It is designed to investigate the physics of strongly interacting matter at extreme energy densities, where quark-gluon plasma forms. For this reason each year LHC provides some weeks with lead-ions collisions. ALICE collects also proton-proton collision data to calibrate and to support the other detectors measurements.

LHCb experiment is specialized in B hadron study and aims to investigate their properties which would help in understanding the difference in the quantity of matter and anti-matter in the universe. Like ALICE, the LHCb experiment is asymmetric and mainly focused on the forward region.

1.3 LHC 2017 achievements and future perspectives

After the Higgs boson discovery in 2012 [8, 9], the LHC experiments continued to study the Higgs boson properties to confirm the Standard Model predictions and search for new physics phenomena.

During the 2017, the LHC experiments published several new results [12]. Both ATLAS and CMS used data from 2015 and 2016 to establish evidence for Higgs boson decays to two bottom quarks [13] and CMS also presented a "5-sigma" observation of Higgs boson decays to two tau particles [14]. Furthermore, both ATLAS and CMS saw evidence of "ttH production", one of the rarest processes measured at the LHC [15, 16]. The two experiments combined their top quark measurements from proton-proton collisions, on the production of a top quark and a Z boson, a rare electroweak process [17]. In addition, CMS observed top quarks from proton-lead collisions [18] and ATLAS presented high-precision measurements of the top quark mass [19] and of the W-boson mass [20]. CMS also measured the forward-backward asymmetry in Z boson decays to electrons and muons [21, 22].

In order to continue exploiting the physics frontier, upgrades of the LHC and higher luminosity are planned for the next years [23].
Further studies and measurements of the properties of the Higgs boson will be done through the possible Higgs boson production processes and decay final states (some studies are already available here [24]).
The enhancement of the performance of the calorimeter and tracker systems in the forward region foreseen for the Phase-II upgrade will be crucial to study Higgs bosons produced via Vector-Boson Fusion (VBF). To distinguish this production mode, forward jets (in the region $|\eta| \geq 2.5$) are reconstructed.
Study on the Vector-Boson Scattering (VBS) are foreseen as well. This process is characterised by high-p_T jets and it will be crucial to determinate if other mechanisms, together with the 125 GeV Higgs boson, are responsible for maintaining unitarity.
Finally, the studies on SUperSYmmetry (SUSY) and Beyond Standard Model (BSM) Physics will continue.

1.4 LHC plan

In Figure 1.2 a detailed schedule of LHC is shown:

- Long Shutdown 1 (LS1): 2013-2014. This shut-down was used to consolidate machine elements (repairing the magnet splices and upgrading the collimation scheme) in order to achieve the design beam energy and luminosity.
- Run 2: 2015-2018. The LHC is colliding beams at $\sqrt{s} = 13$ TeV since 2015, and has exceeded the design peak luminosity of $L = 10^{34} cm^{-2} s^{-1}$ (about $2 \times 10^{34} cm^{-2} s^{-1}$ as of October 2017). An integrated luminosity of 75-100 fb^{-1} is expected to be delivered to ATLAS and CMS during the 3 years of running in Run 2.
- Long Shutdown 2 (LS2): 2019-2020. The Linac4 will be connected into the injector complex, and the injection beam energy of the Proton Synchrotron Booster will be

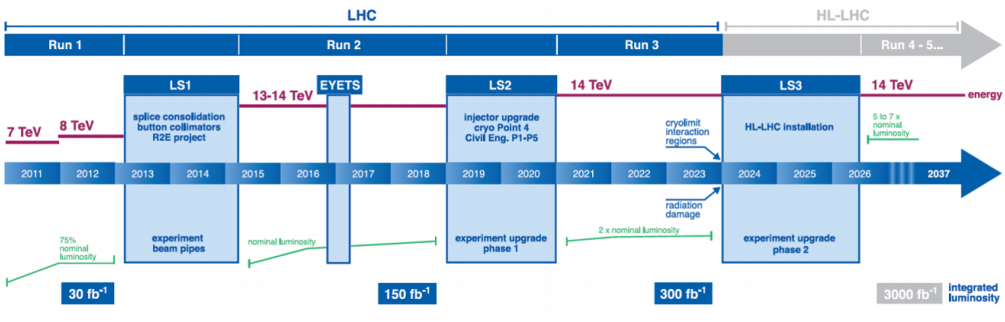

Figure 1.2: LHC plan [25].

upgraded in order to reduce the beam emittance. New cryogenics plants will be installed to separate the cooling of the superconducting radio frequency modules and the magnet cooling circuit.

- Run 3: 2021-2023. The LHC design parameters should allow for an ultimate peak instantaneous luminosity of $L \sim 2.2 \times 10^{34} cm^{-2} s^{-1}$ (Phase-I operation) and for delivering an integrated luminosity of ~ 300 fb^{-1} during Run 3.

- Long Shutdown 3 (LS3): 2024-2026. The LHC will undergo a major upgrade of its components, like low-β quadrupole triplets and the use of crab cavities at the interaction regions.

- High-Luminosity LHC (HL-LHC): 2026-2030 and beyond. The LHC complex will deliver levelled instantaneous luminosity up to $L = 7.5 \times 10^{34} cm^{-2} s^{-1}$ (Phase-II operation) with the goal of providing an ultimate integrated luminosity of up to 4000 fb^{-1} (since Figure 1.2 was produced, the value has been updated) after a period of about 12 years.

1.5 ATLAS upgrade plan

In the next years, ATLAS will have to cope with luminosities beyond the LHC nominal design value, maintaining the same physics performance.

The detector upgrade foreseen for the Phase-I operation will allow ATLAS to maintain low transverse momentum (p_T) trigger thresholds by increasing the granularity of the calorimeters involved in the Level-1 trigger and by introducing new muon trigger in the forward direction. Fast accurate tracking information provided near the start of the Level-2 trigger processing will lead to much more effective identification of the events. The full functionality of all the detector elements has to be preserved to keep the excellent capabilities of ATLAS. Simulations and detector performance extrapolations suggest that changes and upgrade of the sub-detectors should not be required until Phase-II [26].

From 2026, the ATLAS detector will have to meet the challenges and take advantage of operating at the HL-LHC. This will present a unique opportunity to extend the mass reach in searches for many signatures of new physics and to significantly extend the study of the properties of the Higgs boson. The current Inner Tracker will be no longer suitable for long term operations. It will be replaced with new silicon trackers to maintain the performance. The very high luminosities also present significant challenges to the operation of the rest of the detector systems as well as of the trigger. A new

trigger architecture will be implemented exploiting the upgrades of the detector readout systems that will improve the event selection [27].

CHAPTER 2

ATLAS

2.1 Structure of the ATLAS detector

ATLAS is one of the two general-purpose detectors at the Large Hadron Collider (LHC). It has a cylindrical shape, with a diameter of 25 m and a length of 46 m. It weighs approximately 7000 tons. Figure 2.1 shows the structure of the detector:

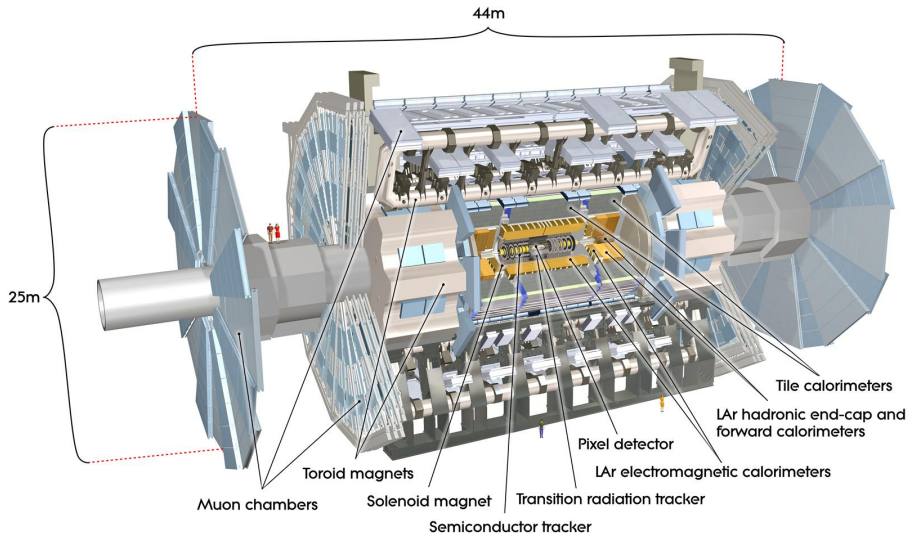

Figure 2.1: A schematic view of the ATLAS detector [28].

ATLAS is organised in layers of detectors and consists of three major components: the inner tracker, the calorimeters and the muon system. A solenoidal magnet is placed between the inner tracker and the calorimeters, while three toroidal magnet systems are used to identify and measure the momentum of muons. ATLAS has been built to cover as much as possible of the solid angle around the interaction point.

2.1.1 Coordinate system

A common coordinate system is used in the ATLAS experiment. The interaction point is defined as the origin of the coordinate system, the beam direction defines the z-axis and the x–y plane is transverse to the beam direction. The positive x-axis points from

the origin of the coordinates to the center of the LHC ring and the positive y-axis points upward to the surface of the earth, as shown on the left side of Figure 2.2. Important physics quantities like the transverse momentum p_T, the transverse energy E_T and the missing transverse energy E_T^{miss} are all defined in the x–y plane. The positive z-values

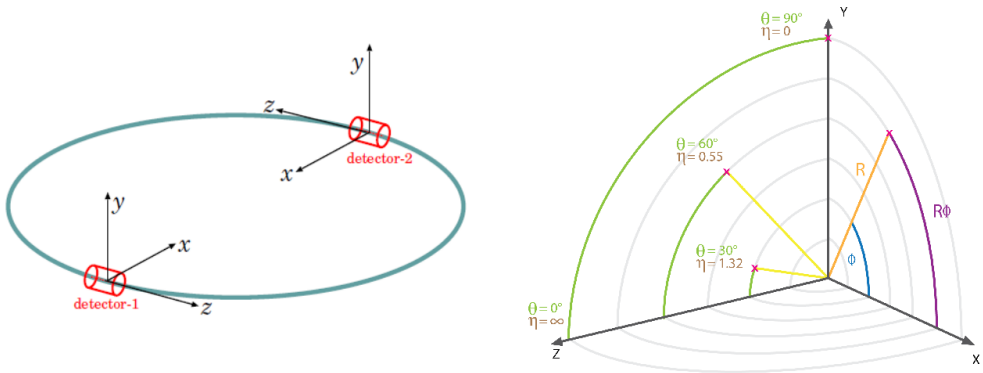

Figure 2.2: Common coordinate system used in the LHC experiments.

are usually referred to the half of the detector called "A-side" while the negative values refer to the other half, called "C-side". Furthermore, the transverse plane can be also described with $r - \phi$ coordinates.

As shown on the right side of Figure 2.2, the azimuthal angle ϕ is measured from the positive x-axis. The polar angle θ is defined as the angle from the positive z-axis, along the detector. The radial coordinate r, describes the distance from the beam.

Usually the pseudorapidity η is used instead of the polar angle θ:

$$\eta = -\log \tan \frac{\theta}{2}, \tag{2.1}$$

and the angular distance ΔR between two objects in the $\eta - \phi$ space:

$$\Delta R = \sqrt{\Delta\eta^2 + \Delta\phi^2}. \tag{2.2}$$

The difference in pseudorapidity η, as well as ΔR, are important as they are Lorentz invariant under boosts along the z-axis.

2.2 Inner Detector

The Inner Detector (ID) [29] covers a pseudorapidity range of $|\eta| < 2.5$ and performs particle tracking in order to measure particle transverse momentum and trajectories. The ID diameter is 2.1 m and the total length is 6.2 m. A schematic picture of the ID is shown in Figures 2.3 and 2.4. The ID is made of three subsystems: Pixel detector [30], Semi-Conductor Tracker [31] and Transition Radiation Tracker [32]. During 2014, a new pixel detector layer was inserted, the Insertable Barrel Layer detector [33]. In the barrel region the high-precision detectors are arranged in concentric cylinders around the beam axis, while the end-cap detectors are mounted on disks perpendicular to the beam axis. The high-resolution detectors at the inner radii and the tracking elements at the outer radii are both inside a solenoidal magnet with a central magnetic field of 2 T.

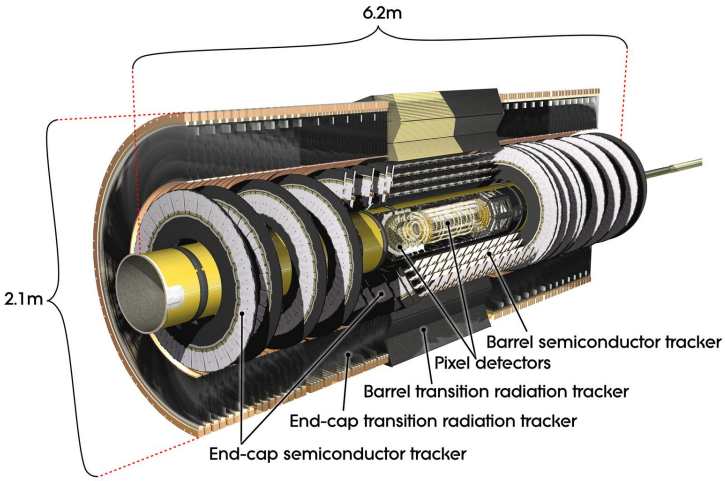

Figure 2.3: A schematic view of the Inner Detector [34].

2.2.1 Pixel detector

The pixel detector system is the innermost element of the ID. It is designed to provide a very high-granularity, high-precision set of measurements as close to the interaction point as possible. Silicon pixels, with their excellent spatial resolution, play a very important role in the identification and reconstruction of primary and secondary vertices, respectively from the proton-proton interaction and from the decay of particles, for example containing a b-quark or for b-tagging of jets.

The system consists of three barrels at average radii of $\sim$50.5, 88.5 and 122.5 mm, and four disks on each side, between radii of 110 and 200 mm, which complete the angular coverage. All pixel modules are identical, the silicon wafer size is 16.4 x 60.8 mm and 250 μm thick. Each sensor has about 46 thousand pixels sensors of size 50×400 μm. There are 1744 modules in the Pixel Detector for nearly 80 million channels. The resolution is of 14 μm in $R - \phi$ and 115 μm in $z(R)$ in the barrel (end-cap).

2.2.2 Insertable B-Layer

The performance of the innermost layer of the Pixel detector is fundamental for vertexing and b tagging. This layer was exposed to a very high radiation dose and this had an impact on its performances, even if it was designed with radiation hardness techniques. For this reason, an additional pixel layer inside the B-layer, the Insertable B-Layer (IBL), has been inserted during the LHC shut down in 2013.

In the IBL, two different silicon sensor technologies have been used: planar n-in-n and 3D with passing through columns. The planar pixel technology is used in the barrel region while the 3D technology is used in the forward region. The two types of sensors have respectively a thickness of 200 and 230 μm and a pixel size of 50×250 μm^2.

The inner radius of the IBL detector is 31 mm while the outer radius is 40 mm. The insertion gap between the inner supporting tube and the IBL detector is only 0.2 mm and the gap between the supporting tube and the Pixel is 1.9 mm. To make the integration possible the diameter of the beam pipe was reduced.

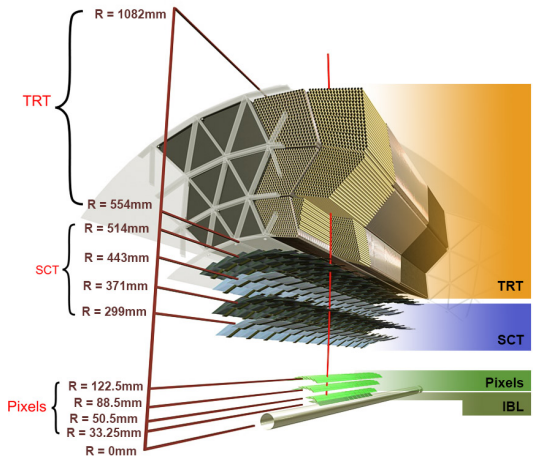

Figure 2.4: A schematic view of the Inner Detector, including the new IBL [34].

2.2.3 Semi-Conductor Tracker

The Semi-Conductor Tracker detector (SCT) is positioned outside of the Pixel detector. It comprises four central barrel layers at radii of 299, 371, 443 and 514 mm and two end-caps which contain nine disks each.

Each silicon sensor is 63.56 $\times$ 63.96 mm^2 with 768 readout strips each with 80 μm pitch. The barrel part is made of 2112 modules while the end-cap carries 1976 modules. Each barrel module includes four silicon microstrip sensors and the readout electronics: two pairs of sensors are mounted back-to-back with a 40 mrad stereo angle to make a double sided module. The forward modules are very similar in construction but use tapered strips. The detector contains 61 m^2 of silicon detectors, with 6.2 million readout channels. The space-point resolution per module is 17 μm in $R - \phi$ and 580 μm in $z(R)$ in the barrel (end-cap).

2.2.4 Transition Radiation Tracker

The Transition Radiation Tracker (TRT) is the outermost of the three tracking subsystems of the ID. The TRT is a gaseous straw-tube tracker and consists of drift tubes with a diameter of 4 mm. In the centre of each tube there is a gold-plated tungsten wire of 31 μm diameter. When a charged particle traverses the TRT, it ionises the gas inside the straws and the resulting free electrons drift towards the wire, where they are collected. The TRT detector combines tracking capabilities with particle identification based on transition radiation photon detection. The particle identification functionality gives significant discrimination power between electrons and charged pions with energy in the range 1 GeV $<$ E $<$ 100 GeV.

The tube wall is kept at a voltage of -1.5 kV and the wire at ground potential, so that each tube acts as a small proportional counter and is filled with a non-flammable xenon-based gas mixture of 70% Xe, 20% CO_2 and 10% CF_4. During Run 1, due to some leaks which developed in the gas system, in some regions the Xenon gas was replaced by a much cheaper Argon-based gas mixture.

The barrel section is built of individual modules between 329 and 793 straws each, covering the radial range from 554 to 1082 mm. Each end-cap consists of 18 wheels

covering a radial range from 640 to 1030 mm, and 830 to 3400 mm along the z-axis.

The total number of TRT readout channels is approximately 420000. The TRT only provides $R - \phi$ information in the barrel and $z - \phi$ in the end-cap, with a resolution of about 130 μm per straw.

2.3 Calorimeters

As shown in Figure 2.5, the components of the ATLAS calorimetry system are: the Liquid Argon (LAr) calorimeter [35] and the Tile hadronic calorimeter [36].

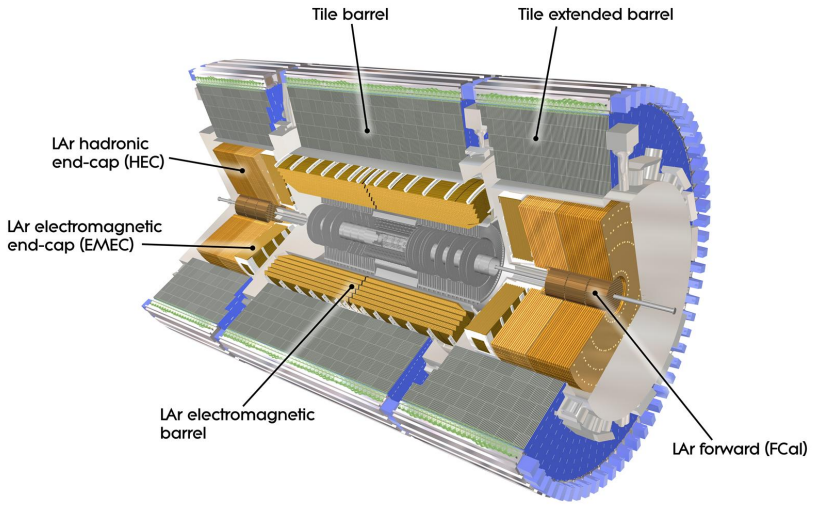

Figure 2.5: A schematic view of the ATLAS calorimeters [37].

2.3.1 LAr calorimeter

The LAr electromagnetic calorimeter uses a sampling liquid argon/lead technology in the barrel and end-cap electromagnetic calorimeter and liquid argon/copper in the hadronic end-caps. It is hosted by three large aluminium cryostats. The LAr calorimeter can be divided in four major parts: the Electro-Magnetic Barrel (EMB), the Electro-Magnetic End-Cap (EMEC), the Hadronic End-Cap (HEC) and the Forward Calorimeter (FCal).

The EMB is 6.4 m long, 53 cm thick and covers the pseudorapidity range $|\eta| < 1.475$ providing signals from about 110000 channels. The EMB consists of two barrels of 1024 lead absorbers, interleaved with the same number of electrodes held in place by a 2.1 mm high honeycomb structure, which defines the liquid argon gap size. The radiation length (X_0), which will be described in section 3.2, in this η-region is about $24X_0$.

The EMEC extends the pseudo rapidity coverage to the range $1.375 < |\eta| < 3.2$, and, as the EM barrel, is built with accordion shaped absorbers and electrodes, providing an excellent hermeticity, a complete azimuthal coverage and longitudinal segmentation. There are 256 absorbers for the inner wheel ($1.375 < |\eta| < 2.5$), and 768 for the outer

wheel ($2.5 < |\eta| < 3.2$). The EMEC is 0.632 m thick and has a radius of 2.077 m. In this η-region the radiation length value is about $30X_0$.

The HEC calorimeters consist of two cylindrical wheels in each end-cap. Each wheel is made of 32 wedge-shaped modules. Each of them is an alternation of 25 mm thick copper absorbers (50 mm in the second wheel), and signal collection gaps. The HEC wheels have a radius of 2.09 m.

The FCAL calorimeter uses cylindrical electrodes consisting of rods positioned in the center of tubes parallel to the beam axis with liquid argon filling the small gaps between the rod and the tube wall. The tubes are substained by a metal matrix. The FCAL consists of three modules: the one closest to the interaction point uses copper as absorber and is optimized for electromagnetic measurements. The other two modules use tungsten and are optimized for hadronic measurements [38].

2.3.2 Tile calorimeter

The Tile hadronic calorimeter is a large sampling calorimeter which makes use of iron as the absorber material and scintillating tiles as the active medium. The Tile calorimeter consists of a cylindrical structure with an inner radius of 228 cm and an outer radius of 423 cm. It is subdivided into a 564 cm long central barrel and two 291 cm extended barrels, each composed of 64 modules along ϕ. The barrel covers the region -1.0< $|\eta|$ <1.0, and the extended barrels cover the region 0.8< $|\eta|$ <1.7. The Tile calorimeter has an effective nuclear interaction length of $\lambda = 20.7$ cm, with a total depth of $7.4\ \lambda$ at $\eta = 0$. Furthermore, the magnetised steel of the hadronic calorimeter and its girder structure serve to return the flux of the solenoid magnetic field.

2.4 Muon system

The ATLAS outermost sub-detector is the muon spectrometer [39]. It provides muon momentum measurements using the muon track deflection by three large superconducting toroid magnets. A schematic view of the muon system is shown in Figures 2.6.

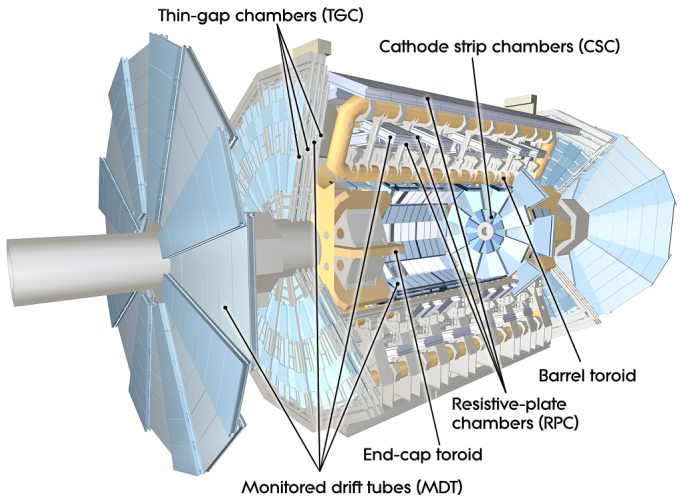

Figure 2.6: A schematic view of the ATLAS muon system [40].

In the barrel region, tracks are measured in chambers arranged in three cylindrical layers (stations) around the beam axis. In the transition and end-cap regions, the chambers are installed vertically, in three stations as well. A precision measurement of the track coordinates in the principal bending direction of the magnetic field is provided by Monitored Drift Tubes (MDTs) over most of the pseudorapidity range. At large pseudorapidities and close to the interaction point, Cathode Strip Chambers (CSCs) with higher granularity are used to sustain the very high rate and the background conditions. In order to meet the very stringent momentum resolution requirements, an optical alignment system has been designed [41].

The trigger system covers the pseudorapidity range $|\eta|$ <2.4. Resistive Plate Chambers (RPCs) and Thin Gap Chambers (TGCs) are used respectively in the barrel and in the end-cap regions. Both types of trigger chambers also provide a second-coordinate measurement of tracks orthogonal to the precision measurement, in a direction approximately parallel to the magnetic field lines.

In the pseudorapidity range $|\eta|$ <1.0, the bending is provided by a large barrel magnet made of eight coils surrounding the hadron calorimeter. For 1.4< $|\eta|$ <2.7, muon tracks are bent in two smaller end-cap magnet systems inserted into both ends of the barrel toroid. In the interval 1.0< $|\eta|$ <1.4 the deflection is provided by a combination of barrel and end-cap fields. This configuration provides a magnet field that is mostly orthogonal to the muon trajectories and minimizes the degradation of resolution due to multiple scattering.

The muon system design provides a good momentum measurement of muons even at high energy: the transverse momentum p_T can be measured with a resolution of $\Delta p_T/p_T = 10\%$ even at $p_T = 1$ TeV.

2.5 Magnet system

The main sections of the magnet system are the central solenoid magnet [42], the barrel toroid [43] and the end-cap toroids [44], as shown in Figure 2.7.

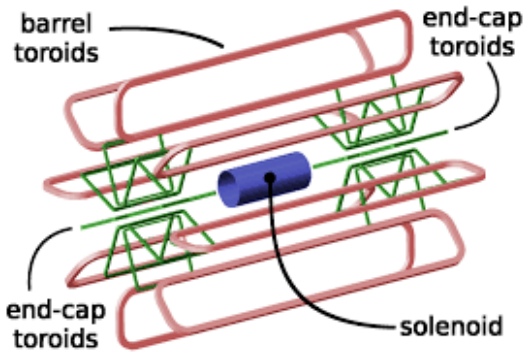

Figure 2.7: A schematic view of the ATLAS magnet system. The toroidal magnet is made of the red and the green part. The solenoid is shown in blue [45].

The solenoidal magnet is placed between the inner detector and the calorimeter system. It's 5.3 m long, with a diameter of 2.4 m and a weight of 5 ton. The solenoid produces a 2 T magnetic field in the central tracking volume. The operating temperature of 4.5 K is maintained by a cryostat shared with the barrel electromagnetic calorimeter.

The barrel toroid consists of 8 flat superconducting race-track coils, each 25.3 m long and 5 m wide. The 8 coils in the torus are kept in position by 16 support rings. In total the barrel toroid is 830 tons. The magnets are cooled down to 4.5 K by liquid helium and operate at a nominal current of 20.5 kA.

Two end-cap toroids, positioned inside the barrel toroid, at both ends, provide the required magnetic field across a radial span of 1.5 to 5 m. Each end-cap toroid has a weight of 240 tons. The coil system of the end-cap toroid is rotated by an angle of 22.5 with respect to the barrel toroid coil. In this way, radial overlap between the two coil systems is provided and the bending power optimized.

2.6 Forward detectors

The ATLAS forward region is covered by a group a small sub-detectors. Figure 2.8 shows their position along the beam line, from the closest (left) to the most distant (right) from the collision point: LUCID (Luminosity measurement using Cherenkov Integrating Detector) [46], ZDC (Zero-Degree Calorimeter) [47], AFP (ATLAS Forward Proton) [48] and ALFA (Absolute Luminosity For ATLAS) [49].

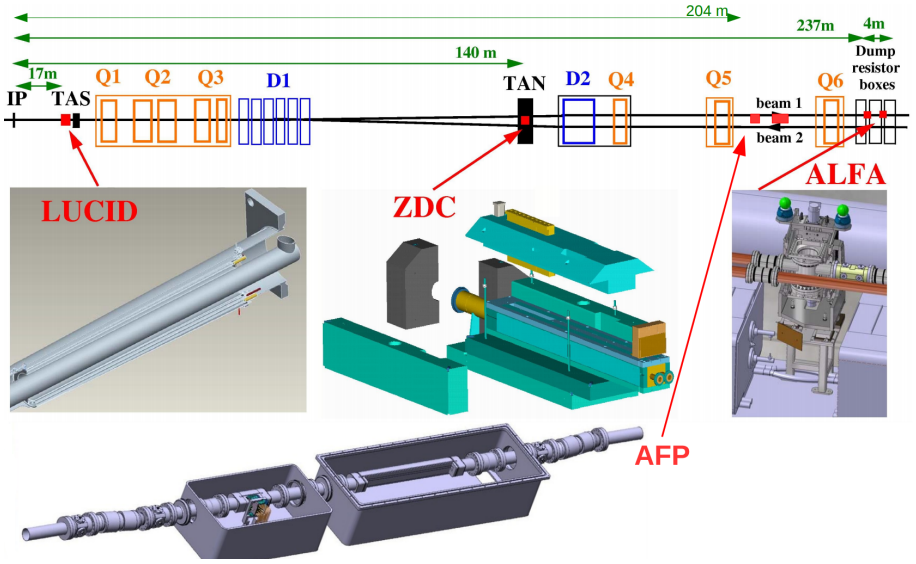

Figure 2.8: The ATLAS forward detectors and their distance from the collision point [50].

LUCID is a Cherenkov counter, monitoring the luminosity delivered by the LHC accelerator, in ATLAS. Two detectors are symmetrically placed in both forward regions at 17 m from the interaction point. Each LUCID detector is made of 16 photomultiplier tubes and 4 quartz fiber bundles. The photomultipliers detect charged particles passing through the quartz windows, where Cherenkov light is produced. Cherenkov light is produced in the fiber bundles as well and carried to the photomultipliers.

ZDC has the role of detecting forward neutrons, for $|\eta| > 8.3$, in both pp and heavy-ion collisions. It is placed on both sides of the ATLAS detector, at a distance of 140 m. Each detector has one electromagnetic module (about 29 radiation lengths thick), and 3 hadronic modules, composed of tungsten with an embedded matrix of quartz rods, attached to photomultiplier tubes.

AFP aims to measure transfer momentum and energy loss of protons emitted from the collision point interaction in very forward directions. Two detectors are located per each ATLAS sides, at 204 m and 217 m, along the beam line. Each detector contains a 3D silicon tracker and a time-of-flight detector in the far stations.

ALFA is the furthest detector, placed at 237 m from the interaction point, on both ATLAS sides. It aims to measure the elastic pp scattering at small angles. The set-up is installed in Roman Pot stations [51], vessels that are connected to the accelerator vacuum via bellows. In this way the detector can approach the beam very close without entering the machine vacuum. Each detector is made of staggered layers of square-shaped scintillating fibers, read out by photomultiplier tubes.

2.7 Trigger and data acquisition

The Trigger and data acquisition (TDAQ) system [52], shown in Figure 2.9, is a fundamental component of the ATLAS detector. Data from collisions is moved from the detector readout electronics into front-end buffers at the bunch crossing rate. The task of the Trigger and DAQ system is to select a few hundred events per second for recording to a permanent storage for later study. The DAQ system has to transport and assembly the event data from the front-end buffers to the recording on disk.

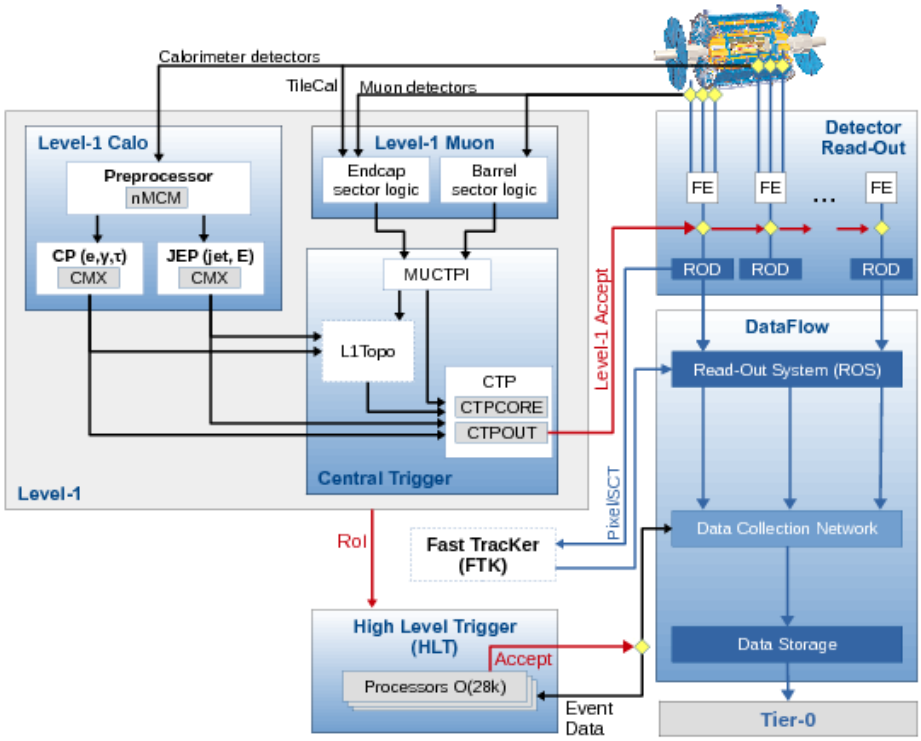

Figure 2.9: The ATLAS trigger and data acquisition system in Run 2 [53].

The bunch crossing rate at the LHC was 20 MHz during Run 1 and was increased to 40 MHz for Run 2. The trigger system has to face high event rates and pile-up levels,

maintaining high efficiency for selecting rare events. A two-level trigger system is used in ATLAS, with each level accepting only a subset of data to be passed on to the next.

The first level trigger (L1) is hardware based and processes data from the calorimeter and the muon detectors. The successive trigger level, called High-Level Trigger, is operated in software and works on a large farm of commercial computer processors.

The L1 trigger decision is taken by the Central Trigger Processor (CTP), which receives inputs from the L1 calorimeter (L1Calo) and L1 muon (L1Muon) triggers as well as several other subsystems like the Minimum Bias Trigger Scintillators (MBTS) [54], LUCID Cherenkov counter and the ZDC. After the L1 trigger acceptance, the events are buffered in the Read-Out System (ROS) and processed by the HLT. The HLT receives Region-of-Interest (RoI) information from L1, which can be used for regional reconstruction by the trigger algorithms. After the events are accepted by the HLT, they are transferred to local storage at the experimental site and exported to the Tier-0 facility at CERN's computing centre for offline reconstruction.

CHAPTER 3

Liquid Argon Calorimeter

3.1 Introduction

In particle physics, calorimeters are detectors used to measure the energy of impinging particles. The typical usage of calorimeters is for energy measurements but they can be used as well for providing position and timing information in case of a proper segmentation or particle identification. In this dissertation I give a synopsis of the characteristic of these detectors, while a complete description can be found in [55] and in [56].

The main features of calorimeters can be summarized in the following points:

- The intrinsic energy resolution for a calorimeter increases as $1/\sqrt{E}$, where E is the particle energy, in contrast with magnetic spectrometers in which the momentum resolution deteriorates with particle momentum.
- Calorimeters are sensitive to both charged and neutral particles, again in contrast to the behaviour of a magnetic spectrometer. These detectors, if hermetic, can provide an indirect transverse measurement of undetected particles like neutrinos through the so-called "missing energy", through energy balance.
- Calorimeters are versatile detectors. They could be segmented and, in addition to energy, they could provide position information, timing and particle identification information.
- Calorimeters can be used for triggering, as they provide fast signals.
- The shower length increases only logarithmically with the energy of the incident particle. As a consequence, the size of a calorimeter, which aims to contain an entire shower, scales logarithmically with the incident particle energy. On the other side, the dimension of an electromagnetic spectrometer scales with $\sqrt{p}$ (where p is the particle momentum) for a given momentum resolution.

Calorimeters can be divided in electromagnetic calorimeters and hadronic calorimeters. Electromagnetic calorimeters measure mainly electrons, positrons and photons through their electromagnetic interactions with the calorimeter material. Hadronic calorimeters are used to measure mainly hadrons through their strong and electromagnetic interactions. In both cases, the interaction of the impinging particle in the calorimeter produces a cascade of secondary particles, called shower, with degraded energy. The energy deposited by the shower in the calorimeter active medium can be detected and is used to estimate the energy of the incident particle. A calorimeter is linear when there is a constant proportion between the energy deposited and the response: electromagnetic

calorimeters are in general linear while hadronic calorimeters are not, unless they are compensated (more details in section 3.3.2).

Calorimeters can be also divided into two other categories, depending on their construction: homogeneous and sampling calorimeters. The material of an homogeneous calorimeter is not only used to degrade the incident particle energy but is also entirely sensitive to the energy deposited by the shower. In a sampling calorimeter, two different materials are used for energy degradation and signal generation. Usually the absorption material is a passive and high-Z medium while the signal is detected in an active low-Z medium.

The rest of this chapter will be mostly dedicated to electromagnetic calorimeters.

3.2 Electromagnetic calorimeter

Electrons and photons over 1 GeV mainly interact through bremsstrahlung mechanism (incident electron) and pair production (incident photon) [55]. These processes creates secondary processes where electrons (or positrons) and photons create a cascade of particles with decreasing energy. At a certain point, the particle energy falls below the threshold for pair production and the remaining energy is dissipated by excitation and ionization.

The energy at which radiation loss and collision loss are equally likely is called "critical energy" (ϵ) and is given by [55]:

$$\epsilon(MeV) \approx \frac{800}{Z + 1.2}. \tag{3.1}$$

3.2.1 Electromagnetic cascade

An incident electron loses its energy through bremsstrahlung mechanism at an average rate of $dE/dx = -E/X_0$ where X_0 is called radiation length. This is the mean distance over which a high energy electron loses all but $1/e$ of its energy and is defined by this equation [55]:

$$X_0 = \frac{(716.4\ g\ cm^{-2})A}{Z(Z+1)ln(287/\sqrt{Z})}, \tag{3.2}$$

where Z and A are the atomic number and atomic weight of the material.

A similar concept exists also for incident photons, the mean free path X_γ that is the average distance which a high energy photon can travel before converting into e^+e^- pair. This is the mean distance over which a group of identical photons lose all but $1/e$ of their number by pair production and can be defined as:

$$X_\gamma = \frac{9}{7}X_0. \tag{3.3}$$

In general, the longitudinal distribution of an electromagnetic shower scales with the radiation length and the shower depth at which the largest number of secondary particles occur is located at

$$t_{max}(X_0) = \left[ln\frac{E_{inc}}{\epsilon} + C_i\right], \tag{3.4}$$

where $C_e = -0.5$ and $C_\gamma = +0.5$.

The transverse size of the electromagnetic shower is mainly due to scattered electrons and positrons. Bremsstrahlung photons can also partially contribute to the shower spread. The spread of the electromagnetic shower is given by the Moliere radius [55]:

$$\rho = X_0 \frac{21MeV}{\epsilon}. \tag{3.5}$$

It represents the average lateral deflection of electrons at the critical energy after one radiation length.

A short description of the most important processes, shown in Figure 3.1, is given in the following paragraphs.

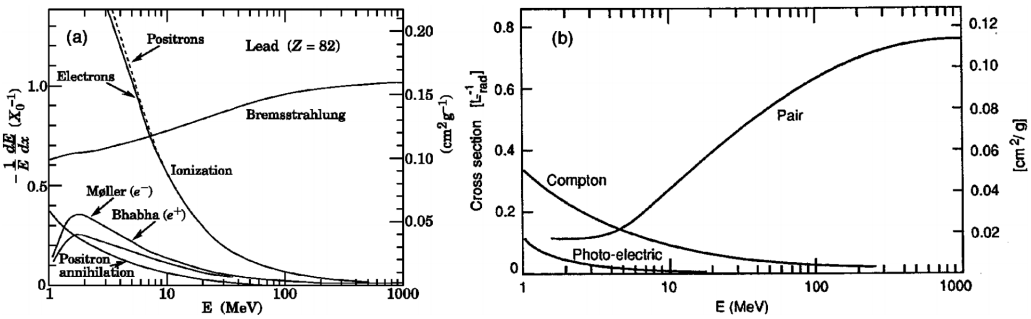

Figure 3.1: (a) Fractional energy lost in lead by electrons and positrons as a function of energy. (b) Photon interaction cross section in lead as a function of its energy [56].

Bremsstrahlung

An electron or a positron with initial energy E_{init} loses part of its energy trough the deceleration caused by the deflection due to the Coulomb field of an atomic nucleus. The moving particle loses kinetic energy, which is converted into a photon, which usually carries only a fraction of the initial energy of the particle. This radiation is called bremsstrahlung, that in german means "braking radiation" [57].

Pair production

If the energy of a photon is at least twice the rest mass of an electron (512 keV), it can produce an electron-positron pair. The cross-section for pair production increases with increasing energy of the incident particle and reaches a plateau for energies above 10 GeV. This process requires the nearby presence of a massive atom for momentum conservation, but the atom recoil energy is negligible [57].

Photoelectric effect

In the photoelectric absorption process, a photon is absorbed by an atom, leaving it in an excited state. The atom returns to the ground state by emitting an electron, in this case called photoelectron. Photoelectric absorption is more significant for low-energy photons ($\approx$ 100 keV), increases with the atomic number of the absorber and decreases with the photon energy. The total energy of the process is always conserved [57].

Compton effect

In the Compton scattering, a photon hits an atomic electron in the absorbing material. The result is a less energetic photon together with a scattered electron, carrying the energy lost by the photon. At the end, the overall momentum of the system is again conserved. The probability of compton scattering is higher at MeV energy range. If the scattered photon still has enough energy, the process may be repeated [57].

3.2.2 Energy resolution

The energy resolution for a calorimeter can be written as [56]:

$$\frac{\sigma(E)}{E} = \frac{a}{\sqrt{E}} \oplus \frac{b}{E} \oplus c \tag{3.6}$$

where $\oplus$ indicates a quadratic sum.

The first term on the right-hand side is the "stochastic term" and depends on the fluctuations related to the physical development of the shower. The stochastic term in an homogeneous electromagnetic calorimeter is small, few percent in units of $1/\sqrt{E(GeV)}$ because the energy deposited in the active volume by a monochromatic beam does not change event by event. In sampling calorimeters this is not the case, because the active layers are interleaved by absorber layers. These fluctuations, called "sampling fluctuation" represent the most important limitation in energy resolution in this type of calorimeters.

The second term, the "noise term", comes from the electronic noise of the readout chain and depends on the detector technique and on the electronics feature of the readout. Calorimeters in which the signal is collected in the form of light such as scintillators or homogeneous calorimeters, can have a small level of noise if a photosensitive device providing a high-gain multiplication of the signal is used as a first element of the readout. The noise will be larger in the detectors which collect the signal in charge form, using a preamplifier as a first element of the chain. In sampling calorimeters the signal to noise ratio can be improved increasing the sampling fraction (an higher sampling fraction implies a larger signal from the active medium).

The third term is the "constant term" and does not depend on the particle energy. It sums up effects caused by material non-uniformities, imperfections of mechanical structures, temperature gradients, radiation damage, etc. These effects can be cured, to a certain extent, by system calibrations. With the energy increase in the accelerators, this term becomes more and more important to the energy resolution. Typically, the constant term of an electromagnetic calorimeter should be around 1% or even smaller and this is particularly true for the homogeneous calorimeters because of the small stochastic term.

3.3 Hadronic calorimeters

The first hadronic calorimeters were used by the end of the 1950s to study the cosmic-ray spectrum [58]. The energy E was assumed to be related to the hadronic shower multiplicity $n(x)$ of fast charged particles versus the shower depth x through the specific ionization ϵ, so that $E = \epsilon \int n(x)dx$. This estimation was correct within a factor of $\simeq 2$.

Nowadays, such detectors are built and optimized with a good understanding of the hadronic cascades also thanks to the development of Monte Carlo codes of the showering processes [56, 59, 60].

3.3.1 Hadronic cascade

Showers generated by strongly interacting particles are much more complex than electromagnetic showers. Few high energy particles are carrying half of the energy of the cascade while the other half goes into multi-particle production. The resulting shower is mostly made of nucleons and pions, of which about $1/3$ are π^0. The parametrization of the longitudinal development of an hadronic shower is, as a consequence, more complex and the general form can be described by two terms: the first term corresponds to the electromagnetic energy deposition while the second to the hadronic energy deposition. Even if Monte Carlo algorithms provide good basis for electromagnetic shower simulations, there is no universal standard for the hadronic showers (more references can be found in [56]).

It is anyway possible to define the depth at which there is the maximum energy deposition in the hadronic shower:

$$t_{max}(\lambda_I) \approx [0.2ln(E) + 0.7], \tag{3.7}$$

where E is expressed in GeV, λ_I is the interaction length, and so the position of the shower maximum depends logarithmically on the energy. The 95% of a shower energy is contained in:

$$t_{95\%}(\lambda_I) \approx t_{max} + 2.5(E), \tag{3.8}$$

but since the fluctuations in energy deposition are very large, it is important to design a calorimeter with additional $\sim 3\lambda_I$ of material [55].

3.3.2 Energy resolution and compensation

A fraction of the non-electromagnetic component of an hadronic shower mainly consists of the break of nucleon bindings and the kinetic energy of recoil nuclei that do not lead to a signal in the calorimeter. This is called invisible energy. As a result, on average it is expected to have, with the same incident energy, a lower response to hadrons respect to the electromagnetic shower. The invisible energy may represent a large part of the total non-electromagnetic energy with large event-to-event fluctuations [55].

