The Central Trigger Processor of the ATLAS Experiment at the LHC and its Monitoring.

Written by

Gorm Galster

The Niels Bohr Institute & CERN

Supervised by

Thilo Pauly

CERN

Mogens Dam

The Niels Bohr Institute

This thesis has been submitted to the PhD School of The Faculty of Science, University of Copenhagen.

Abstract

For the second run period of the Large Hadron Collider (LHC), the ATLAS Central Trigger Processor (CTP) was upgraded. The CTP is the first part of a two-layer trigger system and is responsible for making the initial trigger decision, applying experimental dead-time, and distributing the Trigger Timing and Control (TTC) signals to all sub-detectors. Using 512 trigger items for selecting collisions with high-p_{T} charged leptons or missing transverse energy ($\not{E}_{\mathrm{T}}$), the CTP reduces the 40 MHz event rate to 100 kHz.

The CTP is monitored extensively as the monitoring data provides crucial information about the state of the experiment, the dead-time incurred, and the bandwidth utilisation. The monitoring data is further used for calculating the dead-time correction factors needed in order to normalise the recorded collision data to the number of delivered collisions.

This thesis details the upgrade of the CTP infrastructure with a particular focus on monitoring and its use. Archived monitoring data is used to demonstrate its utility for detecting operational issues. The data is further used to perform two cross checks of the calculation of the dead-time correction factors. Lastly, a new rule-based automation framework for the operation of the trigger is presented. The framework makes extensive use of the monitoring data available during data-taking and is currently used to detect and automatically mitigate certain detector issues.

Synopsis

For den anden driftsperiode ved Den Store Hadron-kollider (LHC) blev ATLAS' Centrale Trigger-Processor (CTP) opgraderet. CTPen er den første del af et to-lags trigger system, og er ansvarlig for at træffe den første triggerbeslutning, håndtere eksperimentel dødtid, samt at distribuere trigger-, tids- og kontol-signaler (TTC) til alle under-detektorene. Ved brug af 512 triggere, til udvælgelse af kollisioner med høj-p_{T} ladede leptoner eller en observerbar energi-ubalance ($\not{E}_{\mathrm{T}}$), reducerer CTPen begivenhedsraten fra 40 MHz til 100 kHz. CTPen monitoreres ekstensivt, da monitoreringsdataen giver kritisk information om eksperimentets tilstand og dødtid samt om båndbredeudnyttelsen. Monitoreringsdataen er yderligere brugt til at beregne dødstidskorrektionsfaktorene, der er nødvendige for at relatere det gemte kollisionsdatasæt til antallet af leverede kollisioner.

Denne afhandling beskriver opgraderingen af CTP-infrastrukturen med særligt fokus på monitoreringen og dens brug. Arkiveret monitoreringsdata er brugt til at demonstrere dennes nytte i forhold til detektering af operationelle problemer. Dataen er desuden brugt til to krydstjek af beregningen af dødstidskorrektionsfaktorene. Til sidst præsenteres et nyt reglebaseret automatiserings-system til styring af trigger-driften. Automatiserings-systemet gør udstrakt brug af den monitoreringsdata, der er tilgængelig under datatagning, og bruges i øjeblikket til at detektere og afbøde visse detektorproblemer.

Acknowledgements

As part of a large collaboration, and working on the upgrade of probably the most central part of the ATLAS detector, I have had the pleasure of working with so many dedicated, inspiring and knowledgeable people. I would like to thank all my friends and colleagues in ATLAS, especially those involved with the operations effort, for making CERN an unforgettable and amazing place to work. Without these people, little would have been the same, and likely, there would have been no experiment.

I would like to thank the ATLAS Trigger and Data Acquisition group that took me in, taught me a lot, gave me responsibility, and eventually trusted me with this project.

Of all the people at CERN, I would in particular like to thank Thilo Pauly, who served as my CERN supervisor. Never was he too busy to discuss a matter and any issue that ever arose was met with well contemplated and constructive feedback. Neither the upgrade project nor this thesis would have been a reality without his supervision.

At my home institute, the Niels Bohr Institute, I owe a great thanks to Jens Jørgen Gaardhøje and Mogens Dam, who supervised the writing of this thesis and who, in moments of need, each fought earnestly to help and support me. In addition, I would like to thank Hass AbouZeid and Kristian Gregersen, for countless fruitful discussions, only a fraction of which are relevant to this thesis.

A heartfelt thank you is also due for my friend Robin Feldman, who, with conviction, took it upon herself to correct my less than perfect English. You are a trooper and a lightsaber.

I will forever be thankful for the endless love and support of Eva Bertels, my anchor in this adventure and many more to come.

Everything not saved will be lost.

— Nintendo *"Quit Screen"*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1. Foreword

The importance of monitoring is often overlooked and forgotten, in much the same way that we live our daily lives in ignorance of the unrelenting work our eyes, ears, and the rest of our organs are doing. At least until something goes wrong.

At the Large Hadron Collider (LHC) protons are collided at unprecedented energies and at unprecedented luminosity, in an ongoing pursuit of new physics. ATLAS is one of four main detector experiments used to study the collision debris at one of the four interaction points where the proton beams are brought to collide. The high interaction rate (and thus data rate), and the extreme rarity of collision events of interest or use for analyses, make the use of a trigger system an absolute necessity: by selecting online what events to store (and not) the signal-to-noise ratio can be improved and the recorded data volume reduced to manageable levels.

In order to make sense of the recorded data set, and relate it to the set of all collisions, additional meta-data about the selection criteria and the state of the trigger system are needed. ATLAS use a two-stage trigger system: a hardware trigger (Level 1) that makes a fast decision based on partial information and a High Level Trigger (HLT) using conventional computers for making the final decision based on the full event information. The Central Trigger Processor (CTP) is at the heart of the Level 1 trigger system and is responsible, amongst other things, for making the first trigger decision. The CTP was upgraded for Run II in Long Shutdown I, between 2013 and 2015, as part of a number of planned upgrades to ATLAS for Run II of the LHC run schedule.

My main contribution to the upgrade is the design of the CTP software infrastructure and the design and implementation of the monitoring software. The CTP monitoring software provides the essential meta-data needed to make sense of any of the recorded data and also provides the operators in ATLAS control room with invaluable information about the current state of the experiment.

This thesis is the extension of my mid-term report, and masters thesis, and has a stronger focus on physics motivation and the technical challenges, that inspired the implementation detailed in the last report. My previous report concluded where we, as a collaboration, had just celebrated the successful start of Run II, marked by the first beam collisions. Since then, an unprecedented amount of physics data has been recorded, leading to new discoveries and tighter limits on physics Beyond the Standard Model (BSM). As (meta-)data is now available in abundance, I will demonstrate how this meta-data can be used to calculate and cross check one of the essential correction factors: the dead-time correction factor. The dead-time correction factor provides the normalisation between the collisions events observed and the collisions events produced. As the last part of the thesis, I will present an automation framework and discuss and demonstrate how the monitoring data from the CTP as well as other systems in ATLAS, can be effectively combined to provide an automated feed-back loop for the operation of the trigger. This project was originally intended to automatically treat the symptoms of a particular operational issue, but has since found more use-cases. The discussion include the extension of the framework necessary to obtain a fully automated operation of ATLAS trigger.

1.2. Thesis Outline

The thesis is structured as follows:

Chapter 2 provides an overview of experimental high energy physics at hadron colliders, from which the need for an efficient trigger system arises. The chapter further motivates the need for splitting the trigger system into a hardware-based trigger and a software-based trigger. The general strategy and physics motivation for allocating the available bandwidth in order to maximise the sensitivity to new physics is also briefly covered. The chapter

also provides an introduction to the LHC and the sub-detectors of ATLAS with a focus on their role for the trigger system.

Chapter 3 provides a description of the ATLAS trigger system, the physics quantities available at each level and how these are combined into trigger items. Through the discussion of an archetype detector, the interplay between ATLAS sub-detectors and the hardware trigger is discussed.

Chapter 4 describes the capabilities and functionality of the CTP. Particular emphasis is given to functionality relevant to the CTP's central role in the trigger path as well as in the distribution of trigger, timing and control signals. As the CTP is the only place in ATLAS where dead-time is applied, the busy tree, with the CTP at its root, is also covered. The upgrade of the CTP included a requirement for the support of up to three simultaneous users. As there is only one CTP this requirement had important implications for the implementation of the upgrade.

Chapter 5 motivates the need for monitoring of the trigger system and of the dead-time. The discussion extends to how the capabilities of the CTP, as outlined in Chapter 4, can be utilised for monitoring. The chapter motivates the division of experimental time into "luminosity blocks" of approximate experimental stability, and outlines the need for dead-time correction factors as well as luminosity corrections, to make sense of the recorded data.

Chapter 6 documents the motivation and design considerations for the implementation of monitoring in the upgraded CTP. The chapter opens with an assessment and recap of the requirements and constraints and presents a solution. The design considerations of this solution for monitoring of the CTP is discussed in detail. For the general considerations of the upgrade of the CTP infrastructure as a whole, please see my previous thesis [1].