The response to the electromagnetic energy deposition E_e can be characterized by a factor called *detector efficiency* ϵ_e and the same for the hadronic part, E_h which is characterized by an efficiency ϵ_h. In general, $\epsilon_h < \epsilon_e$ because of the nucleon processes which do not yield visible energy, as explained before. The calorimeter response to hadronic shower can be describe by [55]:

$$R_h = E_h\epsilon_h + E_e\epsilon_e \tag{3.9}$$

As described in Figure 3.2, depending on the component with the major part of energy, the gap between the hadronic and electromagnetic detection efficiency will vary.

A calorimeter with equal response to both parts would improve the situation. This type of calorimeters are called "compensated" and are characterized by $\epsilon_e/\epsilon_h = 1$. Compensated calorimeters have no gap between electromagnetic and hadronic components, while for non-compensated calorimeters the gap is present. The larger this gap is, the wider the spread for the detected energy will be. In the figure the response for electromagnetic events ("Type E events") and the response for hadronic events ("Type H events") are indicated with solid line, while the overall calorimeter response ("all events") is drawn with the dashed line.

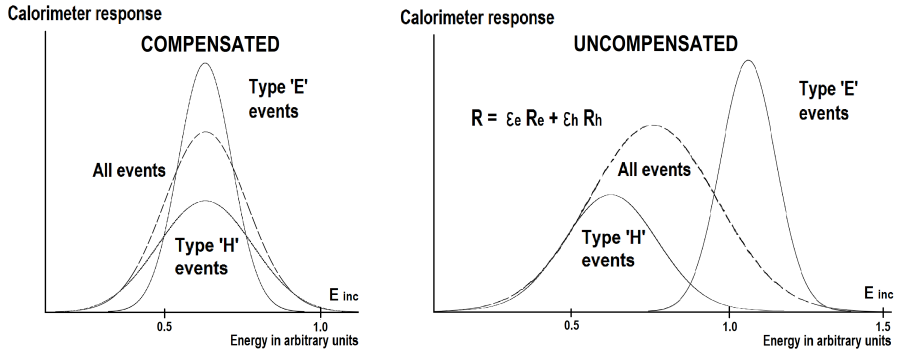

Figure 3.2: Calorimeter compensated response (left) and not compensated response (right) [55].

3.4 Structure of ATLAS calorimeters

The ATLAS calorimetry system consists of an electromagnetic calorimeter covering the pseudorapidity region $|\eta| < 3.2$, a hadronic barrel calorimeter covering $|\eta| < 1.7$, hadronic end-cap calorimeters covering $1.5 < |\eta| < 3.2$, and forward calorimeters covering $3.1 < |\eta| < 4.9$.

A two-dimensional view of both electromagnetic and hadronic calorimeters is shown in Figure 3.3 (more details can be found in Sections 2.3 and 3.5). All calorimeters used in ATLAS are sampling calorimeters.

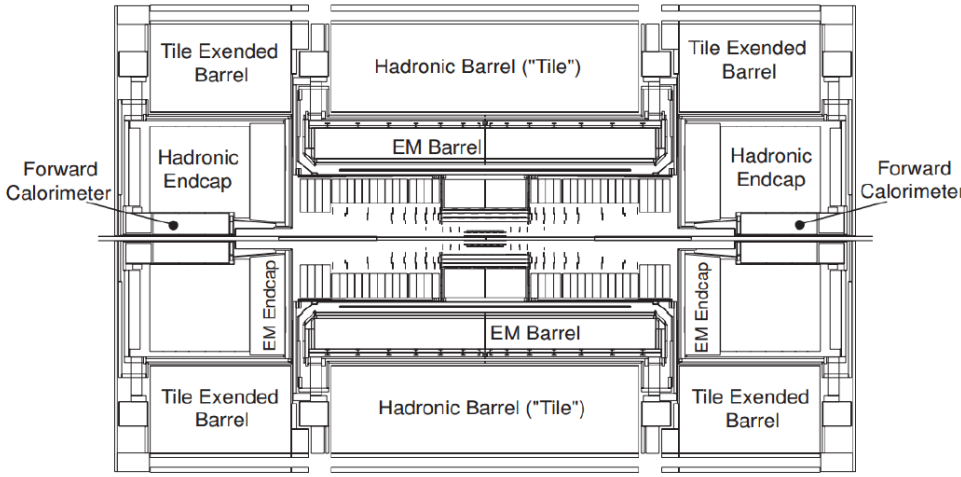

Figure 3.3: 2D view of the ATLAS calorimeters, LAr and Tile [35].

The electromagnetic calorimeter has in addition a presampler detector over the pseudorapidity range $|\eta| < 1.8$, installed immediately behind the cryostat cold wall, and used to correct for the energy lost in the material upstream of the calorimeter [35].

The barrel electromagnetic calorimeter is contained in a barrel cryostat, shown in Figure 3.4, which surrounds the inner detector cavity. The solenoid which supplies the 2 T magnetic field to the inner detector is integrated into the vacuum of the barrel cryostat

and is placed in between the tracker and the electromagnetic calorimeter. Two end-cap cryostats house the end-cap electromagnetic and hadronic calorimeters, as well as the forward calorimeters.

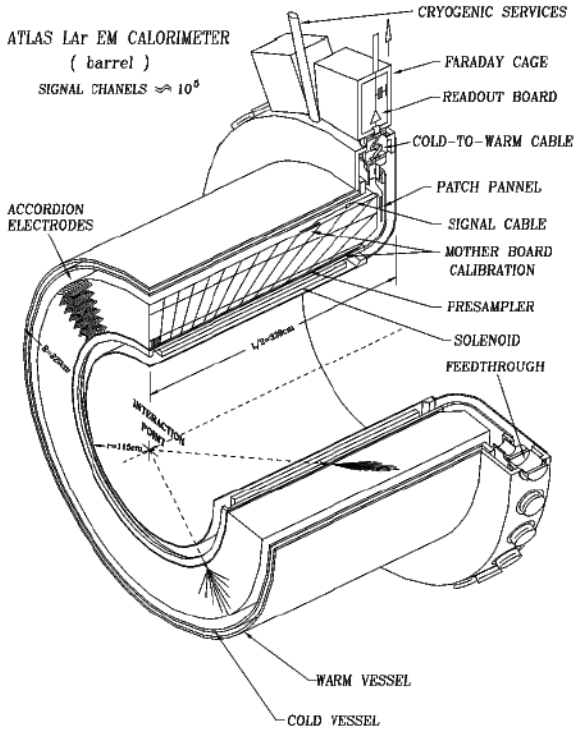

Figure 3.4: Barrel cryostat [35].

3.5 LAr ATLAS Calorimeter

Calorimeters play a central role in ATLAS. In the difficult environment of the collision produced by the LHC, the calorimeters are designed to trigger and to provide precision measurements of electrons, photons, jets, and E_T^{miss} [61, 62, 63].

The electromagnetic calorimeter is a lead-liquid argon detector with accordion-shaped kapton electrodes and lead absorber plates over its full coverage. The accordion geometry provides complete ϕ symmetry without azimuthal cracks. The total thickness of the electromagnetic calorimeter is about 24 radiation lengths (X_0) in the barrel and about 30 X_0 in the end-caps.

3.5.1 Barrel

The barrel part of the electromagnetic calorimeter is contained inside a cryostat, which is 6.8 m long, with an outer radius of 2.25 m and an inner radius of 1.15 m. The barrel calorimeter consists of two identical half-barrels, with a gap of a few millimetres in between. Each half-barrel consists of 1024 lead stainless-steel absorbers with three layers of copper separated by polyamide readout-electrodes in between, as shown in Figure

3.5. The readout electrodes, which are 300 μm thick, are centred by honeycomb spacers, defining two liquid argon gaps of 2.1 mm each (thickness of the active material).

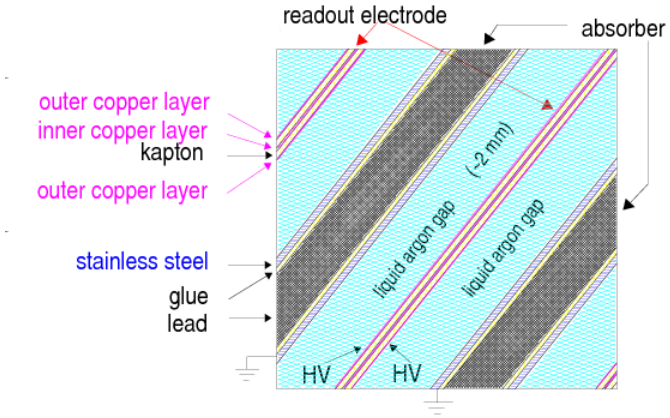

Figure 3.5: LAr readout electrodes made of three layers of copper and two liquid argon gaps for the active material [64].

The readout cells are defined in η by etching of the copper on the readout boards and in ϕ by grouping together four adjacent boards. Read-out and calibration signals are routed through cold-to-warm feedthroughs located at each end of the cryostat. The barrel cryostat has 32 feedthroughs at each end.

3.5.2 End-cap

Figure 3.6 shows the end-cap cryostat which contains the electromagnetic wheel, the two hadronic wheels, and the forward calorimeter [65]. Here, the accordion waves scale with the radius. Each end-cap electromagnetic wheel is made of two concentric wheels, the big one is covering the pseudorapidity interval from 1.4 to 2.5, and the small one from 2.5 to 3.2. The big wheel has 768 plates (3 consecutive planes are grouped together to form a readout cell of 0.025 in ϕ) while the small wheels has 256 plates. As for the barrel, the end-cap cryostats are built out of aluminium, and are vacuum insulated.

3.5.3 Presampler

The material located in front of the calorimeter created the necessity of a presampler layer [66]. The role of this layer is to correct the energy lost in front of the calorimeter. The barrel presampler has 1 cm of LAr active layer, with electrodes roughly perpendicular to the beam axis, while the end-cap presampler has 5 cm of LAr and the electrodes are parallel to the beam axis. No lead absorber is present.

The transition region between barrel and end-cap, around $|\eta| = 1.4$, is particularly critical, and a scintillator layer, between the two cryostats, has been installed to provide signals which can be used to correct for the energy loss due to the inactive material (cable, services, etc...) in the gap. Beyond a pseudorapidity of 1.8, the presampler is no longer necessary given the more limited amount of dead material and the higher energy of particles for a given p_T [67].

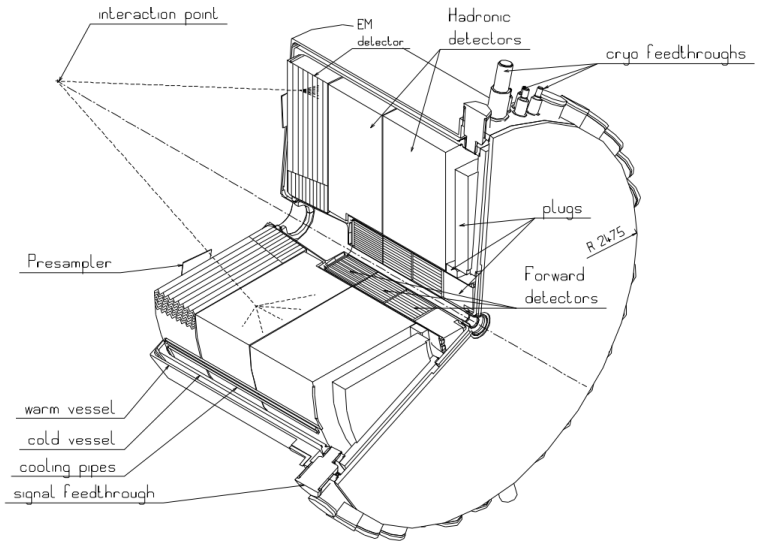

Figure 3.6: End-cap cryostat [35].

3.5.4 Granularity of the samplings

The LAr calorimeter granularity is optimized such that it can provide good physics performance along the covered pseudorapidity. The highest granularity is in the central part of the detector, as shown in Table 3.1, matching the Inner Detector coverage that extends up to $\eta = 2.5$

η range	0 to 1.4	1.4 to 1.8	1.8 to 2.0	2.0 to 2.5	2.5 to 3.0
Presampler	0.025×0.1	0.025×0.1			
Front	0.003×0.1	0.003×0.1	0.004×0.1	0.006×0.1	0.1×0.1
Middle	0.025×0.025	0.025×0.025	0.025×0.025	0.025×0.025	0.1×0.1
Back	0.050×0.025	0.050×0.025	0.050×0.025	0.050×0.025	
Trigger	0.1×0.1	0.1×0.1	0.1×0.1	0.1×0.1	0.2×0.2

Table 3.1: Granularity of the electromagnetic calorimeter ($\Delta\eta \times \Delta\phi$).

The characteristics of front, middle and back layers, in the barrel part, are the following:

- The front layer, placed immediately after the presampler, has a fine granularity to allow neutral pions π^0 rejection up to E_T of 50 GeV and to measure the direction of photons. The depth of this layer is about $6X_0$ (including dead material and presampler) and the granularity is $\Delta\eta \times \Delta\phi = 0.003 \times 0.1$.
- The middle layer is the main layer for reconstructing the energy of electromagnetic showers. It is optimized to be able to contain photon showers up to 50 GeV and has a granularity of $\Delta\eta \times \Delta\phi = 0.025 \times 0.025$. It extends from $6X_0$ to $22X_0$.
- The back layer mostly contain the shower tail and covers the remaining depth of $2X_0$. Close to $\eta = 0$, the total depth is $24X_0$ and in order to leave a minimum of $2X_0$

in the back layer, the middle is reduced to $22X_0$. The back layer has a granularity of $\Delta\eta \times \Delta\phi = 0.050 \times 0.025$.

In total there are about 180 thousand channels in the electromagnetic calorimeter. The trigger itself can't manage all the signals, for this reason a coarser sampling information is sent out for the event selection. Signals coming from the "elementary" cells of the calorimeter are summed to form the "Trigger Towers". These regions are defined as covering 0.1×0.1 in both pseudorapidity and azimuth, except at rapidities larger than 2.5, where they have a size of 0.2×0.2.

Figure 3.7 shows a schematic view of a barrel section where the different longitudinal layers are visible.

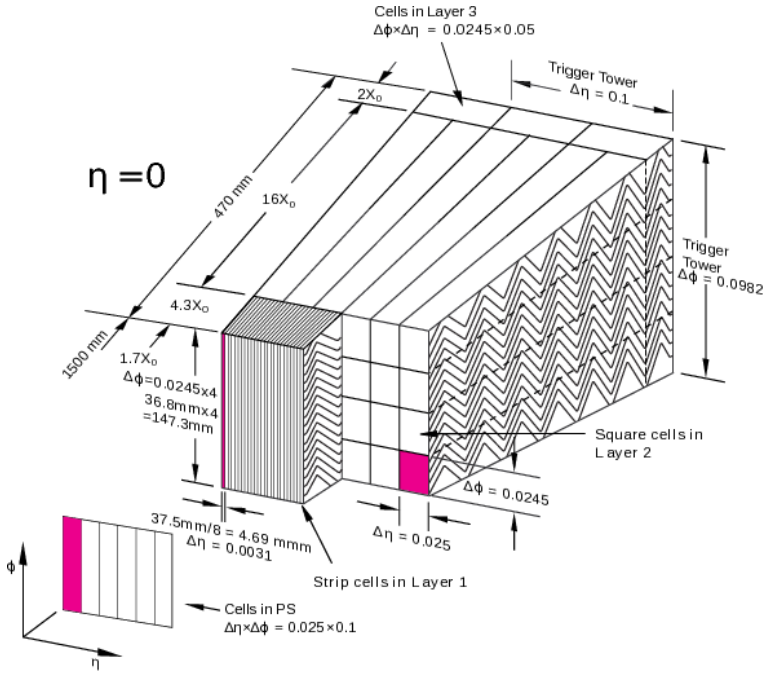

Figure 3.7: Schematic view of LAr calorimeter barrel [64].

3.6 Electronic signal processing

As already mentioned, the LAr calorimeter has lead plates and electrodes folded into an accordion shape and immersed in liquid argon (Figure 3.8, on the left). When charged particles are passing through the liquid argon gap, the liquid is ionized along the tracks (Figure 3.8, on the right), creating electron-ion pairs.

The voltage applied to the outer copper layer of the electrode (while the absorber is kept to ground) separates the electrons from the ions, collecting them on the electrodes. The result of collecting the electrons under the high-voltage field (about 1 kV/mm) between electrodes and absorber, is a triangular shape signal, as shown in Figure 3.9. The peak amplitude of the triangular pulse is proportional to the energy of the incident particle.

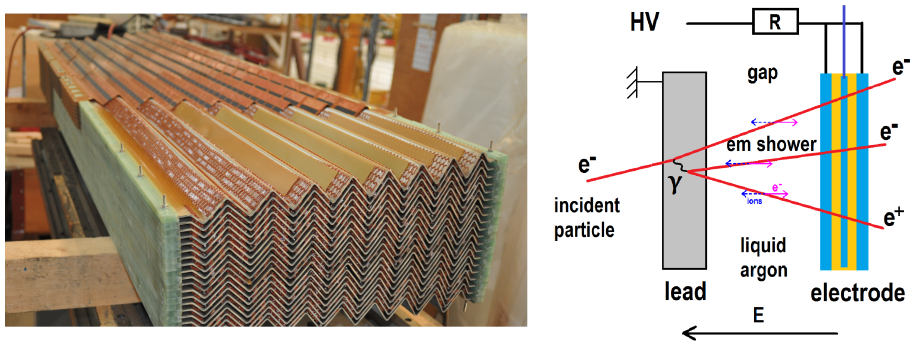

Figure 3.8: Accordion structure of the barrel (left) [68] and readout electrodes behaviour (right) [69].

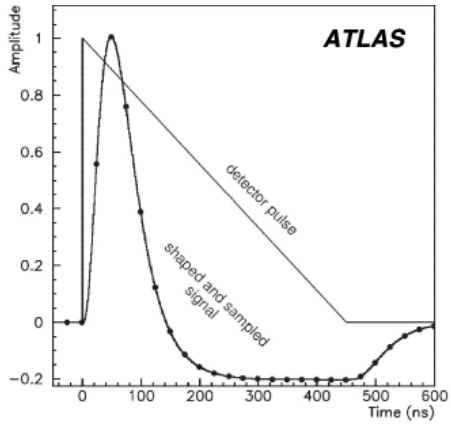

Figure 3.9: Triangular pulse for LAr calorimeter cell. The dots represent the shaped and sampled signal [64].

The ionization signals from all the cells are led outside the cryostats via 114 feedtroughs. Front-end boards housed in crates mounted directly on the feedtroughs, receive the raw signals from up to 128 calorimeter channels. They receive the triangular-shaped signal which is then amplified, split into three gain scales and shaped with a bipolar shaping function. The bipolar-shape pulse is then sampled at the bunch crossing frequency (40 MHz) and the samples stored in analogue form in switched-capacitor array analogue pipelines waiting for the trigger approval. More details about the readout electronics will be given in Section 3.7.

3.7 Readout and trigger system

The current LAr readout system is able to record energies in a range from about 50 MeV to about 3 TeV [70]. The system samples the bipolar-shaped pulses at 40 MHz and sends the digitized information for each bunch crossing upon a Level-1 trigger accept. In Run 1, the maximum Level-1 trigger rate was about 75 kHz, while now, during Run 2, the trigger rate is about 100 kHz.

Figure 3.10 shows a schematic overview of the LAr readout electronics. It can be

mainly divided into two parts, front-end electronics located on the detector and back-end electronics placed in the USA-15 counting room.

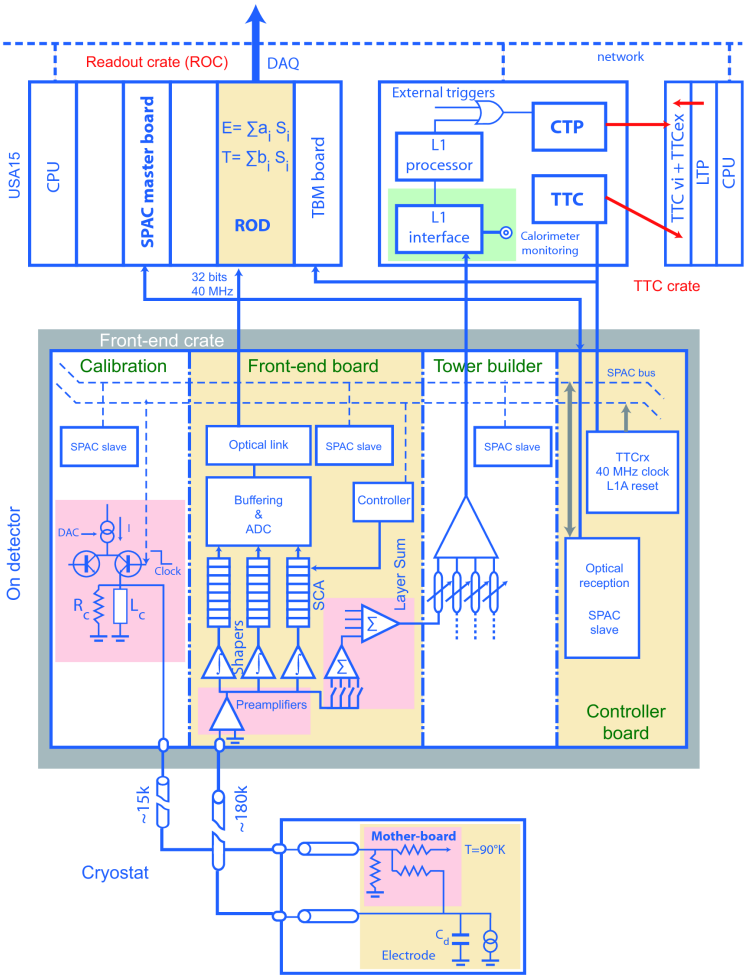

Figure 3.10: Schematic block diagram of the current LAr readout electronics [35].

3.7.1 Front-end electronics

The LAr front-end electronics [71] is mounted directly on the cryostats and it is placed in the gaps between the barrel and the end-cap and on the external face of the end-cap. All the electronics placed here is exposed to radiation, for this reason it has been qualified in terms of radiation tolerance for up to 10 years of LHC operation.

The front-end electronics is placed in 58 front-end crates and each of them contains:

- Front End Boards (FEBs) [72]. The FEBs readout the analogue signals coming from the calorimeter. These are Printed Circuit Boards (PCB) that process signals from 128 channels coming from a specific layer of the calorimeter. On the FEBs, signals are amplified by preamplifier hybrids and then split and sent to shaper chips

that produce three overlapping linear gains, with relative ratio of about 10. The overlapping gains produce an equivalent dynamic range of 16 bit. The shaped signals are sampled at the LHC bunch crossing frequency (40 MHz) and stored on a switched capacitor array analogue pipeline. Upon a L1-Trigger accept, four samples (in Run 2) and five samples (in Run 1) per channel are read out from the switched capacitor array and digitized with a 12-bit analogue-to-digital converter (ADC). The digitized data are formatted, multiplexed, serialized and then transmitted via a 1.6 Gbps optical link off the detector.

- Tower Builder Boards (TBBs) [73]. The analogue signals summed per layer coming from the FEBs are sent to the TBB, where the different layers are summed, forming Trigger Towers with a granularity of $\Delta\eta \times \Delta\phi = 0.1 \times 0.1$. Signals are then sent over analogue cables to the Level-1 calorimeter trigger system.

- Calibration Boards [74]. The calibration boards provide a calibration pulse to the LAr cells. The delivered pulse has an exponential shape, as an approximation of the triangular calorimeter ionization signal.

- Controller Boards [71]. The controller boards receive and distribute the 40 MHz LHC clock and other configuration and control signals.

3.7.2 Back-end electronics

The back-end electronic [75] is placed away from the detector and it is the one that is taking care of digital filtering, formatting and monitoring of the calorimeter signals. Each front-end crate is associated to a VME-based Readout Driver (ROD) crate, which contains:

- Readout Driver Boards (RODs) [76]. In the RODs, signals coming from the FEBs are synchronized with Level-1 trigger and energy, time phase, and quality of the signal are computed.

- CPU Board [75]. The CPU board is a VME processor that controls the ROD crate.

- SPAC Master Board [77]. SPAC stands for a Serial Protocol for ATLAS Calorimeters. The Master Board is able to configure or load parameters into the board of the front-end crate or to read back registers.

- Trigger Busy Module (TBM) [75]. The TBM receives the Trigger, Timing and Control (TTC) signals and delivers them to the RODs, including the 40 MHz LHC clock, Level-1 Accept and more generic commands.

In more detail, groups of eight FEBs are sending data to one ROD, where four Field Programmable Gate Arrays (FPGA) parallelize and check the incoming data. Furthermore, each ROD holds four Processing Units (PUs), each with two DSPs, which calculate through optimal filtering the energy deposited in the calorimeter from the digitized samples [78]. Above a given (programmable) energy threshold, the energy deposition, time and pulse quality are calculated as well. There is also a second energy threshold above which the raw samples values are written out in addition to the results of the optimal filtering algorithm. All these quantities are then sent to the DAQ system via optical fibers.

3.8 L1-trigger readout

A schematic view of the current Level-1 calorimeter (L1Calo) [52] system is shown in Figure 3.11. The trigger receives the analogue Trigger Tower sums from LAr and the equivalent signals from Tile Calorimeter which are then elaborated by the trigger system.

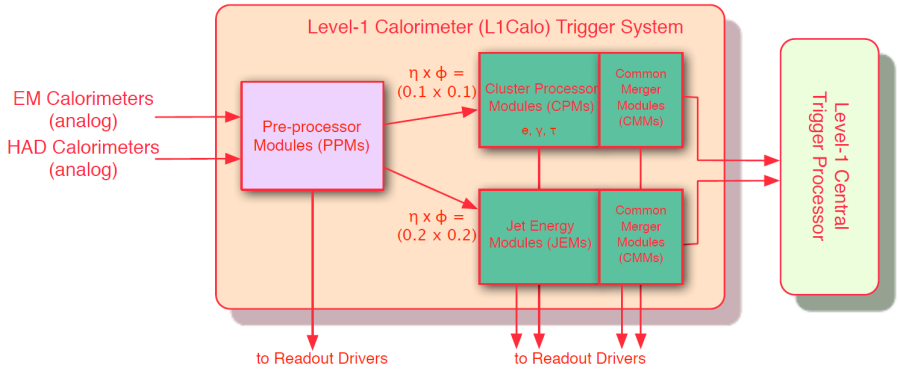

Figure 3.11: Schematic block diagram of the Level-1 Calorimeter (L1Calo) Trigger System [35].

The trigger system consists of:

- Pre-processor Module (PPM) [79]. This module samples the analogue Trigger Tower signals at the TTC clock frequency, 40 MHz, identifies the bunch crossing using the pulse shape and uses a look-up table to compute the transverse energy. In Run 2 a new PPM samples the Trigger Tower at 80 MHz and uses FPGA to compute the transverse energy.

- Cluster Processor Module (CPM) [80]. The CPM receives the digital data from the PPM and isolated electron, photon and τ lepton candidates from the energy deposited in a $\Delta\eta \times \Delta\phi = 0.1 \times 0.1$ Trigger Tower are identified in a given calorimeter region. The Region Of Interest (ROI) is defined using a sliding window algorithm and to compute electromagnetic and hadronic isolation quantities from the surrounding clusters are used.

- Jet Energy Module (JEM) [81]. The JEM receives digital data from the PPM and energetic jet candidates from $\Delta\eta \times \Delta\phi = 0.2 \times 0.2$ jet elements in a given calorimeter region are identified. Also in this case a ROI is defined and the sum of the total transverse energy and the missing transverse energy are also computed.

- Common Merger Module (CMM) [52]. The results from the CPMs and JEMs are transmitted over crate backplanes and summed in CMMs before being sent to the Central Trigger Processor (CTP). In Run 2 this module was replaced by the Common Merger eXtended module (CMX) [82] that in addition sends trigger objects from L1Calo to the topological trigger processor (L1Topo) which also receives data from the Level-1 muon trigger system (L1Muon). In L1Topo combined trigger objects are formed and then sent to the CTP.

After receiving a Level-1 Accept, the L1Calo modules provide readout data and ROIs to the High-Level Trigger (HLT) system via readout drivers.

CHAPTER 4

Liquid Argon Calorimeter Phase-I Upgrade

4.1 Physics requirements and performance

After the Higgs boson discovery in 2012, the LHC experiments started to probe the details of the electroweak symmetry breaking [83]. The Phase-I upgrade of ATLAS will provide new tools to make precision measurements of the Higgs boson and to look for new physics including Supersymmetry and extra dimensions [84].

In order to be able to continue these studies the ATLAS Liquid Argon (LAr) Calorimeter is planning to design, build, and install new trigger readout electronics during the second Long Shutdown (LS2) in 2019 and 2020. The objective of this upgrade is to provide higher-granularity, higher-resolution and longitudinal shower information from the calorimeter to the Level-1 trigger processors, to use shower-shape discriminant in the trigger algorithm for a better event selection. This more detailed information is needed to face the increase of luminosity, to maintain a low p_T lepton threshold (about 25 GeV) and keep the same trigger bandwidth with respect to Run 2 (2015-2018) [85]. Figure 4.1 shows a simulation of the Level-1 trigger rates as a function of the electromagnetic E_T threshold, assuming Run 2 conditions (blue points) and Run 3 conditions for two sets of cuts (green and black triangles). The variables used in the cuts will be described in Section 4.2. From the simulation results, a threshold of 25 GeV, for Run 2 data, corresponds

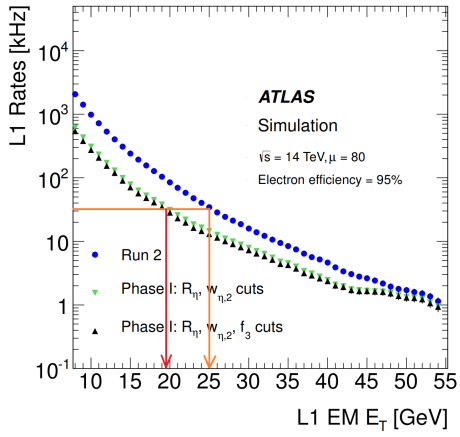

Figure 4.1: Simulation results of Level-1 trigger rates as a function of the electromagnetic E_T threshold, assuming Run 2 conditions (blue points) and Run 3 conditions for two sets of cuts (green and black triangles). In orange and red are highlighted the trigger rate and the threshold values for Run 2 and Phase-I, respectively. [85]

to a trigger rate of about 30 kHz (orange). With the proposed upgrade and maintaining the same trigger rate, the threshold can be lowered by 6 GeV (red).

The performance of the Level-1 trigger, based on the calorimeter information, will be improved by:

- making use of shower-shape variables for a more effective identification of electrons, photons and τ leptons [85]
- and sharpening the electromagnetic, jet, and E_T^{miss} efficiency turn-on curves using advanced reconstruction algorithms and utilizing pileup subtraction techniques [85].

These improvements are possible thanks to the following upgrades:

- a new finer calorimeter segmentation and longitudinal information will be available in the Level-1 trigger. The front-end electronics will take care of summing the calorimeter cells together to form Super Cells.
- Super cell signals digitization precision will improve by at least a factor of 4. The quantization scale and the dynamic range of the digitizers are optimized in each η-region and for each layer.
- The deposited E_T will be calculated at each bunch crossing through optimized algorithms, as it is done currently in the LAr RODs in the main readout.

The existing calorimeter trigger information is based on Trigger Towers, as already mentioned in Chapter 3. They correspond to the sum of the analogue signals along the longitudinal layers of the calorimeter, for an area of $\Delta\eta \times \Delta\phi = 0.1 \times 0.1$.

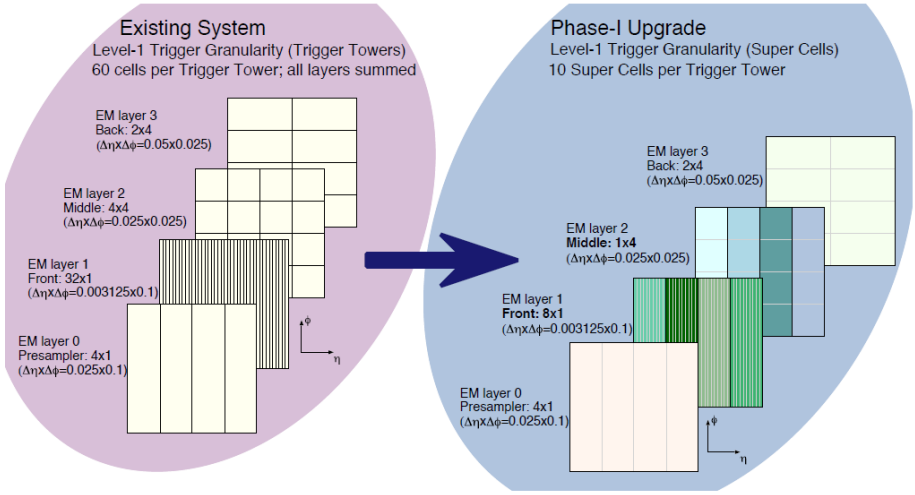

Figure 4.2: Comparison between Trigger Tower (left) and Super Cell (right) segmentation. The signal granularity will be increased of about 10 times [85].

The new finer granularity scheme, as shown in Figure 4.2 and in Table 4.1, is based on the Super Cells, which will provide information for each layer of the calorimeter and a finer segmentation in the front and middle layers of the Electro-Magnetic Barrel (EMB) and End-Cap (EMEC) for $|\eta| \leq 2.5$.

Layer		Elementary Cell	Trigger Tower		Super Cell	
		$\Delta\eta \times \Delta\phi$	$n_\eta \times n_\phi$	$\Delta\eta \times \Delta\phi$	$n_\eta \times n_\phi$	$\Delta\eta \times \Delta\phi$
0	Presampler	0.025×0.1	4×1	0.1×0.1	4×1	0.1×0.1
1	Front	0.03125×0.1	32×1		8×1	0.025×0.1
2	Middle	0.025×0.025	4×4		1×4	0.025×0.1
3	Back	0.05×0.025	2×4		2×4	0.1×0.1

Table 4.1: Comparison of the Trigger Tower and Super Cell granularity in the LAr barrel, in terms of both elementary cells and $\Delta\eta \times \Delta\phi$ [85].

4.1.1 Super cells energy reconstruction

The Super Cell energy reconstruction along the trigger path of the LAr calorimeter has the aim of estimating the energy deposited by particle showers with high precision and assigning this information to the correct bunch crossing. The energy information is estimated from an optimal filtering algorithm based on a linear combination of the signal samples. The algorithm coefficients are calculated to minimize the contribution of electronics and pileup noise thanks to the knowledge of the pulse shape [85].

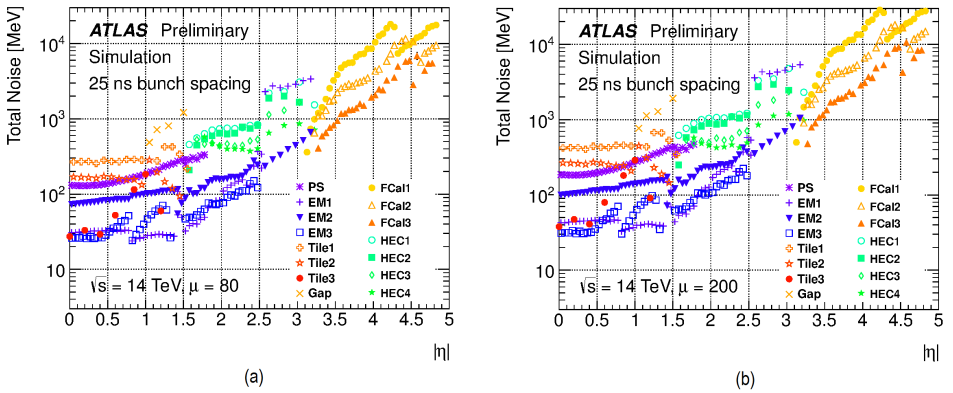

Figure 4.3: Simulated noise as a function of $|\eta|$ for proton-proton collisions at 14 TeV and a bunch spacing of 25 ns. The expected number of pileup events per bunch crossing is $\langle\mu\rangle = 80$ (a) and $\langle\mu\rangle = 200$ (b), corresponding to the nominal Phase-I and Phase-II pileup conditions [85].

As the luminosity increases, the total noise is dominated by the pileup. In Figure 4.3 is shown the expectation for the total noise, for proton-proton collisions at 14 TeV and a bunch spacing of 25 ns, for Phase-I (a) and Phase-II (b) operations.

4.1.2 Super cells energy resolution

The high-granularity and high-resolution data will also allow a better precision in the reconstruction of electromagnetic clusters [85]. As already mentioned in the previous section, the E_T in each Super Cell is obtained through an optimal filtering algorithm applied to the raw input data, which have a quantization scale of 125 MeV in the middle electromagnetic layer and 32 MeV elsewhere. Electromagnetic clusters are then formed from the Super Cells over an area of $\Delta\eta \times \Delta\phi = 0.075 \times 0.2$. Corrections are applied to each layer and as a function of η to account for the material in front of the calorimeter.

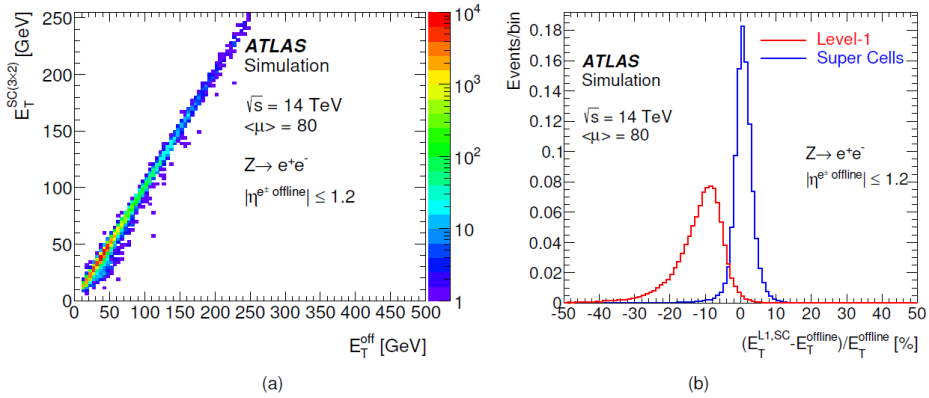

Figure 4.4: (a) Distribution of E_T in 3×2 Super Cells and E_T for offline reconstructed electrons and (b) resolution comparison between Super Cells (blue) and Level-1 (red) for simulated $Z \rightarrow e^+e^-$ events in the LAr barrel [85].

Figure 4.4 (a) is showing the correlation between the resulting Super Cell cluster ($E_T^{SC(3\times2)}$) and the transverse energy of the offline reconstructed electrons (E_T^{off}). Figure 4.4 (b) shows the improved energy resolution using the Super Cell cluster transverse energy, compared to the existing Level-1 E_T. The offset is due to the fact that the Super Cell (blue) plot is calibrated, while no calibration is applied to the Level-1 (red) plot.

4.2 Level-1 trigger performances

As previously mentioned, the trigger rates and instantaneous luminosity expected in the next years at the LHC require more sophisticated tools, for example to separate electrons and photons from low p_T jets. At the trigger level, it will be possible to identify electromagnetic showers due to electrons and photons, thanks to to the higher Super Cells granularity, using shower shape variables.

4.2.1 Shower shape analysis

The usage of Super Cells makes possible to further reduce the jet backgrounds through the use of shower shape variables. In the following, the performances of three variables are presented.

R_η defined, in the middle layer (2), as the E_T measured in a group of Super Cells 7×2 divided by the E_T measured in a group of Super Cells 3×2:

$$R_\eta = \frac{E^{(2)}_{\mathrm{T},\Delta\eta\times\Delta\phi=0.075\times0.2}}{E^{(2)}_{\mathrm{T},\Delta\eta\times\Delta\phi=0.175\times0.2}} \tag{4.1}$$

f_3 is the ratio between the E_T measured in the back electromagnetic layer (3) in an area $\Delta\eta \times \Delta\phi = 0.2 \times 0.2$ with respect to the sum of energy deposited in all three layers; the energies in the front (1) and middle (2) layers are reconstructed in the

area $\Delta\eta \times \Delta\phi = 0.075 \times 0.2$:

$$f_3 = \frac{E^{(3)}_{\mathrm{T},\Delta\eta\times\Delta\phi=0.2\times0.2}}{E^{(1)}_{\mathrm{T},\Delta\eta\times\Delta\phi=0.075\times0.2} + E^{(2)}_{\mathrm{T},\Delta\eta\times\Delta\phi=0.075\times0.2} + E^{(3)}_{\mathrm{T},\Delta\eta\times\Delta\phi=0.2\times0.2}} \tag{4.2}$$

$\mathrm{w}_{\eta,2}$ defines the spread of the shower in the middle electromagnetic layer (2) in a 3×2 Super Cell region. Here the sum are running over the Super Cells:

$$\mathrm{w}_{\eta,(2)} = \sqrt{\frac{\sum\left(E^{(2)}_{\mathrm{T}} \times \eta^2\right)_{\Delta\eta\times\Delta\phi=0.075\times0.2}}{E^{(2)}_{\mathrm{T},\Delta\eta\times\Delta\phi=0.075\times0.2}} - \left(\frac{\sum\left(E^{(2)}_{\mathrm{T}} \times \eta\right)_{\Delta\eta\times\Delta\phi=0.075\times0.2}}{E^{(2)}_{\mathrm{T},\Delta\eta\times\Delta\phi=0.075\times0.2}}\right)^2} \tag{4.3}$$

In Figure 4.5 some examples of the separation power of these variables are shown, in the case of an electrons from $Z \to e^+e^-$ events and background jets.

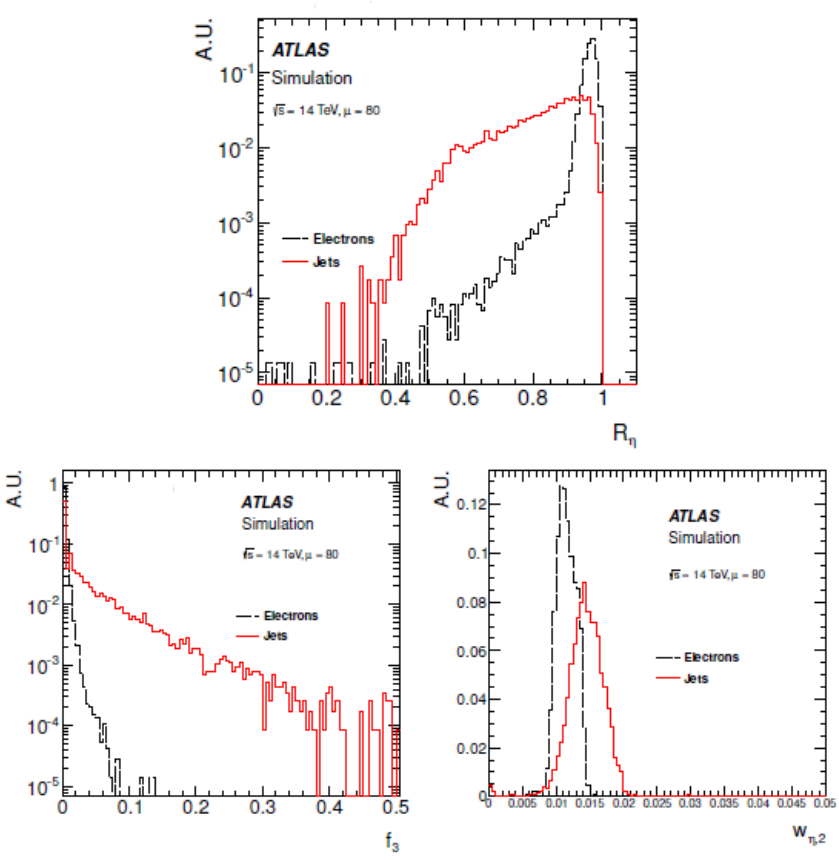

Figure 4.5: Distributions of the variables R_η, f_3 and $w_{\eta,(2)}$, to distinguish between electrons (black) and jets (red), all normalized to unit area. On the y-axis A.U. stands for Arbitrary Unit [85].

R_η distribution shows a narrow peak for the electrons, because the showers generated by them tend to remain inside the 3×2 Super Cells space. For jets the distribution is

much broader. A very similar behaviour is shown for f_3. In this case the electromagnetic shower would be mostly concentrated in the first and second layer, having the distribution with a peak around zero. For what concern $w_{\eta,(2)}$, it describes how broad is the shower inside the 3×2 Super Cells.

4.3 Implementation of the trigger readout system

In order to be able to make use of the improved trigger signals, both the LAr front-end and back-end electronics will be modified and extended during the Phase-I Upgrade. Figure 4.6 shows the calorimeter trigger electronics, with the upgraded and new components outlined in red.

The upgraded trigger branch starts with the signal produced by a linear mixer (providing shaped sum of four input channels), on the FEBs, in the front-end electronics. A new Layer Sum Board (LSB) will be mounted on each FEB and will take care of summing cells to produce Super Cell signals.

These signals, passing through the baseplane, are then sent to the LAr Trigger Digitizer Board (LTDB) where they are digitised and transmitted to the back end via optical links. The thresholds of the Level-1 trigger system are in units of E_T, while the signals from the calorimeters are proportional to energy. In the present system, there are different steps to apply the conversion factor of $\sin\theta$ to the signals: 1) through the choice of preamplifier, 2) the gain of the linear mixer in the FEB, 3) through the η-dependent gains that are applied in the TBB for the electromagnetic calorimeters. In the upgraded system, the choice of preamplifiers and gains in the FEB will remain unchanged. How the η-dependent gains will be applied is not yet decided; they may be included in the analogue section of the LTDB or as a factor in the calibration constant which is applied in the back-end electronics [85].

4.3.1 Front-end electronics

The LAr Calorimeter readout architecture will remain roughly the same. The proposed modifications in the Phase-I upgrade will extend the front-end electronics with the following changes:

- LSB. The current LSB is producing $\Delta\eta \times \Delta\phi = 0.1 \times 0.1$ analogue sums for a certain layer. The new LSB will produce finer Super Cell signals in the front and middle layers.
- Baseplane. The new baseplanes will allocate new slot(s) for the LTDBs, keeping the other front-end electronic board slots intact. Furthermore, more signals will be transferred from the FEBs to the LTDBs (and the legacy signals from the LTDB to the TBB).
- LTDB. The LTDB will digitize Super Cell signals and transmit them to the back-end digital processors. At the same time, the old path will remain partially in place: in the LTDB, before the digitization, the signals are summed and sent to the TBB to recreate the current $\Delta\eta \times \Delta\phi = 0.1 \times 0.1$ towers. This is done to keep the current analogue trigger chain unaffected.

LTDB

The LTDB is the key-board for the Phase-I upgrade. A schematic view of the board is shown in Figure 4.7, while a picture of the board already built and under test is visible

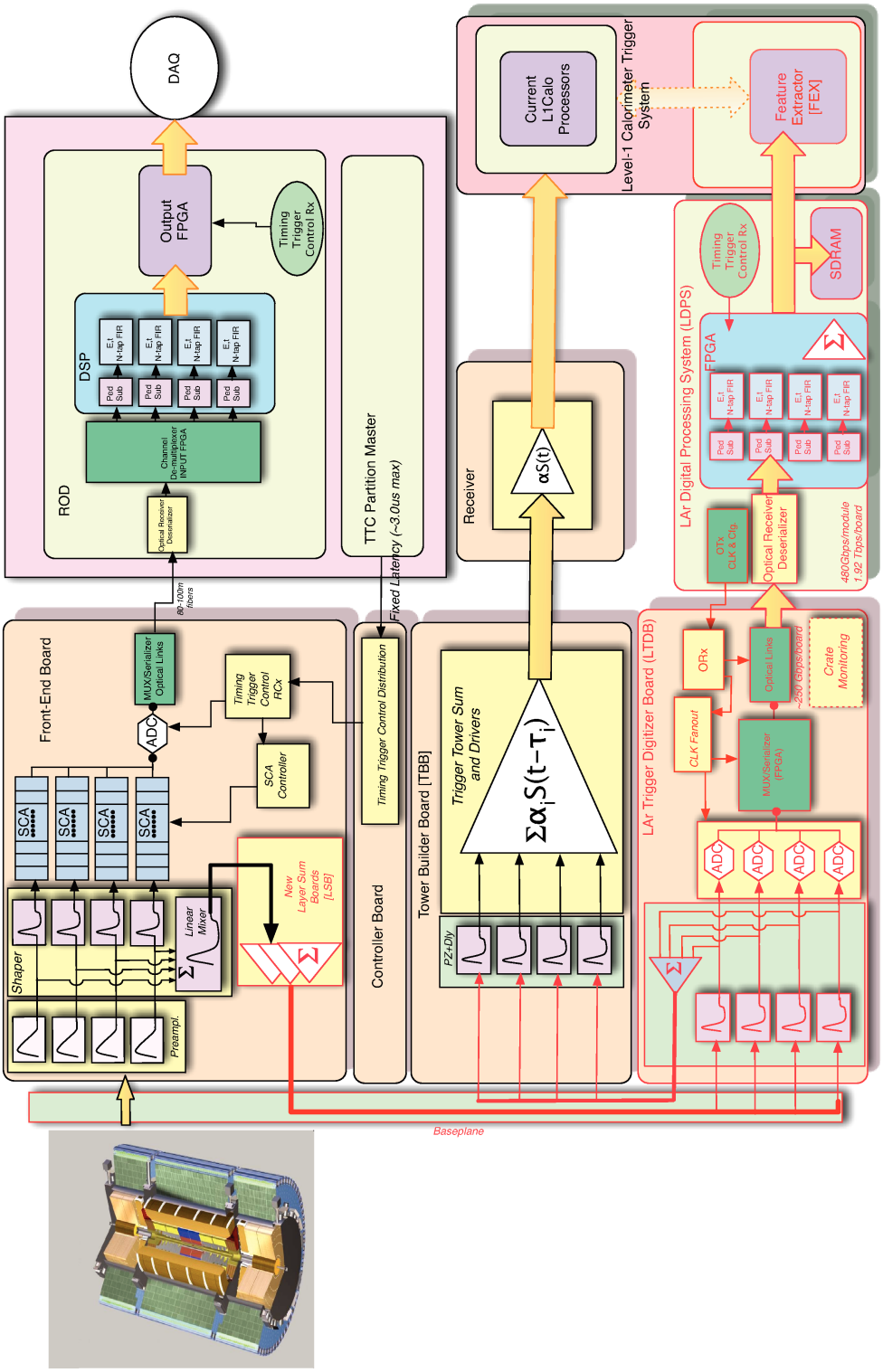

Figure 4.6: Schematic view of the Phase-I upgrade LAr trigger readout architecture [85].

in Figure 4.8.

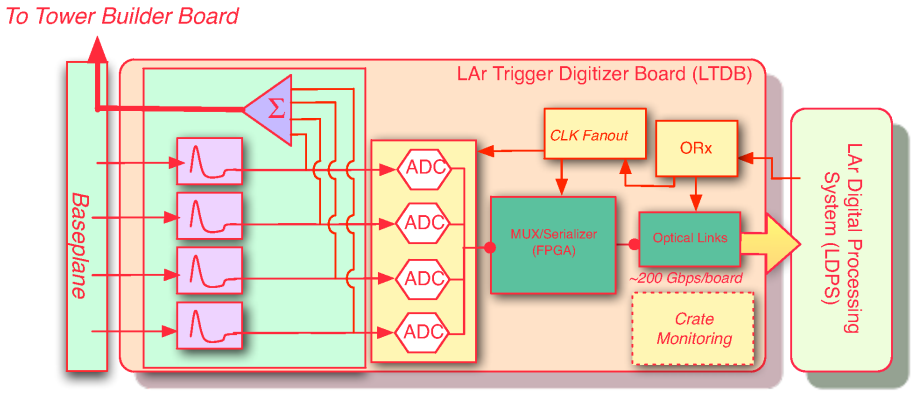

Figure 4.7: Schematic block diagram of the LTDB [85].

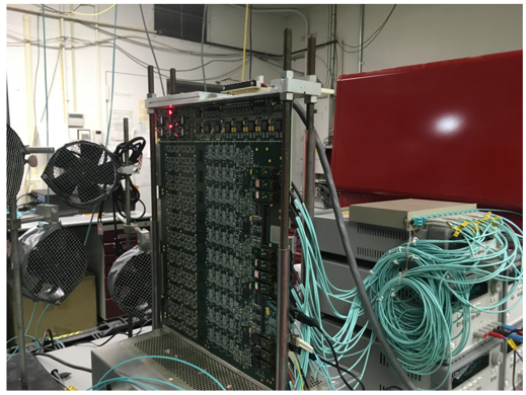

Figure 4.8: LTDB board under test.

There will be 124 LTDB boards and each of them will process up to 320 Super Cell analogue signals. There are typically 10 Super Cells per Trigger Tower in the barrel part, but less in the end-caps. One LTDB per baseplane is present in each EMB, HEC, and EMEC Standard Crate, but in the EMEC Special and FCal crate, two are required.

Analogue signals arrive to the LTDB. The analogue section of the LTDB is important to provide the appropriate gain, shaping, offset and common mode such that the 12-bit dynamic range of the ADC is utilized without compromising the signal to noise ratio. The analogue signals take two different paths: they can be sent to the TBB to recreate the legacy Trigger Tower or to the ADC mounted on the LTDB. Then, on the digital side, as shown in Figure 4.9, two paths are present: the data link and the Timing, Trigger and Control (TTC) link. The digital processing chain drives the data link and is responsible for signal digitization and transmission. Here the Super Cell signals are multiplexed and serialized by a custom chip called LOCx2, and transmitted via optical transmitters over optical fibers. The TTC link is responsible for the transmission of clock signals, slow control and monitoring communications.

As the entire LAr front-end electronics is exposed to a high radiation level, all the

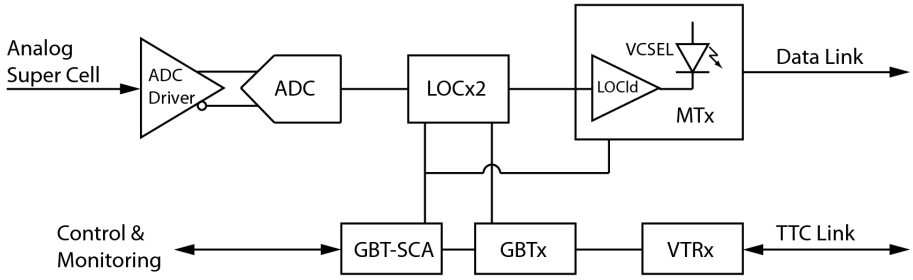

Figure 4.9: Schematic block diagram of the digital section of the LTDB, where two signal flow paths (the data link and the control TTC link) are shown.

components need to undergo extensive radiation qualification tests [86]. The LTDB components have to be tested as well, especially the newly developed ASICs: the ADC, the serializer and the driver for the optical interface.

The ADC, a custom device, is a quad-channel 12-bit 40 MS/s pipeline SAR ADC, designed with 130 nm technology [87]. Recently, the radiation tolerance of the prototype ADC chip has been tested and confirmed up to 10 MRad (the requirement is 100 kRad).

LOCx2 is a dual 8×12 bit serializer, with 5.12 Gbps of output [88]. LOCld is a VCSEL (Vertical Cavity Surface-Emitting Lasers) driver, designed for the optical interface [89]. Both are fabricated with 250 nm Silicon-on-Sapphire CMOS technology and have been irradiated up to 200 kRad. No changes in the output eye diagrams have been observed.

4.3.2 Back-end electronics

The LAr calorimeter back-end electronics system, shown in Figure 4.10, is called the LAr Digital Processing System (LDPS). This system receives the digital Super Cell data from the 124 LTDBs, reconstructs the E_T of each Super Cell and transmits the results to the Level-1 calorimeter trigger system every 25 ns. The LDPS consists of 32 ATCA

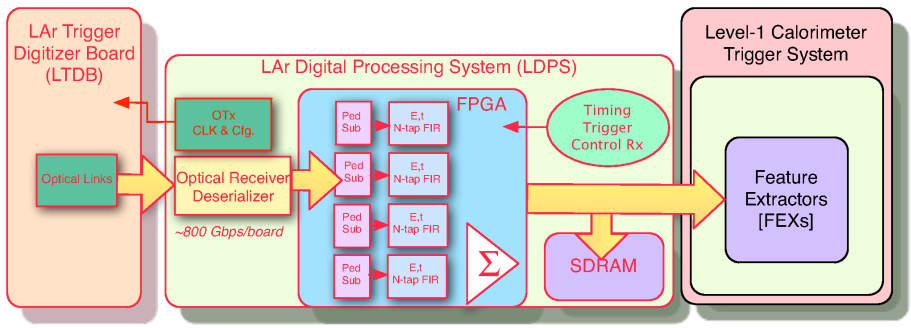

Figure 4.10: Schematic view of the LDPS system [85].

Carrier boards, each one equipped with four Advanced Mezzanine Cards (AMC), called LATOMEs [85].

LATOME board

The LATOME board conforms to the standard ATCA protocol [90] and measures 156 mm $\times$ 73.5 mm to allow four AMCs to fit a Carrier board. The board, shown in Figure 4.11, is built around a high-performance and state-of-the-art commercial FPGA, an ARRIA-10 from Intel FPGA (formerly ALTERA) [91]. The Arria-10 FPGA is built with

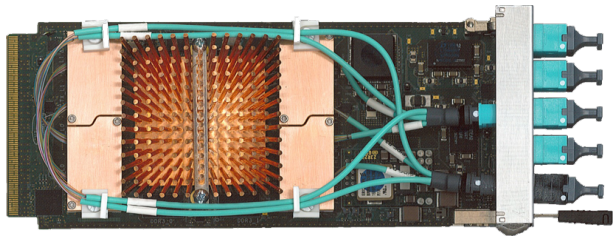

Figure 4.11: LATOME board.

20-nm System-on-Chip (SoC) process technology, with more than 1 million of logic cells, more than 1500 DSPs, about 650 I/O pins and 2700 Block RAM (BRAM), 66 transceiver channels and a maximum speed per each channel of 17.4 Gbps, PCIe and external memory interface for DDR3 and DDR4 [91].

Optical receivers and transmitters are performing the reception and the transmission of the ADC data from the LTDB to L1Calo. To maximize the density of components, and minimize the size of the system, each AMC will handle 48 fibers for reception and 48 for transmission.

Between the AMC and the Carrier board various interfaces are defined:

- power conversion from the AMC 12 V supply to the necessary voltages for components
- one GigaBit Transceiver (GBT) [92] based link to the Carrier board
- one GbE link which can be used for the AMC FPGA configuration and monitoring, or the flash memory programming
- four LVDS links where TTC commands and clock are routed to the LATOME
- high-speed links to send monitoring data to the ATCA carrier board
- one single-ended link for signals like a boundary scan bus (JTAG) that allows for the initial programming of the FPGA

The role of the LATOME board is to receive 12-bits ADC data from the LTDBs at 5.12 Gbps/fiber, extract the transverse energy for each Super Cell every 25 ns, and send all the data to the Level-1 calorimeter trigger system at 11.2 Gbps. The board will also have to monitor the data and send reports to central data acquisition system upon request along the 10 GbE network.

More details about the LATOME firmware will be presented in Chapter 6.

Carrier board

Each Carrier board is able to hold four AMCs and provides a variety of different communication functionalities as shown in the schematic diagram in Figure 4.12. In this picture, the main components for the Carrier board are:

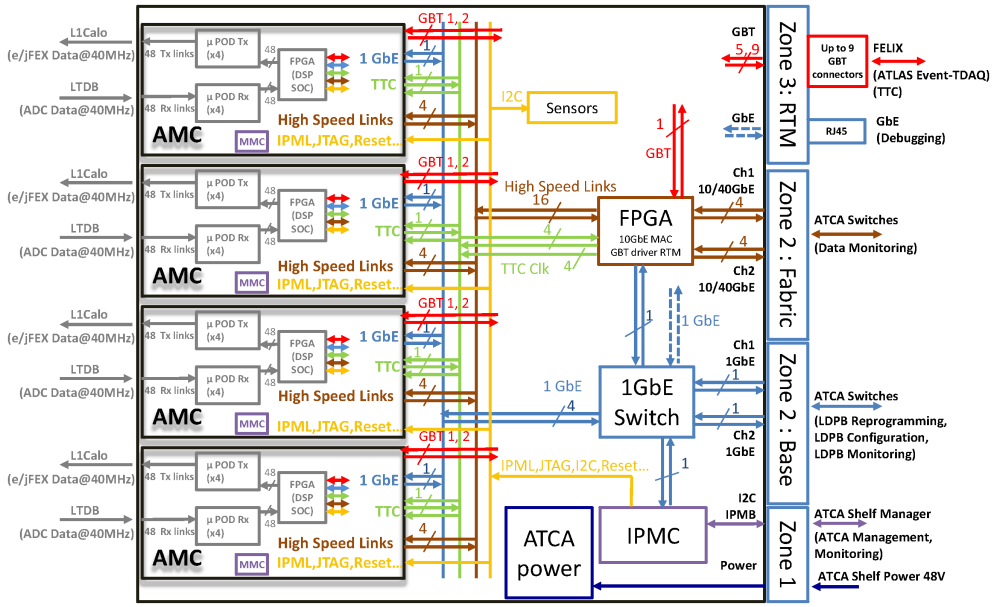

Figure 4.12: Schematic view of a Carrier board [85].

- FPGA (brown) has two main functions: to interface the high-speed links coming from the LATOMEs to the GbE network for the data monitoring, and the transmission of the TTC information (green) from the GBT (red) to all the AMCs.
- GbE switch (light blue) has to take care of the configuration, monitoring and reprogramming of both LATOMEs and carrier FPGA.
- IPMC (purple) is performing the ATCA management.
- ATCA power block (dark blue) provides power to the Carrier components and to the AMCs.

A picture of a Carrier board built and ready for testing is shown in Figure 4.13. More details about the FPGA firmware will will described in Chapter 6.

4.3.3 FEXs

The data coming from the LDPS system will be processed by the upgraded Level-1 Calorimeter (L1Calo) Trigger, with the new Feature EXtractors (FEXs) [93, 94, 95]. Three additional feature-identification subsystems are going to be added to the L1Calo, as shown in Figure 4.14.

- The electromagnetic Feature Extractor (eFEX) identifies isolated e/γ and τ candidates using four processor FPGAs on each board. The eFEX makes use of the Super Cell information from LAr to explore cluster shape and width variables.
- jFEX The jet Feature Extractor (jFEX) uses four processor FPGAs as well for clustering and identification of jets and large-area τ. The jFEX receives the Trigger Tower information from the LAr and Tile calorimeter, as the granularity of the Super Cells is not required for jet reconstruction.

- The global Feature Extractor (gFEX) uses $\Delta\eta \times \Delta\phi = 0.2 \times 0.2$ Trigger Tower sums of Super Cells from the calorimeter data in order to identify large-radius jets.

The FEXs data will be transmitted to L1Topo over optical fibres. In L1Topo, the data will be used as input to topological algorithms and the results are sent to the CTP which will generate the L1A signal in case of interesting events.

The FEXs will improve the existing L1Calo electronics, operating in parallel with the Cluster Processor (CP) and Jet Energy Processor (JEP) systems. Once the outputs of the eFEX, jFEX and gFEX will be validated, the removal of the CP and JEP systems from the L1Calo processing chain will be an option [93].

4.4 Towards the ATLAS LAr calorimeter Phase-II upgrade

The ATLAS LAr calorimeters plan is to install the upgraded trigger electronics during the LS2, starting in 2019. In order to be able to continue the LAr calorimeter system operations at full potential, even during the High-Luminosity LHC (HL-LHC) era, a replacement of the main readout electronics is foreseen for the third Long Shut-down (LS3).

Figure 4.15 shows the readout architecture planned for the Phase-II upgrade. The Phase-I upgrade components will be maintained during the HL-LHC run, while the trigger electronics (grey in the figure) will be decommissioned. The main readout electronics will be completely replaced by new FEB, able to digitize the signals at each bunch crossing and send them over fast optical links to pre-processor readout modules [96].

In the baseline trigger configuration for Phase-II, a single hardware Level-0 trigger system is foreseen. The Level-0 trigger will be made of L0Calo, L0Muon, Global Trigger and CTP. L0Calo will be based on L1Calo system from Phase-I, with minor changes. It will use Super Cells calorimeter data at 40 MHz and will identify physics objects within 10 μs, to respect the trigger requirements. In case of a Level-0 trigger accept (L0A), trigger data and detector data will be transmitted through the DAQ system at 1 MHz to the Event Filter (EF). The event filter will be composed of processing farm and hardware-tracking that must reduce the event rate to 10 KHz, for permanent storage.

The Phase-I upgrade project is fully compatible with the overall ATLAS upgrade long-term plans and can be considered the first step of a single, staged upgrade path.

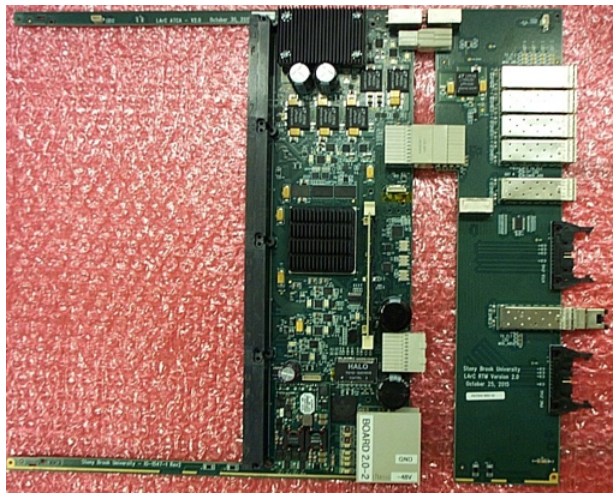

Figure 4.13: Carrier board.

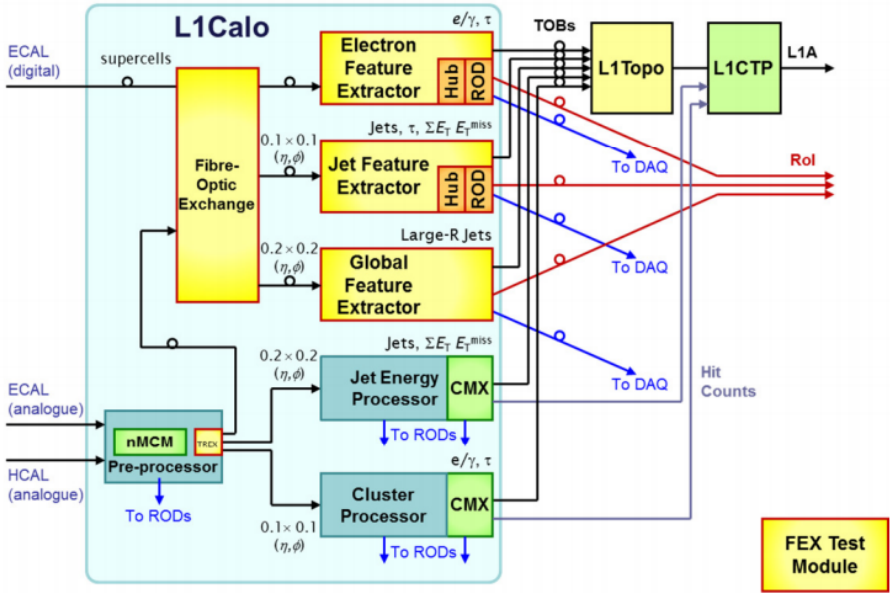

Figure 4.14: ATLAS Level-1 Calorimeter Trigger Phase-I Upgrade [93].

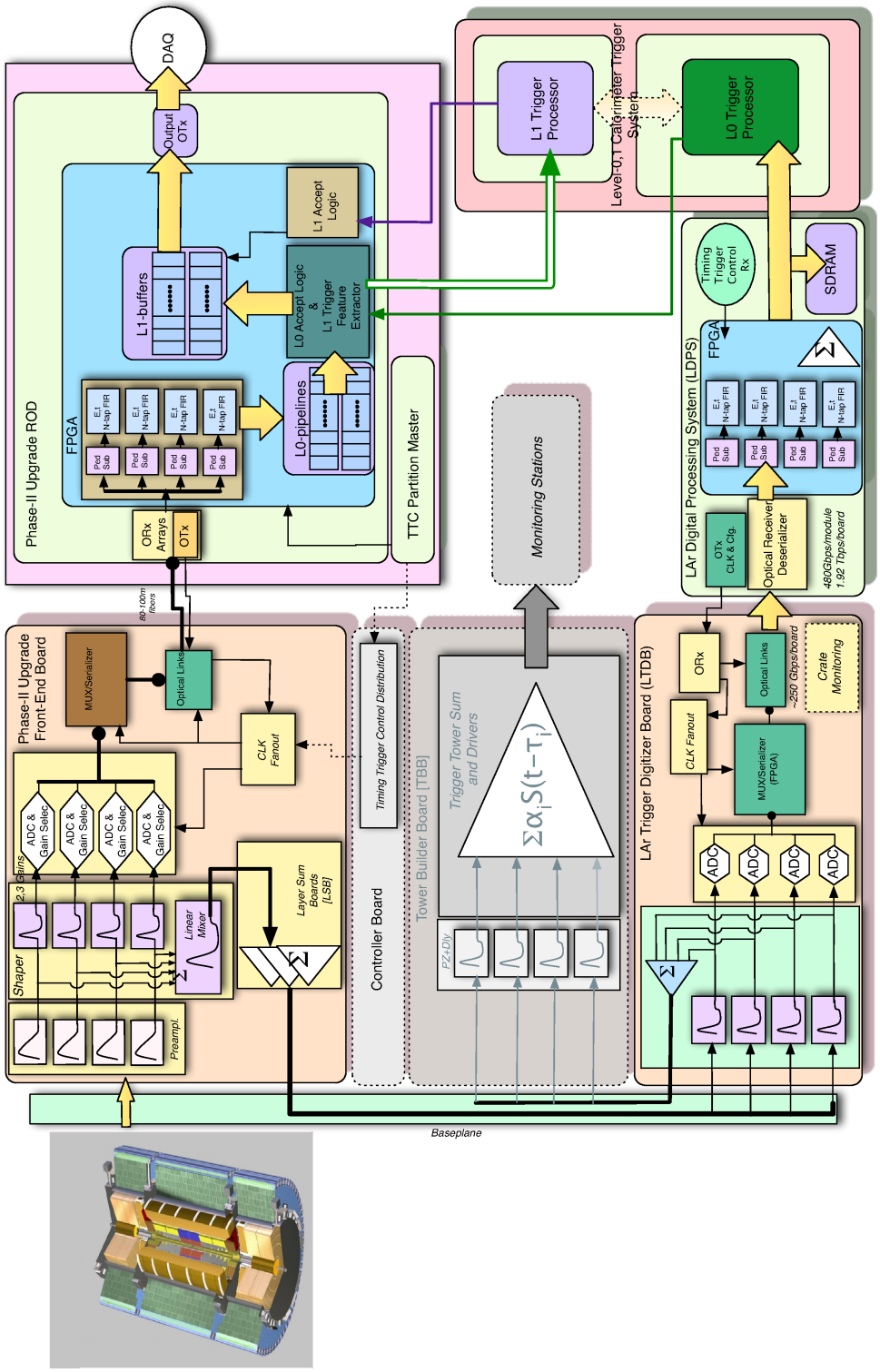

Figure 4.15: Schematic view of the Phase-II upgrade LAr trigger readout architecture [85].

CHAPTER 5

Phase-I Upgrade Demonstrator

5.1 Demonstrator system

In order to evaluate the technical details and performances of the LTDB, a pre-prototype board of the LATOME mezzanine was produced. The pre-prototype is called ABBA (ATCA test Boards for Baseline Acquisition) and mounts three Altera Stratix IV FPGAs [97]. More details about what is an FPGA can be found in Appendix A. ABBA uses a custom protocol interface to communicate with the front-end and 10 Gbps Ethernet to communicate with a PC. A photograph of one this board is shown in Figure 5.1. The two FPGAs on the left side of the board are called *front FPGAs*, as they are receiving the data from the front-end, while the third on the right is called *back FPGA*, preparing and sending the data to the PC.

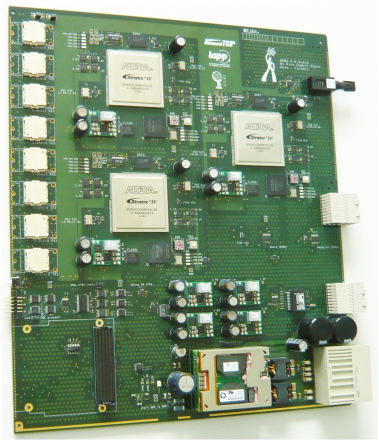

Figure 5.1: The LDPS pre-prototype (ABBA), equipped with three Altera FPGAs.

A demonstrator system made of LTDB pre-prototypes and ABBAs, has been operated first at the LAr Electronic Maintenance Facility (EMF) and then in the Underground Servis hall of ATLAS (USA15), integrated in the ATLAS detector. A simple schema of the demonstrator data readout path is shown in Figure 5.2.

The analogue data, coming from the FEBs of a half barrel crate, are sent to one LTDB, where they are digitized and formatted. The digitised data of one LTDB are collected by two ABBAs and stored in a circular buffer (able to keep them up to 12.8 μs) to be later selected upon a L1A command coming along the TTC. After the selection, data are sent along 10 GbE link to a PC which is running an application called *Abba*. This application

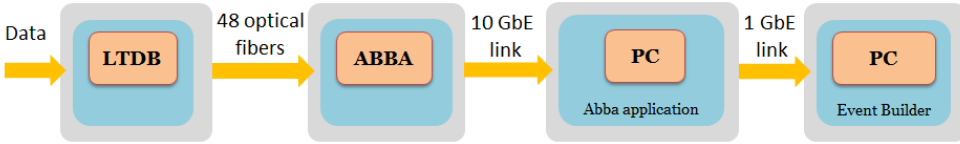

Figure 5.2: Schematic view of readout chain for the demonstrator system.

is the connection between the TDAQ software and the ABBA board. The Abba application has the role of configuring the firmware with the proper settings at the beginning of the data taking, as well as polling the event fragments from the boards, sending Internet Protocol bus (IPbus) [98] requests on the 10 GbE links. Once the Abba application receives an event fragment, it notifies another TDAQ application, called *Event Builder*. The Event Builder receives the event fragments along a 1 GbE link and builds the complete event, storing it on the machine where the application is running. The same readout chain is used both in EMF and in USA15.

5.1.1 The EMF setup

The EMF setup was built to have a full system with easy access and to verify that the new components of the upgraded system would not degrade the performance of the ATLAS main readout. A schematic picture of the EMF setup is shown in Figure 5.3.

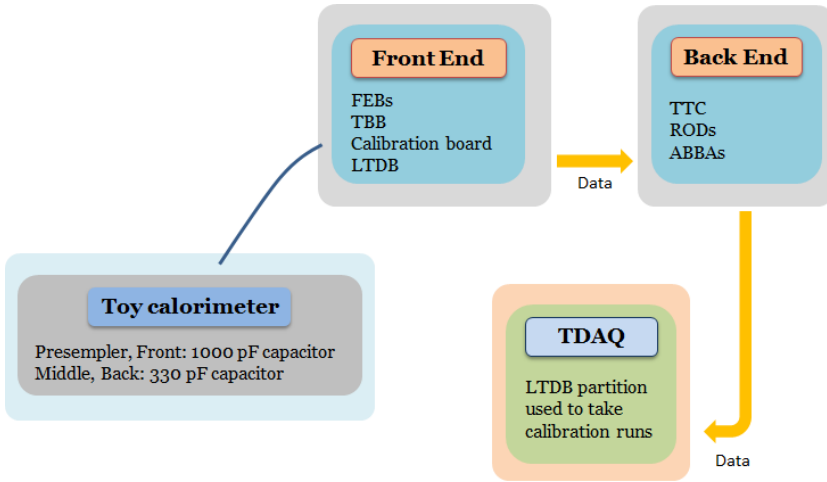

Figure 5.3: Schematic view of readout electronic chain for EMF test system.

A "toy calorimeter" is used to simulate the calorimeter cells. Each cell is simulated by a capacitive load, plugged on the back side of the baseplane. The calibration board pulses the capacitors for injecting signals into the FEBs, as if the signals come from the calorimeter. The FEBs receive, amplify and sum the analogue signals. In the half front-end crate there is one calibration board, one monitor board and one controller board, as also shown in Figure 5.4. One FEB (PS) reads out the pre-sampler of the toy calorimeter, six FEBs (F0-5) read out the front layer, four FEBs (M0-3) read out the cells from the middle layer and two FEBs (B0,1) receive the signals from the back layer. The baseplane

used currently in the half front-end crate in EMF is an EMEC standard baseplane (Figure 5.4, on the right). The new EMEC baseplane has the same number of slots (19) as the current EMEC baseplane used in ATLAS. Additionally, there is one TBB, used to sum the toy-calorimeter signals to form the trigger towers and send them to the trigger receiver crate. One LTDB pre-prototype is placed in the front-end crate, digitizing and formatting the calibration signals for ABBA.

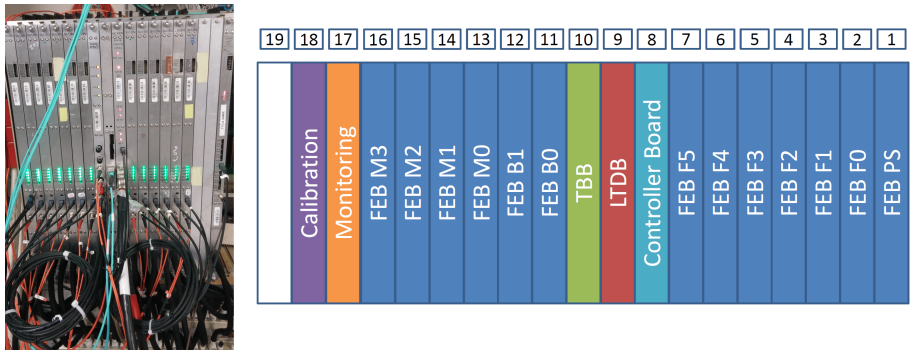

Figure 5.4: Left: half front-end crate of the demonstrator system in EMF. Right: slot assignments for the new EMEC standard baseplane, used for the crate in EMF.

The mapping of the calibration board to the FEBs is one to one, meaning that channel 1 of the calibration board pulses channel 1 of all FEBs, channel 2 pulses channel 2 of all FEBs, etc. On the detector, this is done in a different way in order to minimize cross-talk effects between the calorimeter cells when the calibration is performed [99].

The standard ATLAS TDAQ software is used for the configuration and the control of all electronic components and handles the data monitoring and readout. It can run independent partitions, defined depending on the hardware system. A special partition is set up in the software, for EMF.

At the moment, two ABBAs are placed in an EMF ATCA crate, as shown on the left side of Figure 5.5. On the right side of the figure, the machines used for monitoring and

Figure 5.5: Left: two ABBAs in the ATCA crate, one connected to the optical fibers (leftmost board in the crate) and one disconnected (rightmost board in the crate). Right: machines used for monitoring and reconfiguration.

reconfiguration are shown. One of them is dedicated to the communication with the GLIB, a board which allows the reconfiguration and monitoring of the LTDB. The GLIB board is described in Section 5.1.2. The second machine is where the *Abba* application is running. From this machine it is possible to reset and monitor the ABBAs, thanks to standalone scripts described in Section 5.7.1.

The reconfiguration of the ABBA FPGAs is done from a laptop placed close to the boards. USB cables connect ABBAs to the laptop where the Quartus II software [100], provided by Altera, is installed.

LTDB to ABBA connection

Currently, the EMF LTDB is connected to one ABBA. The data coming along 48 optical fibers can be sent to the two front FPGAs (24 optical fibers each) as drawn in Figure 5.6. The connection between front-end and back-end is represented by the solid red line. A second ABBA is installed in the EMF crate, but one of the two front FPGAs are not usable (red). There is anyway the possibility to switch the connection and use the second front FPGA (dashed red line to the green FPGA), as the Event Builder application is able to reconstruct the events, even if they come from different ABBA boards.

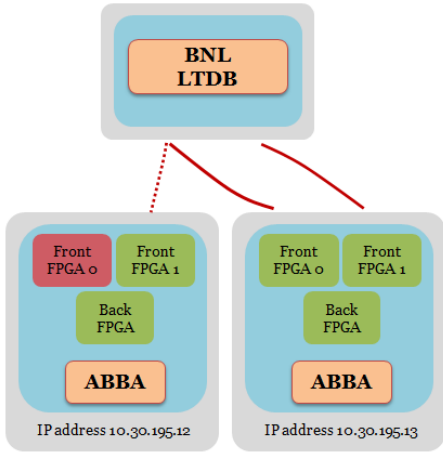

Figure 5.6: LTDB and ABBA boards connection in EMF. The solid line describes the currently used connection. The dashed line shows an alternative connection.

5.1.2 The UX15 and USA15 setup

After the successful tests at the EMF, the demonstrator was installed in ATLAS, with a coverage of $1/32$ of the calorimeter barrel region. Figure 5.7 shows a schematic view of the ATLAS calorimeter and the region of the barrel instrumented by the demonstrator.

To install the fronf-end part of the demonstrator system in the the ATLAS cavern (UX15), the FEBs were extracted from the front-end crate to place the new Phase-I EMB baseplanes. The FEBs were equipped with new LSBs and again inserted, together with two LTDBs.

The two LTDB pre-prototypes, shown in picture 5.8, were designed and installed to digitize the calorimeter data from 284 super cells in the EM barrel. The design of the two LTDB pre-prototypes is different. The LTDB shown on the left side of Figure 5.8 was produced at the Brookhaven National Laboratories (BNL) and based on a digital motherboard working together with analogue mezzanines. The LTDB shown on the right side was produced by Saclay and Laboratoire de l'Accélérateur Linéaire (LAL), and made of an analogue motherboard connected to digital mezzanines. The two LTDB pre-prototypes will be referred in this document as BNL and Saclay/LAL LTDBs.

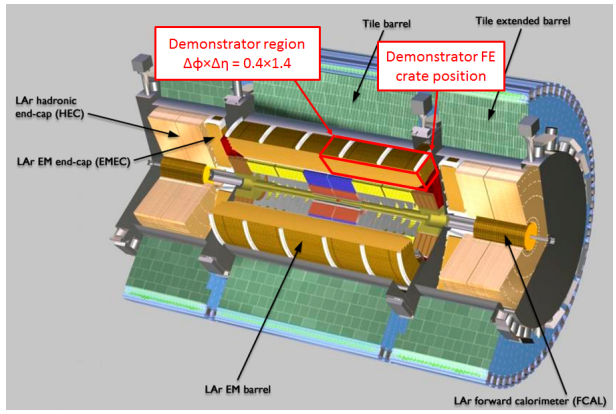

Figure 5.7: A schematic view of the ATLAS calorimeter is shown. The demonstrator region in the EMB is here highlighted.

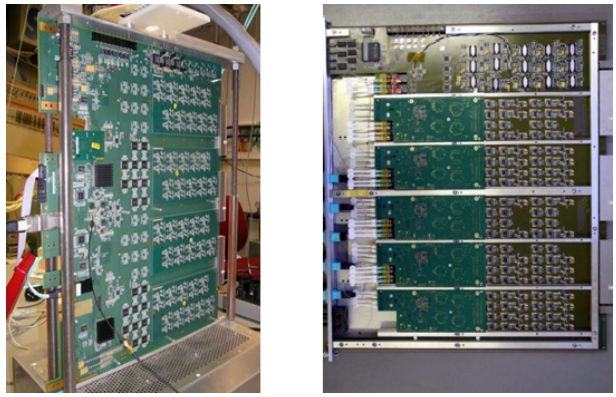

Figure 5.8: Left: LTDB pre-prototype produced by the Brookhaven National Laboratory (BNL). Right: LTDB pre-prototype produced by Saclay and Laboratoire de l'Accélérateur Linéaire (LAL).

On the left side of Figure 5.9, the calorimeter front-end crate where the LTDBs are inserted is shown. On the right side, the slot assignments for the new EMB baseplane, which has the same number of slots (19) as the current barrel baseplane is shown. The space for the LTDB is provided by a previously unallocated slot at one end of the baseplane. The front (F) and PS slots remain unchanged, while the other boards are shifted to allow the LTDB to be located between the TBB and the Controller board, roughly in a central position.

The back-end electronics of the demonstrator system is located in USA15, outside the cavern where the ATLAS detector is installed. An ATCA crate, shown in Figure 5.10, contains three ABBAs and two server machines to handle the communication and the readout of the Super Cell data coming from the calorimeter. The Abba application polling the data from the board is running on one of them. The same machine is also used to reset and program the FPGAs on the ABBA boards. The other machine runs the Event Builder and at the same time it is used to communicate with the GLIB.

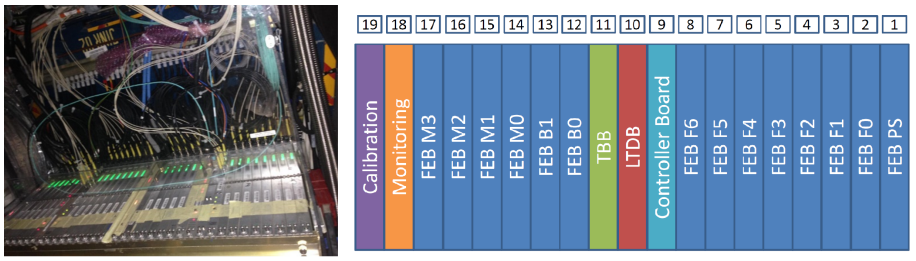

Figure 5.9: Left: LAr front-end crate, sitting on the detector, where the two LTDBs are placed. Right: slot assignments for the new EMB baseplane, used in the demonstrator crate.

Figure 5.10: ATCA crate where three ABBAs and two machines are installed.

GLIB

In the ATLAS environment, the LTDB is sitting in a crate that has no easy physical access. This is the reason why a board called GLIB has been developed, in order to facilitate the communication between the BNL LTDB and the outside world. The operations are triggered from the PC side. A picture of the GLIB and its installation in USA15 (right) is shown in Figure 5.11. An Ethernet cable is connected to the GLIB box. This cable allows to use IPbus to communicate with the GLIB firmware from the PC. A USB cable is also connected to the box and it allows to configure, through JTAG, the FPGAs on the LTDB.

LTDB to ABBA connection

The LTDBs are connected to three ABBAs, as shown in Figure 5.12. Two ABBAs would be enough for the data acquisition. As two FPGAs on two different boards have problems, a third board has been inserted in the crate.

The Saclay/LAL LTDB is connected to two FPGAs on two different boards, as well as the BNL LTDB. The Event Builder is able to reconstruct an entire event coming from different LTDBs, as well as from different ABBAs.

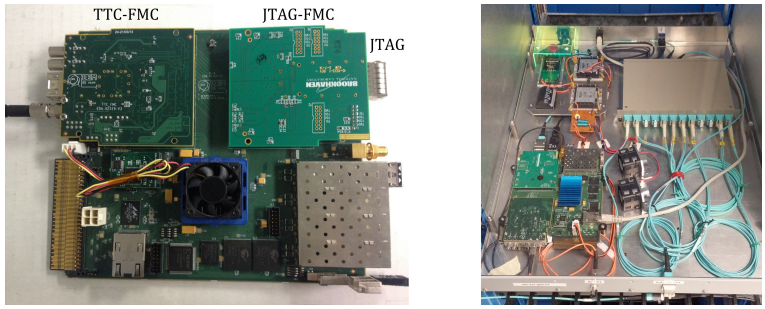

Figure 5.11: Left: GLIB board responsible for the communication between the LTDB and the PCs outside the ATLAS cavern is shown. Right: GLIB installed in USA15, in the ATCA crate.

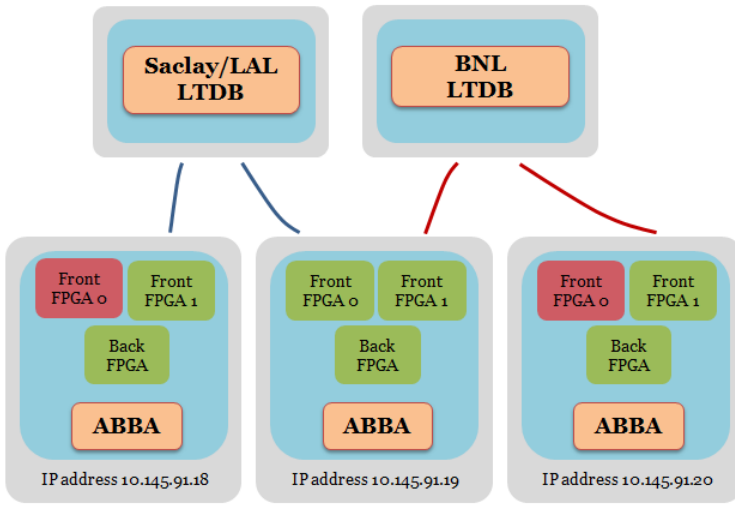

Figure 5.12: Boards connection in USA15.

5.2 ABBA firmware

Front and back FPGAs perform different functions. For this reason two different types of firmware have been developed.

5.2.1 Front FPGA firmware

A schematic description of the front FPGA firmware is shown in Figure 5.13.

The firmware can be divided in different blocks according to their role and functionalities.

The *oscillator configuration* block takes care of the communication with a crystal oscillator component placed on the board, in order to generate the proper clock frequency (120.24 MHz) to handle the LTDB optical transceiver interface. More details about the oscillator configuration block will be given in Section 5.6.

The *TTC* block is receiving the Timing, Trigger and Control signals. More details about the TTC can be found in the Appendix C. In this block L1A trigger, Event Count Reset (ECR), Bunch Count Reset (BCR), Trigger Type (TType) are decoded. ABBA does

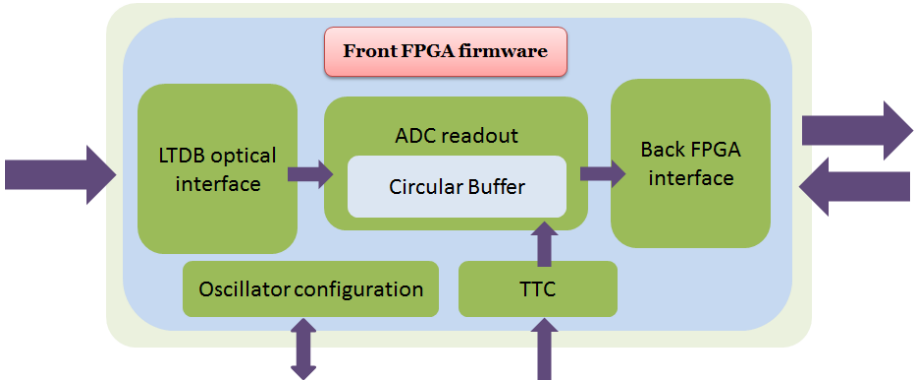

Figure 5.13: Schematic block of front FPGA firmware.

not receive the L1A at the full rate provided by ATLAS: the information is pre-scaled such that the board will receive triggers with a frequency of 1 Hz. Furthermore not all the L1A data are readout but only the ones matching a specific TType: the one with the demonstrator bit set to 1 (more details in Appendix C). For this reason, in the firmware, a mask checking the bits value is setup. The TType is associated to a dedicated topological Level-1 trigger item which selects electromagnetic clusters in the LAr demonstrator region, allowing a comparison with the events of the ATLAS main readout. In the same block, the Event Counter ID (EVID) and Bunch Counter ID (BCID), used to label the trigger data, are calculated and provided to the other blocks of the firmware.

The *LTDB optical interface* block is handling the data coming along the fibers. Each fiber contains 8 channels, each one carrying 12-bit frame data, as shown in Figure 5.14. Highlighted in red in the figure, there are the bits dedicated to the K-comma pattern (also called "K-code") [101]. It is an 8 bit pattern that can be used to find and verify the boundaries of a bitstream and for synchronization. Both LTDBs use the symbol K28.5, which is the default K character for the Xilinx and Altera transceivers.