Chapter 7 shows how the archived monitoring data can be used to verify and cross check the functionality of the CTP. Two cross checks of the calculation of the dead-time correction factors are shown and an analysis of the stability of the experimental conditions is carried out. Through a few examples, the chapter also demonstrates how the trigger system is sensitive to operational issues.

Chapter 8 discusses the implementation of a rule-based framework that can leverage the (trigger) monitoring data, detect issues, and make changes

to the trigger configuration accordingly. The current use of the system is illustrated through two examples, followed by a general discussion of the possibilities of extending the framework beyond its current use.

1.3. Personal Contributions

In modern experimental high energy physics, most research is conducted in large collaborations, such that the experimental challenges of constructing, running and analysing data from an experiment can be shared amongst many. The ATLAS experiment at the LHC is one such collaboration and comprises about 3,000 scientific authors from 181 institutions around the world.

This thesis presents research carried out between September 2013 and October 2016 as part of the ATLAS collaboration – specifically for the planned upgrade of the ATLAS detector for Run II of the LHC run-schedule [2]. In order to present the research in complete form, the work must be presented within the context of the experimental effort as a whole. Furthermore, in order to present the research in a coherent and consistent manner, without repetition or the presentation of superseded results, only the most significant portion of my work is presented here – namely the work relating directly to the upgrade and monitoring of the ATLAS' Central Trigger Processor.

For clarity, this section summarises my personal contributions to the research and effort presented in this thesis. In addition, all figures and tables that I did not produce are indicated using citation or appropriate attribution; figures and tables without attribution are my own work.

The CTP Software Architecture: Part of the upgrade of the CTP was an upgrade of the CTP software infrastructure. I was the architect of this new infrastructure and implemented large parts of the shared functionality. The infrastructure is briefly outlined in Section 6.1 and detailed in [1].

The CTP Monitoring Server: This software is a corner stone for the operation of the CTP and ATLAS and provides the critical monitoring data needed to aid the operators in the control room, for calculating the dead-time correction factors, needed to normalise the recorded

collision data to the delivered collision data, and for post-mortem analysis in case of problems.

I designed and wrote the code for the CTP monitoring server. The design considerations and requirements are the topic of Chapter 6.

Control Room Applications: The monitoring data from the CTP has many operational uses and is presented in various ways in the ATLAS Control Room.

I designed and wrote the "Busy Panel" used to visualise the health of ATLAS data-taking (see Section 6.7.2.2). I also designed and supervised the development of a web based display of the Level 1 trigger rates (see Section 6.7.2.1).

Analysis and Use of Monitoring Data: The cross checks, analysis and illustrative examples detailed in Chapter 7 is solely my work.

The Automatic Prescaler: I proposed, designed and supervised the implementation of the Automatic Prescaler that provides an advanced rule based feed-back loop, allowing the behaviour of the trigger system to be automated based on the observed behaviour of the experiment.

The system is currently in use, though primarily used to automatically handle known detector effects.

Besides the work documented in my thesis, I have had the pleasure during numerous shift periods as "on-call trigger expert" in the ATLAS Control Room to assist in the integration of the Insertable B-Layer (IBL) and the new Micro-Mesh Gaseous Structure (MicroMegas) detector with the CTP and the rest of the ATLAS infrastructure.

Chapter 2

Experimental Particle Physics at the LHC

2.1. Physics Objectives and Goals

The Standard Model of Particle Physics (SM) has, since its formulation in the 1970s, been the theoretical framework for describing the nature and interaction of matter and energy at sub-atomic scales. The SM has undergone extensive experimental testing and accurately describes a large variety of experimental data over large energy ranges. With the 2012 discovery of a particle compatible with the SM Higgs boson, made by A Toroidal LHC ApparatuS (ATLAS) and Compact Muon Solenoid (CMS), all elementary particles predicted by the SM have now been observed. There are however still fundamental questions left unanswered [3] [4], amongst these are, in no particular order:

- Are the fundamental forces of nature unified?
- What is the nature of dark matter?
- What is the origin of baryon asymmetry in the Universe?
- What is the origin of the neutrino masses?
- What is the solution to the hierarchy problem ?

The SM does not give answers to these questions and answers must thus reside in the regime of Beyond the Standard Model (BSM) physics. Many theories and extensions exist for BSM physics. A brief introduction to, and summary of, BSM models, can be found in [4]. Common for the BSM models are that they try to solve one or more problems with the SM and

often introduce new particles in their attempt to do so. Experimentally, there are two approaches to the search for BSM physics:

1. Direct searches – where the existence of new particles is directly observed.
2. In-direct searches – where precise measurement of SM properties are used to put limits on (or rule out) new theories.

Either method relies on a very good understanding of the experimental setup and on the SM background. From a pragmatic point of view, the direct search for any new physics can be considered the careful search for any observable deviation from the SM background.

Since the discovery of the Higgs boson, the measurement of the properties of the new boson has been a key focus at both ATLAS and CMS. The most recent discovery, in June of 2018, is the direct observation and measurement of the Higgs coupling to the top quark [5]. Like so, precise measurement of the Higgs boson's production and decay rates remain an active area of research. The measurement of the Higgs self-coupling, via di-Higgs production [6], would yield an important check of electro-weak symmetry breaking and would be a way to probe several BSM scenarios that allow a large correction to the self coupling [7]. The study of rare decays, such as $H \longrightarrow Z\,\gamma$, or decays involving second generation fermion couplings, could provide valuable insights into flavour physics [8].

Many BSM searches are feasible at the Large Hadron Collider (LHC). Amongst the more prominent BSM searches are searches for Super Symmetry (SUSY) [9] – a "framework" of BSM models that, typically motivated by solving the hierarchy problem, introduces a new symmetry between fermions and bosons. There are many different SUSY models, each of which depends on a set of parameter that can be chosen freely. At the LHC, searches for super-symmetric partners of the quarks (squarks) and gluons (gluinos) are particularly powerful since such particles would be produced in large quantities in the strong primary scattering process. Searches for processes in which *R-parity* is conserved are also an active area of research. The signatures of such SUSY models would be jets from the (cascade) decay of the quarks and missing energy from the Lightest Super-symmetric Particle (LSP) escaping the detector undetected. These signatures are common for other BSM models too.

2.2. The Needle and the Haystack

Experimental high energy physics at a proton collider, such as the LHC, is a search for extremely rare collision events with an overwhelmingly large background. Studying the debris of colliding hadrons is but one way of studying the fundamental particles of nature. Hadron collider experiments can produce collisions at higher energies and at higher rate (luminosity), which makes them our best candidate for producing and discovering heavy particles. In a *pp*-collider experiment, the underlying interactions are primarily strong interactions and as only part of the proton participates in the interaction – with the participating partons carrying a variable fraction of the protons' total energy – the underlying process and the energy available for producing new particles varies from collision to collision. Thus, the high production rate and energies achievable at a *pp*-colliders comes at the price of high collision-to-collision variation with the majority of processes resulting in Quantum Chromo Dynamics (QCD) jets.

2.2.1. Cross Sections

In order to quantify the above discussion, it is instructive to look at the production rate of particles at the LHC. The production rate $\frac{dN}{dt}$ is the product of the instantaneous luminosity, $\mathcal{L}$, and the (energy and process specific) cross section, σ:

$$\frac{dN}{dt} = \sigma \mathcal{L} \, . \tag{2.1}$$

Figure 2.1 shows the total cross section as well as the cross section for a selection of SM processes in proton-(anti)proton collisions as a function of center of mass energy $\sqrt{s}$. The total cross section σ_{tot} is the sum of the cross sections for any SM *pp* interaction and is primarily comprised of QCD processes. The center of mass energy $\sqrt{s} = 7\,\text{TeV}$ for the LHC, as well as the instantaneous luminosity[1] of $\mathcal{L} = 10^{33}\,\text{cm}^{-1}\text{s}^{-1}$ shown in Figure 2.1 reflects the experimental conditions during the first run period, Run I. Collisions at a center of mass energy of $\sqrt{s} = 13\,\text{TeV}$ was reached in the beginning of Run II (2015) and the luminosity has since reached $\mathcal{L} = 2.14 \times 10^{34}\,\text{cm}^{-1}\text{s}^{-1}$ – little more than twice the design-luminosity of

[1] From here on, "luminosity" will be used to refer to the instantaneous luminosity unless otherwise indicated.