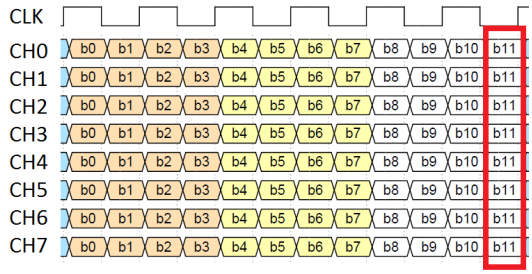

Figure 5.14: LTDB data organisation, with K-comma symbol bits highlighted [101].

The K-comma pattern, representing the decimal value 2048 and shown in Table 5.1, once issued, will replace all the 11th bits . The K-comma pattern is very important because it allows the ABBA FPGAs to be aligned with the front-end electronics but also with each other. This pattern is sent once each bunch revolution (each 3564 bunch crossing) and allows to set a locked condition on the fibers which is reused in other part of the firmware.

If the links between front-end and back-end are properly locked, the ABBA firmware

	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
11th bit	1	0	1	1	1	1	0	0

Table 5.1: K-comma pattern for data alignment.

is able to receive and decode the data. At this stage, the *ADC readout* block is playing a very important role. A procedure checking the alignment of the fibers has been put in place, so that the calorimeter pulse on a single fiber will always appear in the same position and will not fluctuate. More details about this part will be given in Section 5.8. After the alignment data are stored in a circular buffer until a L1A (matching the demonstrator TType), coming with a fixed latency respect to the data, is received by the board. If the fibers connected to the LTDB are properly locked, the data accepted by the trigger are picked up from the circular buffer and associated to the correct EVID, BCID and TType information. If a fiber is not locked, a specific pattern is inserted instead of the data.

The *Back FPGA Interface* block is handling the XAUI (10 Gigabit Attachment Unit Interface [102]) interface, used to communicate with the back FPGA firmware. At the same time, this block contains the state machines handling the IPbus communication.

5.2.2 Back FPGA firmware

The back FPGA communicates with both front FPGAs and prepares the data for the Ethernet transmission. A schematic view of the firmware blocks is shown in Figure 5.15.

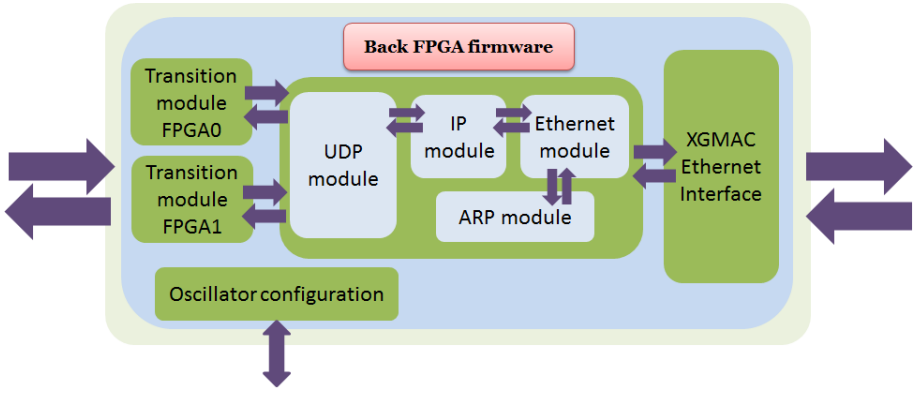

Figure 5.15: Schematic block of back FPGA firmware.

The content of the front FPGAs is read via Ethernet protocol, a very well established working standard, widely used. Each piece of information (requests, replies, addresses, etc.) sent on the Ethernet network is inserted in a packet. In Table 5.2 an example of how an Ethernet frame is defined is shown. The Media Access Control address (MAC address) is a unique identifier assigned to network interfaces for communications. In the demonstrator environment, each board is identified by its own MAC address. Then, the Ethernet Type value indicates the packet protocol, for example `0x0800` signals that the frame contains an IPv4 (Internet Protocol version 4) datagram, while `0x0806` indicates

Part	Definition	Data size
MAC Header	Destination MAC Address	6 bytes
	Source MAC Address	6 bytes
	Ethernet Type	2 bytes
Data	Payload	46 - 1500 bytes
Frame Check	Cyclic Redundancy Check (CRC)	4 bytes

Table 5.2: Definition of the Ethernet frame format.

an ARP (Address Resolution Protocol) frame.
The payload encapsulates further protocols: in ABBA firmware ARP, IP, ICMP and UDP are used. Each board has its own IP address and each FPGA is label with a specific UDP port, such that each Ethernet request can arrive to the desired FPGA.
The frame check is done with a CRC (Cyclic Redundancy Check) code, an error-detecting code used to detect accidental changes to raw data.

Starting from the right side of the firmware schematic, the Ethernet requests arriving to the back FPGA are handled, as a first step, by the *ALTERA 10 GigabitMedia Access Control (XGMAC) Intellectual Property (IP) core* [103]. This core grants the access to all the parts of the Ethernet packet and can be configured to do the CRC calculation and check.

A group of *Ethernet-dedicated Modules* allows the Ethernet frame to be unpacked, stripping the headers of the different protocols, until the UDP and IPbus levels are reached. The payload of the UDP frame is the IPbus request (or reply) and depending on the UDP port (1 or 2) it is distributed to one of the front FPGAs.

The *Transitions Module* manages the interface and contains the transceiver for the communication between front and back FPGAs. There is a transition module per each front FPGA [104].

As for the front FPGA, an *Oscillator Configuration* module is used to set the frequency coming from an oscillator. For the back FPGA the requested frequency is 156.25 MHz, used by the XGMAC core transceivers.

5.3 Validation of the firmware

New features have been added to the firmware of both front and back FPGA. A validation procedure has been put in place to check the good behaviour of the new versions before deploying them and at the same time to maintain a clean structure of the Git repository, described in Appendix B, where the firmware is stored.

As a first step, the developer has to check that the code is working in simulation and that no compilation errors are present. The developer is working on his developing branch, while a branch for testing in hardware has been created. The procedure is the following:

```
1) Develop the aimed improvements
2) Simulate until they work
3) Run a compilation
  3a) Make sure that timing constraints are met. If not, update the SDC
     (Synopsys Design Constraints) file accordingly
  3b) Also check if the STP (Signal TaP) file has been updated
4) Merge the master branch into the development branch
5) Repeat 3) to verify it is still fine
```

```
  5a) If it does not work, check what has changed in the master branch
    and correct it
  5b) If it works, continue to 6)
6) Open a merge request to the hardware test branch ABBA_HW_test
```

Once the developer checks are over, the responsible of the hardware branch starts the test at CERN, monitoring some key values and taking calibration runs. These are the instructions:

```
7) Check if 4) has been carried out by the developer, then accept the
    merge request from the development branch to ABBA_HW_test
8) Increase the firmware version to a test version
9) Repeat 3) (now in the ABBA_HW_test branch) and check
  - if timing constraints are met and STP file is still fine
  - if the usual operations and aimed new functionality work
10) Take a calibration run to verify the status of the pulses
  10a) In case of negative results go back to the developer
  10b) In case of successful test, increase the firmware version to the
     next release version
11) Open a merge request to branch master branch
```

The last steps are managed by the ABBA Git Masters:

```
11) Check if 9) and 10) have been done, then accept the merge request
    from ABBA_HW_test to master
12) Check with developer: development is finished, can the development
    branch be deleted?
13) Compile the versions needed
14) Ask developer to tag the firmware version (merge commit) with a
    useful tag message
15) Ask the developer to document the tag, including day of test and
    deployment in ATLAS
```

5.3.1 Calibrations runs

The demonstrator is using the same calibration system as the LAr calorimeter. Usually, each FPGA is identified by the last part of the board IP address, together with the UDP port. For example, the FPGA with UDP port 2, mounted on the board with IP address 10.145.91.19 is labelled as 19 : 2. In USA15, each ABBA FPGA reads out the values of cells with the same ϕ (one "ϕ-slice" of the demonstrator region).

- FPGA 18 : 2 for the region $\Delta\eta \times \Delta\phi = [0, 1.475] \times [2.1, 2.2]$
- FPGA 19 : 1 for the region $\Delta\eta \times \Delta\phi = [0, 1.475] \times [2.0, 2.1]$
- FPGA 20 : 2 for the region $\Delta\eta \times \Delta\phi = [0, 1.475] \times [1.9, 2.0]$
- FPGA 19 : 2 for the region $\Delta\eta \times \Delta\phi = [0, 1.475] \times [1.8, 1.9]$

Three types of calibration runs can be collected to check the firmware behaviour and the system status. The pulses shown in Figure 5.16, 5.17, 5.18 are all recorded with the demonstrator system.

- **Pedestal run.** Pedestal runs do not have any injected charge and are used to monitor the noise level and establish a pedestal, the value which would be read out without any input signal. An example of pedestal data collected with the demonstrator is shown in Figure 5.16.

- **Delay run.** Each channel is pulsed N times. A given constant input current with several phases are used for pulsing. This has the effect of interpolating samples in between each bunch crossing interval, providing a smooth curve, from which Optimal Filtering Coefficients (OFCs) can be derived. An example of delay data collected with the demonstrator is shown in Figure 5.17.

- **Ramp run.** Each channel is pulsed N times with a set of different input currents and at a constant phase (delay time). When taking ramp runs on the detector, neighbouring cells are not pulsed at the same time in order to avoid cross-talk effects. These runs are usually helpful to measure electronic gain obtained during the shaping of the pulse. An example of ramp data (in this case combined with a delay run) collected with the demonstrator, is shown in Figure 5.18 where pulses with different amplitudes are visible.

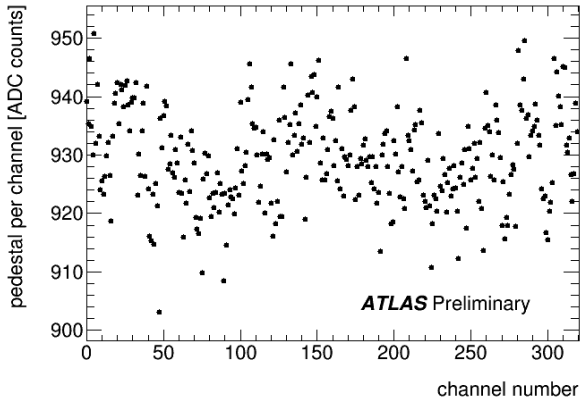

Figure 5.16: LTDB demonstrator pedestal measured on the demonstrator. The pedestal values of the 12-bit ADC of the 320 channels of the LTDB demonstrator measured in USA15 are shown [105].

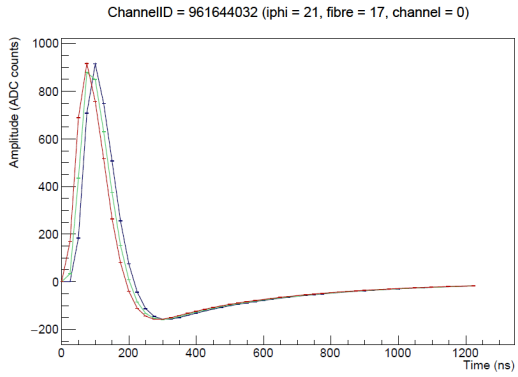

Figure 5.17: Three delay pulses collected with the demonstrator system and injected in between a bunch crossing interval. The different lines represent the pulse shape at delays of 0 ns, 12.48 ns and 23.91 ns.

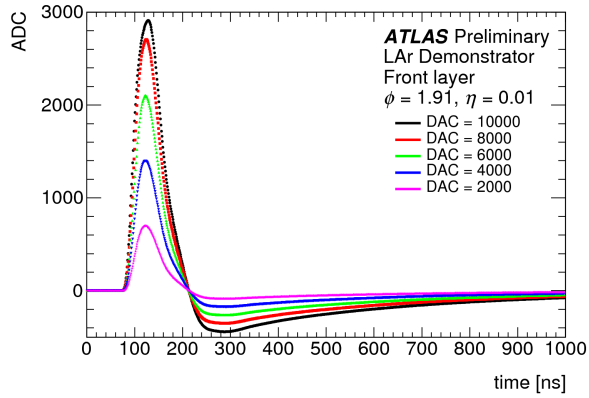

Figure 5.18: Pulse shapes, from a combined ramp and delay run, of a front-layer Super Cell from the LAr Phase-I demonstrator, for injected calibration pulses with different amplitudes [105].

The sanity of the ABBA firmware can be verified with all the three types of run. If the data are not readout correctly, misbehaviours can be observed in the pulse shape as well as in the pedestal value. The most common problems observed until now are bit-flips, sudden jumps in the pedestal value (due to a wrong channel assignment, for example after a reset) or wrong alignment of the pulse in the readout window (due to a wrong readout latency in the circular buffer).

An example of how a calibration run is chosen is shown in Figure 5.19, where a section of the TDAQ calibration panel for LAr is shown. In this case the *Sequence Type* "Ramp" and the *Tag* "High Ramp" are selected. Parameter files include the description of the selected Tag. These files contain the value *# triggers* describing the number of triggers per substep, followed by the *# substeps*, the *Maximum # steps* and the value *Max events* that is the total number of triggers. Furthermore, the files define the pulses amplitude and calorimeter pulsing patterns.

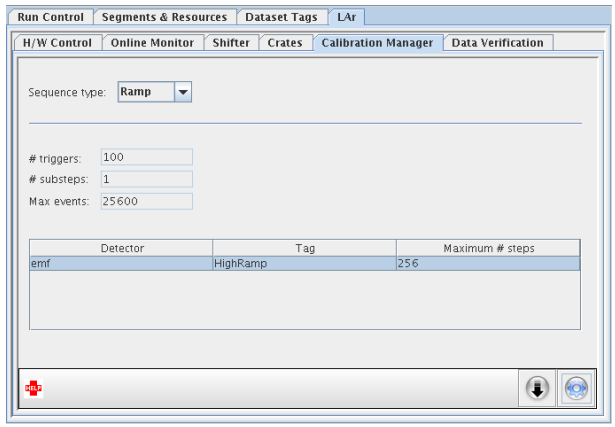

Figure 5.19: TDAQ calibration panel used to set the calibration type.

5.3.2 TDAQ panel for ABBA

All the important ABBA firmware parameters can be modified and checked from the TDAQ software, as shown in Figure 5.20 and 5.21. Each front FPGA can be configured and monitored through the panel.

Figure 5.20 shows the parameters that can be configured inside the firmware for the data taking.

- In the *Main* block the *Samples* determines the dimension of the readout window. The *Peak Sample* is the parameter that influences the pulse peak position inside the readout window. The *sourceID* is the FPGA ID value used in the TDAQ software. The *TType* value can be set in this block: the hexadecimal value `0x9090` means that the collected data are physics data coming from the demonstrator region. The *Timeout* set the time for a bad-response in case of Ethernet communication problems.
- the *Threshold* block allows to set a threshold on the number of locked links between front-end and back-end boards, under which warning or error messages are thrown
- the *Ppod Monitoring* allows the user to select which fiber has to be monitored (light green) and which not (dark green).

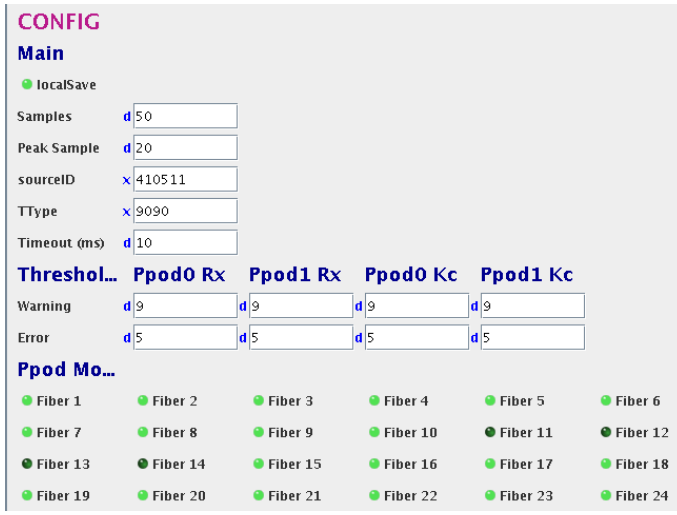

Figure 5.20: Configuration part of the TDAQ panel for the demonstrator.

Figure 5.21 shows the Status TDAQ panel for the demonstrator. It is used to monitor the behaviour and the sanity of the system.

- The *Counters* section allows the user to verify the consistency of the data recording as well as the activity or inactivity of each front FPGA. More details about the counters can be found in Section 5.9.2.
- The *Latency* boxes are showing the latency correction value (described in Section 5.8) applied to the fibers. Each box groups four fibers and shows 4 bytes, one per each fiber.

Figure 5.21: Status part of the TDAQ panel for the demonstrator.

- The *Ppods receiver* and *Ppod K-code* help in debugging which fiber (connected with the front-end) is not locked (not transmitting data).
- The *Error counter* counts the unlocked status of each fiber. It is a one byte information.
- The *Abba Status* block shows the details about the FPGA firmware version, IP address and other general information of the FPGA monitored in the panel.

5.4 Physics runs with the demonstrator

The demonstrator system has been running in parallel with the ATLAS data taking during proton-proton and heavy ion collisions. The main idea of the parasitic running was to compare events from the ATLAS main readout with events recorded by the demonstrator readout. At the beginning, every time that ATLAS was starting a run to collect the data, a demonstrator run had to be started separately. Recently ABBA has become a sub-segment of ATLAS TDAQ system and, once an ATLAS run is started, also the demonstrator is starting a run.

A pulse recorded with the demonstrator readout, during a proton-proton (*pp*) run is shown in Figure 5.22.

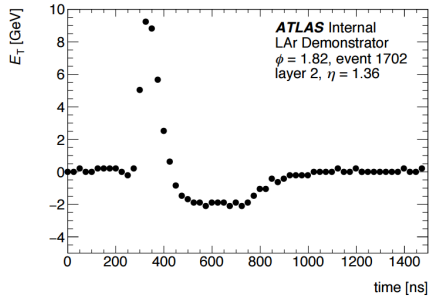

Figure 5.22: Physics pulse recorded with the demonstrator system during a *pp* run [99].

The matching of events is done by comparing the EVID, the BCID and the TType from the event header of the demonstrator data and ATLAS data. Figures 5.23 and 5.24 show an event recorded during a *pp* physics run (between June 27 and June 28, 2017) as observed in the demonstrator and as obtained summing the LAr cell energies from the ATLAS main readout.

The geometrical coverage of the demonstrator system is partially shown in Figure 5.23, while the full geometrical coverage is shown in Figure 5.24. The volume of the depicted boxes is proportional to the deposited energy.

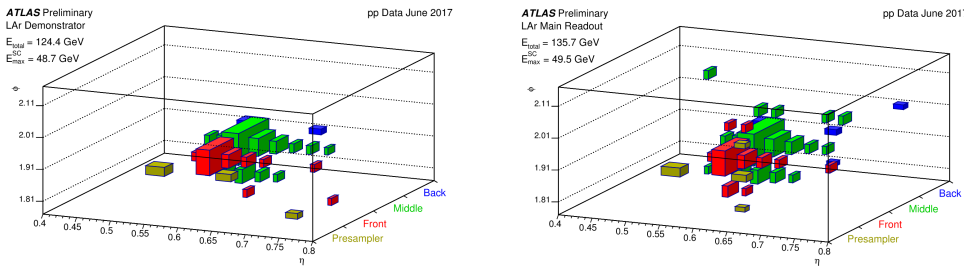

Figure 5.23: Event display of the partial demonstrator region showing the event as observed in the LAr demonstrator (left) and from LAr main readout (right) [105].

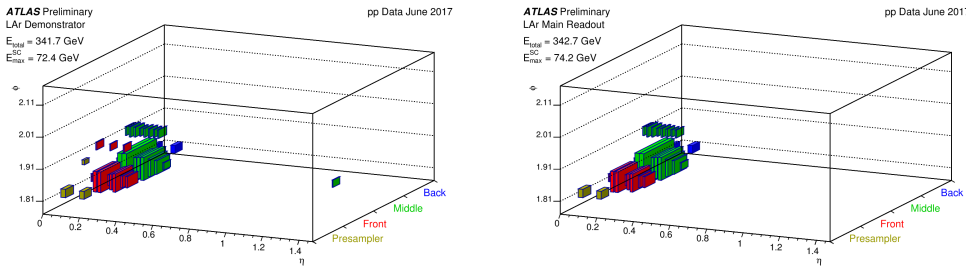

Figure 5.24: As Figure 5.23, but showing the full demonstrator region [105].

Particle showers in the demonstrator region are compared to the same shower in the main read-out. The shower shape can be well reconstructed and the total deposited

energy, as well as the maximum Super Cell energy, agree typically within 10%. An agreement within this range is good enough to allow to use the Super Cells for trigger purposes.

5.4.1 Trigger coverage

In order to induce the demonstrator back end to read out the data, a demonstrator bit is declared inside the TType. When an event is happening in the demonstrator region, the demonstrator bit is raised to 1 (more details about the TType are in Appendix C). From the 3rd June 2016 the demonstrator TType is looking at the barrel region shown in Figure 5.25. The plot shows electrons in the ATLAS main readout, with $p_T > 10$ GeV and the required TType `0x90` from the L1Topo stream.

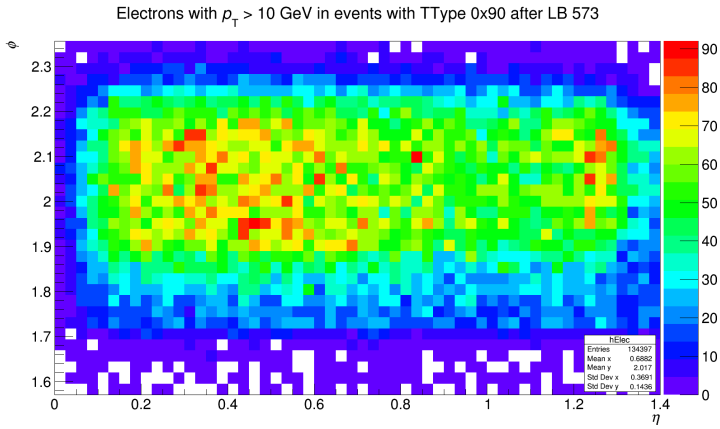

Figure 5.25: Demonstrator trigger coverage: electrons in ATLAS main readout with $p_T > 10$ GeV in events with the required TType. The scale on the right side shows the number of entries per bin [106].

The BNL LTDB is reading the region $\Delta\eta \times \Delta\phi = [0, 1.475] \times [1.8, 2.0]$, while the LAL/Saclay LTDB is reading $\Delta\eta \times \Delta\phi = [0, 1.475] \times [2.0, 2.2]$.

5.4.2 Linearity checks

The Super Cell energy (E_{SC}) from the demonstrator has been compared to the summed LAr cell energies in ATLAS, for $E_{SC} > 1$ GeV. The Super Cells in the presampler, front, middle and back layer consist of 4, 8, 4 and 8 LAr cells, respectively. As shown in Figure 5.26, a good agreement is observed between the two read-outs.

Data from the demonstrator readout are converted into energies using the dedicated demonstrator Super Cell calibration and then compared with the main read-out (the sum of the constituent cell energies of the respective Super Cell). Both calibrations are derived independently. The resulting distribution follows closely a 1-to-1 relation (red line), and a good agreement between the two readout streams is achieved.

Figure 5.27 shows another comparison between the measured Super Cell energies of the LAr Phase-I demonstrator and the summed LAr cell energies in ATLAS main readout. The comparison has been done by calculating the difference $E_{SC} - \sum_{SC} E_{cells}$ for $E_{SC} > 2$ GeV. Good agreement is observed between the two readout streams. The

width of the distribution is compatible with the expected noise level. The shift of the means is due to the preliminary calibration of the Super Cells.

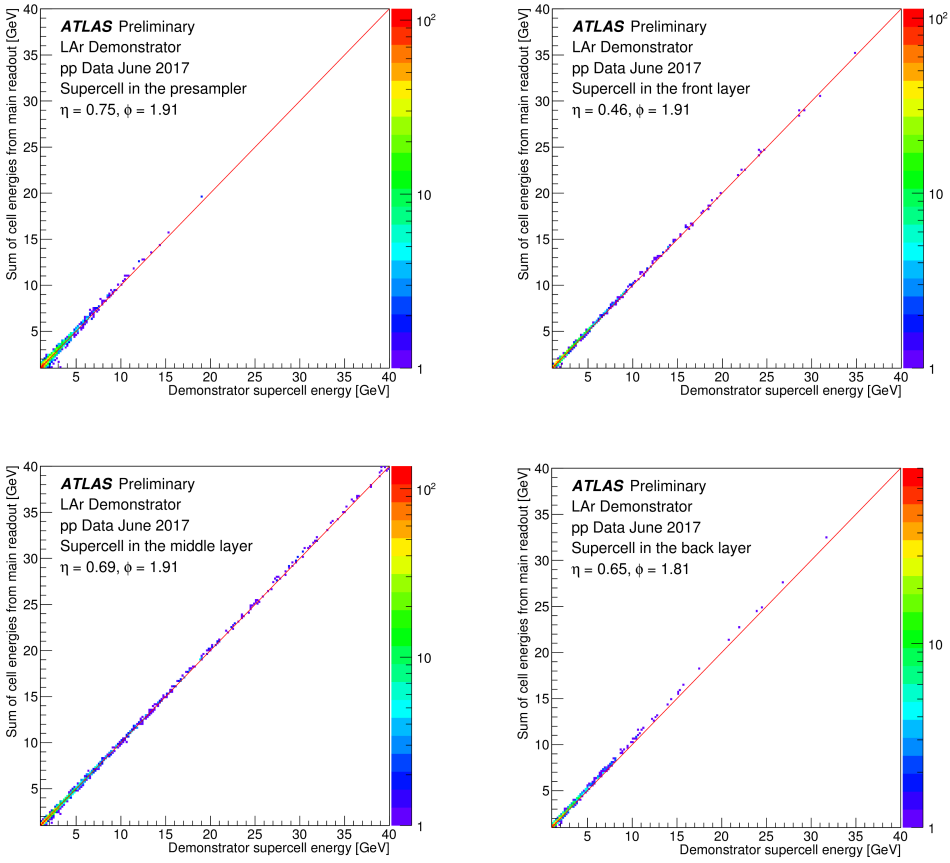

Figure 5.26: Correlation of energy measurements for each layer measured using the Super Cell energies and the ATLAS main readout [105].

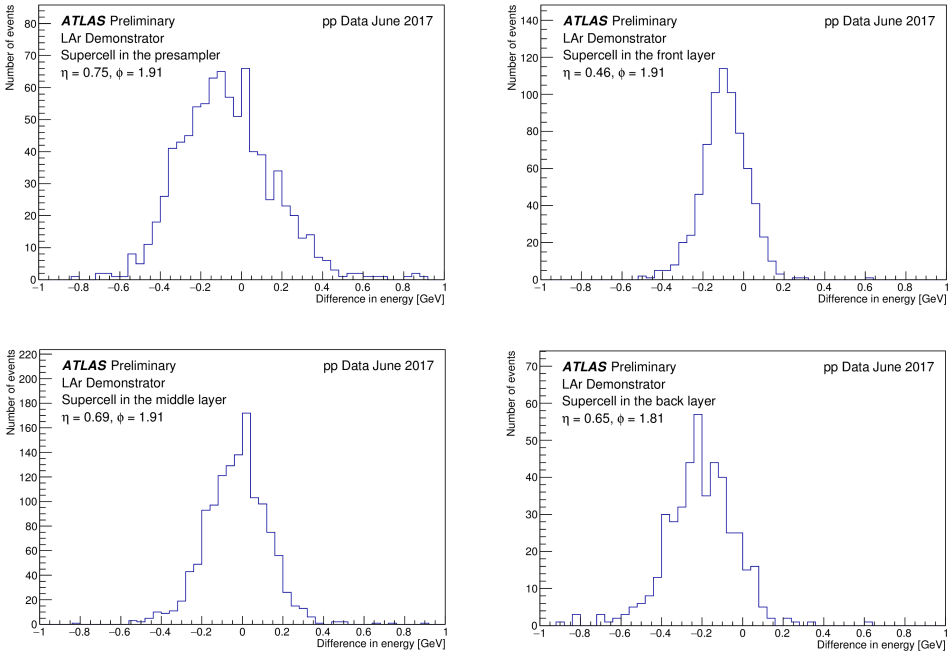

Figure 5.27: Difference of energy measurements for each layer: the measured Super Cell energies of the LAr Phase-I demonstrator are compared to summed LAr cell energies in ATLAS by calculating the difference $E_{SC} - \sum_{SC} E_{cells}$ for $E_{SC} > 2$ GeV. [105].

5.5 Firmware contribution

In the next sections, a description of my contributions to the demonstrator firmware is reported.

I implemented an I2C-bus protocol to integrate the I2C communication in the main firmware and generate specific clock frequencies from a crystal oscillator. The low jitter clock generated from the crystal is then used by the transceivers inside both front and back FPGAs.

The reset is a key component of the firmware. It allows to clean errors and bad behaviour, bringing back the system to a clear condition or to the initial state. I led the effort to rethink the reset tree, to be able to reach the critical blocks independently and cure the on-line instabilities.

In addition, I implemented a latency auto-correction to ensure the alignment of the pulse inside the readout window. The misalignment was happening because of the uncertainty of the phase of the hardware transceivers locking, after each reset, at the beginning of the data taking.

5.6 I2C-bus project

In the original implementation of the firmware, to work properly, ABBA needed two steps. First, the embedded NIOS II processor [107] firmware needed to be loaded: this firmware was used to set the reference clock frequency via I2C communication with a crystal oscillator. After this step, the main firmware was loaded. The aim of my project is to integrate the frequency settings in the main firmware, having only one loading step.

On the ABBA board each FPGA is equipped with two external Si570 chips from Silicon Laboratories [108], as shown in Figure 5.28, that allow to freely choose two reference clocks for the FPGA transceivers.

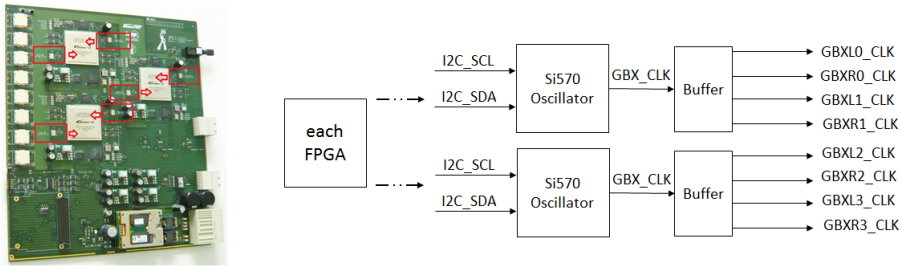

Figure 5.28: Position of the oscillators on ABBA (left) and a simplified schematic view of the signals involved per each FPGA (right).

A clean and precise clock is important to ensure a good quality high-low signal transition and periodicity. For this reason crystal oscillators with a very low jitter were mounted on ABBA.

The jitter is the deviation from the true periodicity of the clock, like it is shown in Figure 5.29. It is a significant, and usually undesired, effect that can be caused, for example, by electromagnetic interference and crosstalk.

A crystal oscillator is an electronic oscillator circuit that uses the mechanical resonance of a vibrating crystal of piezoelectric material to create an electrical signal with a

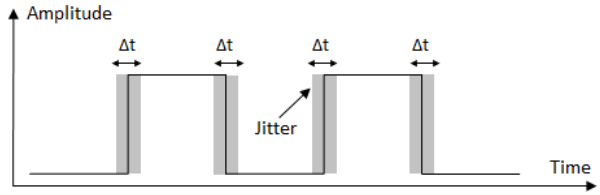

Figure 5.29: Jitter effect on a clock signal.

precise frequency. For this purpose an Si570 oscillator, shown in Figure 5.30 has been used [108].

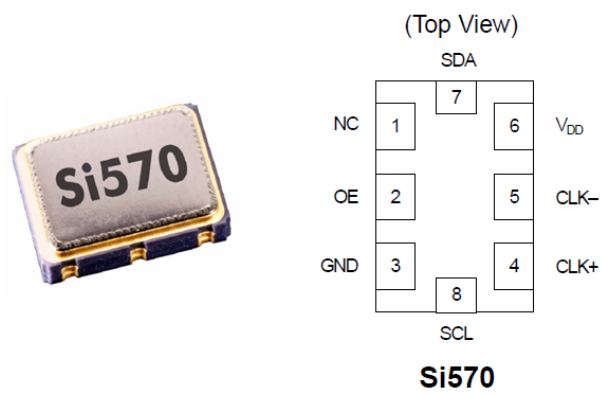

Figure 5.30: Si570 chip (left) and top view of its pin connection (right) [108].

The Si570 is a any-frequency low jitter oscillator, with a typical phase jitter of 0.3 ps and the following features:

- it provides a low-jitter clock, with a stable and reliable frequency making use of advanced DSPLL, an IP technology to simplify the clock multiplication
- it is user-programmable to any output frequency from 10 to 945 MHz and selected frequencies up to 1400 MHz
- it can be programmed via an I2C serial interface.

Figure 5.30 shows the Si570 chip and its pin connection. A brief description of the pins, taken from the datasheet, can be found in Table 5.3.

5.6.1 I2C serial communication

The I2C communication protocol is a popular and powerful bus, used for communication between a master (or multiple masters) and a single (or multiple) slave device. A master device initializes and terminates a transfer, and generates the clock for the communication. A slave is the device addressed by the master [109] [110] [111].

I2C protocol uses two bidirectional lines, Serial Data Line (SDA) and Serial Clock Line (SCL). The data on the SDA line must be stable during the `HIGH` period of the clock.

Pin	Name	Type	Function
1	NC	N/A	No Connect. Make no external connection to this pin
2	OE	Input	Output Enable
3	GND	Ground	Electrical and Case Ground
4	CLK+	Output	Oscillator Output
5	CLK–	Output	Complementary Output
6	VDD	Power	Power Supply Voltage
7	SDA	Bidirectional Open Drain	I2C Serial Data
8	SCL	Input	I2C Serial Clock

Table 5.3: Pin description for crystal oscillator Si570 [108].

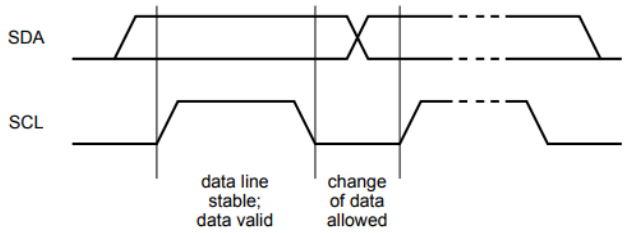

Figure 5.31: Bit transfer on the I2C bus [111].

The `HIGH` or `LOW` state of the data line can only change when the clock signal on the SCL line is `LOW`, as shown in Figure 5.31.

All transactions begin with a `START` (S) and are terminated by a `STOP` (P), as shown in Figure 5.32. A `HIGH` to `LOW` transition on the SDA line while SCL is `HIGH` defines a `START` condition. A `LOW` to `HIGH` transition on the SDA line while SCL is `HIGH` defines a `STOP` condition.

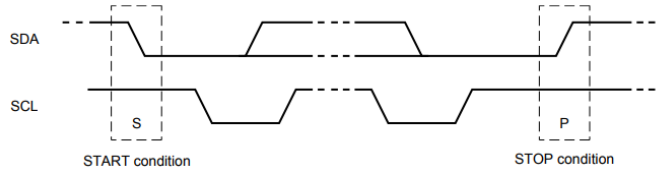

Figure 5.32: I2C start and stop conditions [111].

There is no specific restriction on the number of bytes that can be transmitted. Each byte must be followed by an acknowledge bit. The acknowledge allows the receiver to signal the transmitter that the bytes were transmitted successfully. Data is transferred with the Most Significant Bit (MSB) in the first position, as shown in Figure 5.33.

Data transfers follow the format shown in Figure 5.34. After the `START` condition, a slave address is sent. This address is seven bits long followed by an eighth bit which is a data direction bit (R/W). A “zero” indicates a transmission (`WRITE`), a “one” indicates a request for data (`READ`). A data transfer is always terminated by a `STOP` condition generated by the master. However, if a master still wishes to communicate on the bus,

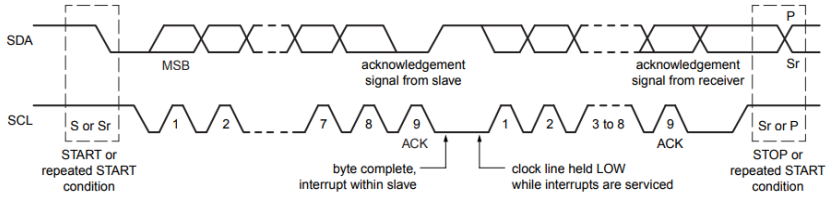

Figure 5.33: I2C data transfer condition [111].

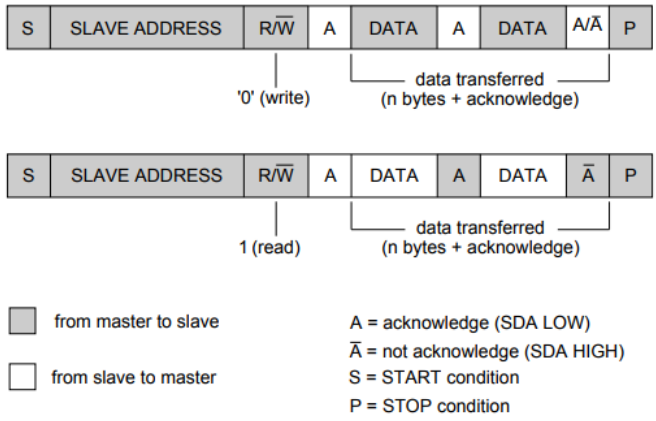

Figure 5.34: I2C data read and write requests [111].

it can generate a repeated `START` condition (Sr) and address another slave without first generating a `STOP` condition, as shown in Figure 5.33.

5.6.2 Procedure

The Si570 oscillator can be programmed to generate any output clock between 10 MHz and 1.4 GHz. The default output frequency of the device can be reprogrammed powering it down, returning the oscillator to its factory-set default output frequency.

The output frequency f_{out} can be programmed setting the Digitally-Controlled Oscillator (DCO) frequency and the device output dividers HS_DIV and $N1$. The output frequency is calculated using the following equation:

$$f_{out} = \frac{f_{DCO}}{Output\_Dividers} = \frac{f_{xtal} \times RFREQ}{HS\_DIV \times N1}. \tag{5.1}$$

The internal crystal frequency f_{xtal} is 114.285 MHz, $RFREQ$ is a high-resolution 38-bit multiplier (10 bit integer part, 28 bit fractional part) and $HS\_DIV \times N1$ is an integer number that depends on the frequency range, as shown in Table 5.4. The lowest value of $N1$ with the highest value of HS_DIV also results in the best power savings. Furthermore, $N1$ can't assume any odd value except for 1.

All the listed values, $RFREQ$, $N1$ and HS_DIV, are sent to the oscillator to generate the required frequencies.

I set the oscillators to generate two different frequencies for the ABBA firmware, as follows:

Parameter	Test Conditions	Min	Max	Unit
Output frequency range	$HS\_DIV \times N1 >= 6$	10	945	MHz
	$HS\_DIV \times N1 = 5$ and $N1 = 1$	970	1134	MHz
	$HS\_DIV \times N1 = 4$ and $N1 = 1$	1.2125	1.4175	GHz

Table 5.4: Conditions for the available frequency ranges [108].

- 120.24 MHz used for the interface between ABBA front FPGA and LTDB
- 156.25 MHz used for the interface between ABBA back FPGA and the PC

Calculation example for 156.25 MHz

Equation 5.1 can be written as:

$$HS\_DIV \times N1 = \frac{f_{DCO}}{f_{out}}, \tag{5.2}$$

to obtain the Output Dividers integer value. Knowing that the DCO frequency is adjustable in the range of 4.85 to 5.67 GHz by setting the high-resolution 38-bit fractional multiplier $RFREQ$ and knowing the output frequency, I calculated the Output Dividers obtaining a value between 31.04 and 36.29. As suggested by the manufacturer, for power saving reasons, $36 = 9 \times 4$ as Output Divider was selected.

Then the correct $RFREQ$ value to be sent to the oscillator registers was calculated using the following equation:

$$RFREQ = \frac{f_{out} \times HS\_DIV \times N1}{f_{xtal}}. \tag{5.3}$$

The decimal value $49,21905761911$ (where both the integer and the fractional portions are important) is obtained. Before entering the fractional number into the $RFREQ$ register it needs to be converted into a 38-bit binary number:

- the integer portion is converted to a 10-bit binary number
- the fractional portion is multiplied by 2^{28}, truncated and converted to a 28-bit binary number

The final binary value sent to the oscillator is the concatenation of the two parts.

Values to be sent via I2C

Table 5.5 shows the values used to program the oscillator for both 156.25 MHz and 120.24 MHz frequencies.

5.6.3 Oscillator registers

Readable and writeable registers are available inside each oscillator. The values previously calculated to request a specific frequency are written in specific registers shown in Figure 5.35.

	Values for 156.25 MHz	**Values for 120.24 MHz**
HS_DIV (3 bit)	11	9
$N1$ (7 bit)	4	4
$RFREQ$ (38 bit)	46.29268932931	49.21905761911

Table 5.5: Necessary values to program the oscillator to generate the frequencies 156.25 MHz and 120.24 MHz.

Register	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
7	High Speed/ N1 Dividers	HS_DIV[2:0]			N1[6:2]				
8	Reference Frequency	N1[1:0]		RFREQ[37:32]					
9	Reference Frequency	RFREQ[31:24]							
10	Reference Frequency	RFREQ[23:16]							
11	Reference Frequency	RFREQ[15:8]							
12	Reference Frequency	RFREQ[7:0]							

Figure 5.35: Register table for setting specific frequencies [108].

5.6.4 Firmware modules

A dedicated block of the firmware is handling the communication with the crystal oscillator, organised as shown in Figure 5.36.

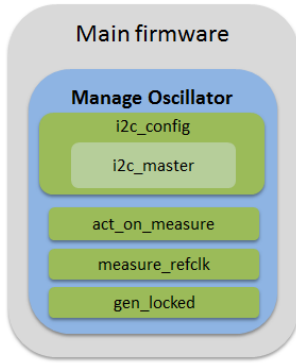

Figure 5.36: Schematic view of the firmware organisation for the oscillator block.

A top level called *Manage Oscillator* is handling the connections between all the I2C entities. Here the target frequency value is set.

The entity *I2C Configuring Oscillator* calculates the values presented before, depending on the the target frequency. Then the entity *I2C Master*, which is the core of the firmware, handles these values. Two state machines are managing the I2C bus. The first state machine sets the timing for the bus clock (SCL) and the data clock (SDA). The values of the input clock from the user logics and of the bus speed are set as generics. The

clock, the reset and the bus info are set as input of the entity. A second state machine prepares the I2C bus. The slave and register addresses, the read and write commands, the acknowledge info and the data are placed in the correct order on the bus, ready to be sent to the desired device. SDA and SCL are the output of this entity.

The I2C command is directly sent to the oscillator which is generating a frequency. The generated clock is then coming back to the same firmware block to be evaluated by the entity called *Measure Reference Clock* (which counts the rising edges of the input clock for a given number of cycles of the reference clock) and send it to the *Act on Measure* entity. The latter entity decides whether the measured frequency is within tolerance (set by the user in the top level, in our case ± 10 kHz). If the frequency is outside the tolerance limit, then a correction (± 5 kHz) is applied. A new request with the values corresponding to the new frequency is sent to the oscillator.

The entity *I2C Configuring Oscillator* receives the command of increasing or decreasing the parameters, to set the new frequency. Inside this entity, the procedure described in the previous section to extrapolate the values to be sent to the oscillator registers is implemented. This implementation allows an easy frequency auto-correction procedure in the firmware. The data are then prepared again in the entity *I2C Master* and an I2C signal is sent to the oscillator.

Once the frequency has reached a value inside the tolerance, the stability is evaluated in the entity *Generation Locked*. The stability is defined as the number of cycles in which no reconfiguration should take place. If after a fixed number of clock cycles the frequency is considered stable, a signal of "lock" is generated and used in the main firmware to start the data acquisition.

5.6.5 Oscillator simulation

I prepared various simulations to verify the good behaviour of the I2C-bus code.

In Figure 5.37, the two communication lines, `sda` and `scl` of the I2C bus (highlighted in orange), carrying the information to be written in the crystal registers, are shown. In particular, `data_wr_0` and `data_wr_1` (highlighted in yellow) are the two data words which has to be transferred along the bus to the slave.

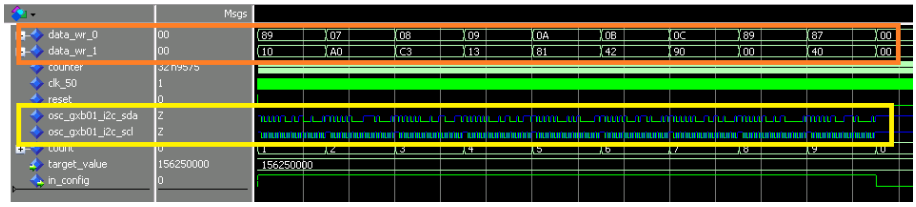

Figure 5.37: Simulation showing the values used to configure the oscillator registers (in orange) and the communication lines of the I2C bus (in yellow).

Figure 5.38 shows the simulation of the locked state. I introduced a "fake" frequency value to have a faster simulation as well as "wrong" frequencies to issue the reconfiguration process. The last value issued in the simulation is the expected value. The locked signal, highlighted in yellow, is raised to one after 3 clock cycles of frequency stability.

Figure 5.39 and 5.40 show the top level simulation, with all the useful signals of the oscillator block. The first figure shows the two I2C bus lines (highlighted in orange) and the "wrong" measured frequencies (highlighted in yellow). The second figure shows

the evaluation of the deviation from the expected value (highlighted in orange) and the reconfiguration process with the new value along the I2C lines (highlighted in yellow).

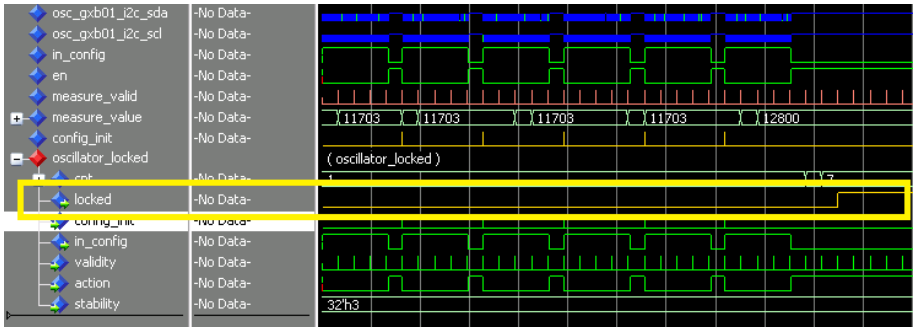

Figure 5.38: Simulation showing the locked signal (in yellow) after the stability is reached.

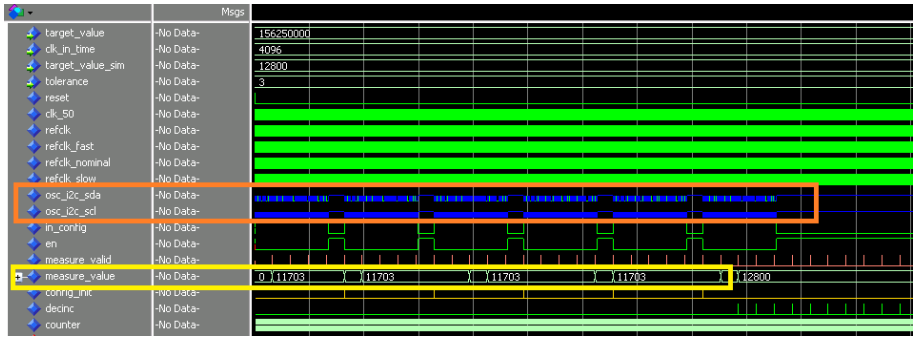

Figure 5.39: Simulation of the top level entity. This part of the simulation shows the data sent to the oscillator via I2C bus (in orange) and the measured frequencies (in yellow).

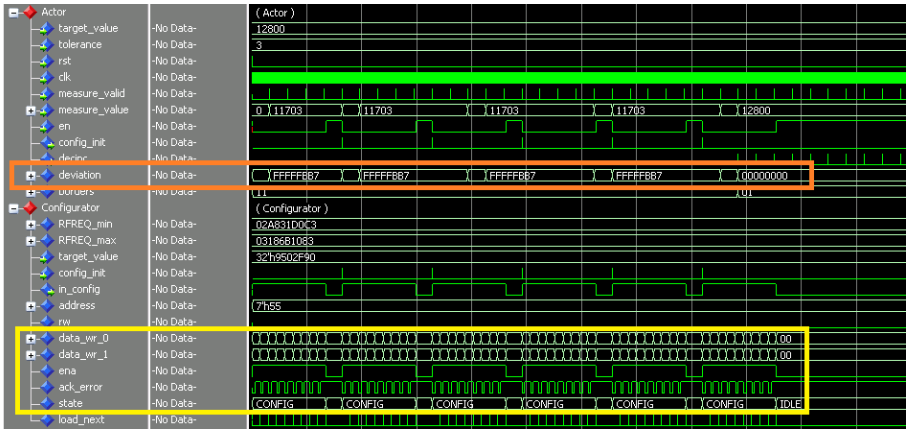

Figure 5.40: Simulation of the top level entity. This part of the simulation shows the deviation from the expected value (in orange) and the reconfiguration of the clock (in yellow).

5.7 Reset structure project

The front FPGA firmware has a complex structure, as already shown in Section 5.2.1. A new reset structure, able to touch all the critical parts of the code, was needed to debug and correct the misbehaviour happening during the data taking.

The reset tree inside a firmware has to be organised carefully. One must make sure, e.g. that no loops are generated when a reset is set to `HIGH`. To have a proper reset procedure, a good knowledge of the firmware and the use of simulation tools are required.

I was asked to take care of the reset structure for the FPGA firmware where data are stored and packets are built. A more detailed sketch of the front FPGA firmware behaviour is shown in Figure 5.41. The circular buffer is where all the data coming from the LTDB are stored. It waits until a L1A signal, associated to the required TType (the special TType associated to the demonstrator), decoded by the TTC block is received. Upon the L1A, the data are readout and stored in a FIFO, waiting for the header, containing the EVID, BCID and other information useful for the data reconstruction. At this point data are stored in another FIFO, waiting for the PC request passing through the Back FPGA interface. In the sketch the data path is drawn in yellow while the TTC information path is in violet. The red stars indicate were some debug counters were inserted to monitor the packet construction.

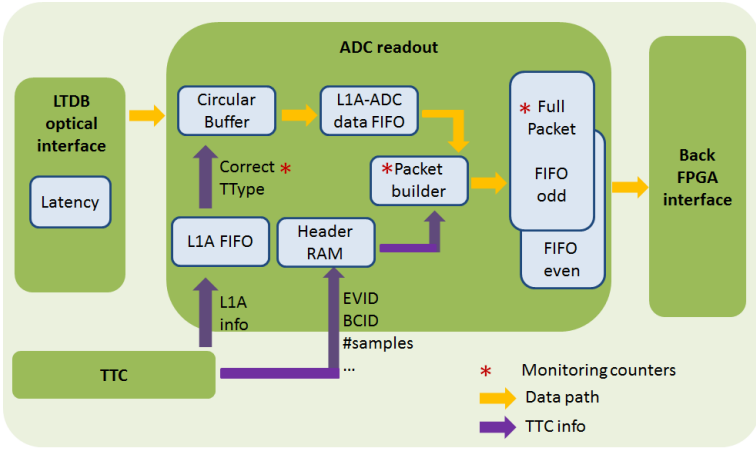

Figure 5.41: Sketch of the front FPGA firmware behaviour, focusing on the ADC readout block.

Figure 5.6 shows how I organised and grouped the reset tree. A 32 bit vector is driving all the resets and is organised in four groups of 8 bit each where different firmware parts can be identified.

The first byte is dedicated to the communication with the LTDB: the bit "0" is resetting the latency settings in the *ADC Readout* block. All the other LTDB resets, from bit 1 to bit 4 are connected to the *Optic Interface* block. The reset in position "1" is resetting the RAMs storing the data coming from the LTDB. The reset "2" is resetting some control registers which are monitoring the communication between LTDB and ABBA. The third reset is handling the process where the periodicity of the BCR is used to allow the data decoding and collection (locked state). The last of the four bits is resetting some counters used for monitoring the packets flow inside the firmware. The second byte is related to the ABBA configuration and memories. These resets are used in the *ADC Readout* block. The bit in position "8" is resetting the address value of the circular buffer where data are

Byte	bit	Reset	Group	Byte	bit	Reset	Group
0	0	Latency settings		**2**	16		
	1	RAMs			17		
	2	Control registers			18		
	3	Lock state	**LTDB**		19		
	4	Error counters	**interface**		20		
	5				21		
	6				22		
	7				23		
1	8	Circ-buff addresses		**3**	24	TTC interface	
	9	Circ-buff and FIFO empty			25	Avalon interface	
	10	Packet header			26	XAUI interface	**Interfaces**
	11	L1A counter	**ABBA**		27	Oscillator interface	
	12	Built packages	**configuration**		28	Oscillator interface	
	13	Package difference			29		
	14	Sent packages			30		
	15				31		

Table 5.6: Reset table: bit per bit division and reset description.

written. This reset combined with a "not properly configured" condition, also resets the precesses for the header creation. Bit "9" is emptying the circular buffer as well as the FIFOs which are collecting the events upon L1A and TType checks. The reset in position "10" is resetting the header information contained in the firmware registers. Resets from "11" to "14" (and recently also the "15") are used to reset some monitoring counters placed along the data path and used for debugging.

At the time of the reset tree implementation, the third byte was empty. Recently the position "16" has been filled with a new reset, dedicated only to the even and odd FIFOs (on the most right side of Figure 5.41) readout by the IPbus.

The fourth byte is dedicated to the firmware interfaces. The reset bit "24" is dedicated to the TTC block, decoding and calculating all the important TTC values. Then, bit "25" is dedicated to the Avalon Intel Altera interface, used to connect components and define interfaces appropriate for reading and writing registers and memories. Bit "26" is used to reset the XAUI interface with the back FPGA. The resets associated to bit "27" and "28" are dedicated to reset the crystal oscillator settings.

5.7.1 Standalone tools

A standalone software is available for the demonstrator. It allows to check the status of the firmware through some registers, readout via IPbus. It can configure the firmware for the data taking, setting for example a specific TType mask for the L1A selection or the latency settings (explained in the next section) and can set to zero all counters and memories to start a clean run.

The standalone tool allows the user to send the resets described above manually to the firmware. Selecting the command *Reset Front* (for the front FPGA only), an hexadecimal parameter is required, for example:

- `1f` to reset LTDB interface
- `7f00` to reset the second byte
- `1000000` to reset the TTC decoder
- `ffffffff` to reset everything

Following this idea, also in the back FPGA the reset structure has been improved and the proper standalone tool has been setup.

5.8 Latency project

The latency is defined as the time interval between the stimulus and the response. In the demonstrator system, the stimulus is an event happening in the calorimeter and the response is the readout of the pulse.

During the readout, it is important not to loose any part of the pulse coming from the calorimeter. Measuring the pulse length in Bunch Crossings (BCs), as the reference clock is the 40 MHz clock, the pulse can be 25 to 30 BCs long. That poses a constraint on the readout window: it must be at least as wide as the pulse length.

The readout window dimension can be modified, accordingly to the quantity of information requested. A simple sketch can be seen in Figure 5.42. Here L_0 is the stimulus coming from the calorimeter and L_{peak} is where the peak of the pulse is readout.

The demonstrator readout window is setup in the firmware through a value called *Samples*, already shown in Section 5.3.2. Usually the readout window is set to 50 samples, that corresponds to 50 BCs (1250 ns).

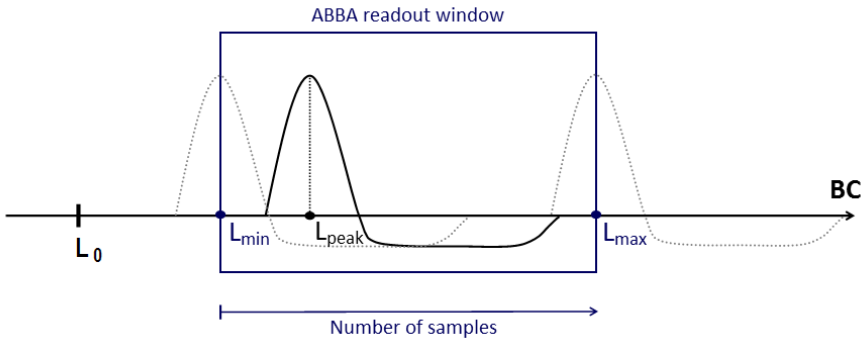

Figure 5.42: Simple sketch of readout window and peak positioning for the demonstrator pulse.

Only one side of the readout window can be reduced or extended. As shown in Figure 5.42, there is a fixed L_{min} and a variable L_{max}. A change in the number of samples will not modify the pulse position in time of the window.

The pulse can be shifted in the readout window, changing the latency value L_{peak} inside the firmware, with the standalone tool or using the ATLAS TDAQ interface (using the value *Peak Sample*). The latency value in TDAQ is a decimal value that corresponds to the desired peak position, considering L_{min} as zero position. A shift of one BC corresponds to a shift of one unit for the decimal value. For example, if the latency value is set at 20, to shift the pulse:

- 10 BC on the right: the value to be written in the TDAQ panel is 30
- 10 BC on the left: the value to be written in the TDAQ panel is 10

This was not the case in the first firmware implementation: the value to be set in the partition was an hexadecimal value that was not giving any hint about the pulse position. For example, to set the pulse in position 20, the user had to set the value `0x56`.

5.8.1 System differences

The position of the pulse peak is not only depending on the value set in the firmware. A difference in latency can be also due to differences in the hardware setup used for collecting the data.

During this work, I studied the differences between calibration and physics runs, as well as between the two LTDBs. Furthermore, there are differences between the setup used on the ATLAS detector and the one used in EMF. I collected a campaign of calibration data to define the latency differences.

- *Physics runs vs. calibration runs in USA15*. The difference between the pulse peak in physics runs and calibration runs is of 12 BC: Phy_{peak} + 12 BC = Calib_{peak}
- *BNL LTDB vs. LAL LTDB in USA15*. The difference between the pulse peak in the BNL LTDB and the one in the LAL LTDB is of 8 BC: BNL_{peak} + 8 BC = LAL_{peak}
- *Calibration in EMF vs. calibration in USA15*. The difference between the pulse peak in calibration runs in EMF and in USA15 is of 15 BC: $\text{EMF}_{Calib,peak}$ + 15 BC = $\text{USA15}_{Calib,peak}$

These values will be used in the next section.

5.8.2 Latency correction

A shift of the pulse peak was observed among different runs. The shift was not due to the latency setting change but to the uncertainty of phase of the hardware transceivers locking after the reset at the beginning of a run. The uncertainty was generating a shift of 1 or 2 BCs in the pulse position, run by run. A sketch of the issue is shown is Figure 5.43

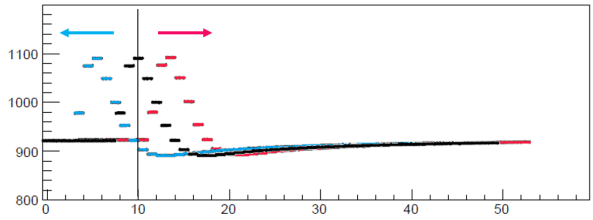

Figure 5.43: Simple drawing of the uncertainty of the peak position between different runs.

The goal of this work is to have a fixed latency upon fiber locking in order to avoid "random" misalignments.

As a first step, I redesigned the TDAQ latency value, as shown in Equation 5.4, dividing the original latency value in different parts:

$$\text{Old TDAQ value} = \text{Fixed value} + \text{LAr global} + \text{LTDB value} + \text{Correction} + \text{New TDAQ value}, \tag{5.4}$$

- *Fixed Value*. This value is set inside the firmware and is calculated to comply to the LAr global parameter and the new TDAQ value: currently it is set to the integer value of -44.

- *LAr global*. This parameter is the value used by LAr to distinguish for example between physics and calibration. This is set via TDAQ panel and written in a firmware register. For physics, it corresponds to the decimal value of 104.
- *LTDB value*. It is set directly in the firmware. The decimal value assigned to the BNL LTDB corresponds to 8 while the value for Saclay/LAL LTDB is 0, matching the latency between the two boards.
- *Correction*. This is the correction for the hardware transceiver lock uncertainty and is set in the firmware. This is explained later in the section.
- *New TDAQ value*. This value allows to choose the pulse position directly from the TDAQ panel. To have the pulse peak at BC 20 in the readout window, this value will be set directly to the decimal value 20 in the panel.

To solve the issue of the pulse misalignment happening run by run, I implemented a latency correction procedure.

As already mentioned, the K-comma symbol sent by the LTDBs has a specific periodicity. It is sent once per each bunch revolution and it comes together with the LTDB data. Also ABBA is receiving the bunch revolution information, that corresponds to the BCR sent along the TTC. From previous works [112] the value of the real hardware latency (L_{real}) was known (in hexadecimal 0x44). At the same time, the on-the-fly latency distance can be measured, knowing the time distance between the K-comma received from the LTDB and the BCR calculated in ABBA (L_{meas}). With these two values, one can calculate run by run the shift of the pulse, as shown in Equation 5.5 (which can be either positive or negative). As this information is available fiber by fiber, it allowed a dedicated correction per each fiber.

$$Correction = L_{real} - L_{meas}. \tag{5.5}$$

The correction value is also monitored inside the TDAQ partition, as shown in Figure 5.44.

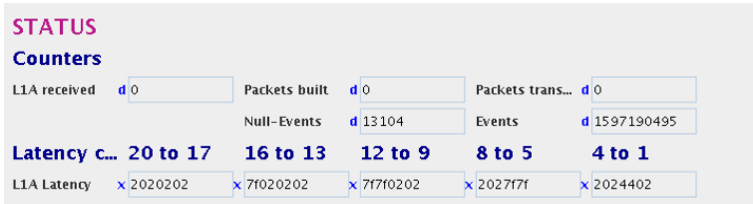

Figure 5.44: Latency correction values monitored in the TDAQ panel.

The correction value together with the other new values allows to pick up the exact data upon L1A from the circular buffer and send them out to be analysed. The results of this work can be seen in Figures 5.45, 5.46 and 5.47.

The figures show channel 7 of fiber 4 and channel 1 of fiber 18 from the same FPGA, labelled as 19:2. Three different calibration ramp runs have been taken in different moments, two on the 1st of March and one on the 2nd of March 2017. It is easy to notice that the pulse peak position is very stable and not changing from run to run. The same result has been confirmed also with other calibration runs and with physics runs.

The online correction was first verified and applied to the data coming from the BNL LTDB. The Saclay/LAL LTDB was missing a piece of hardware for receiving the TTC. Recently, the correction has also been applied to the Saclay/LAL LTDB board.

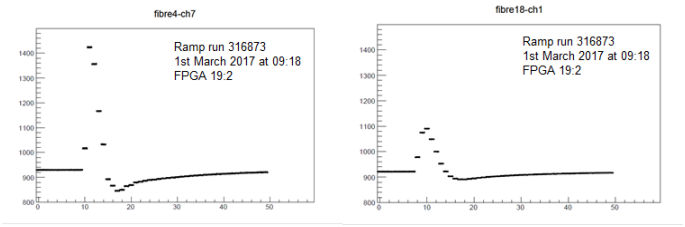

Figure 5.45: Pulse from a ramp run (316873) collected after applying the latency correction. Two channels connected to FPGA 19:2 are shown.

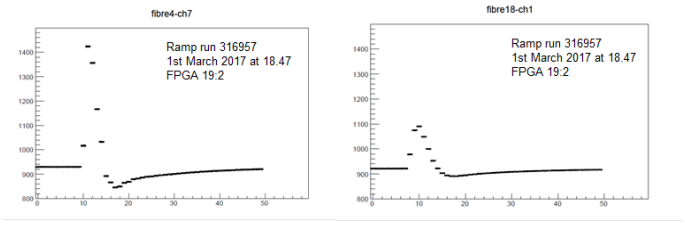

Figure 5.46: Pulse from a ramp run (316957) collected after applying the latency correction. Two channels connected to FPGA 19:2 are shown.

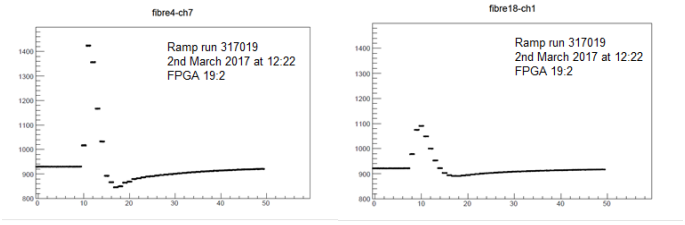

Figure 5.47: Pulse from a ramp run (317019) collected after applying the latency correction. Two channels connected to FPGA 19:2 are shown.

5.9 Identification of firmware failures

I participated to the intense debug of the demonstrator firmware during the data taking.

In this section, the firmware developments implemented in order to fix a failure in the data taking and the lack of some triggers during both calibration and physics runs are described.

5.9.1 Data taking failure

In January 2015, the ABBA data taking after some hours of stable running stopped functioning.

To debug the issue, a tool provided by the Intel Altera software, called SignalTap Logic Analizer [113] was used. This tool is like a probe, compiled together with the firmware and looking at some specific signals requested by the user. In this case, the IPbus requests and responses, passing through the back FPGA, were monitored in order to understand what was causing the blockage.

ABBA uses IPbus protocol over UDP (User Datagram Protocol) as Ethernet protocol

to communicate with the PC. IPbus is a simple packet-based control protocol for reading and modifying memory-mapped resources within FPGA IP-aware hardware [98]. The UDP is one of the core members of the Internet protocol suite, providing checksums for data integrity and port numbers for addressing different functions. Unfortunately it is unreliable as there is no guarantee of delivery, ordering, or duplicate protection. The IPbus-specific information is contained within the payload of the UDP packet, which is wrapped within an IP packet and an Ethernet packet, as shown in Figure 5.48.

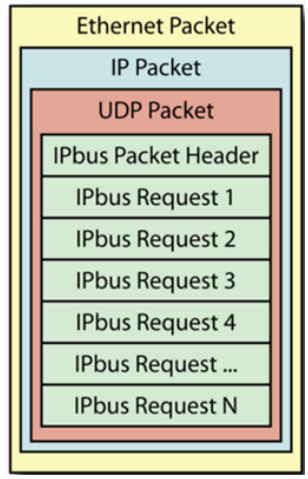

Figure 5.48: IPbus information wrapped within UDP, IP and Ethernet packet.

The Abba application on the PC sends requests (read or write) to the ABBA board. The replies can contain calorimeter data or register information.

The Ethernet frame has a complex structure: it has a minimum number of 64 bytes and a payload length that can vary between 46 and 1500 bytes. All the bits were carefully checked to find some discrepancies with the standard protocol expectations.

I found that some padding data were added to the packet. This extra information is inserted if the packet is shorter than 64 bytes. So, even if the packet was declared to contain a certain number of word, the total length of the entire information was longer, because not removed by the back FPGA firmware. This was causing a misbehaviour of front FPGA state machines in charge of handling the IPbus.

This issue was not there from the beginning and probably it was caused by a default option setup during the regeneration of the XGMAC core using a new version of Quartus II. The removal of the extra padding data from the packet in the back FPGA firmware was implemented.

Moreover, some "garbage" was generated in the front FPGA and was sent out to the outside world. The state machines had some misbehaviour, shown in Figure 5.49 (bottom), getting stuck on some states. The good behaviour is shown in the top part of Figure 5.49. That caused the continuous generation of too long packets, containing random data. When the back FPGA tried to identify the addresses it flooded the system sending broadcast requests to all the other modules and to the other ABBA board as well. The addresses were not recognised and the system was blocked.

To solve this trouble, a new reset structure was implemented, as mentioned in Section 5.7. Furthermore, I changed most of the front FPGA firmware processes from asynchronous to synchronous and I optimized some timing constraints.

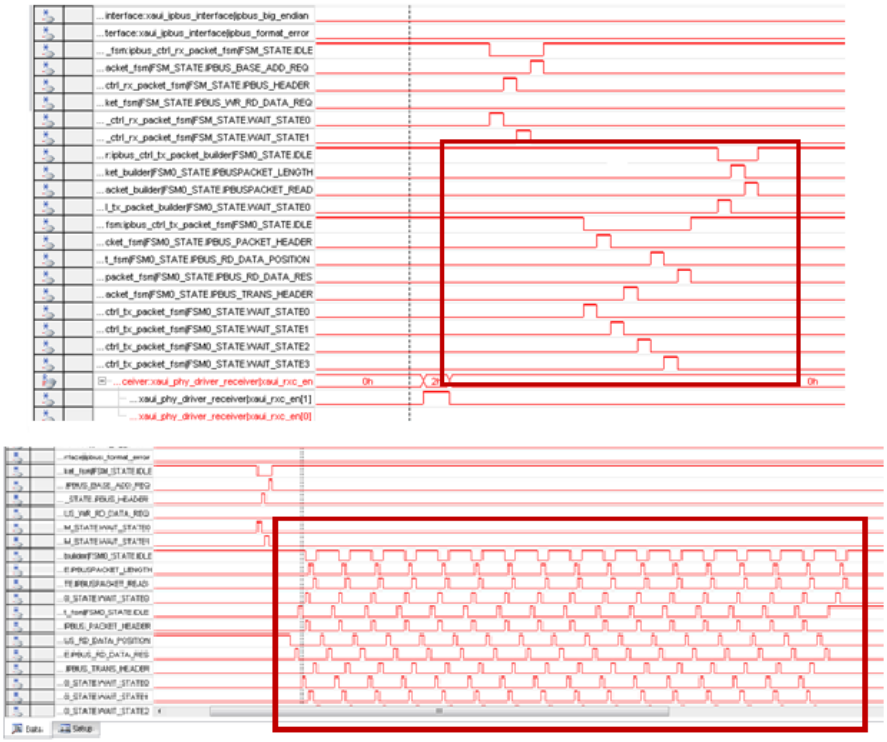

Figure 5.49: Good behaviour (top) and bad behaviour (bottom) of the state machine handling the IPbus requests.

At the same time, a length check in UDP and IP module was inserted to avoid transmission of too long packages containing wrong information. Now each protocol packet contains also a length information and if it is not respected the packet is dropped.

5.9.2 Missing triggers

During the calibration runs, it was observed that not all the events were collected. This is actually a problem when the user wants to analyse the data with the ATLAS offline analysis code (Athena Software [114]) which expect a certain number of events and crashes if this is not respected.

Different tests were done to investigate the origin of this problem. A *L1A received* counter, looking at the L1A signals matching with the required TType, was already in place in the front FPGA firmware. Two more counters along the data path to monitor the events behaviour and loss were implemented. The first counter is monitoring the *packets built* inside the firmware and the second counter is counting the *packets transmitted* to the back FPGA.

The expectation is to see all the counters showing the same number. During the data taking, the first counter was often showing a higher value with respect to the other two.

During the test, I noticed the inconsistency of the monitoring counters in the FPGA 19:1. The three counters were supposed to show the same value but the result was different:

- the L1A received counter was stuck at 0
- the packet built counter was correctly at 3000
- the packet transmitted counter was also stuck at 0

If not all events are recorded, the L1A received must be bigger than the other two. A set of calibrations was taken, excluding FPGA 19:1. All the runs recorded 100% of the events. The number of samples used were 40, 50, 60 and in all the cases was reached the 100% of events recorded.

In addition, I recorded other calibration runs with 3000 events, this time varying both the frequency of the triggers received by ABBA and the number of samples. At the same time FPGA 19:1 was excluded. The expectation was that decreasing the frequency value would have helped in solving the problem. As shown in Figure 5.50, the results were difficult to interpret.

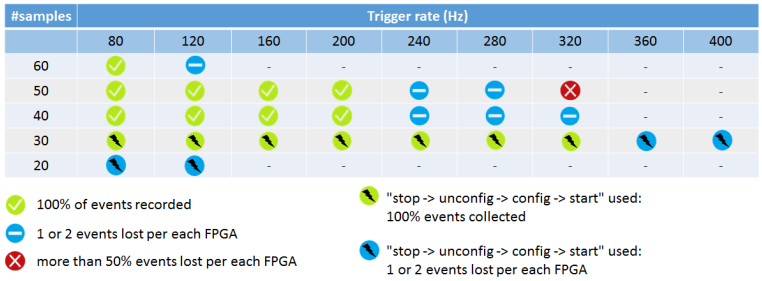

Figure 5.50: Percentage of events successfully recorded for various number of samples and different trigger rates.

The green symbols indicate that it was possible to collect all the events. For what concern the green label with the flash, a special procedure had to be used. To be able to collect all the events, the reconfiguration of all the ABBA settings was necessary. The expectation would have been to collect all the events just starting and stopping a run. The blue symbols indicate that only few events were lost during the data taking. In the particular case of the symbol with the flash, more than a couple of events were lost. Then, for what concern the red symbol, it was impossible to collect more than 50% of the events.

The conclusion from this study was that the cause of the trigger loss was not only due to the number of samples, to the trigger rate or the FPGA misbehaviour.

I continued the investigation, placing some more counters along the data path in the front FPGA firmware, as highlighted in Figure 5.51.

The counters highlighted with the red star are the counters described at the beginning of this section. The blue arrow indicates where the mismatch is placed. I added two more counters to monitor the behaviour of the TTC decoding (green) to make sure that no other events were lost. I inserted three other counters monitoring the behaviour of the FIFO and the packet construction (red).

Making use of the counters, I discovered that the "Full Packet" FIFOs were not emptied by the IPbus request sent from the PC. In Figure 5.52, the packet transfer from the circular buffer to the readout done via IPbus is sketched.

As the "even" and "odd" FIFOs are not emptied, the L1A ADC FIFO is not transmitting the data. This is exactly the point where events were lost in the firmware.

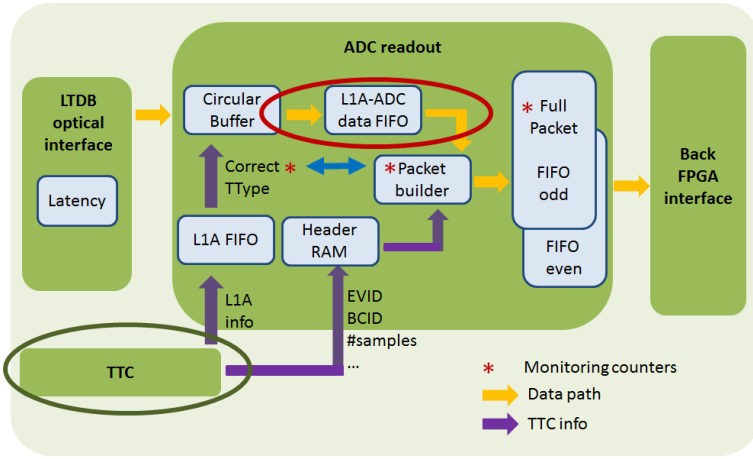

Figure 5.51: Front FPGA firmware sketch with highlighted the parts of the code where some new counters were implemented: in red the readout of the circular buffer and in green the TTC block. The blue arrow indicates the two counters having a mismatch.

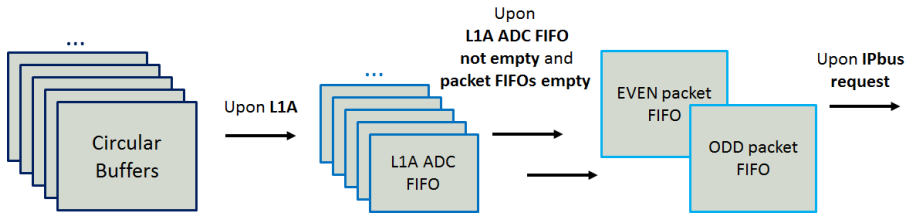

Figure 5.52: Sketch of the packet transfer from the front FPGA circular buffer to the IPbus readout.

More in detail, the PC sends an IPbus request to the even/odd FIFO where the data are stored, as shown in Figure 5.53. A physics event, coming from the front-end, is made of 12 packets: the FIFO contains all the packets forming one event. The software knows how many packets it should receive. An IPbus packet can be lost on the way to the ABBA board, or on the way back to the PC (in a router for example). In the first case, the FPGA would not be affected and the software would only generate a "timeout" message as the reply did not come back. In the second case a packet would be taken from the FIFO: once the packet is lost, the software would readout a number of remaining packets that does not match with the expected value. This would generate a new "timeout" and the even/odd FIFO not to be emptied properly. During this time events are lost, until the software sends a reset to the firmware.

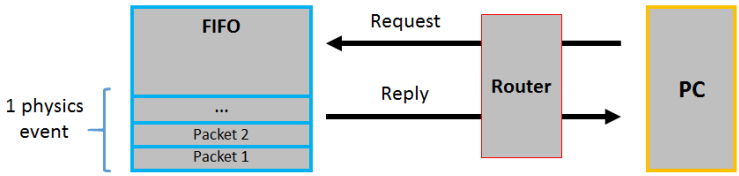

Figure 5.53: Sketch of the packet transfer between the front FPGA FIFO and the PC.

As already mentioned, the UDP protocol is not a secure protocol as it is not tracing the requests and replies. The solution adopted at the moment is to reduce the IPbus timeout duration, so that less events are lost. The more elegant solution would be to replace the UDP protocol with the TCP protocol, slower but with more control on the requests.

CHAPTER 6

Phase-I Upgrade System Test

6.1 The system

A new demonstrator system, with pre-production units of both LTDB and LDPS, will be installed in the ATLAS detector in January 2018. The ABBAs will be removed and a prototype of the LDPS system, one ATCA Carrier board and two LATOMEs, will be installed in the currently used ATCA shelf.

To establish a benchmark with the performance of the future LAr Phase-I setup, two other boards will be installed: FELIX and gFEX. FELIX (Front End LInk eXchange) will be placed close to the back-end, handling the communication with the TDAQ infrastructure, providing the TTC signals and configuring the LTDBs in the front-end. More details about this board can be found later. The gFEX (global Feature EXtractor) is a component of the Level-1 trigger Phase-I upgrade which will be used to select large-radius jets of hadrons. Data coming from the LATOMEs will be processed by the gFEX to validate the new trigger algorithms.

6.1.1 EMF setup

Before installing the system in ATLAS, it is necessary to establish the functionalities of the future LAr back-end Phase-I electronics. For this reason, a test setup is under installation in EMF. The installation started at the end of 2016 and the final setup will be the one shown in Figure 6.1. The Carrier board, on the top part of the figure, will have the role of the LTDB and of the FEXs. It will mount three LATOMEs, inject LTDB data and at the same time read them back, like the trigger system would do. The other three Carriers will mount four LATOMEs each. The LATOMEs will run the user code to compute the E_T and send data to TDAQ. Two of them will run in loopback mode, while the third will be connected to the injector Carrier.

The injector Carrier will be connected to FELIX and in parallel send out the data for monitoring along the 10 GbE link. Only one of the two Carriers in loopback mode will be connected to another FELIX, as only two FELIX PCs are available.

Figure 6.2 shows the present setup installed in EMF. Three Carriers and two LATOMEs are placed inside the central part of the crate, together with two FELIX PCs and all the other necessary components to run correctly the system.

All the functionalities have to be tested in conditions as close as possible to the final usage (in terms of power consumption, temperature and electromagnetic environment).

6.1.2 FELIX

FELIX is a new detector readout component, developed for the ATLAS upgrade. FELIX is a PC-based multi-purpose routing device that can forward data to or from multiple

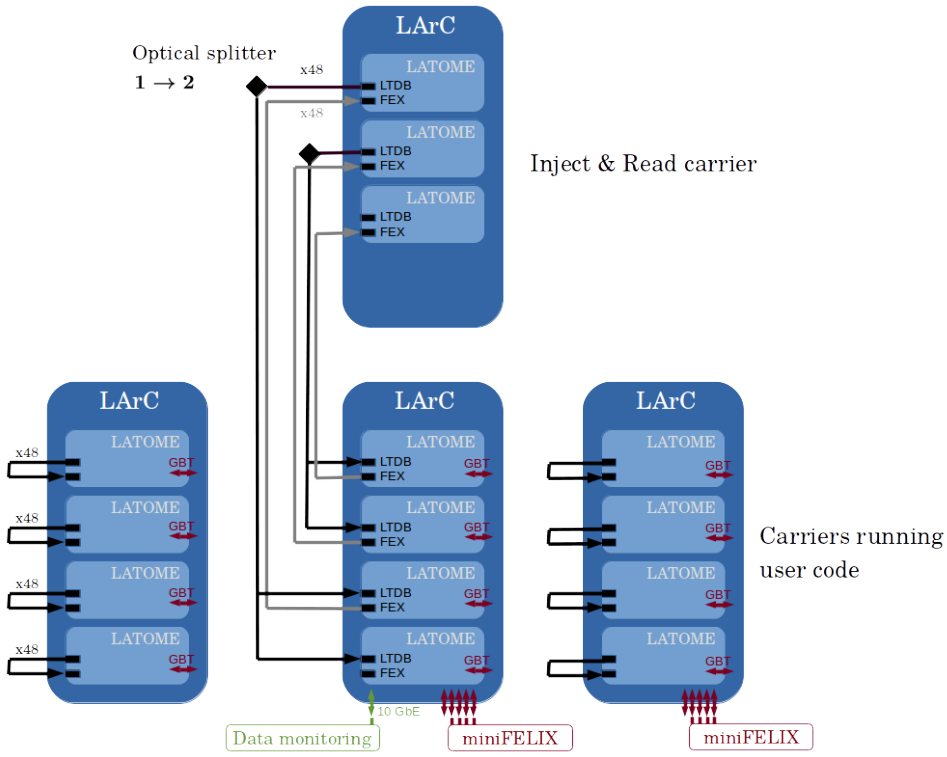

Figure 6.1: Schematic drawing of the final EMF setup.

destinations connected to a network. The hardware platform is based on the Xilinx 209 VC-709 Connectivity Kit.

As well as transferring data to and from front-ends, FELIX also interfaces with the ATLAS TTC system. The transmission of both information is done along Gigabit Transceiver (GBT) links known as E-links.

6.1.3 Project milestones

To validate the full back-end system, all the functionalities have to be tested. For this reason, several milestones (M1-M10) have been identified involving firmware, hardware and software:

- M0: TTC clock recovery and IPMC power management
- M1: 1 GbE communication via IpBus verification
- M2: Detector Control System (DCS) monitoring of LDPS implementation
- M3: FEX data path check
- M4: TTC decoding verification
- M5: TDAQ data to FELIX readout
- M6: monitoring data via 10 GbE readout

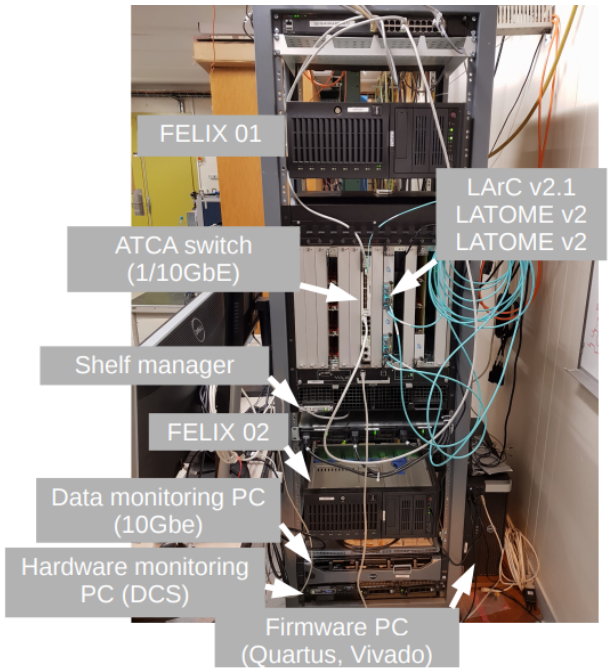

Figure 6.2: Present setup in EMF.

- M7: redo the tests up to M6 with the newest LATOME version
- M8-10: scaling tests to use several LATOME and LArC boards

The project is ongoing and the last milestones will be validated at the beginning of 2018.

6.2 LATOME firmware

Figure 6.3 presents the block diagram of the LATOME firmware. The firmware is built around a low level interface which controls the LATOME hardware components. The main data path, where data are flowing from the LTDB to the FEXs, is organised in four blocks:

- Input Stage: deserializes, demultiplexes and aligns in time the ADC data coming from the LTDB
- Configurable Remapping: remaps the input data following the detector topology
- User Code: applies Finite Impulse Response (FIR) filters to reconstruct the Super Cell E_T and compute the bunch crossing identification
- Output Summing: builds the proper geometrical sums for gFEX and jFEX.

Three other blocks are part of the firmware: the "TDAQ/monitoring" which is used to organise and transfer data to TDAQ and to the local monitoring processes, an "IPbus controller" which is the interface between the LATOME and the PCs, and the "TTC" which decodes and provides signals from the TTC system [115].

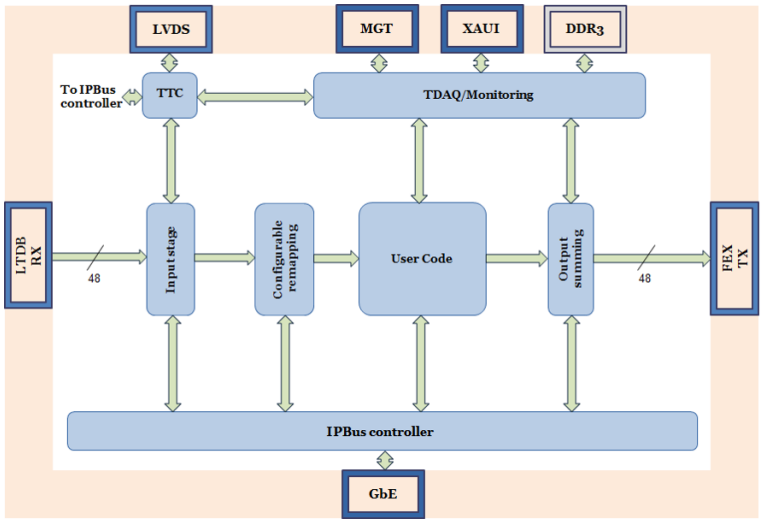

Figure 6.3: Schematic drawing of the LATOME firmware blocks.

6.2.1 Firmware modules

A brief description of the firmware modules, which can be found in the Git repository described in B, will be given in this section.

Low Level Interface

The Low Level Interface (LLI) sits in between the external environment (DDR3, GbE, XAUI, LVDS and LTDB links...) and the FPGA core. The interfaces between the LLI and the internal blocks are designed with standard Avalon Streaming and Avalon memory Mapped interfaces [116]. In this way it is possible to have standard interfaces between each block.

This block is in charge of handling the hardware resets and the clocks. In particular, it has to provide hardware synchronous reset signals in each clock domain, in order to prevent metastability in state machines.

Between the various interfaces, it is interesting to focus on the LVDS. This is the part preparing the signals for the TTC block on which I worked.

There are 4 LVDS links connected to the LATOME and they are used to carry the TTC signals coming from the Carrier. The links are implemented on hardware LVDS transmitters and receivers which can accept LVDS data up to 1.6 Gbps. The TTC information coming along the links is synchronized with 160 MHz clock. The block diagram of how the LVDS interface is organised is shown in Figure 6.4.

Table 6.1 shows how the signals coming from the LVDS lines are organised inside the LLI block. In particular, the RX data bus describes the signals entering the TTC module. A vector of 4-bit data and a one-bit "information valid" carry the necessary information.

Input Stage

The Input Stage (IStage) is fed with 16-bit data at 320 MHz. This module receives and processes the LTDB data stream, tests the pattern generator and checks the alignment of

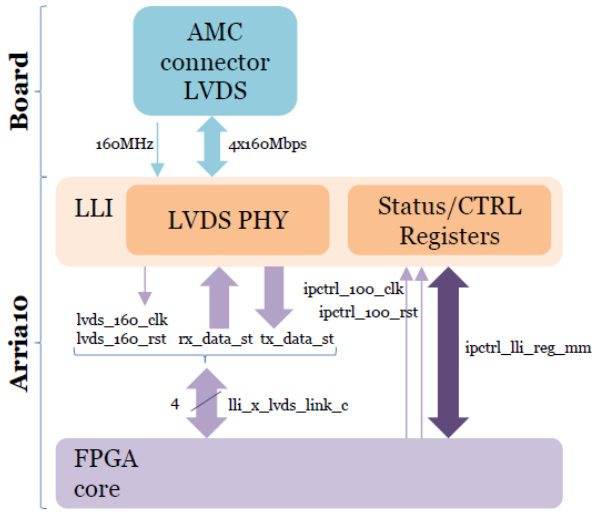

Figure 6.4: LVDS interface block diagram.

Interface		I/O	Width	Description
lli_ttc_lvds_link_c				
rx_data_st				RX data bus
	data_o	O	4	Data bus synchronized to clk_ttc_160clock
	valid_o	O	1	TTC data valid, high active.
tx_data_st				TX data bus

Table 6.1: LVDS interface description

the fibers.

A border pattern is present in each frame, so that the IStage is able to recognise the 16-bit data words and have them properly aligned. A CRC computing code is also present, so that the block can compute it from the data and compare it to the extracted CRC. A partial BCID is also sent along the data. The IStage module computes the entire value and checks that the BCID corresponds to the expected value.

The LTDB data are not arriving synchronous on the different fibers, because of the differences in time of flight and fiber lengths. Data have to be aligned and to do it the module makes sure that the first word written corresponds to the data of BCID zero. The difference between the BCID value from the TTC module and the BCID value from the data will always keep the same difference. The latency is adjusted on these two pieces of information.

The interface used to send to the data from the TTC block to the IStage is described in Table 6.2.

Configurable Remapping

The Configurable Remapping module takes care of reordering the bunch crossing data coming from the IStage, according to the detector geometry. The block groups the data from the Super Cells belonging to the same Trigger Tower.

Interface		I/O	Width	Description
ttc_istage_bcid_st				
	bcid_o	O	12	BCID
	bcr_o	O	1	BCR
	startofpacket_o	O	1	40MHz boundary for TTC information
	valid_o	O	1	Valid data

Table 6.2: TTC interface towards IStage.

The reason why the Super Cells are grouped according to the detector geometry is to prepare the information for the jFEX and gFEX and to eventually use the bunch crossing identification from a neighbouring Super Cell if the pulse shape is distorted because of saturation.

User Code

The User Code receives the data coming from the Reconfigurable Remapping, computes the E_T, the bunch crossing identification and some quality and error bits for the Output Summing block.

The User Code consists of two sub-blocks:

- the "filtering" block does the energy reconstruction and the bunch crossing identification. It runs two parallel tasks:
  - the FIR filtering which converts 12-bit ADC data to E_T
  - the algorithm for the detection of saturated pulse shapes
- the "combine" block combines the outputs from these two tasks and provides the energy in the correct bunch crossing and one quality bit.

The User Code also provides data for monitoring, error detection, histogramming and other functionalities for the calibration run.

Output Summing

The Output Summing block groups the data received from the user code, to calculate the sums over specific $\eta - \phi$ areas. Furthermore, this module adjusts the precision (number of bits) for each FEX and prepares the encapsulation of the data, headers and trailers sent to the low level interface and to the FEXs.

Packaged data are duplicated for selected channels. The set of duplicated channels and the multiplicity of the duplication is set by specific registers of the output summing block. The destination (eFEX, jFEX or gFEX) and area covered by specific optical fibers is set by the registers and by the logic of the selective duplication block.