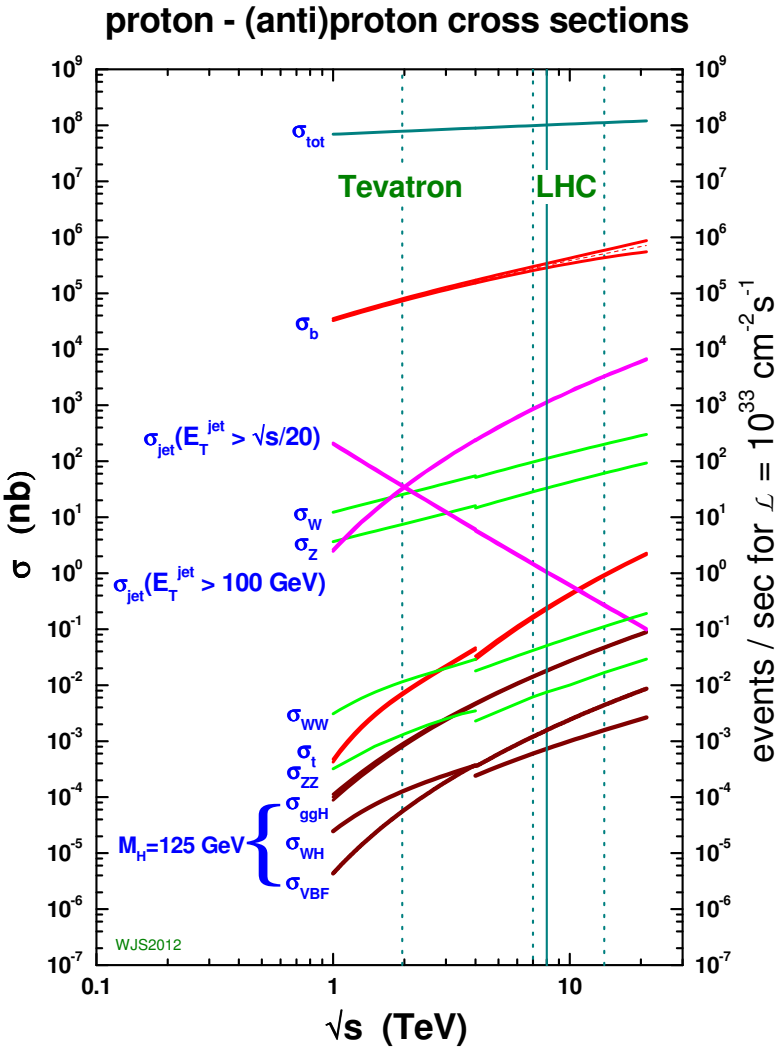


Figure 2.1: Cross section (left) and production rate (right) at, $\mathcal{L} = 10^{33}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$, as function of center of mass energy, $\sqrt{s}$, for selected processes. Source: [10]

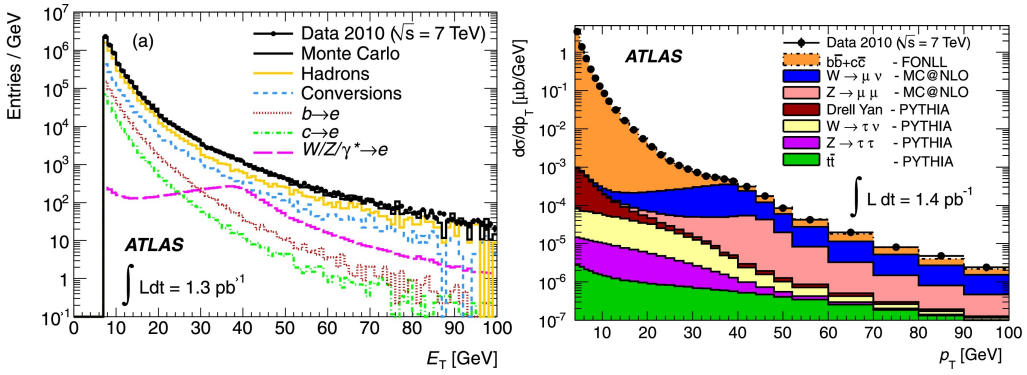


Figure 2.2: Inclusive distributions of transverse electron energy (left) and muon momentum (right). Source: [11].

$\mathcal{L}_{\text{design}} = 10^{34}\,\text{cm}^{-1}\text{s}^{-1}$. This implies that the current interaction rate at the LHC is an order of magnitude above what is detailed in Figure 2.1. From the plot it can be seen that the total cross section σ_{tot} is $\mathcal{O}(10^{10})$ bigger than the total Higgs cross section $\sigma_{\text{H}} \approx \sigma_{\text{ggH}} + \sigma_{\text{WH}} + \sigma_{\text{VBF}}$. It can further be seen that the total cross section σ_{tot} grows slower with $\sqrt{s}$ than the electroweak processes of interest, e.g., σ_{H}, which motivates the increase in collision energy. Collision at the design energy of $\sqrt{s}_{\text{design}} = 14\,\text{TeV}$ has still not been reached. For the study of almost any process, the total cross section implies an overwhelmingly large (QCD) background.

2.2.2. Weak Decay Processes

With an overwhelmingly large background and a sparse signal, effectively identifying processes of interest becomes of paramount importance. Doing so forms the basis for any analysis and for the baseline trigger strategy discussed in Section 3.3.

Most processes of interest include particles produced by the electroweak interaction as opposed to the strong interaction, responsible for the QCD background. As the weakly decaying particle are typically heavy (W, Z, t, H), the decay products tend to have high momentum. The observable signatures for these processes are high-momentum leptons and missing energy from the neutrinos escaping detection: for the typical decay of a Z boson (at rest) the momentum of the decay leptons is expected to be around 45.6 GeV – half of the mass of Z.

Figure 2.2 shows the inclusive spectrum for transverse electron energy (left) and transverse muon momentum (right) as a function of transverse energy and momentum, respectively. The inclusive momentum distributions for both can be seen to drop off rapidly with (transverse) momentum, driven by an initial drop off in hadronic processes. As the longitudinal component is not known from collision to collision, the transverse energy and momentum is commonly used. In both cases, imposing a momentum or energy threshold on the lepton candidate provides a significant rejection of the background processes while maintaining large parts of the contribution from Z and W decays. This improves the signal quality and reduces the sample size of the selection.

2.2.3. Decay Mode and Background

The low production cross section for processes of interest is only part of the experimental challenges: after a particle has been produced, it also needs to be detected. However, unless the particle is long lived enough to make it into the detector, what is actually detected is the debris of the decaying particle.

Figure 2.3 shows the theoretical branching fractions for the decay modes of a SM Higgs as a function of the Higgs mass m_H. In many cases the SM Higgs decays into something that is impossible ($c\bar{c}$, gg) or difficult ($b\bar{b}$, $\tau^+\tau^-$) to distinguish from the 10 orders of magnitude of primarily QCD background. Despite excellent detector resolution, this poses an experimental challenge. The decay channels with the cleanest detector signature are $H \longrightarrow \gamma\gamma$ and $H \longrightarrow ZZ$ with a subsequent decay of $ZZ \longrightarrow 4l$ for $l \in \{e, \mu\}$. With the Higgs mass of $m_H \sim 125\,\text{GeV}$ [13] the total branching ratio of the SM Higgs into either of these channels is $\Gamma_{\text{clean}} \sim 0.24\%$, the main contribution, $\sim 0.23\%$, coming from $H \longrightarrow \gamma\gamma$. This adds an effective factor 400 reduction to an already poor signal-to-noise ratio.

2.3. The Large Hadron Collider

The LHC [15], situated at the European Organization for Nuclear Research (CERN) in Geneva, Switzerland, is the largest and most powerful collider and storage ring in existence. The underground structure of the LHC tunnel is depicted in Figure 2.4. The ring is divided into octets by the eight

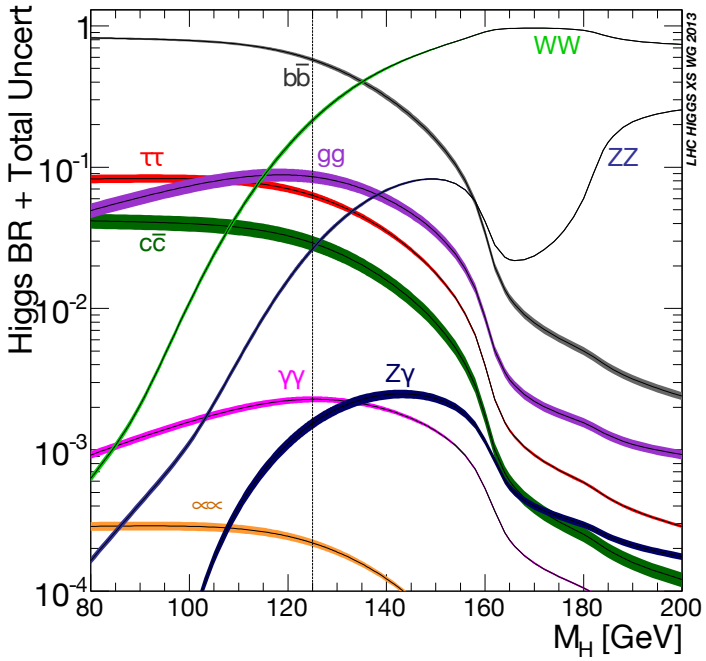


Figure 2.3: Standard Model Higgs boson decay branching ratios as function of Higgs mass m_H. Source: [12]

strategic *Point*s on the ring. Each Point is numbered in the clockwise direction, starting at one with the point closes to the main CERN-site, in Meyrin. The segment between each is called a Sector and numbered based on the adjacent Points. The LHC consists of two beam pipes with Interaction Points (IP) at the center of the four large detector experiments: ATLAS at Point 1, A Large Ion Collider Experiment (ALICE) at Point 2, CMS at Point 5, and LHCb at Point 8. The injection of the beam into the LHC happens in Sector 12 and Sector 81 for the clockwise and the counter-clockwise rotating beam, respectively. The Radio Frequency (RF) cavities used to accelerate the beam are located at Point 4. From the RF cavities the reference bunch clock is extracted. The LHC beam-dump is located at Point 6. For completeness it should be mentioned that Point 3 is used for beam monitoring and collimation and that Point 7 primarily hosts a cooling plant for the LHC cryogenics.