TDAQ and monitoring

The TDAQ and monitoring block has two outputs, one to TDAQ and the other to the monitoring stream for debugging and fast online analysis.

ADC data have to be buffered to wait for a L1A for the TDAQ readout or for a Send Monitor Data request for the second path. The TDAQ data will be sent via high speed serial pair through the Carrier FPGA (that acts as a buffer) and then to the Multi-Gigabit

Transceiver (MGT) chip-set. Monitoring data will be sent via four XAUI pairs to the Carrier FPGA and then via the ATCA shelf to a host PC or farm.

Both TDAQ and monitoring will receive data from the TTC block to select the events or label them with the correct information. Table 6.3 shows the interface description for the data sent from the TTC. The same interface is used in two different clock domains, 320 MHz and 240 MHz.

Interface				I/O	Width	Description
startofpacket_o				O	1	Start of the TTC packet
ttc_istage_bcid_st						
	bcid_st					
		bcr_o		O	1	BCR
		bcid_o		O	12	BCID
		valid_o		O	1	BCID valid
ttc_mon_ttc_data_st						
	l1a_st					
		valid_o		O	1	L1A valid
		event_id				EVID
			l1a_o	O	8	L1A counter
			ecr_o	O	24	ECR counter
	ttype_st					
		valid_o		O	1	TType valid
		value_o		O	8	TType value
		event_id				EVID
			l1a_o	O	8	L1A counter
			ecr_o	O	24	ECR counter

Table 6.3: TTC interface towards TDAQ and monitoring.

Slow control

The Slow Control interface connects the firmware to the PCs and allows the user to read or change all the configuration parameters set in the firmware registers. Furthermore, it monitors the status of the system and implements the integration into the ATLAS Detector Control System (DCS). The transport mechanism for the Slow Control interface is the IPbus protocol over Ethernet. The IPbus controller is available from the CACTUS package [117].

All firmware blocks have a slow control interface, in particular Table 6.4 shows the interface with the TTC.

TTC

The TTC block receives the TTC signals along the LVDS lines, from the Carrier board. It takes care of decoding the relevant TTC information to be provided to the other firmware modules. The TTC block communicates with the IStage, the TDAQ/monitoring and the Slow Control modules and the interfaces have already been shown in the previous tables. More details about functionalities, organisation and registers will be found in the following sections.

Interface		I/O	Width	Description
ipctrl_ttc_reg_mm				TTC registers
	address_o	O	32	Address bus
	clk_o_o	O	1	IP controller 100MHz clock
	read_o	O	1	Read transfer active
	readdata_i	I	32	Read data bus
	readdatavalid_i	I	1	Read data is valud
	rst_o_o	O	1	IP controller 100MHz reset
	waitrequest_i	I	1	Write wait request
	write_o	O	1	Write transfer active
	writedata_o	O	32	Write data bus

Table 6.4: Slow control interface description for TTC block.

6.2.2 Firmware occupancy

The FPGA mounted on LATOME is an ARRIA 10 [91], model 10AX115R3F40E2SG, which is the biggest matrix of the Arria 10 family and has more than one million logic cells. The present firmware occupancy (M5) is shown in Figure 6.5: the compilation included all the blocks, while the debugging part was removed.

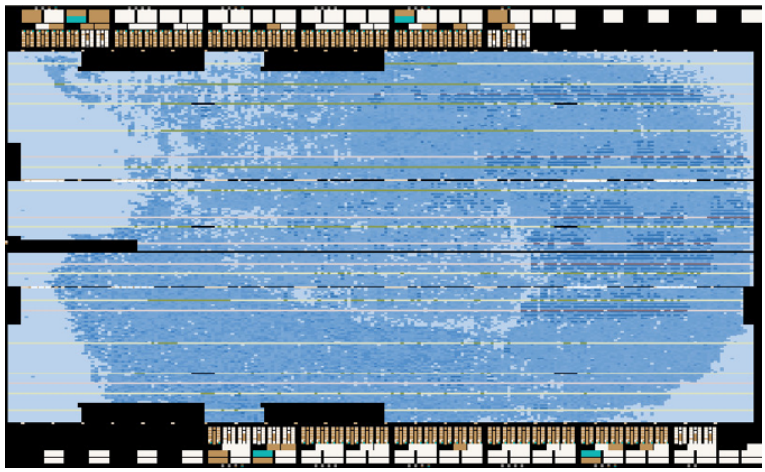

Figure 6.5: LATOME firmware occupancy.

Figure 6.6 shows the resources used by the last compiled and (almost) complete version of the firmware (M5).

6.2.3 Board characteristics and consumption

Three versions of LATOME were produced [118]. The changes are briefly summarized in Table 6.5.

The V2 board has been tested in June 2017 making use of a test firmware containing FIR filter design. Different configurations were tried, setting different numbers of FIR samples. Figure 6.7 shows the firmware occupancy versus the number of samples, with a

Resource	M5	Arria 10 total
Logic utilization (in ALMs)	191127 (45%)	427200
Total pins	284 (44%)	650
Total block memory bits	7123952 (13%)	55562240
Total RAM Blocks	792 (29%)	2713
Total DSP Blocks	248 (16%)	1518
Total HSSI RX channels	53 (80%)	66
Total HSSI TX channels	54 (82%)	66
Total PLLs	76	138

Figure 6.6: Resource usage comparison between complete LATOME M5 firmware and FPGA availability.

Revision	Changes	Date
LATOME V1	Initial release: engineering sample FPGA version	September 2015
LATOME V2	New power scheme and production FPGA version	November 2016
LATOME V3	New I2C bus	September 2017

Table 6.5: LATOME version history.

details description of the DPSs, RAM and logic percentage used per each sample. It was not possible to test more than 28 samples, as the fit procedure during the compilation did not complete.

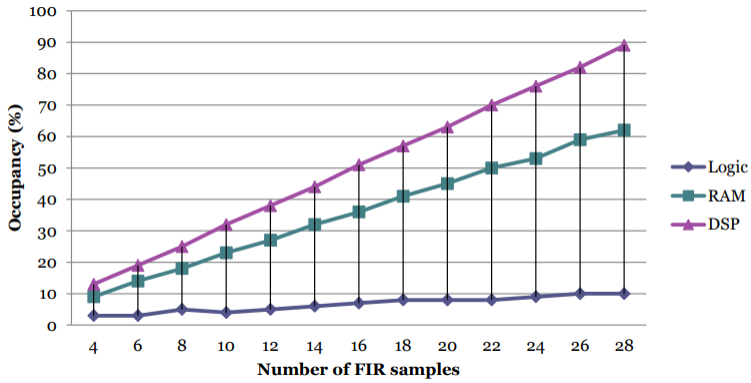

Figure 6.7: Occupancy test with FIR design.

Furthermore power consumption and temperature tests were done on the same board, with the same test firmware and the results are reported in Figure 6.8. The maximum power of the board is limited to 90 W. During the tests the power consumption was estimated to be 79 W for 28 samples, making use of 90% of the available DSP, 60% of RAM, 10% of the logic. The temperature estimation was of 83°C at 79 W, with continuous airflow on the table were the board was tested.

The power consumption and the temperature are not expected to raise with V3 board. The next measurements will be done in the near future with the official LATOME firmware.

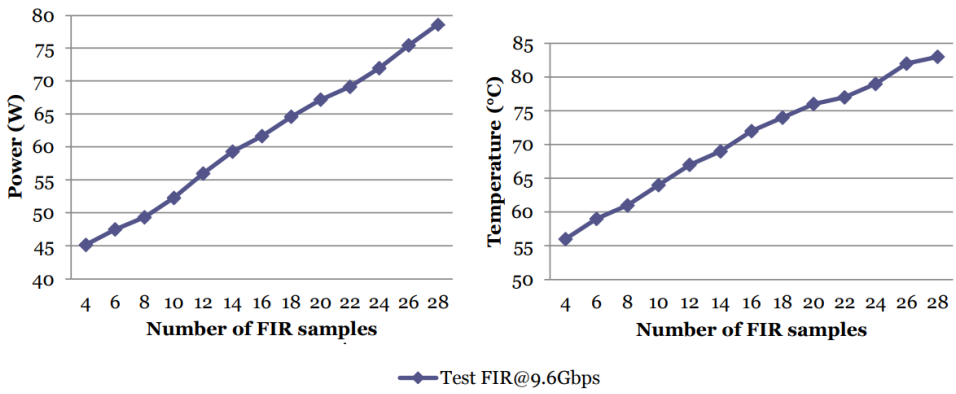

Figure 6.8: Power and temperature tests with FIR design.

6.3 Firmware contribution

I was in charge of the development of the firmware module for decoding the TTC signals and commands. More details about how TTC works can be found in Appendix C.

In EMF the TTC information, channel A and channel B, is delivered by the TTCvi module [119]. In ATLAS, this is the module which interfaces the TTC system to CTP. The module is fully programmable from the VMEbus.

As shown in Figure 6.9, the TTCvi module is not directly connected to the LATOME but there are some intermediate steps. The two TTC channels are sent to FELIX. To decode the channels, FELIX makes use of standard VHDL code developed at CERN [120] and already used in ABBA. The code allows to extract all the important TTC information and commands.

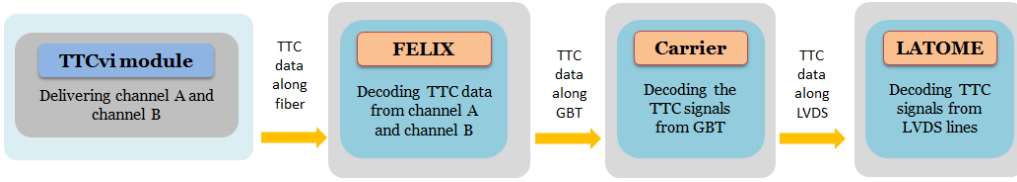

Figure 6.9: TTC chain in EMF.

FELIX is then re-encoding the TTC along the GBT transmission protocol. The link connection is called by FELIX E-link. Any E-link can carry the TTC information. Each of them can transmit 80, 160 or 320 Mb/s to transfer 2, 4 or 8 TTC bits on the 40 MHz TTC clock and with fixed latency. In general, the user can choose between different TTC E-link configurations, thanks to an E-link selector interface. Figure 6.10 shows the different options that can be chosen, depending on the needs of the detector. The one marked in red is the one selected for LAr.

The Carrier board is taking care of decoding the GBT information and re-encoding it for the LATOME. In particular, the Carrier board is encoding the signal in the protocol shown in Figure 6.11. The signals L1A, BCR, ECR and the channel B (bit by bit) arrive to the LATOME every frame. The role of the TTC module in the LATOME is to decode all the signals, and in particular the information coming along the channel B and prepare

#	speed	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	80 Mb/s							**B-chan**	L1A
1	160 Mb/s					**B-chan**	ECR	BCR	L1A
2	160 Mb/s					Brcst[2]	ECR	BCR	L1A
3	160 Mb/s					BCR	BCR	BCR	BCR
4	320 Mb/s	**B-chan**	Brcst[5]	Brcst[4]	Brcst[3]	Brcst[2]	ECR	BCR	L1A
5	320 Mb/s	Brcst[6]	Brcst[5]	Brcst[4]	Brcst[3]	Brcst[2]	ECR	BCR	L1A

Figure 6.10: TTC chain in EMF.

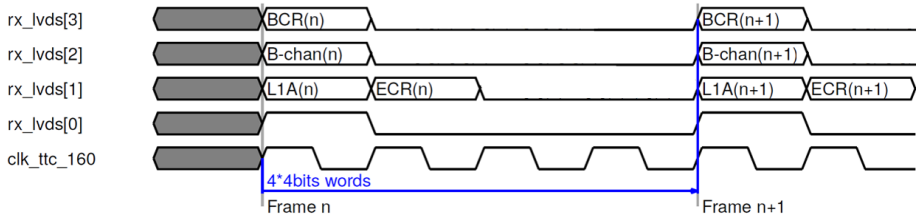

Figure 6.11: Custom protocol used for data transmission along LVDS links.

the counters related to the BCR and ECR.

6.4 TTC firmware module

Signals coming from the LVDS links arrive to the LLI interface which prepare them for the TTC block. A four-bit vector containing the LVDS data and a valid signal enter the block.

As shown in Figure 6.11, LVDS 0 carries a synchronization pattern, used to recognise the first word coming along the frame. LVDS 1 carry the L1A and the ECR on the first and second word respectively. The first clock cycle of the frame, of LVDS 2, corresponds to the channel B bit. Finally, the LVDS 3 on the first word position has the BCR.

Due to a particular hardware configuration in EMF, the ECR and BCR information are swapped. This lead to the fact that in EMF the ECR arrives on LVDS 3, while the BCR arrives on LVDS 1.

Figure 6.12 shows how I organised the TTC block. When the data are valid, the *TTC decoding top level* entity takes care of decoding the LVDS lines.

6.4.1 The decoding

As a first step the entity contains a state machine, shown in Figure 6.13, dedicated to the synchronization of the LVDS 0 signal.

It passes from IDLE to the first state (RST) if a *start of packet* (sop) signal is received. From the RST state the machine goes directly to the SYNC state: a cyclic counter has to align with the LVDS 0 synchronization signal. Once this is true, a *ttc locked* signal is set to `HIGH`. The alignment is monitored with the signal *synch hilo*, and if no alignment is reached after a 32 clock cycles, *synch hilo* is set to `HIGH`, bringing the state machine

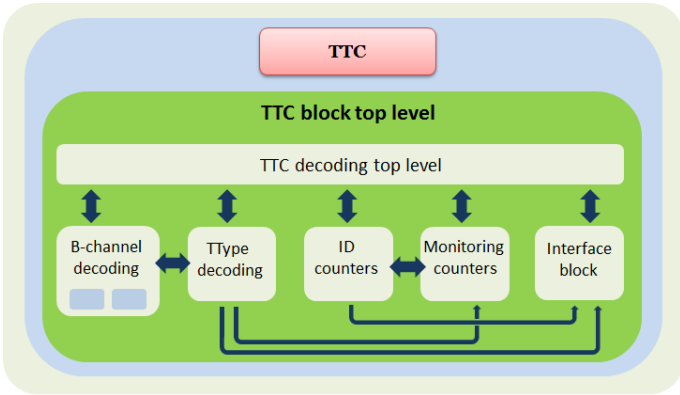

Figure 6.12: Schematic view of the TTC block.

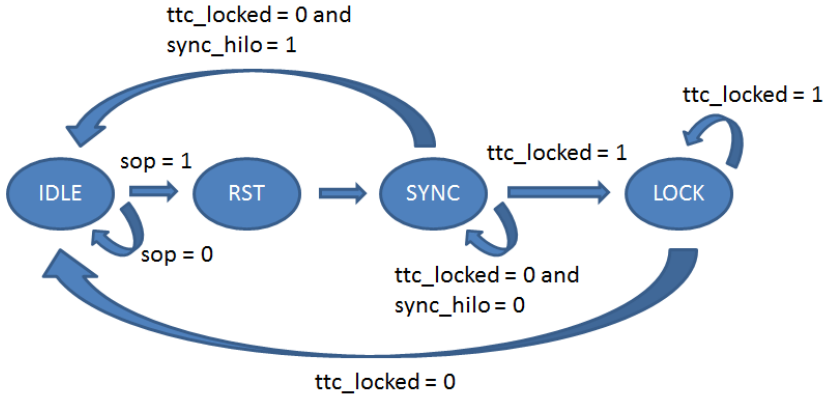

Figure 6.13: Diagram showing the state machine used to synchronize the decoding.

back to IDLE. If the state machine enters the LOCK state, the real decoding can start. In case the synchronisation is lost (due to a misalignment o no data transmission), *ttc locked* is `LOW` and the state machine goes back to IDLE. A second state machine decodes word per word (four per each frame) the LVDS lines. As shown in Figure 6.11, a word is made of 4 bits: `WORD1` corresponds to the first clock cycle of the frame and contains the synchronization information, the L1A bit, the channel B bit and the BCR bit.

The *TTC decoding top level* contains all the other entities. Looking at the schematic, the first module on the left is the *B channel decoding*.

All the data contained in the channel B, both broadcast and long address command are decoded. The channel B information from the top level state machine changes its length from one clock cycle to four clock cycles on the 160 MHz clock, and that is how it enters the entity. This is done to maintain coherence with the 40 MHz clock. An enable signal comes together with the data and is one clock cycle long. A simple entity called *state machine decoding counter* takes care of retrieving bit per bit the data coming along the channel B and put them inside a vector signal. The outputs of the entity contain the broadcast address and data, as well as the long command address, subaddress and data. Furthermore, two signals are set as an output to indicate the presence of a long or a short

channel B. The end-of-channel information is used later in the TType module.

The output of the entity *state machine decoding counter* is an inverse bit counter with an end of counter signal, to be used in combination with a vector in the state machine, such that bit per bit the vector can be filled. The length of the vector can be chosen by the user.

6.4.2 Building the other TTC signals

The long address command data are sent to the *TType decoding* entity. The TType information is written in the data part of the long command and the LATOME firmware requires it to be decoded. In this entity, a process is checking the values of the subaddress and of the data. The TType is present only when the subaddress value is zero. The TType values are decoded and later sent to the monitoring.

The *ID counters* receives from the top level both the BCR and the ECR. From these two values the BCID and EVID are calculated as explained in Appendix C.

6.4.3 TTC monitoring

The *monitoring counters* entity collects all the signals calculated and decoded in the previous entities. Inside this entity, counters to monitor how many L1A, ECR, BCR and TType are received have been implemented. In this way, it is possible to compare, for example, the value displayed in the TDAQ panel with the ones computed in the firmware. Furthermore it is important to always check that the number of L1A matches the number of TType decoded. At the same time, also the BCID and the EVID are two inputs of the entity.

An online monitoring of the BCR period value is implemented and a counter is increasing every time that the period value matches with the expected 3564 (the origin of this value is explained in Appendix C). This allows a direct comparison with the BCR counter: the two values must match. At the same time, a counter looking at value different from 3564 is implemented to make the readout easier.

Finally a TType evaluation is implemented. A process checks which value is contained in the 8 bit and a counter is increased if it matches with the TType required by the user. The counter can be compared with the TType counter to verify the correct behaviour of the firmware.
What makes possible the readout of the data is the interface with the slow control. Writing and reading processes allow the data to be sent to the outside world along IPbus over UDP protocol, as done for ABBA.

6.4.4 The interfaces

The last module is the *interface block*, preparing the interfaces towards the IStage and the TDAQ and monitoring, shown in Figure 6.14 and 6.15.

The IStage protocol, in Figure 6.14, is sent along the 320 MHz clock. A FIFO is handling the clock domain crossing. A start of packet (`startofpacket_o`) tells the start of the frame and a valid signal (`valid_o`) informs the receiver of the validity of the data. Then, the BCR signal (8 clock cycles long) (`bcr_o`) corresponds to a zero on the BCID value (`bcid_o`).

The TDAQ and monitoring protocol, in Figure 6.15, is sent on two different clock domains: 320 MHz and 240 MHz. Again FIFOs are use to do the clock domain crossing. As before, the BCR (`bcr_o`) and BCID (`bcid_o`) values are sent aligned with the other frame information, together with a valid signal (`bcid_st.valid_o`). In addition, a valid info

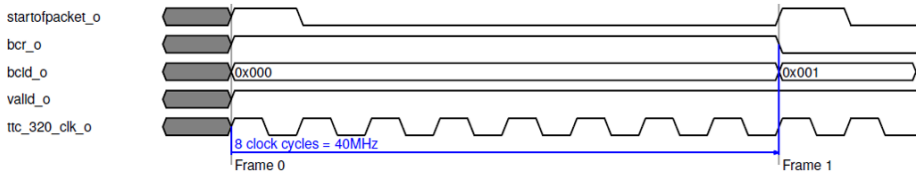

Figure 6.14: Custom protocol used for communication with IStage block. A start of packet (`startofpacket_o`) tells the start of the frame and a valid signal (`valid_o`) informs the receiver of the validity of the data. Then the BCR signal (8 clock cycles long) (`bcr_o`) corresponds to a zero on the BCID value (`bcid_o`).

corresponding to a L1A (`l1a_st.valid_o`) is sent when a L1A is received and decoded, together with the two EVID subcounters: the L1A counter (`l1a_st.l1a_count_o`) and the ECR counter (`l1a_st.ecr_count_o`). Furthermore, the TType valid (`ttype_.valid_o`) info informs of the presence of this signal and as a consequence the 8-bit information is sent in the valid frame (`ttype_st.value_o`). The EVID (`ttype_st.ecr_count_o` and `ttype_st.l1a_count_o`), which matches the previous L1A EVID, is sent together with the TType. This is done to allow the monitoring to match the two information.

6.4.5 Simulations

Per each entity, I built a specific simulation, so that every time that a part of the firmware is modified, the correct behaviour can be immediately checked. Here I will show only one example: the top level simulation.

Figure 6.16 shows how signals are grouped inside the simulation. This simulation represents a summary of the behaviour of the entire block.

Figure 6.17 shows a zoom on two groups, both in the top level decoding entity. The picture shows the locked state and the locked signal (yellow) which allows the decoding to start, and at the same time the LVDS words and their decoding state (orange).

To setup each simulation, a testbench file is needed. In particular for the top level simulation also a data file was required to be able to simulate the LVDS lines. Thanks to an entity already used for simulations in ABBA, called *file reader hex*, I built a data file, where each line corresponds to a word. In this way, it was easy to inject L1A, ECR, BCR and channel B signals and simulate my code.

As a summary of the entire block, this simulation contains also the interfaces towards the other firmware blocks. Figure 6.18 (top), for example, shows the good behaviour of the BCR and BCID signals, with a detailed zoom (bottom) on the good signals alignment. For this simulation a dummy value was used to temporary check the BCR (7 instead of 3563).

Figures 6.19 and 6.20 show the simulation of the interface towards TDAQ and monitoring. In particular, the top picture shows the L1A valid together with the EVID information, while the bottom picture shows the TType value, with the respective valid and EVID. In both cases, both clock domains are shown.

6.4.6 Tests in hardware

The EMF setup is available for testing. I tested my code in hardware to verify again the correct behaviour of my block.

Figures 6.21, 6.22 and 6.23 are produced with SignalTap. The signals shown in the figures are from the *TTC decoding top level*. In particular, the four LVDS lines (lvds_rx_data)

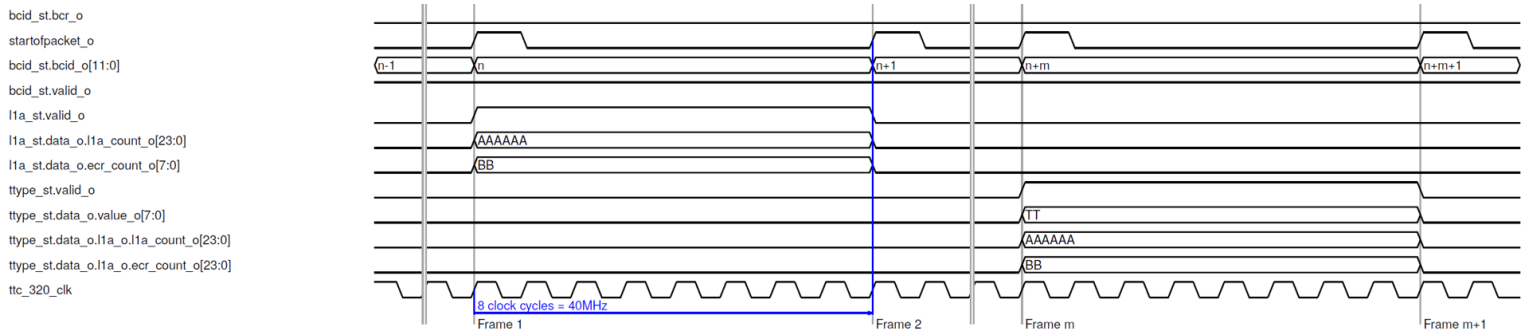

Figure 6.15: Custom protocol used for communication with TDAQ and monitoring blocks. The BCR (`bcr_o`) and BCID (`bcid_o`) values are sent aligned with the other frame information, together with a valid signal (`bcid_st.valid_o`). A valid info corresponding to a L1A (`l1a_st.valid_o`) is sent when a L1A is received and decoded, together with the two EVID subcounters: the L1A counter (`l1a_st.l1a_count_o`) and the ECR counter (`l1a_st.ecr_count_o`). Furthermore the TType valid (`ttype_st.valid_o`) info informs of the presence of this signal and as a consequence the 8-bit information is sent in the valid frame(`ttype_st.value_o`). The EVID (`ttype_st.ecr_count_o` and `ttype_st.l1a_count_o`), which matches the previous L1A EVID, is sent together to the TType.

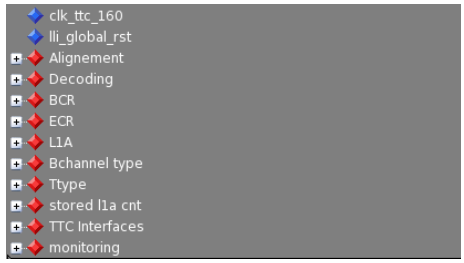

Figure 6.16: Top level simulation, groups of signals.

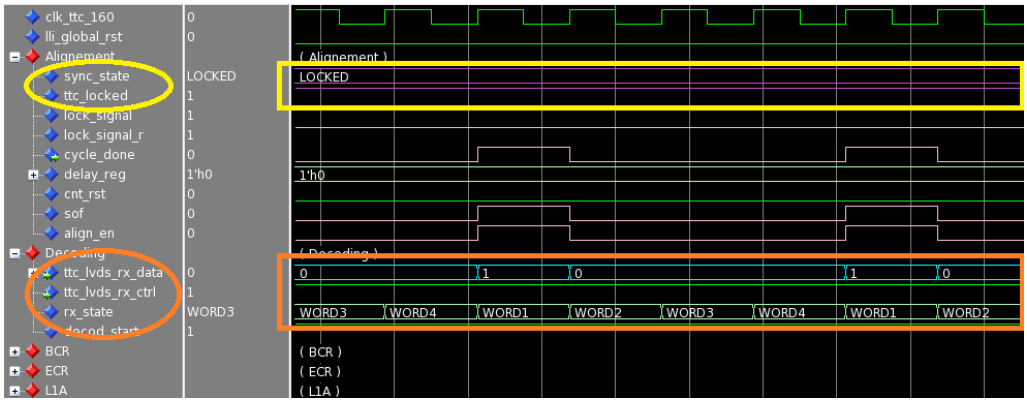

Figure 6.17: Top level simulation, zoom on locked state and signal (highlighted in yellow) and LVDS words and decodeding (highlighted in orange).

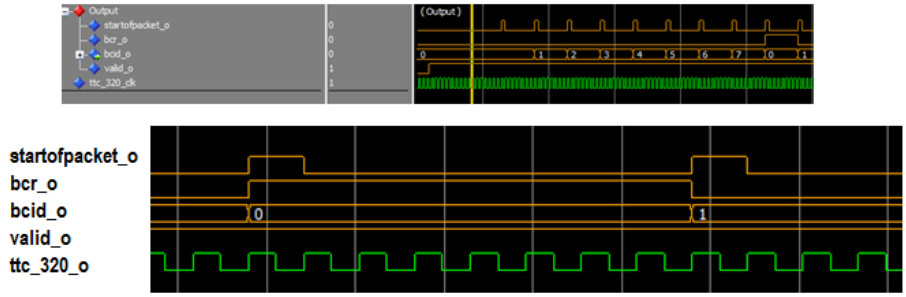

Figure 6.18: IStage simulation details: the top picture shows the simulation of the protocol towards the Istage block. The second picture is a zoom around the BCR event.

with the synchronization pattern on line 0 and the channel B idle on the line 2 are visible. All the signals shown are on the 160 MHz clock.

When a L1A, BCR or ECR is present on the LVDS line (as a 1-bit information), they are decoded as a 4-bit signal, to maintain the coherence with the TTC clock.

The output signals of the TTC block, the interfaces towards the other modules, have also been tested in hardware.

Figure 6.24 shows the correct behaviour of the signals sent along the IStage protocol. In this case, it is possible to see that the BCID value reaches the correct period value

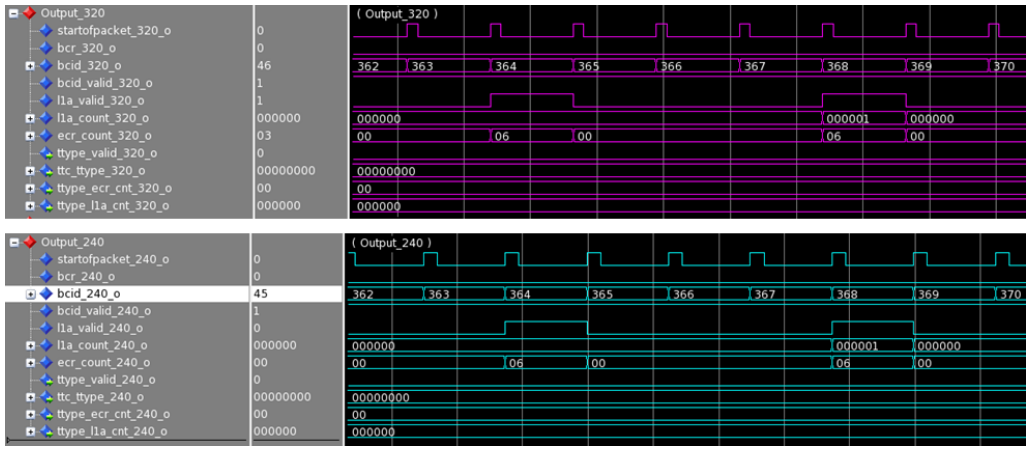

Figure 6.19: TDAQ and monitoring simulation, detail on L1A transmission. The top figure shows the protocol on the 320 MHz clock while the bottom figure shows the protocol on the 240 MHz clock.

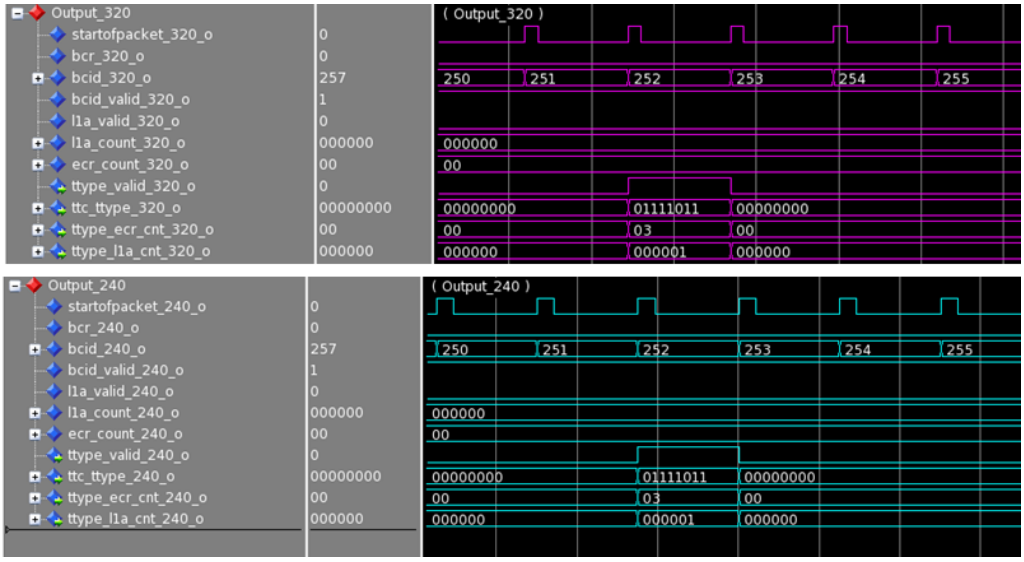

Figure 6.20: TDAQ and monitoring simulation, detail on TType transmission. The top figure shows the protocol on the 320 MHz clock while the bottom figure shows the protocol on the 240 MHz clock.

(DEB). The BCR comes immediately after this value and resets the counter to zero. The signals shown here are on the 320 MHz clock domain.

Figures 6.25, 6.26 and 6.27 show the results on the TDAQ and monitoring protocol. To top part of each figure shows the signals on the 320 MHz clock domain, while the bottom part always shows the protocol on the 240 MHz clock domain.

Figure 6.25 shows the BCR and BCID signals correct behaviour, as shown also for the IStage. In both cases, the BCR is arriving immediately after the value 3563 and then the counter is properly reset.

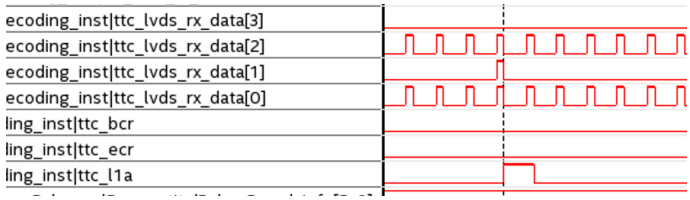

Figure 6.21: L1A detection and decoding in the TTC top level. The four LVDS lines (lvds_rx_data) with the synchronization pattern on the LVDS line 0, the channel B idle on the LVDS line 2 and the L1A on the LVDS line 2 are visible.

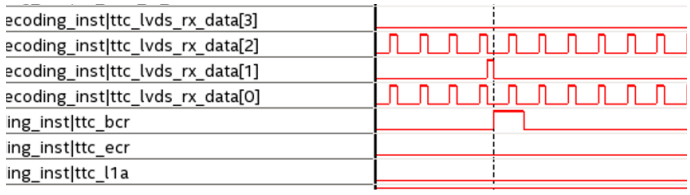

Figure 6.22: BCR detection and decoding in the TTC top level. The four LVDS lines (lvds_rx_data) with the synchronization pattern on the LVDS line 0, the channel B idle on the LVDS line 2 and the BCR on the LVDS line 1 are visible.

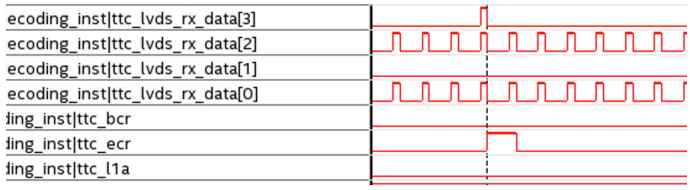

Figure 6.23: ECR detection and decoding in the TTC top level. The four LVDS lines (lvds_rx_data) with the synchronization pattern on the LVDS line 0, the channel B idle on the LVDS line 2 and the ECR on the LVDS line 3 are visible.

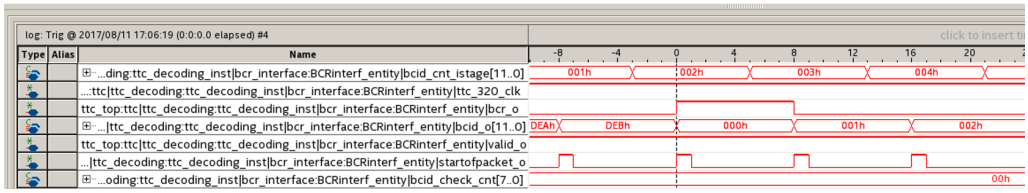

Figure 6.24: IStage protocol tested in hardware: the BCID value reaches the correct period value (DEB) and the BCR comes immediately after to resets the counter to zero.

Figure 6.26 shows the L1A information, associated to the EVID counter again on both clock domains. The information is properly aligned. The same can be said about Figure 6.27, which shows the TType value and the EVID associated, again on both clocks.

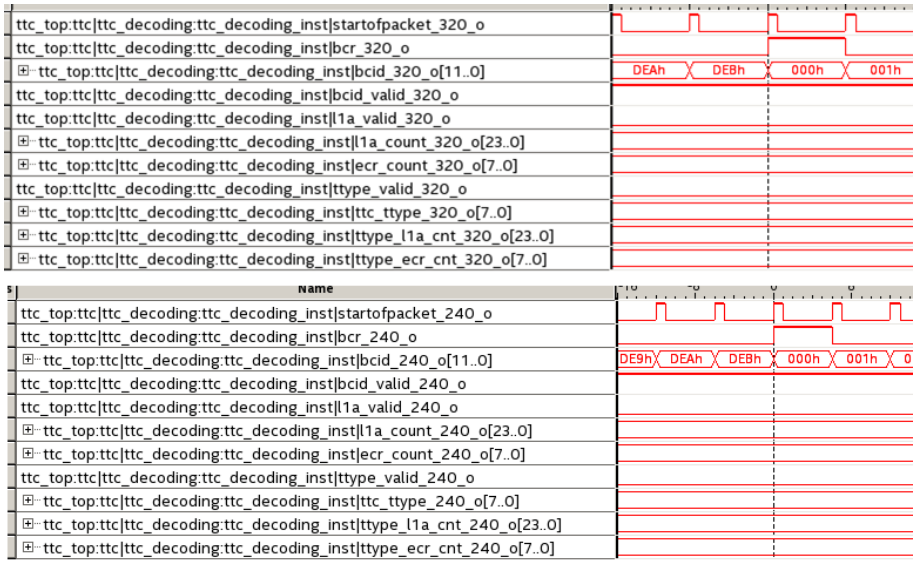

Figure 6.25: TDAQ and monitoring protocol tested in hardware: correct transmission of BCR and BCID signals. The top picture shows the 320 MHz clock domain, the bottom picture shows the 240 MHz clock domain.

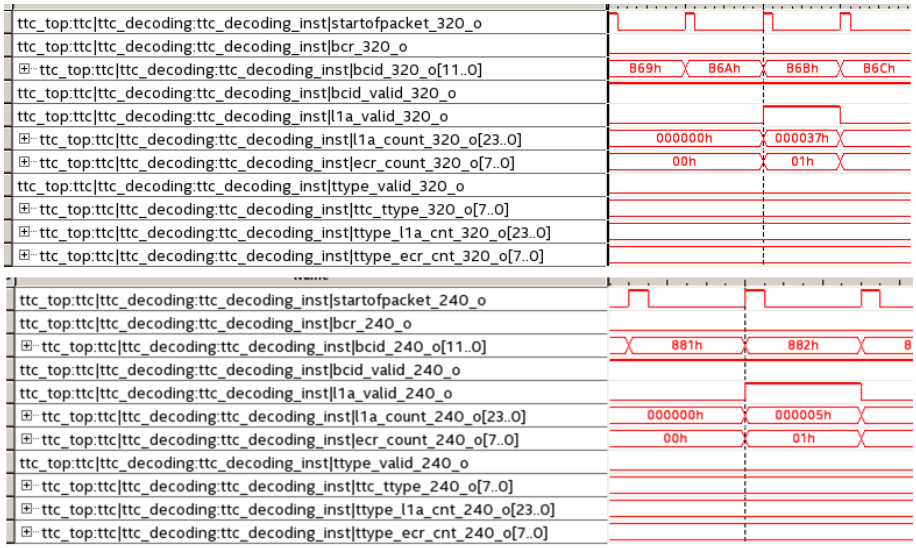

Figure 6.26: TDAQ and monitoring protocol tested in hardware: correct transmission of L1A and EVID signals. The top picture shows the 320 MHz clock domain, the bottom picture shows the 240 MHz clock domain.

6.4.7 Registers

As already mentioned for the TTC monitoring block, I implemented some registers to store the most relevant information. Upon IPbus requests, the data are readout from the

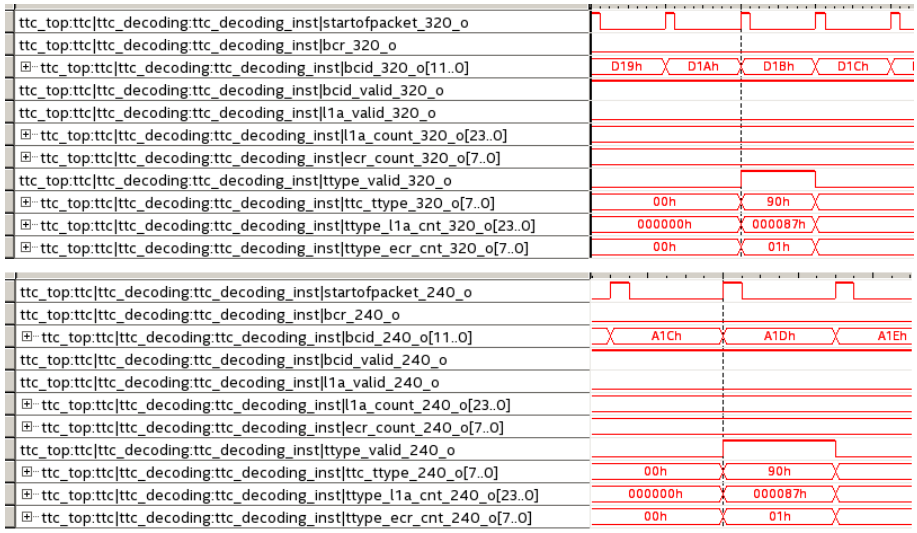

Figure 6.27: TDAQ and monitoring protocol tested in hardware: correct transmission of TType and EVID signals. The top picture shows the 320 MHz clock domain, the bottom picture shows the 240 MHz clock domain.

firmware.

I implemented some python scripts to create register constants and read the registers. A *TTC.py* file contains the implementation and the description of all the registers. This file is implemented together with the firmware and generates some constants (name for the register addresses, length of the registers...) to be used in the code. An example of the code is reported here:

```
'ttc_soft_reset': { 'description': 'soft reset', 'type': 'register', 'cardinality': 1, '
     order': 1,
  'permission': 'rw',
  'definition': {
    'reserved':            { 'description': 'Reserved, do not use', 'bits': '31:3'},
    'soft_reset_partition':{ 'description': 'partition', 'bits': '2', 'reset_value': '0'},
    'soft_reset_code':     { 'description': 'code', 'bits': '1', 'reset_value': '0'},
    'soft_reset_mon_cnt':  { 'description': 'mon', 'bits': '0', 'reset_value': '0'},
  },
},
```

Some of the constants obtained from this block are:

```
TTC_REGISTERS_TTC_SOFT_RESET_REGISTER_BASE_ADDRESS = 0

TTC_REGISTERS_TTC_SOFT_RESET_REGISTER_RESERVED_FIELD_WIDTH = 29

TTC_REGISTERS_TTC_SOFT_RESET_REGISTER_SIZE = 1

TTC_REGISTERS_TTC_SOFT_RESET_REGISTER_SOFT_RESET_CODE_FIELD_BIT = 1
```

The first constant corresponds to the address of the soft reset register. The second constant gives the number of bits contained in the register reserved field, 29 as described above. The third constant describes the size of the register, that is 1 in this specific case. Then, the last constant gives the bit position in the soft reset vector of the code reset, so bit 1 as set before.

A script interface has been created per each firmware block. Here are reported the scripts used by the TTC.

The *TTC signals monitoring counters* block contains other sub-blocks. Each sub-block is made of two registers of 32 bit. The counters shown here in the firmware correspond to a 64-bit information. They are split in the upper and lower 32 bits, as the maximum available register length is 32 bits. Here the L1A, BCR, ECR, TType and TType value are checked and compared.

The *BCR periodicity monitoring* block contains again the BCR counter. I also implemented a period monitoring: every time that the BCR is appearing the value of the BCID is checked. In addition, another counter is checking whether the BCR appears in presence of a wrong BCID.

The *TType selector* register is a 32-bit register, where only 8 bit are used. This is a writable register where a TType value can be written and used in the monitoring entity. In the code, a process makes a comparison between the selected TType and the decoded one, incrementing a counter every time that there is a good match.

The *TType readout* register allows to read which TType has been set by the user for the monitoring.

The *Reset* is a 32-bit register where, for the moment, only 3 bits are used. Each of them can raise a specific reset inside the firmware. The reset in position 0 resets all the monitoring counters. Bit 1 resets the entire code. Bit 2 will be connected to the TDAQ panel (for example, it will be issued before starting the data acquisition) and reset again the monitoring counters.

Then, the *BCID, EVID and TType Status* block is printing out the TType, BCID and EVID value upon read request.

Shadow registers are implemented to read simultaneously the counters: when a script is launched, the information contained in a firmware register is transferred to another register which is then read through IPbus.

A screenshot of the script interface, showing scripts for LLI, TTC, FPGA info and Istage, is shown in Figure 6.28.

```
 1.                : List nodes
 2.                : Read/Write nodes
 3.                : Ping
 4. LLI            : Soft global reset
 5. LLI            : TFW: LLI: Soft reset 160MHz clock domain
 6. LLI            : TFW: LLI: Soft reset 100MHz clock domain
 7. LLI            : IO Pll soft reset
 8. LLI            : Status
 9. LLI            : Stop LTDB data
10. LLI            : Start LTDB data
11. LLI            : Disable LTDB inputs
12. LLI            : Enable LTDB inputs
13. TTC            : TTC signals monitoring counter
14. TTC            : BCR periodicity monitoring
15. TTC            : TType selector
16. TTC            : TType readout from register
17. TTC            : Reset of the monitoring counters
18. TTC            : BCID, EVID and TType Status
19. fpga           : Git information
20. fpga           : Latency values
21. istage         : Status
22. istage         : Status (live)
23. istage         : Clear all channels active
```

Figure 6.28: Script interface to readout LLI, TTC, FPGA info and Istage registers.

6.4.8 Standalone validation

At the end of October 2017, an integration week has been organised to validate some of the project milestones. One of the main points of the week was to validate the TTC block. The good functionality was already established during the previous integration week in September. The aim of these new tests was to check them over long runs in standalone mode, so only with TTC part of the firmware.

The test plan required to check the following:

- the numbers of L1A, Ttype, BCR and ECR, making use of the TTCvi scripts and of the TDAQ partition
- the BCR period
- the TType value.

I recorded runs with increasing frequency and number of events to have a clear picture of the system. Before starting the data acquisition, all the counters were reset to zero, to make sure that nothing was interfering with the system. In addition, the counters were checked only after the stop of the run to avoid misreading due to IPbus time. The counters were checked both with the SignalTap tool and with the python scripts.

TDAQ partition runs

As a first test, a run with 3000 events at a rate of 40 Hz (the minimum available) was collected with the TDAQ software. Figure 6.30 shows in the top part the SignalTap results. In dark blue, the number of BCRs and the number of good BCR periods received by LATOME are highlighted. As expected, the two counters are matching, showing the stability of the BCR reception. In light blue, the number of L1As, the total number of TTypes and the number of TType with value $0x11$, are shown. The default TType value in the TDAQ partition corresponds to $0x11$ and is the value used to label the calibration runs. The three counters match as expected and Table 6.6 shows a summary of the results.