The LHC is situated in the old $\sim 27\,\text{km}$ long Large Electron Positron collider (LEP) tunnel. The LEP ceased operation in the year 2000 as the loss to synchrotron radiation restricted the energies that could be reached.

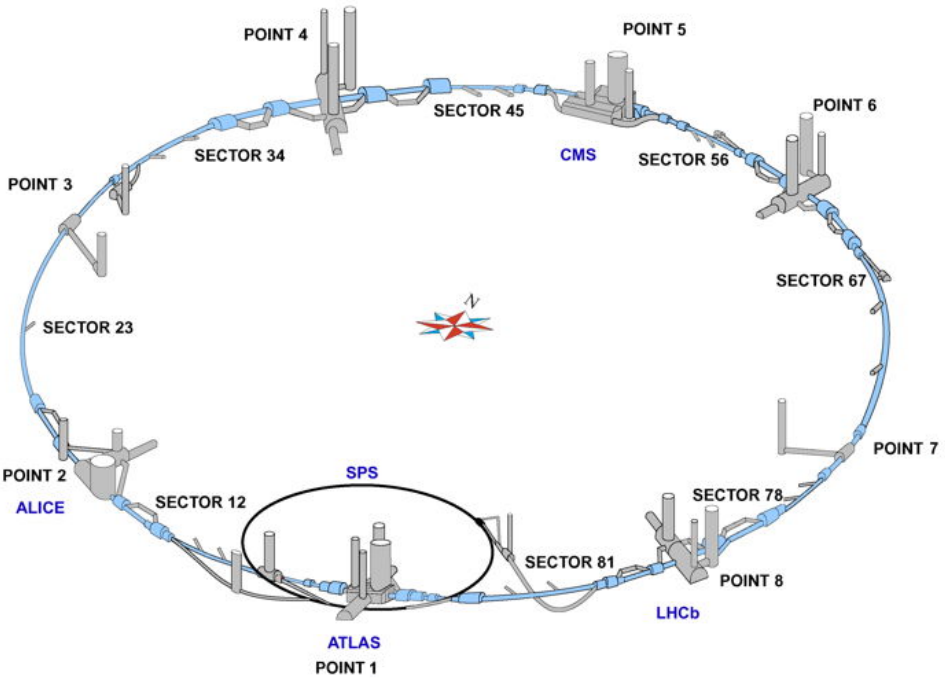


Figure 2.4: The underground structure of LHC. Edit of image from [14]

The energy loss to synchrotron radiation Φ is proportional to the cube of the Lorentz-gamma factor $\gamma = \frac{E}{m}$ and thus inversely proportional to the cube of the mass m of the accelerated charged particle: $\Phi \propto m^{-4}$. The rest mass of the proton is $\sim 1,800$ times that of an electron and as such, synchrotron radiation at proton colliders plays a negligible role. The limiting factor at the LHC is the forces required to maintain the circular motion of the beam. This is provided by superconducting dipole magnets. There are a total 1,232 dipoles around the ring, each 15 m long and producing a vertical magnetic field of 8.3 T to keep the proton beams in orbit. Another 7,903 magnets are used to focus and steer the beam. The protons are injected at an energy of 450 GeV and from there accelerated to currently 6.5 TeV before being brought to collision. The LHC design energy of 7 TeV per beam will be reached in Run III starting in 2021.

The various design parameters relevant for the further discussion is summarised in Table 2.1.

Parameter	Value
Length	26.7 km
Number of Filled Bunches	2808
Protons per Bunch	1.1×10^{11}
Beam energy at collision	7 TeV
β^*	0.55 m
Normalised emittance	3.75 μm
Nominal bunch spacing	24.95 ns
Instantaneous Luminosity	$10^{34}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$

Table 2.1: Summary of the LHC design parameters. [14]

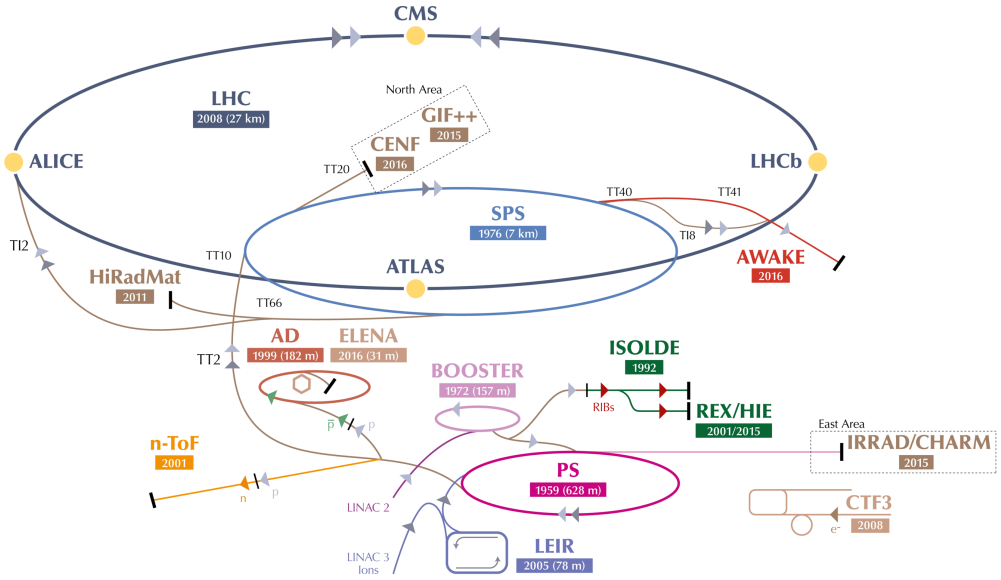


Figure 2.5: The CERN's accelerator complex. Based on [16]

2.3.1. The CERN Accelerator Complex

Figure 2.5 shows the full accelerator complex at CERN. The acceleration of protons starts at the LINear ACcelerator (LINAC) where ionised hydrogen is accelerated to an initial energy 50 MeV before reaching the Booster. The Booster accelerates the protons to 1.4 GeV before injecting them into the Proton Synchrotron (PS). The PS accelerates the protons to 25 GeV before transferring the (bunched) protons to the Super Proton Synchrotron (SPS). The SPS then accelerates the protons to 450 GeV before they are injected into either of the two rings of the LHC.

In order to maximise the number of protons in each beam, each circular accelerator acts as both a storage ring and an accelerator, with several batches of protons from the previous acceleration step being injected into the accelerator before the actual acceleration and later transfer (or collision) takes place.

2.3.2. Bunch Structure

The two proton beams circulate the LHC at a revolution frequency of $f_{\text{rev}} \sim 11.245\,\text{kHz}$ and a nominal bunch spacing of 25 ns. The harmonic number of the LHC is

$$h = \frac{f_{\text{BC}}}{f_{\text{rev}}} = 3564 \approx \frac{27\,\text{km}}{c \cdot 25\,\text{ns}}\,. \tag{2.2}$$

The bunch clock signal $f_{\text{BC}} \sim 40.08\,\text{MHz}$ is extracted from the RF cavities used to drive the beam. The harmonic number defines the total number of bunches that the beam protons get grouped into. The distribution of protons in the different bunches is not uniform, and several bunches will typically be empty. The exact fill pattern, what bunches are populated and the number of protons in each bunch, is subject to change. The fill pattern is subject to operational constraints and protocols that allow stable and safe operation of the beam [17]. Such constraints include:

- a gap of min. $3\,\mu$s (120 bunches) must exist, to allow the beam extraction kicker magnet to turn on to allow the beam to be extracted (dumped) safely. This is called the *abort gap*.
- a gap of $0.95\,\mu$s (38 bunches) between adjacent batches injected for the rise time of the injection kicker.
- an (approximate) four-fold symmetry to ensure the collision of all bunches in all four interaction points.

Each of the circular accelerators in the injection chain are subject to similar constraints. This influences the possible filling patterns. The filling pattern of the LHC is the product of the intricate interplay of the filling patterns used through out the acceleration process.

Figure 2.6 shows the standard filling scheme for LHC operation. The correspondence between the PS, SPS and finally the LHC fill patterns can

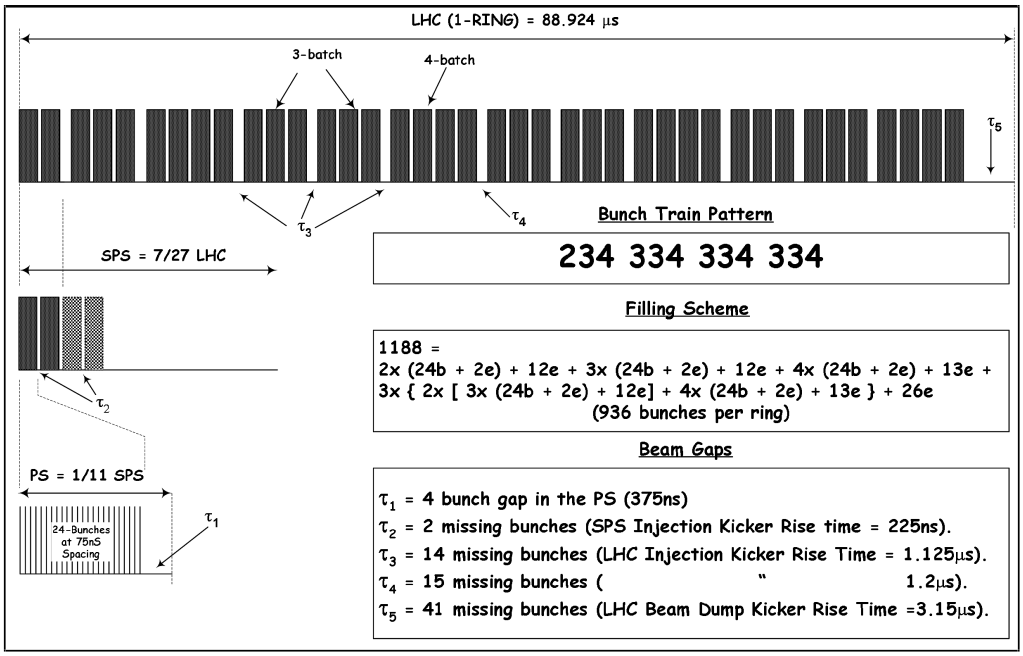


Figure 2.6: LHC Standard fill pattern for 25 ns spacing. Source: [18]

be seen on the figure on the left. Using this filling pattern, 2,808 of the 3564 bunches are filled. It should be stressed that the standard filling patterns are idealised and are not used: while they provide a baseline, other arguments affect the actual filling pattern.

2.3.3. Luminosity and Pileup

For any given fill pattern of the two LHC beams, 3564 unique bunch crossings exist. The luminosity of bunch i, $\mathcal{L}_i$, can be expressed as:

$$\mathcal{L}_i = \frac{f_{\text{rev}}\, N_i^{(1)}\, N_i^{(2)}}{4\sqrt{\epsilon_i^{(1)}\,\epsilon_i^{(2)}}\,\beta^*} \tag{2.3}$$

where $N_i^{(1)}$ and $N_i^{(2)}$ are the number of protons in each beam at the i^{th} bunch counting along a ring, and where $\epsilon_i^{(1)}$ and $\epsilon_i^{(2)}$ are the per-bunch transverse emittance of each beam and β^* is amplitude function at the interaction point. In the case where either of the beams are empty, $N_i^{(1)} = 0$ or $N_i^{(2)} = 0$, the bunch luminosity is $\mathcal{L}_i = 0$. The total luminosity is the sum of the contributions from each bunch:

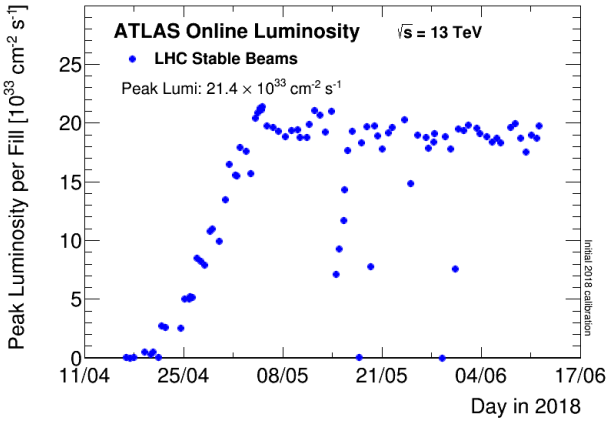


Figure 2.7: Peak instantaneous luminosity by fill in 2018. Source: [19]

$$\mathcal{L} = \sum_i \mathcal{L}_i \,. \tag{2.4}$$

The peak luminosity per fill in 2018 is shown in Figure 2.7. The highest recorded luminosity, $\mathcal{L} = 21.4 \times 10^{33}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$, is a little more than double the design luminosity of $\mathcal{L} = 10^{34}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$.

With k colliding bunches the mean interaction rate per collision is:

$$\left\langle \frac{dN_i}{dt} \right\rangle = \frac{1}{k} \sum_{j \in \{\text{filled}\}} \sigma_{\text{tot}}\, \mathcal{L}_j \,. \tag{2.5}$$

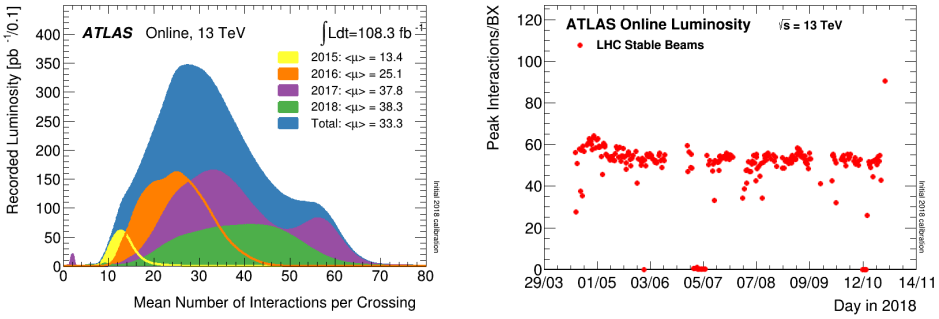
The mean number of interactions per collision μ can thus be expressed as:

$$\mu = \frac{1}{f_{\text{rev}}} \left\langle \frac{dN_i}{dt} \right\rangle = \frac{\sigma_{\text{tot}}}{k\, f_{\text{rev}}} \sum_{j \in \{\text{filled}\}} \mathcal{L}_j \,. \tag{2.6}$$

Similarly, the total luminosity can be expressed in terms of the mean number of interactions per bunch crossing:

$$\mathcal{L} = \frac{1}{\sigma_{\text{tot}}}\, \mu\, k\, f_{\text{rev}} \,. \tag{2.7}$$

Figure 2.8a shows the luminosity weighted distribution of the mean number of interactions per bunch crossing, up until the 12$^{\underline{\text{th}}}$ of November 2018. The mean number of interactions per bunch crossing can be seen to have increased from an average of $\langle\mu\rangle = 13.4$ in 2015 to an average of $\langle\mu\rangle = 38.3$



(a) (b)

Figure 2.8: Pileup distribution for the 2015-2018 runs (a) and peak pileup in the 2018 fills prior to 12$^{\underline{\text{th}}}$ of November 2018 (b) [19].

in 2018. The peak number of interactions per bunch crossing as a function of day in 2018 is shown in Figure 2.8b, and can be seen to be $\mu \sim 60$. After the high luminosity upgrade of the LHC, the luminosity is expected to exceed $\mathscr{L} = 5 \times 10^{34}\,\text{cm}^{-2}\text{s}^{-1}$ and the mean number of interactions per bunch crossings is expected to exceed 100 [20].

The experimental conditions at the LHC are harsh: a high interaction rate with many simultaneous collisions and short intervals between collisions are accepted to obtain an acceptable production rate of rare processes.

2.4. The ATLAS Detector

A Toroidal LHC ApparatuS (ATLAS) [21] is one of four large experiments at the LHC and one of two general purpose experiments: its main aim is to discover and study the properties of any (new) particles produced at the LHC. ATLAS consists of two magnet systems and a number of sub-detectors, for measuring different properties of the particles created in the collisions. By combining the measurements from the different sub-detectors the nature of the collision processes can be inferred. A two-level trigger system is used to select what collisions are to be recorded.

This section provides a quick introduction to the sub-detectors of ATLAS with special emphasis on the trigger detectors – the detectors that provide the basis for the first part of ATLAS trigger system, discussed in Chapter 3 and throughout the thesis. For a more exhaustive description of the ATLAS detector, please see [21] and [22].

The main design goals of the ATLAS detector are:

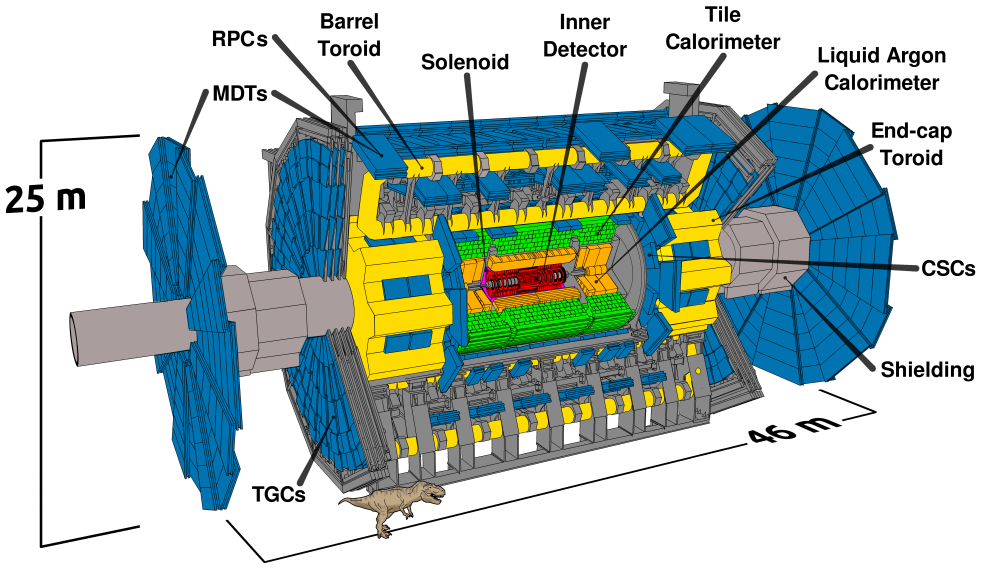


Figure 2.9: Cut-away view of the ATLAS detector with a T-Rex for scale. Source: [23]

- Efficient tracking of charged particles at high instantaneous luminosities over the full momentum range. Furthermore, tracking need to be precise enough to determine secondary vertices for particles with finite lifetimes and the collision impact parameter;
- Excellent calorimetry for identification and measurement of photons and electrons as well as a full coverage hadronic calorimeter for measurement of jet and missing energy;
- High precision muon momentum determination over a large momentum range as well as unambiguous charge determination;
- A large acceptance in (pseudo)-rapidity and full azimuthal coverage to minimise the amount of particles, escaping detection;
- High granularity in both space and time, to deal with the very high interaction rate and to suppress the effects of overlapping events.