The second run contains 256000 events. The event rate was 2.5 kHz. Again, in dark blue the number of BCRs and the number of good BCR periods received by LATOME are highlighted. In light blue the number of L1As, the total number of TTypes and the number of TType with value $0x11$, are highlighted. As shown in Figure 6.30 and Table 6.7, there is good match between the counters.

The last test with the TDAQ partition was done again collecting 256000 events. The maximum possible frequency with this type of run was limited at 6.5 kHz. As before, in dark blue the number of BCRs and the number of good BCR periods received by LATOME are highlighted. In light blue, the number of L1As, the total number of TTypes and the number of TType with value $0x11$ are highlighted. A good agreement between the counters is observed in Figure 6.31 and the values are summarized in Table 6.8.

TTCvi script runs

As with the TDAQ software it was not possible to have a rate of events higher than 6.5 kHz, I run some test directly setting the TTCvi module. The scripts used are described in Appendix C.

With these scripts I was able to set the event rate at 100 kHz. I took a first run of about 15 minutes to verify that the counters were still matching with the higher rate.

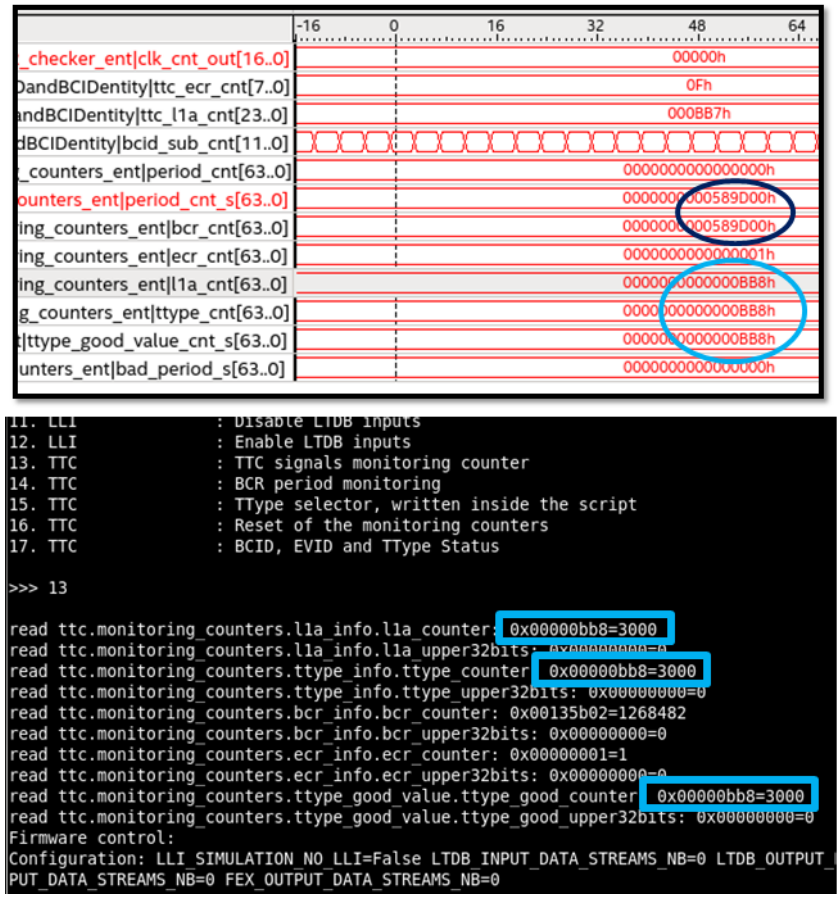

Figure 6.29: Calibration run taken with TDAQ partition: 3000 events at 40 Hz delivered. SignalTap tool (top) and firmware registers (bottom) pictures show the correct behaviour of the TTC firmware block: the number of BCRs received and BCR with good periodicity are matching (dark blue) as well as the number of L1A, TType and TType with expected value (light blue).

Signal	Hex value	Dec value
# L1A	BB8	3000
# TType	BB8	3000
# 0x11 TType	BB8	3000
# BCR	589D00	5807360
# good BCR periods	589D00	5807360

Table 6.6: Summary table: 3000 events collected with TDAQ partition at 40 Hz rate.

The results are shown in Figure 6.32, with the same scheme color used before. The values between the counters were again matching, as summarized in Table 6.9.

As a second test I let the system run for about two hours. The available scripts give also the possibility to check the number of events sent from the module, but unfortunately they give only the lowest 3 bytes. This is anyway enough to establish a compar-

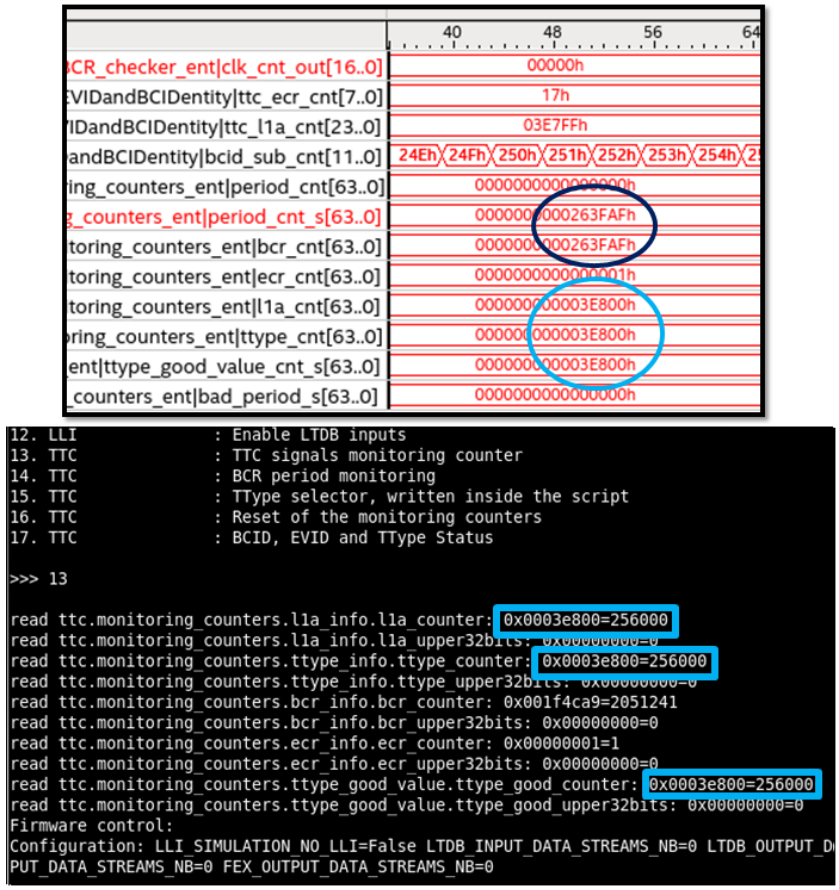

Figure 6.30: Calibration run taken with TDAQ partition: 256000 events at 2.5 kHz delivered. SignalTap tool (top) and firmware registers (bottom) pictures show the correct behaviour of the TTC firmware block: the number of BCRs received and BCR with good periodicity are matching (dark blue) as well as the number of L1A, TType and TType with expected value (light blue).

Signal	Hex value	Dec value
# L1A	3E800	256000
# TType	3E800	256000
# 0x11 TType	3E800	256000
# BCR	263FAF	2506671
# good BCR periods	263FAF	2506671

Table 6.7: Summary table: 256000 events collected with TDAQ partition at 2.5 kHz rate.

ison, as shown in Figure 6.33. The correct result of these test is summarized in Table 6.10.

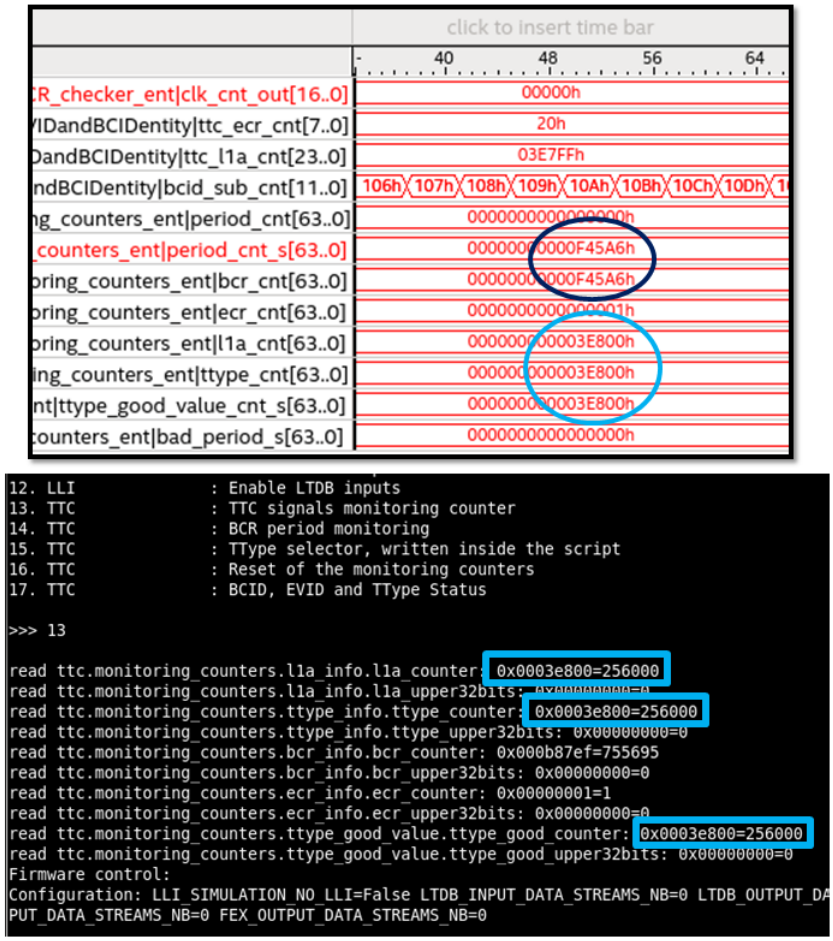

Figure 6.31: Calibration run taken with TDAQ partition: 256000 events at 6.5 kHz delivered. SignalTap tool (top) and firmware registers (bottom) pictures show the correct behaviour of the TTC firmware block: the number of BCRs received and BCR with good periodicity are matching (dark blue) as well as the number of L1A, TType and TType with expected value (light blue).

Signal	Hex value	Dec value
# L1A	3E800	256000
# TType	3E800	256000
# 0x11 TType	3E800	256000
# BCR	F45A6	1000870
# good BCR periods	F45A6	1000870

Table 6.8: Summary table: 256000 events collected with TDAQ partition at 6.5 kHz rate.

ECR check with TDAQ software

Calibration runs do not contain ECRs. To complete the decoding tests, the TDAQ software was set to produce also ECRs.

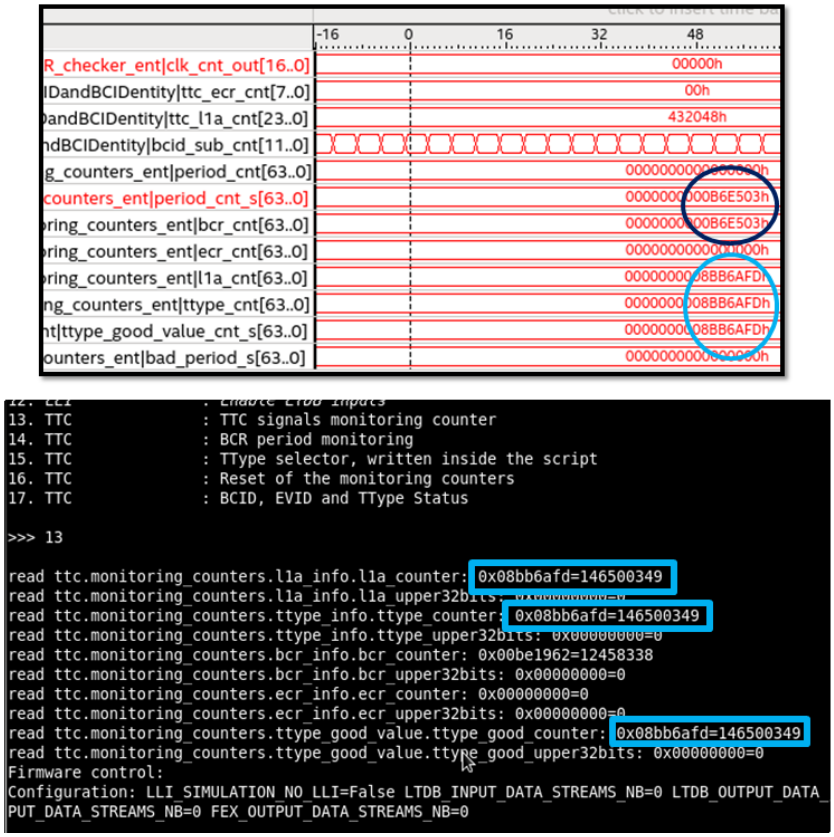

Figure 6.32: Events collected at a rate of 100 kHz. SignalTap tool (top) and firmware registers (bottom) pictures show the correct behaviour of the TTC firmware block: the number of BCRs received and BCR with good periodicity are matching (dark blue) as well as the number of L1A, TType and TType with expected value (light blue).

Signal	Hex value	Dec value
# L1A	8BB6AFD	146500349
# TType	8BB6AFD	146500349
# 0x11 TType	8BB6AFD	146500349
# BCR	B6E503	11986179
# good BCR periods	B6E503	11986179

Table 6.9: Summary table: events collected for 15 minutes at 100 kHz rate.

A first test of one hour, setting the software to have one ECR every 5 ms, was done. Figure 6.34 shows a mismatch: but, as highlighted in red, it can be noticed that the lowest part of the hexadecimal values matches. It means that the partition counter reached the overflow, as only 16 bit are available.

A second test of 25 minutes, with the same ECR settings, was done. The results from the TDAQ software and from the script are shown in Figure 6.35. The lowest part of the hexadecimal values, highlighted in red, matches and the decimal values, highlighted in

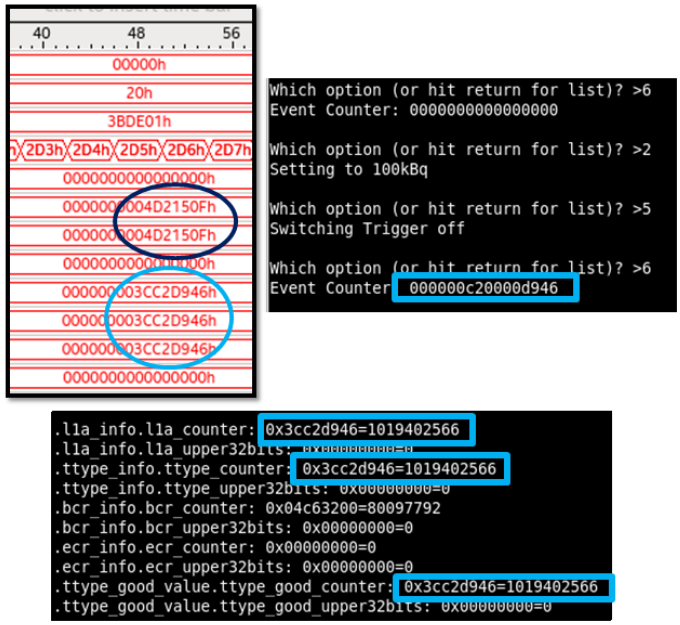

Figure 6.33: Two hours test at 100 kHz. SignalTap tool (top-left), TTCvi script (top-right) and firmware registers (bottom) pictures show the correct behaviour of the TTC firmware block: the number of BCRs received and BCR with good periodicity are matching (dark blue) as well as the number of L1A, TType and TType with expected value (light blue).

Signal	Hex value	Dec value
# L1A	3CC2D946	1019402566
# TType	3CC2D946	1019402566
# 0x11 TType	3CC2D946	1019402566
# L1A from TTCvi	C2D946	-
# BCR	4D2150F	80876815
# good BCR periods	4D2150F	80876815

Table 6.10: Summary table: events collected for two hours at 100 kHz rate.

blue, are the same. The results are summarized in Table 6.11.

Milestone 4 firmware validation

As a final confirmation of the system and code stability, the complete LATOME firmware was compiled, including the TTC block. The plan was to let the system run for 12 hours and verify the counters behaviour, making use of the TDAQ software to handle the TTC. The TDAQ software was set in order to send both L1A and ECR, with higher rate respect to the calibration mode.

During the first 12 hours run, the software handling the system crashed and it was impossible to understand if the mismatch shown in the counters was due to a firmware issue or to the instability of the system.

During the second attempt, FELIX had a misconfiguration: the clock switched from

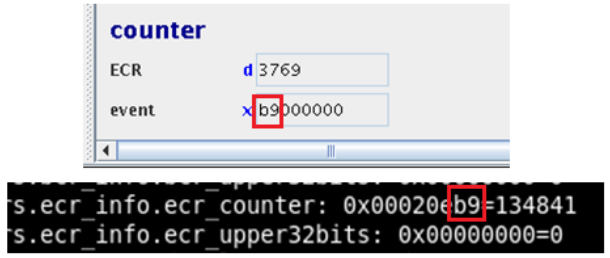

Figure 6.34: ECRs collected setting the partition rate at 50 ms and running for about 60 minutes. The top figure is a screenshot of the TDAQ panel and the bottom figure is a screenshot from the scripts. As highlighted in red, the lowest part of the hexadecimal values matches.

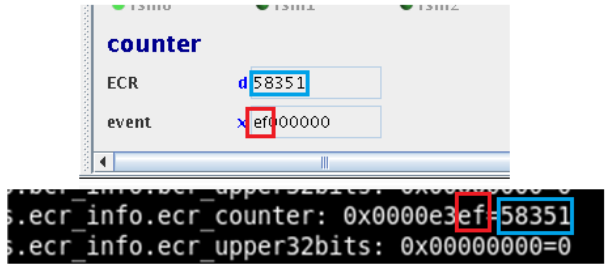

Figure 6.35: ECRs collected setting the partition rate at 50 ms and running for about 25 minutes. The top figure is a screenshot of the TDAQ panel and the bottom figure is a screenshot from the scripts. As highlighted in red, the lowest part of the hexadecimal values matches. Also the decimal values, highlighted in blue, are the same.

the TTC clock, to local clock. As a result, the TTC signal was corrupted, as shown in Figure 6.36. The group of counters monitoring the BCR periodicity shows that some bad periods (different from the expected `DEB` value) were recorded. The bad periodicity can explain also the mismatch between the L1A and the TType counters.

The third night run also had some troubles. In the morning the FPGA was not pinging anymore, like if no firmware was loaded. A quick further test (roughly 6 hours) showed again some instabilities during the data taking. Figure 6.37 shows a bad BCR periodicity, causing the mismatch in the other counters.

In the final test, I let the system run for about 60 hours, making use of the TTCvi module (without using the TDAQ software), generating 100 kHz rate of L1A. This attempt was successful, as shown in Figure 6.38. The system was very stable, all the BCR signals were received with the good periodicity. The L1A and TType counters were matching.

Test time	Signal	Hex value	Dec value
$\sim$ 60 minutes	# ECR (TDAQ partition)	03B9	3769
	# ECR recorded	203B9	134841
$\sim$ 25 minutes	# ECR (TDAQ partition)	E3EF	58351
	# ECR recorded	E3EF	58351

Table 6.11: ECR recorded by the system, summary of the values for the two runs.

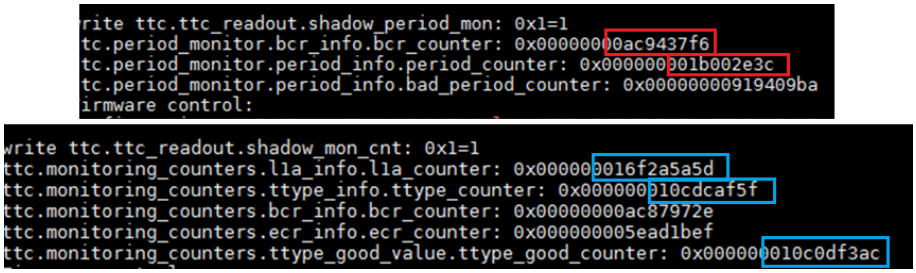

Figure 6.36: Monitoring counters showing a mismatch due to a system misconfiguration at the end of the run: in the top figure is shown the mismatch (in red) of the BCR counters, while in the bottom figure the mismatch (in blue) of the L1A and TType counters.

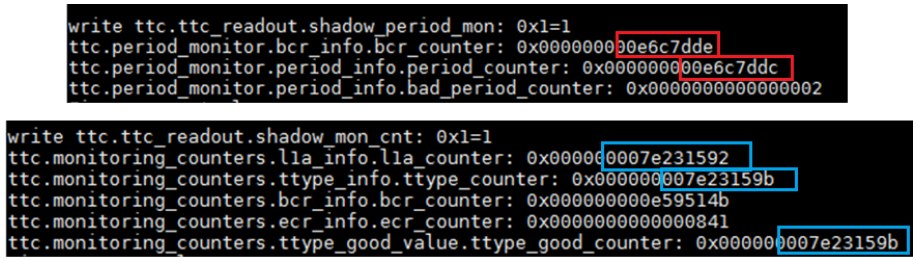

Figure 6.37: Monitoring counters showing a mismatch due to BCR periods detected by the system: in the top figure is shown the mismatch (red) of the BCR counters, while in the bottom figure the mismatch (blue) of the L1A and TType counters.

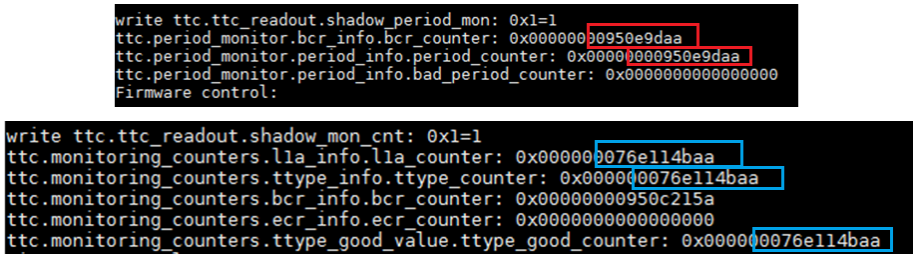

Figure 6.38: Monitoring counters showing correct results after week end test: in the top figure is shown the match (red) of the BCR counters, while in the bottom figure the match (blue) of the L1A and TType counters.

The problems encountered with the TDAQ software are still object of investigation.

6.5 Results from IStage block

As already mentioned, the TTC block has to provide BCR and BCID information to the IStage. This information allows the incoming data to be aligned properly.

Figure 6.39 shows the data coming along different fibers. The data come from a pattern generator based on a RAM memory which contains data for 256 BCID only. For the other part of BCID (256 to 3563), null data are sent.

As the RAM content is limited to 256 BCIDs, it is possible to observe that data stop

for 0x0FF value. The pattern generator sends then its own computed null data.

In Figure 6.39, no delay is set in the pattern generator. Between the BCID in the IStage and the BCID in the TTC block, a real delay of 6 clock cycles (counting on the 40 MHz clock) is experienced. This is the reason why the last 0x0FF data are arriving on the TTC BCID with value 0x106.

Figure 6.40 shows a delay, set in the pattern generator where the clock used is the 320 MHz. In this case, a delay of 100 is set, and it results in a delay of 12 on the 40 MHz clock. In this case the delay has been set per each fiber: the last data 0x0FF are appearing at the BCID 0x112, corresponding to the expected value.

In both figures, it is possible to see a good alignment of the fibers, thanks to the correct reception of BCR and BCID information.

6.6 Final considerations

The TTC system used for the back-end in EMF is complex. Different boards are involved, decoding and encoding several times the information. This complexity makes the debug more difficult.

As a first step, I had to understand which TTC signals where available from FELIX, along the GBT links. As also FELIX was beginning its development, retrieving the correct information took some time. Then, in collaboration with the LATOME and Carrier group we decided a custom protocol to be used along the LVDS links. This required some iterations, also because at the beginning it was not clear which TTC signals should have been used inside the LATOME firmware.

A major difficulty was encountered when the signals had to be sent along the LVDS lines. Along these physical connection, at the very beginning, the signals where not arriving synchronously. As first step, the timing constraints were set on both Carrier and LATOME firmware, to make sure that the distance between I/O pins and the first available registers in the the firmware is fixed.

This partially solved the problem. During the first tests, between the TTC block and the IStage, a not fixed value of the BCR period was observed, as well as some mismatches in the register counters. In the end, we realised that the two FELIX PCs were configured differently and were using two different clock configurations. After setting the proper TTC clock on FELIX in EMF, the signals became stable and the decoding produced correct results.

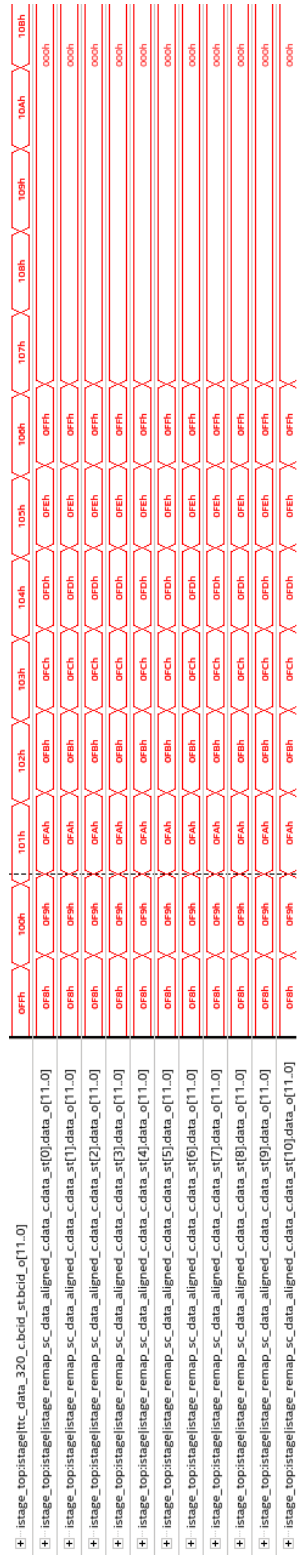

Figure 6.39: Data alignment on different fibers in the IStage block.

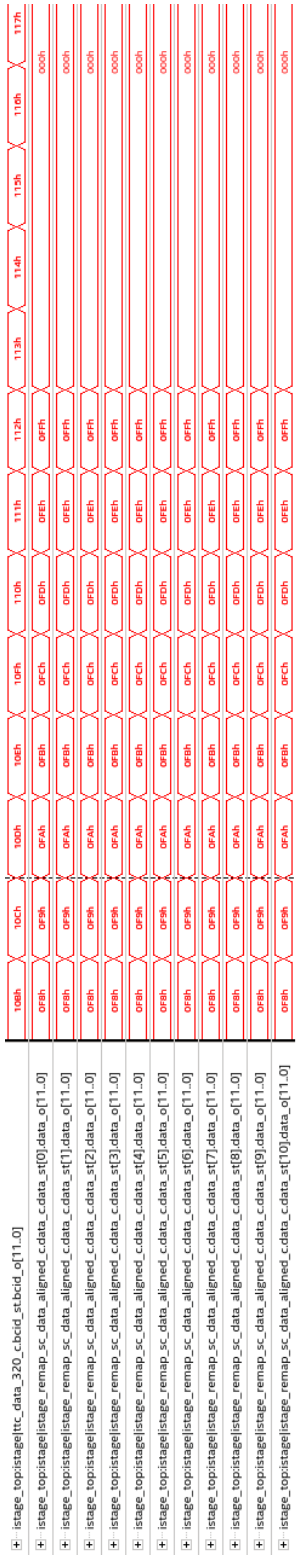

Figure 6.40: Data alignment on different fibers in the IStage block setting an internal delay.

CHAPTER 7

Conclusions

During my Ph.D. studies I contributed to the development of the FPGA readout firmware for the LAr back-end Phase-I upgrade electronics.

The installation of the LAr demonstrator system (LTDB and ABBA) in the ATLAS detector, allowed me to gain experience in the perspective of the Phase-I upgrade. I made significant contributions to the LAr Phase-I back-end firmware to improve the system stability and the data quality.

I was one of the few people involved in keeping the demonstrator operational and I contributed to the system data taking. Thanks to all the improvements, it was possible to collect proton-proton data and compare them with the ATLAS main readout.

While the demonstrator system was smoothly reading out collision data, a system test for the upgraded back-end electronics was built in the EMF facility. The idea of this system is to start combining the Phase-I back-end boards, Carriers and LATOMEs, and verify their behaviour.

I was in charge of the LATOME firmware block decoding the TTC data, used to synchronize the entire ATLAS detector and sub-detectors. The LATOME board is one of the core components of the LAr back-end trigger readout electronics and has the role of reconstructing the energy of the digital pulses coming from the calorimeter Super Cells.

To establish a benchmark with respect to the future LAr Phase-I setup, an upgraded demonstrator system will be installed in the ATLAS detector. All the problems encountered and the solutions adopted in the first demonstrator will be taken into consideration for the installation of the new and complete demonstrator in January 2018. The system will use prototype versions of the LTDB and LDPS boards.

To verify the installation, calibration data will be taken both via the standard readout and with the LTDB within the LAr calibration infrastructure. The standard readout will be checked to make sure that the new system is not interfering with the main data taking. After the 2017 End-Of-Year shutdown, LHC will restart the proton-proton collision operations and the new upgraded demonstrator will collect data again in parallel with the ATLAS detector.

APPENDIX A

FPGAs and VHDL

A.1 Field Programmable Gate Array

Field Programmable Gate Arrays (FPGAs) were first introduced in the 80's. Since that time FPGA technology grew and became a popular implementation media for digital circuits.

FPGAs are silicon devices that can be electrically programmed to become almost any kind of digital circuit or system. They contain:

- Programmable logic blocks able to implement logic functions
- Programmable routing to connect the logic functions
- I/O blocks connected to logic blocks to make off-chip connections

An example of FPGA architecture is shown in Figure A.1. The Configurable Logic Blocks (CLBs) are arranged in a two dimensional matrix and interconnected by programmable routing. I/O blocks are arranged at the matrix border and they are also connected to the programmable routing.

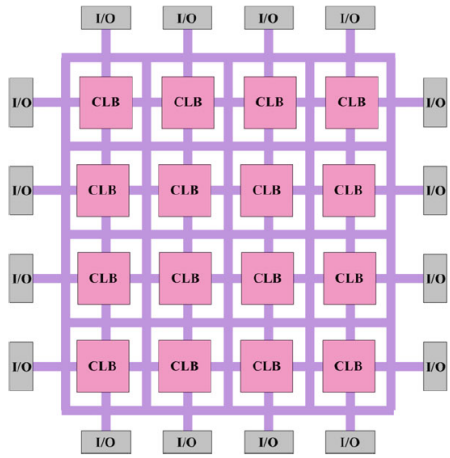

Figure A.1: FPGA architecture overview [121].

The reconfigurability of FPGAs is based on their ability to implement a new function after a previous implementation was completed. Programmability is based on a programming technology which can cause a change in behaviour of the chip.

Different technologies have been developed in the last years and the most known are static RAM, flash and anti-fuse FPGAs. In this section I will concentrate only on the static memory technology. Most vendors use SRAM-based FPGAs. Static memory cells are the basic cells where data are stored. As SRAM is a volatile memory, it can't keep data without power source, for this reason FPGAs must be reconfigured upon start.

Commercial vendors use LookUp Table (LUT) based CLBs which provide a good compromise between too fine and too coarse grained logic blocks. A CLB can be made of a single Basic Logic Element (BLE) or a cluster of BLE. A simple BLE consists of a LUT, and a Flip-Flop, as shown Figure A.2.

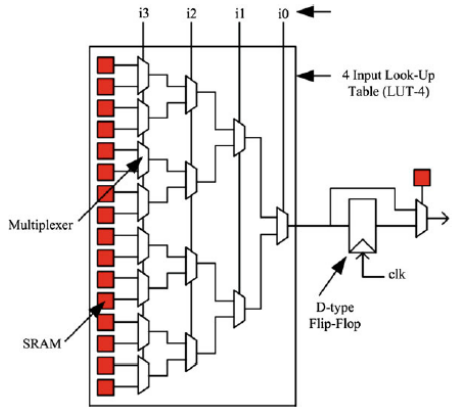

Figure A.2: FPGA basic logic element [121].

The BLE shown has four input LUT (LUT-4) and a D-type Flip-Flop. The LUT-4 uses 16 SRAM bits to implement any four inputs boolean function. The output of LUT-4 is connected to an optional Flip-Flop. A multiplexor selects the BLE output to be either the output of a Flip-Flop or the LUT-4.

The computing functionality depends on the programmable logic blocks and how these blocks are connected to each other through programmable routing network. The routing has to be flexible so that it can accommodate a wide variety of circuits but it's also a critical point as it plays a very important role in the overall efficiency of the architecture. From the point of view of the global arrangement of routing resources, FPGA architectures can be categorized as either hierarchical or island-style.

The most commonly used architecture among commercial FPGAs is the island-style, shown in Figure A.3.

In this configuration logic blocks look like islands in a sea of routing interconnections. The routing network has wiring segments and programmable switches that are organized in horizontal and vertical routing channels. The Figure shows how the horizontal and vertical routing tracks are interconnected through Switch Boxes (SB) and how logic blocks are connected to the routing through Connection Boxes (CB). The flexibility of a connection box (Fc) is the number of routing lines of adjacent channel which are connected to the pin of a block. The flexibility of switch box (Fs) is the total number of tracks with which every track entering in the switch box connects to. A switch and connection box examples are shown in Figure A.4.

In this figure, the switch box has Fs$=3$ as each track incident on it is connected to 3 tracks of adjacent routing channels. Similarly, the connection box has Fc(in)$=0.5$ as each

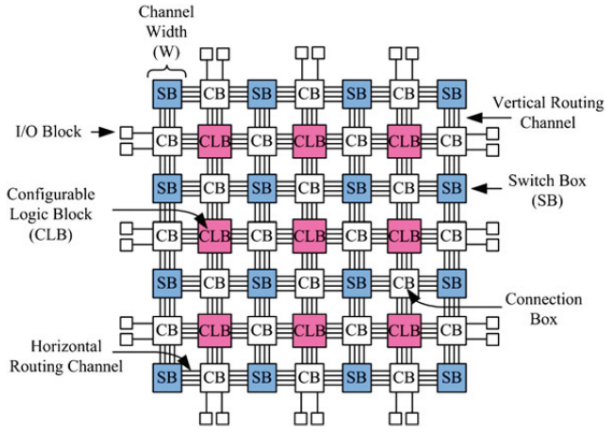

Figure A.3: Island-style FPGA architecture [121].

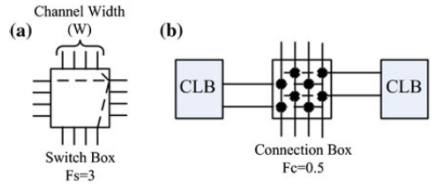

Figure A.4: Switch and connection box examples [121].

input of the logic block is connected to 50% of the tracks of adjacent routing channel [121].

To handle all these complex interconnection systems and translate what the user wants in hardware, many software tools have been developed. The software flow takes a design description in a Hardware Description Language (HDL) and converts it to a stream of bits that is eventually programmed on the FPGA. How HDL works is described in the next section.

A.2 Very high speed integrated circuits Hardware Description Language (VHDL)

Hardware Description Languages (HDLs) are used to give the possibility to describe a digital system at different levels of abstraction. Figure A.5 shows the Y-chart used to described the different type of representations as well as the different levels per each type.

The three possible system representation are the behavioural, the structural and the physical view. The behavioural representation describes the functionality of the system and it focuses on the relations between the input and the output signals. This type of description is never unique. The structural view is done specifying the components to be used and how they should be connected. It's more like a schematic or diagram description of the system. Then the physical representation specifies the components and their location on a board, for example a printed circuit board layout.

For each representation different, levels of abstraction can be identified. The purpose

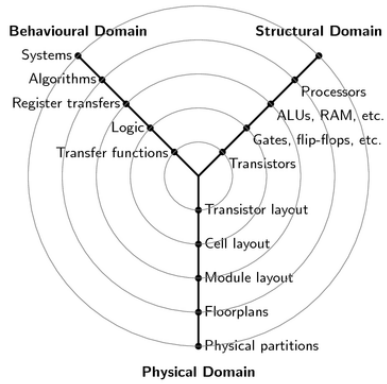

Figure 1: Gajski-Kuhn Y-chart

Figure A.5: Domains and levels of abstraction for a digital system [122].

of the abstraction is to reduce the amount of data so that only the important information is presented. On the other side, although it is more complex, the low-level of abstraction is more accurate and closer to the real circuit.

The main characteristics of a digital circuit created with HDL are entities, connectivity, concurrency and timing. Entity is a basic and self-consistent building block. Connectivity allows the connection and interaction between the entities. Concurrency describes the fact that many entities can be active at the same time and perform parallel operations. Then timing, which is related to concurrency as it handles schedule and order of the operations.

VHDL and Verilog are the two most widely used HDLs. In this section I will concentrate only on VHDL description.

VHDL can be used for pure structural or pure behavioural descriptions. Some examples will be shown in the following. Before entering in the details of the two view, it is important to know how the code is in general organised. Here is an example of a simple program:

```
entity AND_ent is
port(
   x: in  std_logic;
   y: in  std_logic;
   F: out std_logic
);
end AND_ent;

architecture and_arch of AND_ent is
begin
   F <= x and y;
end and_arch;
```

Two major parts can be seen in the previous lines: the *entity declaration* and the *architecture body*. The entity declaration specifies the input and output signals, in this case two inputs (x and y) and one output (F). The architecture describes the operations done by the circuit, in this case a simple *and* operation between the two input signal and connected to the output.

A.2.1 Structural view

The structural view is made of smaller parts and is essentially a schematic representing a circuit diagram. In the VHDL the structural view is done making use of components, declared and instantiated in the architecture body.

```
entity top_entity is
port(
   input1:        in  std_logic;
   input2:        in  std_logic;
   input3:        in  std_logic;
   signal_out1:   out std_logic;
   signal_out2:   out std_logic
);
end top_entity;

architecture struct of top_entity is
begin
   ent1_inst : entity ent_lib.ent1
      port map(
         signal_in1  => input1,
         signal_in2  => input2,
         signal_out1 => signal_out1,
      );

   ent2_inst : entity ent_lib.ent2
      port map(
         signal_in1  => input1,
         signal_in2  => signal_out1,
         signal_out2 => signal_out2,
      );
end struct;
```

Here is an example of a structural view. The entity contains two components, instantiated inside the architectural body. The two components belong to the library *ent_lib* and are connected to the input and output signals of the main entity. As shown in the second component, the output of one entity can become the input of another.

The structural description helps to have a hierarchical design and to divide complex systems into smaller subsystem. Furthermore it allows to use already made entities and IP cores, treating them as black boxes [122] [123].

A.2.2 Behavioural view

Large designs can be very complex. For this reason it is easier to focus on the system operation instead of the construction. VHDL provides language constructs that resemble sequential semantics, usually encapsulated in *processes*. This type of code is usually referred to the behavioural description.

```
entity top_entity is
port(
   signal_en:  in  std_logic;
   signal1:    in  std_logic;
   signal2:    in  std_logic;
   signal_out: out std_logic
);
end top_entity;

architecture behav of top_entity is
begin
   example_proc : process(signal_en)
```

```
   begin
      if signal_en = '1' then
         signal_out <= signal1 and signal2;
      else
         signal_out <= signal1 or signal2;
      end if;
   end process;
end behav;
```

This small program shows a behavioural view in the architecture and in particular a process. Inside the process the commands are sequential. Between the parenthesis is shown the sensitivity list, for which the process is triggered and used, in this case an enabling signals. If the enable is HIGH then an *and* operation is applied between the two input signals, else an *or* operation is used. The output of the top entity is driven by the process [122] [123].

A.3 Testbench

Simulating a VHDL program is an important step to study the functionality of a circuit and to verify the correctness of a design. It is like doing a real experiment in which the inputs of a circuit are connected to some stimuli.

A testbench consists of an entity declaration and an architecture body. Since the testbench is self-contained, no port is specified in the entity declaration. Concurrent statements can be placed in the architecture body, like component instantiations and processes.

```
entity entity_tb is
end entity_tb;

architecture tc_arch of entity_tb is
   signal signal_en  : std_logic;
   signal signal1    : std_logic;
   signal signal2    : std_logic;
   signal signal_out : std_logic;

begin
   topentity_inst : entity ent_lib.top_entity
      port map(
         signal_en      => signal_en,
         signal1        => signal1,
         signal2        => signal2,
         signal_out     => signal_out
      );

   signal_en <= '1';

   stimul_proc1: process
   begin
      signal1 <= '0';
      wait for 200 ns;
      signal1 <= '1';
      wait for 200 ns;
      signal1 <= '0';
      wait;
   end process;

   stimul_proc2: process
   begin
      signal2 <= '0';
```

```
      wait for 100 ns;
      signal1 <= '1';
      wait for 100 ns;
      signal2 <= '0';
      wait;
   end process;
end tc_arch;
```

In the example listed above a testbench of the behavioural entity is described. The top entity is instantiated as a component and runs in parallel with two process. The I/O pins of the entity are connected to internal signals, defined before the architecture begin. The stimuli are generated by two process and one signal assignment. The processes take the input signals and assign them different values depending on the simulation time, thanks to the command *wait* which holds the value [122] [123].

APPENDIX B

Git

B.1 How it works

Git is a version control system designed to handle and coordinate commercial and open source projects among multiple people. It was developed in 2005 by Linus Torvalds, the creator of the Linux operating system kernel. In particular, git is a distributed version control system, as there is not only one single place for the full version history of the software, but each developer's copy of the code is a repository that can contain the full history of the project.

The version control feature allows to record changes done in a file or set of files over time, so that it is possible to recall them later. For example git allows you to revert the changes done in file, revert an entire project, know who has done a modification and also compare changes done in different moments.

A more clear idea of how git is organised is shown in Figure B.1:

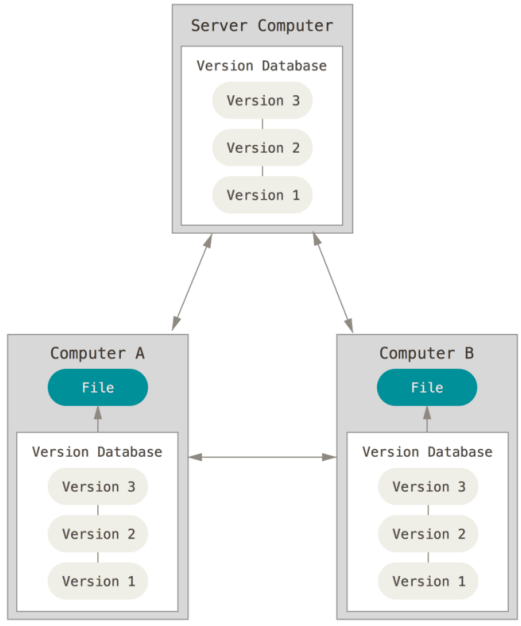

Figure B.1: Git as a distributed version control system [124].

The server computer is storing the repository as well as the entire history of the files. But at the same time, all the local copies are not simply a check out the latest snapshot,

but they fully mirror the repository. In this way, if a server dies, any local repository can be copied back to the server to restore it.

Git files can have three states: committed, modified, and staged. Committed means that the data is safely stored in the local database. Modified means that there are some changes in the file, but it is not yet committed to the database. Staged means that a file has been marked as modified in its current version, such that it is possible to go into your next commit snapshot. A schema of these definitions and of the work-flow in Git is shown on the left side of Figure B.2. When a file a modified in the working tree, in order

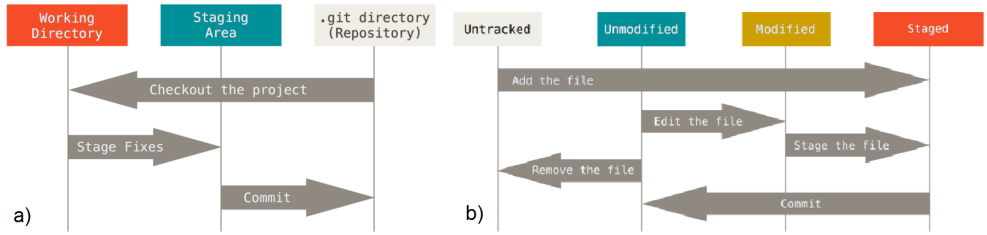

Figure B.2: a) Three possible states for a Git file. b) More general view of lifecycle of a file involved in a Git repository [124].

to save the changes in git, the file has to be staged. A snapshot of the file is taken in this way. Then, with a commit, the file is taken from the staging area and put in the local Git directory. The previous description is valid only for files already in Git. But there can also be new files in the folders, for example added after a clone of the repository on a local machine. This type of files is called "untracked", as they are not recognised by the Git repository.

In Git there is the possibility to work on different branches. A branch is a path where the work-flow is developed and the default branch name is "master". New branches can be created from master (or from another branch as well) such that a parallel development can continue respect to the master, like shown in Figure B.3.

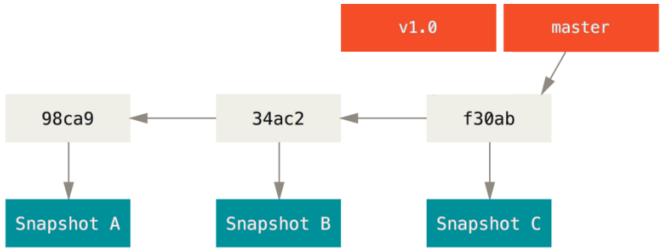

Figure B.3: Example of branches development and commits (snapshots) in git: in this figure a new branch is created from the master [124].