The ATLAS detector measures 25 m in diameter and 46 m from end-cap to end-cap and is composed of several sub-detectors systems, most of which are arranged in a cylindrical geometry around the LHC beam

pipe with the interaction point at the center of the cylinder. Figure 2.9 shows a cut-away view of the ATLAS detector. ATLAS uses a right-handed coordinate system. The origin is at the nominal interaction point, the x-axis pointing towards the center of the LHC, the z-axis clock-wise tangential to the LHC, and the y-axis pointing upwards, towards the surface. Cylindrical coordinates (r, ϕ) is used in the transverse $((x, y))$ plane, with ϕ being the azimuthal angle around the beam pipe. Pseudo-rapidity η is commonly used as a substitute for the polar angle θ. The relation between the pseudo-rapidity and the polar angle is

$$\eta = -\ln\left[\tan\frac{\theta}{2}\right] . \tag{2.8}$$

The midsection of the cylindrical geometry is often referred to as the *Barrel* region, while the flat regions on either side are referred to as the *end-caps*.

2.4.1. Detector Elements

Most of the information presented in the following section is extracted from [24] and [21].

2.4.1.1. Magnet System

ATLAS has two magnet systems, a solenoid system and a toroid system. The solenoid magnet, barely visible in pink in Figure 2.9, produces a 2 T field in the direction of the z-axis in the volume occupied by the inner tracking detectors. The purpose of the solenoid field is to deflect charged particle trajectories. From the curvature the momentum and sign of charge can be determined. The toroid magnets, visible in yellow in Figure 2.9, creates an azimuthal field used to deflect muons in η direction. The integrated field strength is roughly constant in η with the exception of a drop in the transition from the barrel to the end-cap toroid magnets and is above 4 Tm for most η-ranges.

2.4.1.2. The Inner Detector

The Inner Detector consist of three parts: The pixel detector, the Semiconductor Tracker (SCT), and the Transition Radiation Tracker (TRT). The detectors are confined within a cylindrical space of radius $r \approx 1.1$ m and

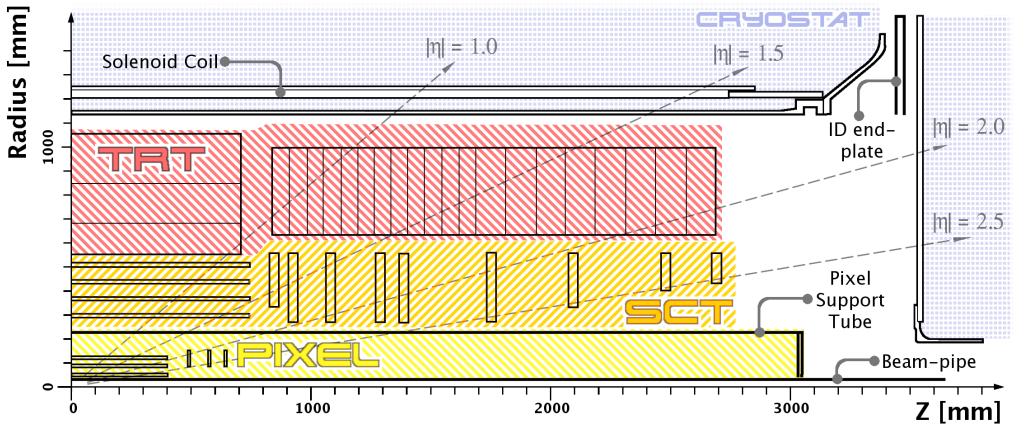


Figure 2.10: The ATLAS tracker showing the location of the pixel detector, the Silicon Tracker (SCT) and the Transition Radiation Tracker (TRT). Source: [23]

length, $l \approx 7\,\text{m}$. The layout and extent of each detector type is depicted in Figure 2.10.

Closest to the beam pipe is the semiconductor pixel detector. This detector consist of the original pixel detector as well as an additional layer, the Insertable B-Layer (IBL), that was installed during Long Shutdown I. The original pixel detector consists of 1,744 modules with 47,232 pixels with a nominal size of $50\,\mu\text{m} \times 400\,\mu\text{m}$ on each module. The IBL consists of 14 staves distributed around the beam pipe with 32 modules of 26,880 pixel cells on each module [25]. The combined detector provides $\sim 10^8$ pixels distributed over four barrel layers and 2×3 end-cap layers, covering an η range up to $|\eta| < 2.5$. The installation of the IBL required the replacement of a segment of the beam pipe. The new beam pipe is made of beryllium. Beryllium was chosen as it is light (low atomic number and density) and has properties that makes it suitable as a beam pipe, such as a long radiation length ($65.19\,\text{g}\,\text{cm}^{-2}$ [26]) as well as being non-magnetic, radiation hard, and thermally and dimensionally stable.

The small radius of the new beam-pipe ($r = 26.5\,\text{mm}$) allow the sensor elements of the IBL to be located at a mean distance of 33.25 mm from the center of the beam-pipe.

The SCT, like the pixel detector, covers up to $|\eta| < 2.5$. The SCT consists of 4,088 silicon strip detector modules – 2,112 of which make up the 4 concentric layers in the barrel and the remaining 1,976 distributed on

the 2×9 end-cap disks. Each module (in disk or barrel) provides 2 strip measurements with the strips arranged at a small stereo angle to allow the construction of space points. The SCT consist of a total of 6.3 million strips, providing a space point resolution of $17\,\mu$m in the cross-strip direction ($r\phi$) and $580\,\mu$m in the longitudinal direction (z in the barrel, r in the end-caps) – the large difference being due to the small stereo angle between the strips [27].

The TRT is a straw detector consisting of about 300,000 straw tubes each with a diameter of 4.0 mm. Each straw provides a position measurement in the bending plane with an accuracy of $\sim 100\,\mu$m. The detector extends in radius from $r_{\text{inner}} = 56\,\text{cm}$ and $r_{\text{outer}} = 107\,\text{cm}$. The TRT further provides electron identification via the transition radiation created by highly relativistic electrons passing dedicated volumes of radiator material inserted between the straws The TRT was designed to be operated with a (70 %) xenon gas mixture. Xenon was chosen for its ability to absorb transition radiation photons with a typical energy of $6 - 15\,\text{keV}$ [28]. Due to gas leakage in some parts of the detector and the high cost of xenon the affected sectors of the TRT is currently operated with an argon-based gas mixture, which can not absorb the transition radiation [29]. The affected sectors are two (out of a total of 28) straw wheels in the end-cap region – one at each side of ATLAS – as well as the two (out of three) innermost module layers in the barrel region [30].

Neither of the tracking detectors are used for the triggering at Level 1 during normal operation[2], even though the detectors themselves are relatively fast. The reason is that the detector hits do not provide any discrimination before they have been fitted to tracks. The computational complexity of constructing tracks combined with the tight latency requirement at Level 1, has so far prevented the use of track information at Level 1.

2.4.1.3. The Calorimeters

ATLAS calorimeter system, illustrated in Figure 2.11, consist of an electromagnetic calorimeter surrounded by a hadronic calorimeter. Liquid Argon (LAr) calorimeters are used for the electromagnetic calorimeter in the range $|\eta| < 3.2$ and for hadronic calorimeter in the range $1.5 < |\eta| < 4.9$. The

[2]For cosmic data-taking the TRT does provide a trigger input based on the detector occupancy.

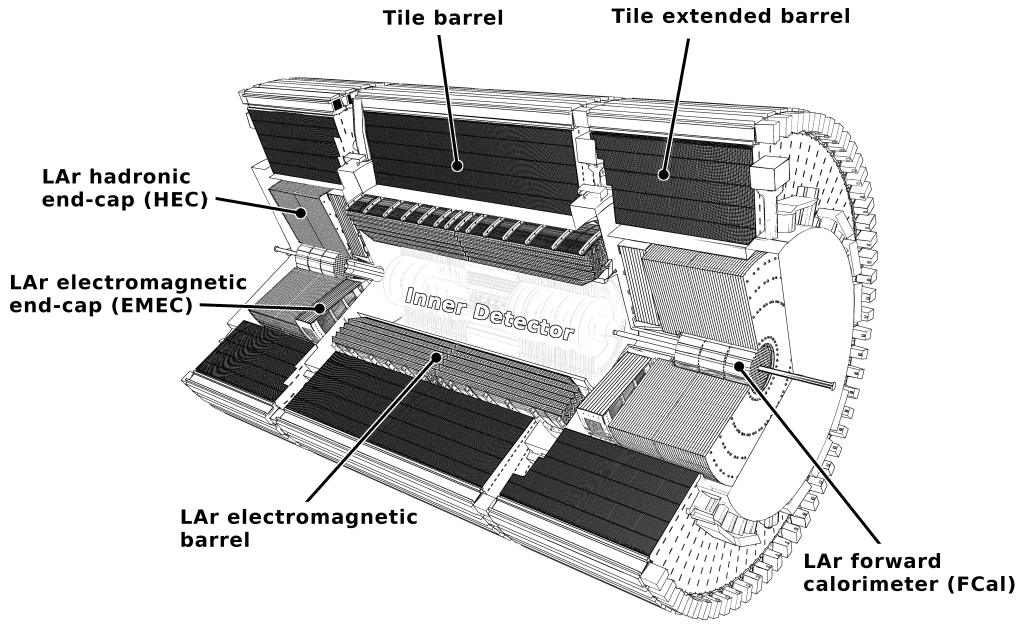


Figure 2.11: The calorimeter systems. The Liquid Argon (LAr) calorimeters are located inner most, with an electromagnetic calorimeter in the barrel region and two hadronic end-caps. The LAr calorimeters are surrounded by the tile calorimeter. The tile calorimeters has no end-cap but rather an extended barrel region. Source: [23]

LAr calorimeters, consist of a barrel and two end-caps, each in a separate cryostat. The hadronic calorimeter further consists of three segments of scintillating tile calorimeters covering the range $|\eta| < 1.7$. The material choice and design of the active elements of the LAr detectors varies with the type of calorimeter (electromagnetic or hadronic) as well as the location within ATLAS. Common for all, are alternating layers of electrodes and grounded absorber plates emerged in liquid argon with an applied high voltage field. Incident particles produce a shower of particles which in turn ionise the argon. The electrons and ions then drift, inducing a pulse at the electrode [31].

The tile calorimeter, ATLAS's hadronic calorimeter, is divided into three segments, a central barrel segment and the two "extended barrel" segments – one on each side of ATLAS – as illustrated in Figure 2.11. The three segments are each subdivided into 64 smaller modules and consist of 3 mm

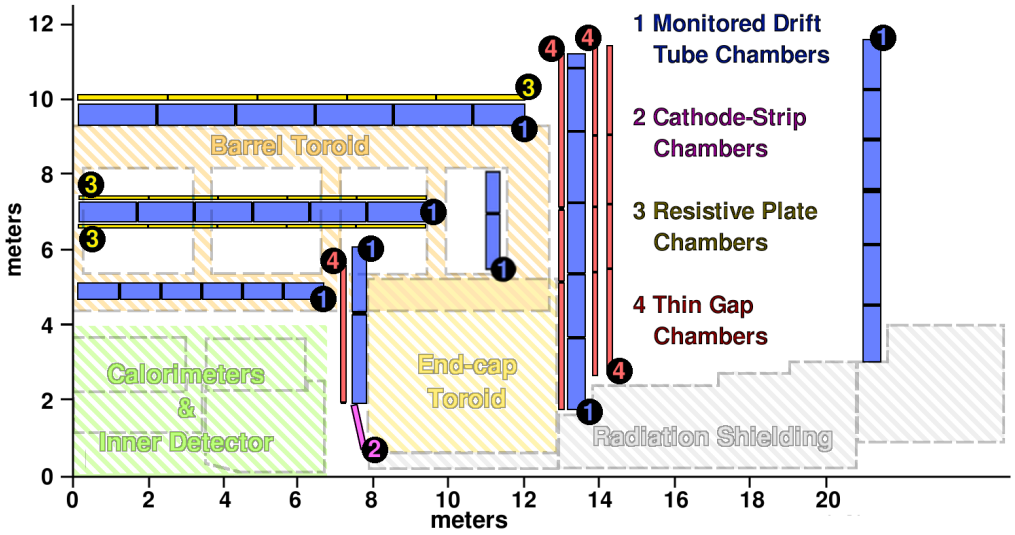


Figure 2.12: The ATLAS muon system showing the location of the MDT, CSC, RPC, and TGC sub-systems Source: [23]

layers of iron plates and scintillating plastic tiles, with a mass ratio of iron to scintilator of roughly 5 : 1. The scintillation light produced in the tiles is transferred via wavelength shifting fibers to Photo Multiplier Tubes (PMTs) for readout. The tiles of the calorimeter are grouped into roughly 5,000 cells, with two PMTs per cell. The calorimeter timing resolution per collision is ~ 1 ns for cell energies above 5 GeV.

For the LAr calorimeter, the raw ionisation signals from all cells are led out of the cryostat via 114 feed-throughs. The signals are then processed by front-end boards sitting directly at the feed-through. These feed-through boards processes and splits the signal from each cell into two: a fast analogue sum that is used for triggering, and a slower but better calibrated representation of the cell energy [31]. The same division of signals happens at the tile calorimeter that uses front-end-boards directly on the detector to split the raw signal into a "fast" signal for triggering and a "slow" signal for precision measurement [32].

2.4.1.4. The Muon System

ATLAS makes use of 4 different muon detector technologies for its muon spectrometer, Figure 2.12:

- Monitored Drift Tubes (MDT) ;

- Cathode Strip Chambers (CSC) ;
- Resistive Plate Chambers (RPC) ; and
- Thin Gas Chambers (TGC) ;

The MDT and the CSC provide precision measurements, but long read-out times makes them unsuitable as trigger detectors. The RPCs and the TGCs, are fast enough that they are suitable for triggering while they provide less precise p_{T} resolution for high-p_{T} muons. The RPCs and the TGCs still contribute to the muon measurements by providing additional space points to the muon tracks.

The spectrometer is designed to measure the momentum of charged particles leaving the calorimeter in the range $|\eta| \leq 2.7$. The calorimeters act as an effective absorber in front of the muon system with very little hadronic punch-through into the muon spectrometer. This ensures a clean signal in the muon spectrometer.

The MDT consist of 1,150 chambers containing a total of 354,000 pressurised drift tubes with a diameter of 3 cm. All the drift tubes in a MDT chamber are parallel. This implies that a MDT chamber can only provide a one dimensional measurement. The MDT chambers in ATLAS are aligned perpendicular to the bending plane of muons and thus provide a measurement of the η-component of the track. The MDT chambers are distributed such that each muon will pass through three MDT chambers. The MDT allows the determination of the sagitta[3] of a muon track with an accuracy of 60 μm corresponding to a momentum resolution of 10% at $p_{\mathrm{T}} = 1\,\mathrm{TeV}$.

The CSC consists of multiwire proportional chambers with the cathode plane segmented into strips. The CSC chambers are used in the very forward region, $2 < |\eta| < 2.7$, and are installed in what is commonly referred to as the "small wheel". Each ATLAS end-cap has 16 chambers, each with 4 precision layers with 192 strips in each layer and 4 coarse layers with 48 strips in each layer. The layers are orthogonal and oriented such that the highest precision is achieved on the η direction, yielding a hit position resolution of 60 μm in η and 5 mm resolution in $r \cdot \phi$. The CSC is used in the forward region due to the high rate capabilities of the detector, 1000 Hz/ cm compared to $\sim$ 150 Hz/ cm for the MDT.

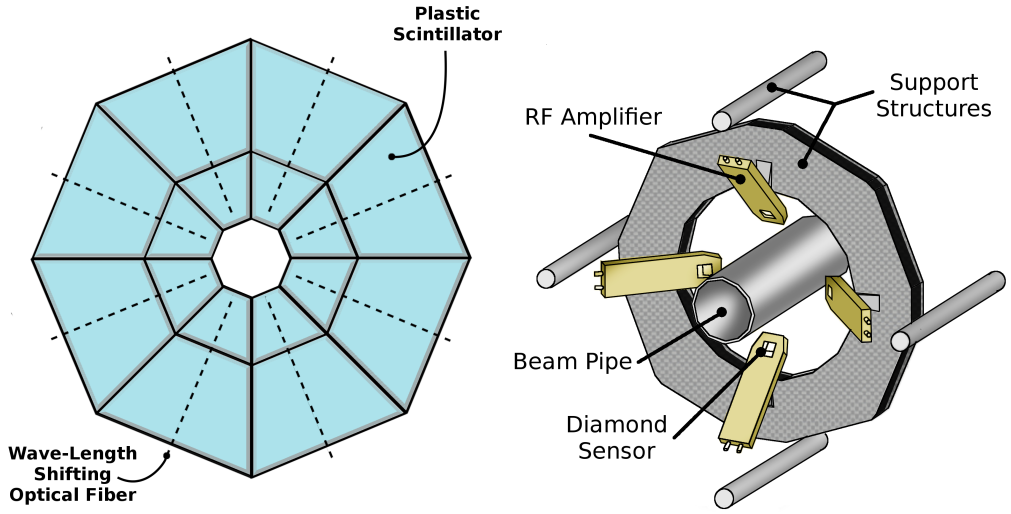
[3]The sagitta is defined as the height of an arc as measured perpendicular from the midpoint of the arc's chord to the arc itself.