As said at the beginning, the remote repository can be stored on server. In this case, in order to have the changes not only in the local repository but on the remote as well, it's necessary to do a push, as shown in Figure B.4. Different people can work on the same repository, developing their branch a pushing to the remote. To pick up someone else changes from the remote to a local machine, a pull is necessary [124].

There are a lot of different ways to use Git. There are command line tools but also many graphical user interfaces. From the command line all the Git commands can be

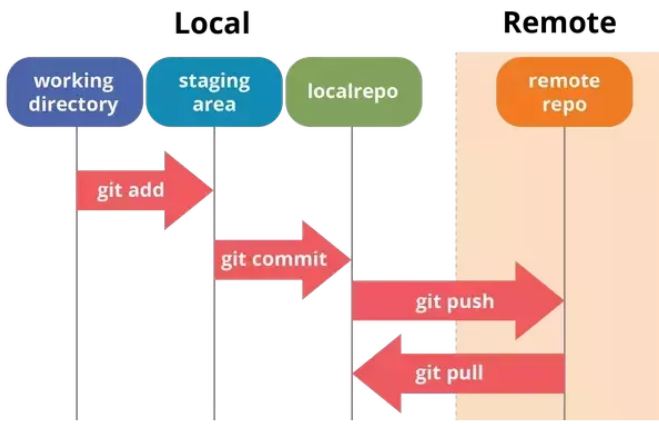

Figure B.4: Steps to move the changes from the local to the remote repository [124].

used, while most of the GUIs only implement a subset of Git functionalities.

Some of the most used commands are summarized here:

- `git status`: to see modified (and new/deleted) files and to show also untracked files
- `git add myfile`: to add a file for the next commit
- `git commit -m "message"`: to commit the file directly with a message
- `git diff`: to show the changes of files before *git add*
- `git checkout -- myfile`: to revert only *myfile* since the last *git add myfile*
- `git branch`: to show available local branches, the current branch is marked with an asterix
- `git checkout -b branchname`: to create branch *branchname* (from the current branch)
- `git push`: to push the commit to the remote repository on the respective branch
- `git pull`: to pull the new commits from the remote from the respective branch

B.2 LDPS firmware repository

The source code for the ATLAS LAr LDPS firmware is stored on the GitLab repository at CERN. The LDPS firmware repository can be accessed directly through the CERN Web interface: `https://gitlab.cern.ch/atlas-lar-ldpb-firmware`

As shown in Figure B.5 and Figure B.7, the folders *altera*, *common* and *env* are shared between the two projects: ABBA and LATOME. The *altera* folder contains the Altera libraries for both projects. In particular it has two subdirectories, named with the Quartus II version used to compile the firmware: for ABBA the version is 13.1 while for LATOME it is 17.0. The *common* folder contains all the common file used in both ABBA and LATOME projects, for example some components like FIFOs, RAMs, delay chains, but also the oscillator code and some Altera and Xilinx tools. The *env* folder contains the HDL Make based environment for building VHDL/Verilog projects for Altera FPGAs.

B.2.1 ABBA project

The ABBA firmware can be accessed through the CERN Web interface: `https://gitlab.cern.ch/atlas-lar-ldpb-firmware/ABBA`
The first level of directories in the repository is shown in Figure B.5. In this project the

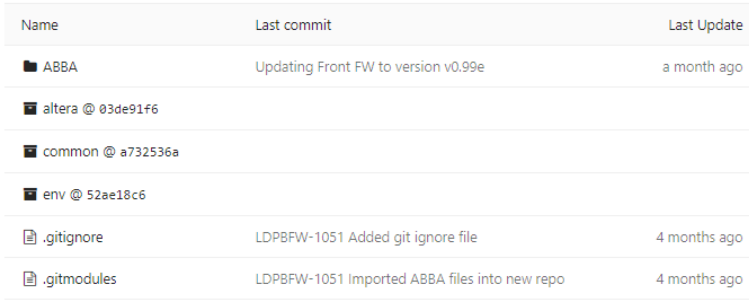

Figure B.5: First level of folder in the ABBA repository.

folder called $ABBA$ is the main folder, containing the firmware of both Front and Back FPGA. More in detail and as shown if Figure B.6 the $ABBA$ folder contains many sub-folders. In $back\_fpga\_src$ and $front\_fpga\_src$ respectively the source file for Front and

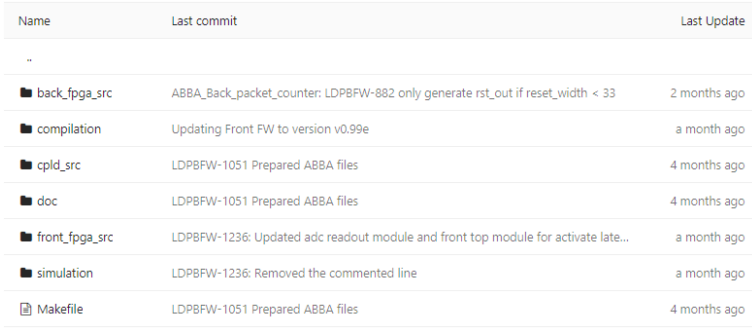

Figure B.6: Folder organisation for the ABBA folder.

Back FPGAs are contained. In $compilation$ there are the two sub-folders for the compilation of Back and Front FPGA while in $simulation$ there are the sub-folders for the simulation of the different entities. In folder $cpld\_src$ the are the source files to compile the CPLD firmware that allows the communication with a flash memory where a copy of the firmware is stored.

B.2.2 LATOME project

The LATOME firmware can be accessed through the CERN Web interface: `https://gitlab.cern.ch/atlas-lar-ldpb-firmware/LATOME`
The first level of directories in the repository is shown in Figure B.7. The LATOME repository is more complex than the ABBA repository. The LATOME repository contains sub-repositories, as shown in Figure B.8, where each folder is a project with a master branch and development branches. Each folder corresponds also to a module inside the firmware.

Name	Last commit	Last Update
Arria10-Devkit @ 8b5e01fc		
LATOME	Merge branch 'system_test_m3' of ssh://gitlab.cern.ch:7999/atlas-lar-ldp...	a month ago
altera @ 3120dd69		
common @ dd9b4fbe		
env @ 144c8bf9		
testbench @ 43b20a70		
.gitignore	LDPBFW-1051 Added git ignore file	4 months ago
.gitmodules	LDPBFW-1182 Added Arria-10 devkit repo	2 months ago
Makefile	Added Makefile	4 months ago
initialize_repository.sh	Update init script	a month ago
update_repository.sh	LDPBFW-1263 Added update script	a month ago

Figure B.7: First level of folder in the LATOME repository.

Name	Last commit	Last Update
..		
fex_checker @ 2a879a6e		
fpga @ 4e821763		
ipctrl @ 5bbf049d		
istage @ 0ce2b494		
latome_testbench	LDPBFW-1173 Use UVVM instead of Bitvis	a month ago
lli @ f2ed18cb		
mon @ 9ff320dd		
osum @ 8a2a20cd		
pattern_generator @ 0647b35e		
remap @ cc21f791		
test @ 406ffed6		
tools	LDPBFW-1220 Added library for 8b10b	a month ago
ttc @ 145c38ae		
user @ 761f2d47		

Figure B.8: Sub-project organisation for the LATOME folder.

APPENDIX C

TTC System

Synchronization is a critical aspect in the LHC experiments. Timing, Trigger and Control (TTC) signals must be distributed to all the subsystems and to their electronics, located all along the ring. Synchronization has to be achieved and monitored for proper operation of the system. More details about TTC system can be found in [125].

In each LHC experiment the TTC system role is to control the detector synchronization and deliver messages to the front-end electronics, in phase with the LHC clock and with the orbit. Important signals coming with the TTC are the 40.08 MHz bunch-crossing clock, the Level-1 Accept (L1A) trigger decision, the bunch crossing and event resets, as well as the broadcast commands [126].

Figure C.1 shows a schematic view of how the TTC system is organised. The electrical signals (synchronisation signals, L1A, LHC Orbit, LHC clock, etc.) are sent to the TTC crate from the TTCvi (channel A and channel B) and the LHC machine. Modules in the crate perform the encoding and the electrical-to optical conversion. The channel A is entirely dedicated to the L1A trigger, a one bit information delivered every bunch crossing. The channel B is used to broadcast data to all or specific system destinations [127].

C.1 TTC data

The L1A represents the first level trigger event acceptance. This is the 1 bit information sent along the channel A, if the Central Trigger Processor (CTP) forms a L1A, according to the Level-1 trigger menu, after it receives information from the calorimeter and the muon Level-1 triggers.

To protect the system from being flooded by L1A, ATLAS makes use of some dead-time rules. Preventive dead-time is introduced by the CTP to stop the front-end buffers from overflowing, and is calculated in two ways: simple and complex.

The simple dead-time is the number of bunch crossings (BC) after each L1A used to avoid overlapping readout windows: for LAr this value is set to 4 in standard physics runs. The complex dead-time uses a leaky bucket model to emulate a front-end buffer. In this model when the bucket is full there is dead-time. It is defined by X (in units of L1A) being the size of the bucket and R (in BC) being the time it takes to leak 1 L1A. For LAr X$= 9$ while R$= 351$ [128].

The data stream coming along the channel B can be of two types: broadcast commands or individually-addressed commands/data.
Broadcast commands are used to deliver messages to all TTC destinations in the system while individually-addressed commands/data are used to transmit user-defined data and commands over the network to specific addresses and sub-addresses. These two types of command have a specific frame format, as shown in Figure C.2:

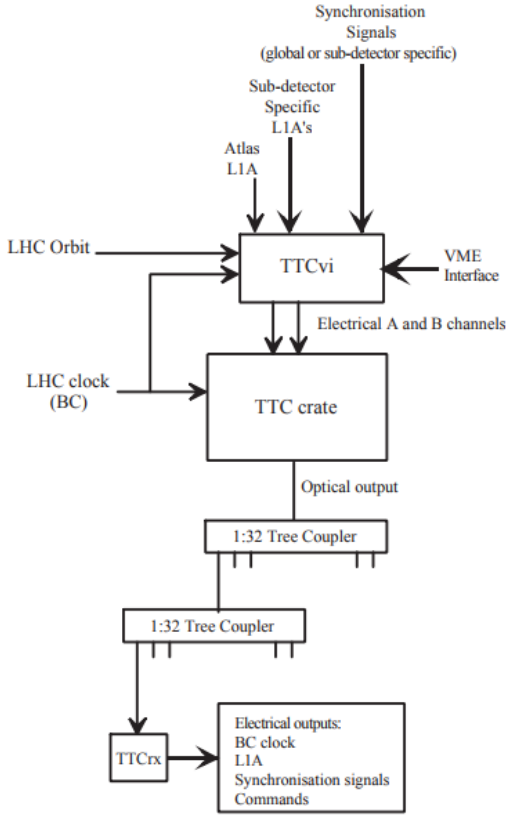

Figure C.1: Schematic view of the TTC system [127].

More in detail, the channel B IDLE state is set to 1. When a sequence of commands is sent, the data transmission changes from 1 to 0. After a first zero it is already possible to distinguish between the short broadcast and the long address commands: if the second bit in the stream is a 0 then the command is a short broadcast, if it is a 1 then the command is a long address.

```
IDLE=111111111111
        Short Broadcast, 16 bits:
        00TTDDDDEBHHHHH1:
                T=test command, 2 bits
                D=Command/Data, 4 bits
                E=Event Counter Reset, 1 bit
                B=Bunch Counter Reset, 1 bit
                H=Hamming Code, 5 bits
        Long Addressed, 42 bits
        01AAAAAAAAAAAAAAE1SSSSSSSSDDDDDDDDHHHHHHH1:
                A= TTCrx address, 14 bits
                E= internal(0)/External(1), 1 bit
                S=SubAddress, 8 bits
                D=Data, 8 bits
                H=Hamming Code, 7 bits
```

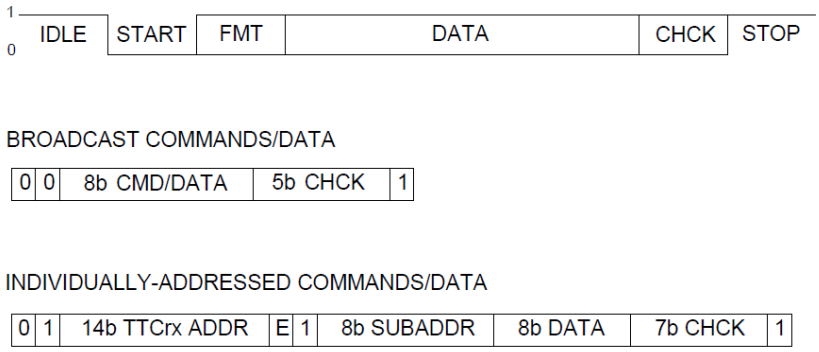

Figure C.2: Frame formats for the channel B commands [125].

Along the short broadcast command two important values are sent: the Bunch Counter Reset (BCR) and the Event Counter Reset (ECR).

The BCR is used to reset the bunch crossing counter which is increased every clock cycle on the 40 MHz clock. This is a twelve bit counter also called BCID. A BCR command is sent roughly every 89 μs, that corresponds to the time that a bunch needs to do an entire round along LHC. During this time the BCID counter reach its maximum value, 3564 counts.

The ECR is used to increase the event counter. The periodicity of this reset is decided by the experiment, for ATLAS it is set to 5 seconds. The event reset counter combined with the L1A counter gives the EVID, a 32 bit vector:

```
Event_reset_counter : std_logic_vector(31 downto 24);
L1A_counter: std_logic_vector(23 downto 0);
```

Every time that a ECR is received the upper counter is increased by 1 and the lower part is reset to -1. Every time that a L1A is received the lower part is increased by 1.

BCID and EVID values are used as a label for the data accepted by the trigger.

Along the long address command another important information is sent: the Trigger Type (TType). Each L1A information is followed, with a variable latency, by an 8 bit TType word. This word is generated inside the LVL1 Central Trigger Processor (CTP) and distributed from the CTP to the TTCvi modules for each of the TTC zones of the experiment via the corresponding LTP modules.

The presence of a Trigger Type along the long address command is announced by a sub-address (8 bit information) set to 0.

Sub-Trigger	physics	ALFA	FTK	LAr demonstrator	Muons	Calorimeter	ZeroBias	Random
bit	7	6	5	4	3	2	1	0

Table C.1: Trigger type 8 bit word: each bit represent the sub-detector which fired the trigger or the data type.

As shown in Table C.1, each bit has a specific role. Bits 3 to 6 are used to indicate which sub-detector or subsystem fired the trigger. Bit 7 represent the physics trigger-mode when set to 1, while it describes the calibration mode when set to 0. In calibration

mode, bits 0 to 2 can be used to distinguish between up to eight different possible types of calibration trigger within each sub-detector.

ABBA makes use of a mask on the TType: 0x90 = 10010000. This mask allows to monitor only physics events accepted by the demonstrator. A second condition checks which bit has been set to 1: the condition is again 0x90 = 10010000, as the demonstrator wants to record at least the physics happening in the demonstrator region. This condition means that the possible TTypes approved inside the firmware are 0x90, 0x98 or 0x94, where the muons or the calorimeter can be fired.

For the calibration runs the mask is set to 0x01 = 00000001, while the second condition is 0x11 = 00010001. These two conditions together allow ABBA to record random triggers only coming from the demonstrator region.

C.2 How to communicate with TTCvi module

In the LAr Electro Magnetic Facility (EMF) a TTCvi module is used to have the proper TTC signals. It handles the generation of channel A and channel B, using a 40.08 MHz clock signal, for the test setup used in EMF.

In particular some scripts have been developed to manual control the TTCvi to send specific TTC data to FELIX, Carrier and LATOME. The following interface is available:

```
1) Check VME Access
2) Set Trigger Rate 100kBq
3) Set Trigger Rate 5kBq
4) Set Trigger Rate 1Bq
5) Trigger Off
6) Show Event Counter
7) Reset (local) Event Counter
8) Short Command (0x03) on B-Channel
9) Long Command (0x90) on B-Channel
10) Quit
```

The list of available actions is in the following lines.

- check presence of VME board and correct function of VME driver on SBC (1)
- set the average random trigger rate (L1A and TType) (2,3,4)
- switch off the trigger (5)
- display the lower 3 bytes or reset the internal TTCvi event counter (6,7)
- send channel B commands:
  - short command: BCR, ECR (bits 0 and 1), other sequences. Default (0x03) is to send ECR and BCR signal (8)
  - long command: used to set the TType. Default is 0x90 (9)
- quit the tool (10)

A direct communication with the TTCvi module has been very helpful during the debugging of the code. For example, it allowed to select and test different trigger types in the channel B, to verify the reception of ECR and BCR on the proper LVDS lines independently, as well as to make use of the maximum trigger rate (100 kHz).

Bibliography

[1] M. Krause, *CERN: how we found the Higgs boson*. Hackensack, NJ: World Scientific, Nov 2014, german edition with the title : Wo Menschen und Teilchen aufeinanderstossen : Begegnungen am CERN. [Online]. Available: https://cds.cern.ch/record/1748524

[2] T. S. Pettersson and P. Lefèvre, "The Large Hadron Collider: conceptual design," Tech. Rep. CERN-AC-95-05-LHC, Oct 1995. [Online]. Available: https://cds.cern.ch/record/291782

[3] ATLAS Collaboration, "The ATLAS Experiment at the CERN Large Hadron Collider," *JINST*, vol. 3, p. S08003, 2008.

[4] CMS Collaboration, "The CMS Experiment at the CERN LHC," *JINST*, vol. 3, p. S08004, 2008.

[5] LHCb Collaboration, "The LHCb Detector at the LHC," *JINST*, vol. 3, p. S08005, 2008.

[6] ALICE Collaboration, "The ALICE experiment at the CERN LHC," *Journal of Instrumentation*, vol. 3, no. 08, p. S08002, 2008. [Online]. Available: http://stacks.iop.org/1748-0221/3/i=08/a=S08002

[7] *LHC Guide*, Mar 2017. [Online]. Available: https://cds.cern.ch/record/2255762

[8] ATLAS collaboration, "Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC," *Phys. Lett. B*, vol. 716, no. arXiv:1207.7214. CERN-PH-EP-2012-218, pp. 1–29. 39 p, Aug 2012. [Online]. Available: https://cds.cern.ch/record/1471031

[9] CMS collaboration, "Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC," *Phys. Lett. B*, vol. 716, no. CMS-HIG-12-028. CMS-HIG-12-028. CERN-PH-EP-2012-220, pp. 30–61. 59 p, Jul 2012. [Online]. Available: https://cds.cern.ch/record/1471016

[10] L. R. Evans and P. Bryant, "LHC Machine," *J. Instrum.*, vol. 3, p. S08001. 164 p, 2008, this report is an abridged version of the LHC Design Report (CERN-2004-003). [Online]. Available: https://cds.cern.ch/record/1129806

[11] M. E. Peskin and D. V. Schroeder, *An introduction to quantum field theory*. Boulder, CO: Westview, 1995, includes exercises. [Online]. Available: https://cds.cern.ch/record/257493

[12] Kate Kahle, "LHC experiments highlight 2017 results," December 2017. [Online]. Available: https://home.cern/about/updates/2017/12/lhc-experiments-highlight-2017-results

[13] "Evidence for the $H \to b\bar{b}$ decay with the ATLAS detector," CERN, Geneva, Tech. Rep. ATLAS-CONF-2017-041, Jul 2017. [Online]. Available: https://cds.cern.ch/record/2273847

[14] A. M. Sirunyan *et al.*, "Observation of the Higgs boson decay to a pair of tau leptons," 2017.

[15] G. Petrucciani, "Evidence for ttH production with 13 TeV data?" CERN, Geneva, Tech. Rep. CMS-CR-2017-144, May 2017. [Online]. Available: https://cds.cern.ch/record/2267575

[16] R. Narayan, "Evidence for ttH production at ATLAS experiment," Nov 2017. [Online]. Available: https://cds.cern.ch/record/2293116

[17] M. Aaboud *et al.*, "Combination of inclusive and differential $t\bar{t}$ charge asymmetry measurements using ATLAS and CMS data at $\sqrt{s} = 7$ and 8 TeV," *Submitted to: JHEP*, 2017.

[18] A. M. Sirunyan *et al.*, "Observation of top quark production in proton-nucleus collisions," *Phys. Rev. Lett.*, vol. 119, no. 24, p. 242001, 2017.

[19] "Measurement of the top quark mass in the $t\bar{t} \to$ lepton+jets channel from $\sqrt{s}$=8 TeV ATLAS data," CERN, Geneva, Tech. Rep. ATLAS-CONF-2017-071, Sep 2017. [Online]. Available: https://cds.cern.ch/record/2285809

[20] M. Aaboud *et al.*, "Measurement of the W-boson mass in pp collisions at $\sqrt{s} = 7$ TeV with the ATLAS detector," 2017.

[21] G. Pasztor, *Electroweak measurements in CMS*, ser. Moriond QCD, 2017.

[22] A. Apyan, *Electroweak precision measurements with Z and W bosons at the LHC*, ser. Moriond EW Session, 2017.

[23] "ATLAS Phase-II Upgrade Scoping Document," CERN, Geneva, Tech. Rep. CERN-LHCC-2015-020. LHCC-G-166, Sep 2015. [Online]. Available: https://cds.cern.ch/record/2055248

[24] "Projections for measurements of Higgs boson signal strengths and coupling parameters with the ATLAS detector at a HL-LHC," CERN, Geneva, Tech. Rep. ATL-PHYS-PUB-2014-016, Oct 2014. [Online]. Available: https://cds.cern.ch/record/1956710

[25] G. Apollinari, I. Béjar Alonso, O. Brüning, M. Lamont, and L. Rossi, *High-Luminosity Large Hadron Collider (HL-LHC): Preliminary Design Report*, ser. CERN Yellow Reports: Monographs. Geneva: CERN, 2015. [Online]. Available: https://cds.cern.ch/record/2116337

[26] "Letter of Intent for the Phase-I Upgrade of the ATLAS Experiment," CERN, Geneva, Tech. Rep. CERN-LHCC-2011-012. LHCC-I-020, Nov 2011. [Online]. Available: https://cds.cern.ch/record/1402470

[27] ATLAS Collaboration, "Letter of Intent for the Phase-II Upgrade of the ATLAS Experiment," CERN, Geneva, Tech. Rep. CERN-LHCC-2012-022. LHCC-I-023, Dec 2012, draft version for comments. [Online]. Available: https://cds.cern.ch/record/1502664

[28] J. Pequenao, "Computer generated image of the whole ATLAS detector," Mar 2008. [Online]. Available: http://cds.cern.ch/record/1095924

[29] *ATLAS inner detector: Technical Design Report, 1*, ser. Technical Design Report ATLAS. Geneva: CERN, 1997. [Online]. Available: https://cds.cern.ch/record/331063

[30] N. Wermes and G. Hallewel, *ATLAS pixel detector: Technical Design Report*, ser. Technical Design Report ATLAS. Geneva: CERN, 1998. [Online]. Available: https://cds.cern.ch/record/381263

[31] Y. Unno, "ATLAS silicon microstrip detector system (SCT)," *Nucl. Instrum. Meth.*, vol. A511, pp. 58–63, 2003.

[32] E. Abat *et al.*, "The ATLAS Transition Radiation Tracker (TRT) proportional drift tube: Design and performance," *JINST*, vol. 3, p. P02013, 2008.

[33] M. Capeans *et al.*, "ATLAS Insertable B-Layer Technical Design Report," Tech. Rep. CERN-LHCC-2010-013. ATLAS-TDR-19, Sep 2010. [Online]. Available: https://cds.cern.ch/record/1291633

[34] J. Pequenao, "Computer generated image of the ATLAS inner detector," Mar 2008. [Online]. Available: http://cds.cern.ch/record/1095926

[35] *ATLAS liquid-argon calorimeter: Technical Design Report*, ser. Technical Design Report ATLAS. Geneva: CERN, 1996. [Online]. Available: https://cds.cern.ch/record/331061

[36] *ATLAS tile calorimeter: Technical Design Report*, ser. Technical Design Report ATLAS. Geneva: CERN, 1996. [Online]. Available: https://cds.cern.ch/record/331062

[37] J. Pequenao, "Computer Generated image of the ATLAS calorimeter," Mar 2008. [Online]. Available: http://cds.cern.ch/record/1095927

[38] Henric Wilkens and the ATLAS LArg Collaboration, "The ATLAS Liquid Argon calorimeter: An overview," *Journal of Physics: Conference Series*, vol. 160, no. 1, p. 012043, 2009. [Online]. Available: http://stacks.iop.org/1742-6596/160/i=1/a=012043

[39] *ATLAS muon spectrometer: Technical Design Report*, ser. Technical Design Report ATLAS. Geneva: CERN, 1997. [Online]. Available: https://cds.cern.ch/record/331068

[40] J. Pequenao, "Computer generated image of the ATLAS Muons subsystem," Mar 2008. [Online]. Available: http://cds.cern.ch/record/1095929

[41] S. Aefsky, "Alignment of the Muon Spectrometer in ATLAS," *Physics Procedia*, vol. 37, no. Supplement C, pp. 51 – 56, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1875389212016641

[42] *ATLAS central solenoid: Technical Design Report*, ser. Technical Design Report ATLAS. Geneva: CERN, 1997, electronic version not available. [Online]. Available: https://cds.cern.ch/record/331067

[43] J. P. Badiou, J. Beltramelli, J. M. Baze, and J. Belorgey, *ATLAS barrel toroid: Technical Design Report*, ser. Technical Design Report ATLAS. Geneva: CERN, 1997, electronic version not available. [Online]. Available: https://cds.cern.ch/record/331065

[44] ATLAS collaboration, "ATLAS endcap toroids: Technical design report," 1997.

[45] A. Yamamoto *et al.*, "The ATLAS central solenoid," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 584, no. 1, pp. 53 – 74, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0168900207020414

[46] D. Caforio, "Luminosity measurement using Cherenkov Integrating Detector (LUCID) in ATLAS," in *Astroparticle, particle and space physics, detectors and medical physics applications. Proceedings, 10th Conference, ICATPP 2007, Como, Italy, October 8-12, 2007*, 2008, pp. 413–417.

[47] P. Jenni, M. Nessi, and M. Nordberg, "Zero Degree Calorimeters for ATLAS," CERN, Geneva, Tech. Rep. LHCC-I-016. CERN-LHCC-2007-001, Jan 2007. [Online]. Available: https://cds.cern.ch/record/1009649

[48] L. Adamczyk *et al.*, "Technical Design Report for the ATLAS Forward Proton Detector," Tech. Rep. CERN-LHCC-2015-009. ATLAS-TDR-024, May 2015. [Online]. Available: https://cds.cern.ch/record/2017378

[49] S. Jakobsen, P. Fassnacht, P. Hansen, and J. B. Hansen, "Commissioning of the Absolute Luminosity For ATLAS detector at the LHC," Dec 2013, presented 31 Jan 2014. [Online]. Available: https://cds.cern.ch/record/1637195

[50] P. Hamal, "Physics prospects with the ALFA and AFP detectors," May 2013. [Online]. Available: https://cds.cern.ch/record/1548109

[51] A. Khalek *et al.*, "The ALFA Roman Pot Detectors of ATLAS. The ALFA Roman Pot Detectors of ATLAS," *JINST*, vol. 11, no. arXiv:1609.00249, p. P11013. 36 p, Sep 2016, 37 pages, 22 figures, final version published in JINST. [Online]. Available: https://cds.cern.ch/record/2212402

[52] P. Jenni, M. Nessi, M. Nordberg, and K. Smith, *ATLAS high-level trigger, data-acquisition and controls: Technical Design Report*, ser. Technical Design Report ATLAS. Geneva: CERN, 2003. [Online]. Available: https://cds.cern.ch/record/616089

[53] M. Aaboud *et al.*, "Performance of the ATLAS Trigger System in 2015," *Eur. Phys. J. C*, vol. 77, no. CERN-EP-2016-241. 5, p. 317. 76 p, Nov 2016. [Online]. Available: https://cds.cern.ch/record/2235584

[54] M. Dano Hoffmann, "Commissioning and Initial Run-2 Operation of the ATLAS Minimum Bias Trigger Scintillators," CERN, Geneva, Tech. Rep. ATL-DAQ-PROC-2015-020, Jun 2015. [Online]. Available: https://cds.cern.ch/record/2025200

[55] F. Sauli, *Instrumentation in High Energy Physics*, ser. Advanced series on direction in High Energy Physics. P O Box, Farrer Road, Singapore 9128: World Scientific, 1993.

[56] C. W. Fabjan and F. Gianotti, "Calorimetry for Particle Physics," *Rev. Mod. Phys.*, vol. 75, no. CERN-EP-2003-075, pp. 1243–1286. 96 p, Oct 2003. [Online]. Available: https://cds.cern.ch/record/692252

[57] K. S. Krane, *Introductory Nuclear Physics*. 111 River Street, Hoboken, NJ 07030: John Wiley and Sons, 1987.

[58] M. V.S., "Progress in Elementary Particle and Cosmic Ray Physics," *North-Holland Publ. Comp.*, vol. IX, pp. 245–303, 1967.

[59] P. Adragna, *et al.*, "Measurement of pion and proton response and longitudinal shower profiles up to 20 nuclear interaction lengths with the ATLAS Tile calorimeter," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 615, no. 2, pp. 158 – 181, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0168900210001051

[60] E. Abat *et al.*, "Response and Shower Topology of 2 to 180 GeV Pions Measured with the ATLAS Barrel Calorimeter at the CERN Test-beam and Comparison to Monte Carlo Simulations," CERN, Geneva, Tech. Rep. ATL-COM-CAL-2010-004, May 2010. [Online]. Available: https://cds.cern.ch/record/1263858

[61] A. Airapetian *et al.*, "ATLAS calorimeter performance Technical Design Report," 1996.

[62] H. De La Torre, "Status of the Atlas Liquid Argon Calorimeter and its Performance after three years of LHC operation," CERN, Geneva, Tech. Rep. ATL-LARG-PROC-2013-004, Aug 2013. [Online]. Available: https://cds.cern.ch/record/1581520

[63] J. Benitez, "Performance of the ATLAS Liquid Argon Calorimeters in LHC Run-1 and Run-2," May 2016. [Online]. Available: https://cds.cern.ch/record/2153383

[64] N. Nikiforou, "Performance of the ATLAS Liquid Argon Calorimeter after three years of LHC operation and plans for a future upgrade," in *Proceedings, 3rd International Conference on Advancements in Nuclear Instrumentation Measurement Methods and their Applications (ANIMMA 2013): Marseille, France, June 23-27, 2013*, 2013. [Online]. Available: http://inspirehep.net/record/1240499/files/arXiv:1306.6756.pdf

[65] T. Barillari, "The ATLAS liquid argon hadronic end-cap calorimeter: construction and selected beam test results," *Nuclear Physics B Proceedings Supplements*, vol. 150, pp. 102–105, 2006.

[66] J. Hostachy, "Progress report on the ATLAS liquid argon presampler," *Nuclear Physics B - Proceedings Supplements*, vol. 61, no. 3, pp. 89 – 94, 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0920563297005446

[67] F. Teischinger, M. Aleksa, and C. Fabjan, "Energy Measurement with the ATLAS Electromagnetic Calorimeter at the Per Mill Accuracy Level," Feb 2014, presented 19 Mar 2014. [Online]. Available: https://cds.cern.ch/record/1699838

[68] "ATLAS Liquid Argon Calorimeter Module Zero. Module Zero du calorimètre à Argon liquide d'ATLAS," 1993. [Online]. Available: https://cds.cern.ch/record/2254084

[69] D. Banfi, L. Carminati, and L. Mandelli., "Electron and photon energy reconstruction in the EM calorimeter of ATLAS," 2008. [Online]. Available: http://slideplayer.com/slide/11194378/

[70] A. Bazan, "The ATLAS liquid argon calorimeter read-out system," *IEEE Trans. Nucl. Sci.*, vol. 53, pp. 735–740, 2006. [Online]. Available: https://cds.cern.ch/record/1035600

[71] N. J. Buchanan *et al.*, "ATLAS liquid argon calorimeter front end electronics," *Journal of Instrumentation*, vol. 3, no. 09, p. P09003, 2008. [Online]. Available: http://stacks.iop.org/1748-0221/3/i=09/a=P09003

[72] N. J. Buchanan *et al.*, "Design and implementation of the Front End Board for the readout of the ATLAS liquid argon calorimeters," *Journal of Instrumentation*, vol. 3, no. 03, p. P03004, 2008. [Online]. Available: http://stacks.iop.org/1748-0221/3/i=03/a=P03004

[73] P. Borgeaud, X. De la Broise, E. Ferrer-Ribas, A. Le Coguie, B. Mansoulié, and J. Pascual, "The LArg Tower Builder Board: calculation, simulation, measurements." 2002.

[74] J. Colas *et al.*, "Electronics calibration board for the ATLAS liquid argon calorimeters," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 593, no. 3, pp. 269 – 291, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0168900208007407

[75] The Liquid Argon Back End Electronics collaboration, "ATLAS liquid argon calorimeter back end electronics," *"Journal of Instrumentation"*, vol. 2, no. 06, p. P06002, 2007. [Online]. Available: http://stacks.iop.org/1748-0221/2/i=06/a=P06002

[76] F. Henry-Coüannier, "The ATLAS liquid argon calorimeters Read Out Driver (ROD) system," 2000. [Online]. Available: https://cds.cern.ch/record/619214

[77] M. Dhellot, M. Escalier, B. Laforge, O. Le Dortz, D. Martin, and J. M. Parraud, "SPAC Master VME64 boards User Guide," 2004.

[78] C. G. Ruiz, "Calculation of the Optimal Filtering Coefficients and check of the signal reconstruction for the ATLAS Electromagnetic Calorimeter," March 2008.

[79] B. Asman *et al.*, "The ATLAS Level-1 Calorimeter Trigger: PreProcessor implementation and performance," *Journal of Instrumentation*, vol. 7, no. 12, p. P12008, 2012. [Online]. Available: http://stacks.iop.org/1748-0221/7/i=12/a=P12008

[80] G. Anagnostou *et al.*, "Prototype cluster processor module for the ATLAS level-1 calorimeter trigger," 2002. [Online]. Available: https://cds.cern.ch/record/593970

[81] R. Achenbach *et al.*, "Commissioning of the Jet/Energy-sum and Cluster Processors for the ATLAS Level-1 Calorimeter Trigger System," CERN, Geneva, Tech. Rep. ATL-DAQ-CONF-2008-004. ATL-COM-DAQ-2007-045, Oct 2007. [Online]. Available: https://cds.cern.ch/record/1068289

[82] D. Edmunds *et al.*, "Atlas Level 1 Calorimeter Trigger Common Merger eXtended module (CMX) Hardware Description," 2013.

[83] S. Dawson, "Introduction to electroweak symmetry breaking," in *Proceedings, Summer School in High-energy physics and cosmology: Trieste, Italy, June 29-July 17, 1998*, 1998, pp. 1–83.

[84] F. Quevedo, S. Krippendorf, and O. Schlotterer, "Cambridge Lectures on Supersymmetry and Extra Dimensions," 2010.

[85] M. C. Aleksa *et al.*, "ATLAS Liquid Argon Calorimeter Phase-I Upgrade Technical Design Report," Tech. Rep. CERN-LHCC-2013-017. ATLAS-TDR-022, Sep 2013, final version presented to December 2013 LHCC. [Online]. Available: https://cds.cern.ch/record/1602230

[86] N. J. Buchanan *et al.*, "Radiation qualification of the front-end electronics for the readout of the ATLAS liquid argon calorimeters," *JINST*, vol. 3, 2008.

[87] H. Xu, "The Trigger Readout Electronics for the Phase-I Upgrade of the ATLAS Liquid Argon Calorimeters," CERN, Geneva, Tech. Rep. ATL-LARG-PROC-2016-003, Nov 2016.

[88] L. Xiao *et al.*, "LOCx2, a low-latency, low-overhead, 2 $\times$ 5.12-Gbps transmitter ASIC for the ATLAS Liquid Argon Calorimeter trigger upgrade," *Journal of Instrumentation*, vol. 11, no. 02, 2016.

[89] F Liang and others, "The design of 8-Gbps VCSEL drivers for ATLAS liquid Argon calorimeter upgrade," *Journal of Instrumentation*, vol. 8, no. 01, 2013.

[90] (2008, March) PICMG 3.0 Revision 3.0 AdvancedTCA Base Specification. [Online]. Available: https://www.picmg.org/wp-content/uploads/PICMG_ATCA_3_0R3_0.pdf

[91] Intel FPGA. (2017, July) Intel Arria 10 Device Overview. [Online]. Available: https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/arria-10/a10_overview.pdf

[92] P. Moreira *et al.*, "The GBT Project," 2009. [Online]. Available: https://cds.cern.ch/record/1235836

[93] Aad, Georges and others , "Technical Design Report for the Phase-I Upgrade of the ATLAS TDAQ System," Tech. Rep. CERN-LHCC-2013-018. ATLAS-TDR-023, Sep 2013. [Online]. Available: https://cds.cern.ch/record/1602235

[94] H. Chen, "The Prototype Design of gFEX – A Component of the L1Calo Trigger for ATLAS Phase-I Upgrade," CERN, Geneva, Tech. Rep. ATL-DAQ-PROC-2016-046, Dec 2016. [Online]. Available: https://cds.cern.ch/record/2239824

[95] B. Bauss *et al.*, "Development of the jet Feature EXtractor (jFEX) for the ATLAS Level 1 Calorimeter Trigger Upgrade at the LHC," CERN, Geneva, Tech. Rep. ATL-DAQ-PROC-2017-032, Oct 2017. [Online]. Available: https://cds.cern.ch/record/2289210

[96] A. Collaboration, "Technical Design Report for the Phase-II Upgrade of the ATLAS LAr Calorimeter," CERN, Geneva, Tech. Rep. CERN-LHCC-2017-018. ATLAS-TDR-027, Sep 2017. [Online]. Available: https://cds.cern.ch/record/2285582

[97] ALTERA. (2016, January) Stratix IV Device Handbook. [Online]. Available: https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/stratix-iv/stx4_siv51001.pdf

[98] Robert Frazier, Greg Iles, Marc Magrans de Abril, Dave Newbold, Andrew Rose, David Sankey and Tom Williams. (2013, March) The IPbus Protocol. [Online]. Available: http://ohm.bu.edu/~chill90/ipbus/ipbus_protocol_v2_0.pdf

[99] A. Milic, L. Hervas, and C.-E. Wulz, "Demonstrator System for the Phase-I Trigger Readout Upgrade of the ATLAS Liquid Argon Calorimeter at the LHC," Sep 2016, presented 20 Oct 2016. [Online]. Available: http://cds.cern.ch/record/2229980

[100] Intel FPGA. (2017) Quick-Start for Intel Quartus Prime Pro Edition Software.

[101] LAr trigger Digitizer Board (LTDB) group. (2014, February) Proposal format of data link from LTDB to LDPB. [Online]. Available: https://twiki.cern.ch/twiki/pub/LAr/LArDemonstrator/format_of_data_link.pdf

[102] ALTERA, now part of Intel. (2017) V-Series Transceiver PHY IP Core User Guide. [Online]. Available: https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/ug/xcvr_user_guide.pdf

[103] ALTERA. (2014) 10-Gbps EthernetMACMegaCore Function - User Guide. [Online]. Available: http://www.altera.com/literature/ug/10Gbps_MAC.pdf

[104] S. Stärz and A. Straessner, "Energy Reconstruction and high-speed Data Transmission with FPGAs for the Upgrade of the ATLAS Liquid Argon Calorimeter at LHC," Feb 2015, presented 19 May 2015. [Online]. Available: https://cds.cern.ch/record/2030122

[105] LAr community, "Public Liquid-Argon Calorimeter Plots on Upgrade." [Online]. Available: https://twiki.cern.ch/twiki/bin/view/AtlasPublic/LArCaloPublicResultsUpgrade

[106] LAr Demonstrator community, "LAr Demonstrator Twiki." [Online]. Available: https://twiki.cern.ch/twiki/bin/view/LAr/LArDemonstrator

[107] ALTERA now part of Intel. (2016) Nios II Classic Processor Reference Guide. [Online]. Available: https://www.altera.com/en_US/pdfs/literature/hb/nios2/n2cpu_nii5v1.pdf

[108] Silicon Laboratories. (2014) Si570/Si571 - 10 MHZ TO 1.4 GHZ I2C PROGRAMMABLE XO/VCXO. [Online]. Available: https://www.silabs.com/documents/public/data-sheets/si570.pdf

[109] Texas Instruments. (2015) Understanding the I2C Bus. [Online]. Available: http://www.ti.com/lit/an/slva704/slva704.pdf

[110] Philips Semiconductors. (2003) I2C MANUAL. [Online]. Available: https://www.nxp.com/docs/en/application-note/AN10216.pdf

[111] NXP Semiconductors. (2014) I2C-bus specification and user manual. [Online]. Available: https://www.nxp.com/docs/en/user-guide/UM10204.pdf

[112] Nicolas Dumont Dayot. (2015, February) ABBA demonstrator Front FPGA. [Online]. Available: https://indico.cern.ch/event/376592/contributions/1799532/attachments/751002/1030296/ATLASLar_LAPP_2015_0203_ABBA.pdf

[113] Altera Corporation. (2009, November) Design Debugging Using the SignalTap II Embedded Logic Analyzer. [Online]. Available: https://www2.pcs.usp.br/~labdig/material/15.%20Design%20Debugging%20Using%20the%20SignalTap%20II%20Embedded%20Logic%20Analyzer.pdf

[114] ATLAS Software Project, "ATLAS Athena Guide," April 2017.

[115] G. Aad, A. Camplani, N. Chevillot, B. Dinkespiler, N. Dumont-Dayot, Y. Enari, R. Hentges, K. Johns, I. Wingerter and V. Zhulanov, "ATLAS LAr Calorimeter trigger electronics phase I upgrade: LATOME Firmware Specifications," August 2017, aTLAS Internal Note. [Online]. Available: https://gitlab.cern.ch/atlas-lar-ldpb-firmware/LATOME/blob/master/LATOME/doc/LAr-LATOME-FW/LAr-LATOME-FW.pdf

[116] Intel. (2017, May) Avalon Interface Specifications. [Online]. Available: http://www.altera.com/literature/manual/mnl_avalon_spec.pdf?GSA_pos=4&WT.oss_r=1&WT.oss=avalon%20interface

[117] The Cactus Team. Cactus 4.0, User Guide. [Online]. Available: http://cactuscode.org/documentation/UsersGuide.pdf

[118] Nicolas Dumont Dayot, "LATOME reference manual," August 2017, aTLAS Internal Note. [Online]. Available: https://gitlab.cern.ch/atlas-lar-ldpb-firmware/LATOME/blob/latome_lli/LATOME/doc/board/latome_reference_manual.pdf

[119] Ph. Farthouat and P. Gällnö . (2000, May) TTC-VMEbus INTERFACE, TTCvi - MkII. [Online]. Available: http://ttc.web.cern.ch/TTC/TTCviSpec.pdf

[120] Sophie Baron. (2014, February) Timing, Trigger and Control (TTC) Systems for the LHC. [Online]. Available: http://ttc.web.cern.ch/TTC/

[121] U. Farooq, Z. Marrakchi, and H. Mehrez, *Tree-based Heterogeneous FPGA Architectures: Application Specific Exploration and Optimization*. New York: Springer, 2012. [Online]. Available: https://cds.cern.ch/record/1503787

[122] P. P. Chu, *RTL hardware design using VHDL: coding for efficiency, portability, and scalability*. Newark, NJ: Wiley, 2006. [Online]. Available: https://cds.cern.ch/record/1141261

[123] P. J. Ashenden, *The designer's guide to VHDL; 3rd ed.*, ser. The Morgan Kaufmann Series in Systems on Silicon. Burlington, MA: Morgan and Kaufmann, 2008. [Online]. Available: https://cds.cern.ch/record/1167307

[124] Scott Chacon and Ben Straub, *Pro Git*, 2nd ed. Apress, November 2014.

[125] S. Baron. (2014) Timing, Trigger and Control (TTC) Systems for the LHC. [Online]. Available: http://ttc.web.cern.ch/TTC/

[126] J. Christiansen *et al.*, "TTCrx Reference Manual," CERN, Geneva, Tech. Rep., October 2004. [Online]. Available: http://ttc.web.cern.ch/ttc/ttcrxmanual3.9.pdf

[127] *ATLAS level-1 trigger: Technical Design Report*, ser. Technical Design Report ATLAS. Geneva: CERN, 1998. [Online]. Available: https://cds.cern.ch/record/381429

[128] S. Ask *et al.*, "The ATLAS central level-1 trigger logic and TTC system," *Journal of Instrumentation*, vol. 3, no. 08, p. P08002, 2008. [Online]. Available: http://stacks.iop.org/1748-0221/3/i=08/a=P08002

Acknowledgements

First of all, I would like to thank my thesis supervisors, Valentino Liberali e Francesco Tartarelli. Thanks to their support I had the chance to pursue the physics studies, being involved in challenging research activities. Without their experience and precious advices I could not have accomplished this work.

I want to express a special thank you to Mauro Citterio, who played an important role after my master thesis. He introduced me to the FPGAs world and gave me the opportunity to work in an international environment. He was never lacking kind words and good advices.

I would like to explain my deep gratitude to Luis Hervas, who gave me the opportunity to stay at CERN for two years and work on the Phase-I upgrade of the ATLAS LAr calorimeter.

Thanks to the people of the Demonstrator, LATOME and LAr Online groups. I truly enjoyed the time we spent working together: from all of them I learned some good lessons. Thanks for the enthusiasm and the friendship. This work was possible also thanks to their efforts.

Thanks to my office mates, colleagues and friends, in Milan and at CERN: the days would have passed far more slowly without them. I never felt alone.

Some more special and personal thanks:

I am grateful to Adriana, first person and friend met at CERN: without her this experience would have not been the same.

I would like to thank Max, accidentally met in Amsterdam: always encouraging and supporting me throughout the final steps of this experience and for my future. Your love and kindness are precious to me.

Finally, I wish to thank my lovely parents, always present: their support has been unconditional all these years. They helped me at every stage of my personal and academic life, to see all my achievements come true. This thesis is dedicated to you.