The RPCs provides a fast muon signal in the barrel region. The gas volume in an RPC panel is 2 mm thick and sandwiched between two restive plates, each of 2 mm thickness. The size of RPC elements varies from $\sim 1.3\,\mathrm{m}^2$ to $\sim 2.3\,\mathrm{m}^2$ depending on their position in the detector. High voltage is applied between the two plates to cause electrons to drift and create and avalanche in case an incident particle causes ionisation of the gas. The avalanche signal is picked up by readout strips on the backside of the resistive plates. The strips on the plates have a typical width of $\sim 3\,\mathrm{cm}$. The orientation of the strips on the plates are orthogonal to allow for a space point measurement. There are $\sim 8,000$ RPC modules in ATLAS.

The TGC provides a fast muon signal in the end-caps using multiwire proportional chambers. They are designed and oriented in a manner that ensure that the drift time is short enough that they can be used for triggering. There are two types of TGC modules: a doublet and a triplet chamber. Each chamber in a module consists of a gas-volume with between 6 and 31 anode wires. The number of wires varies as a function of η to meet the required momentum resolution. Pick-up strips oriented orthogonal to the anode wires provide the second coordinate. The gas chambers are separated by a honeycomb structure. The chambers within each module are oriented such that the wires of all chambers are parallel. There are 3,600 TGCs in ATLAS.

The muon system implements a good deal of trigger logic on a sector to sector basis, based on coincidence between stations as detailed in [33]. This greatly helps to suppress fakes from the cavern background and reduces the amount of signal processing needed by the Level 1 trigger. The muon detectors are grouped into 208 sectors – 64 for the barrel, 96 for the end-cap and 48 for the forward region. Each sector provides the Level 1 trigger with $(\eta, \phi, p_{\mathrm{T}})$ information for up to two muon candidates per sector with priority being given to high-p_{T} muons. The p_{T} values are discretised into 6 programmable thresholds, typically three low-p_{T} thresholds in the approximate range $6\,\mathrm{GeV} < p_{\mathrm{T}} < 9\,\mathrm{GeV}$ and three high-p_{T} thresholds in the approximate the range $9\,\mathrm{GeV} < p_{\mathrm{T}} < 35\,\mathrm{GeV}$.

A couple of Micro-Mesh Gaseous Structure (MicroMegas) elements were installed in the small wheel during Long Shutdown I to allow parasitic data-taking, test, and commissioning [34]. As MicroMegas is currently not used for data-taking with ATLAS its relevance for this thesis is limited to



(a) (b)

Figure 2.13: Illustrations of an MBTS disk (left) and a BCM module (right). Source Source: [23].

its integration with the central trigger system as a new part detector in ATLAS.

2.4.2. Forward Detectors

A couple of other detectors deserve mentioning here as they provide input for the Central Trigger Processor (CTP) and plays a role in the later discussions of timing and luminosity.

2.4.2.1. BPTX

The Beam Pickup (BPTX) [35] are a set of two electrostatic beam pickup stations – one per beam pipe – located at a distance of 175 m from the interaction point along the LHC beam pipe. The BPTX are located "in front" of ATLAS and sees the incoming beam. The BPTX are provided and operated by the LHC beam instrumentation group and are used by ATLAS as beam monitor and for timing purposes, as will be discussed in Section 4.2.3. The BPTX signals are used to extract per-bunch timing signals which are compared to the LHC bunch clock. The timing resolution provided by the BPTX allows the measurement of the phase between the bunch clock and the bunch encounter at the interaction point with an uncertainty of less

than 30 ps [36]. This in turn allows ATLAS to correct the LHC-provided bunch clock. The signals from the BPTX are, after discrimination, also used to provide a Level 1 trigger input when bunches pass through ATLAS.

2.4.2.2. *MBTS*

The Minimum Bias Trigger Scintillators (MBTS) consist of two scintilator disks, located at $|z| \approx 3.6\,\text{m}$. Each disk contains an inner ring and an outer ring, both consisting of 2 cm thick scintilator material, as illustrated in Figure 2.13a. The outer ring cover the range $2.08 < |\eta| < 2.78$ while the inner ring covers the range $2.78 < |\eta| < 3.75$. Wavelength shifting fibers direct the emitted light radially outwards to eight PMTs per disk. The MBTS is used as input for the trigger system for minimum bias triggering and timing.

2.4.2.3. *BCM*

The Beam Conditions Monitor (BCM) is a diamond detector consisting of two stations at $|z| = 184\,\text{cm}$ and $r = 55\,\text{mm}$, equivalent to $|\eta| \sim 4.2$. Each station is equipped with 4 modules as illustrated in Figure 2.13b. The stations are mounted on the supporting frame of the original pixel detector. The main purpose of the detector is to protect the silicon-based trackers from beam incidents [37]. The fast response time, $\mathcal{O}\,(\,\text{ns})$, also makes it suitable as minimum bias and timing trigger. The detector can be used to discriminate between collisions, beam background events and spray particles from beam incidents. The detector can also be used for luminosity measurements, but is, due to a bunch-to-bunch effect, only used for cross-checks or during fills with few filled bunches. The information from each sensor module is delivered as input to the Level 1 trigger on a hit-or-miss basis [38].

2.4.2.4. *ALFA*

Absolute Luminosity For ATLAS (ALFA) is used to measure the beam profile for luminosity and to detect deflected protons from elastic collisions during special runs for forward physics[4]. During these runs, ALFA is used

[4]These runs require a very different beam optics. β^* is typically $\mathcal{O}\,(1\,\text{km})$ during these runs, compared to $\sim 0.4\,\text{m}$. This implies very parallel beams with little squeezing (or focusing) of the beams at the interaction point, consequently implying low $\mathcal{L}$ and low μ.

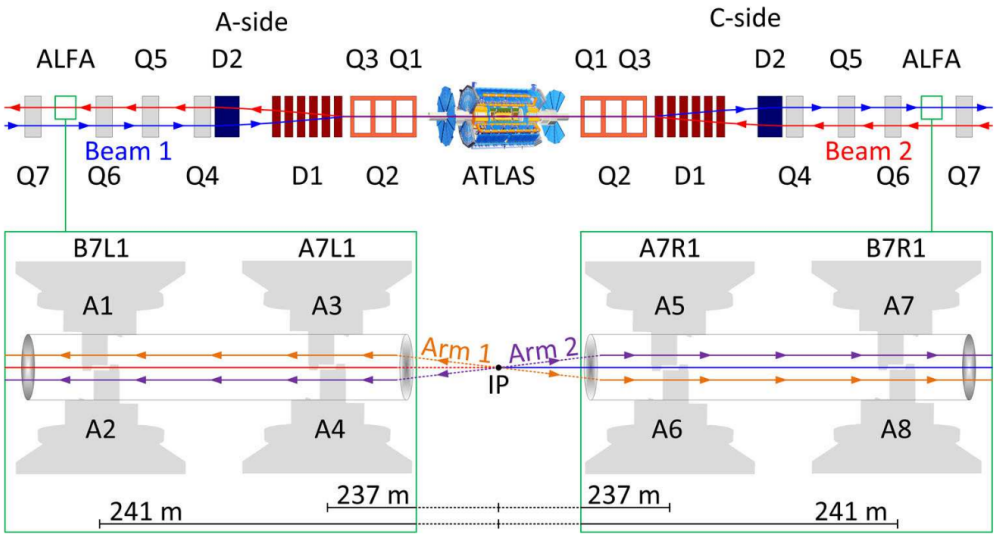


Figure 2.14: Illustration of ALFA's roman pots and the detectors location w.r.t. ATLAS. Source: [39]

as a trigger detector for ATLAS. ALFA consists of two Roman Pot stations integrated as part of the LHC beam pipe $\sim 240\,\text{m}$ from the interaction point on both sides of ATLAS as illustrated in Figure 2.14. The Roman Pots are allowed to move within the LHC beam pipe and each carries multi-layer scintillation-based tracking detectors. The scintillation signals are amplified and extracted using multi-channel PMTs. The trigger signals are created on each station by requiring coincidence in two neighbouring roman pots. The trigger signals are transmitted back to ATLAS, using air-core cables to minimise the latency [40].

2.4.3. Particle Identification

The following is a simplified explanation of particle identification in ATLAS, but serves the purpose for later discussion about consequences of imperfect information at Level 1.

Figure 2.15 depicts a cross-section slice of the ATLAS detector in the (x, y)-plane. The typical signatures of common particle types have been overlaid. Inner-most are the tracking detectors, subject to the solenoid field. The tracking detectors register small energy deposits (so-called *hits*) caused by charged particles passing through the active elements of the detectors. From the hits, tracks and their curvature can be fitted. The curvature of

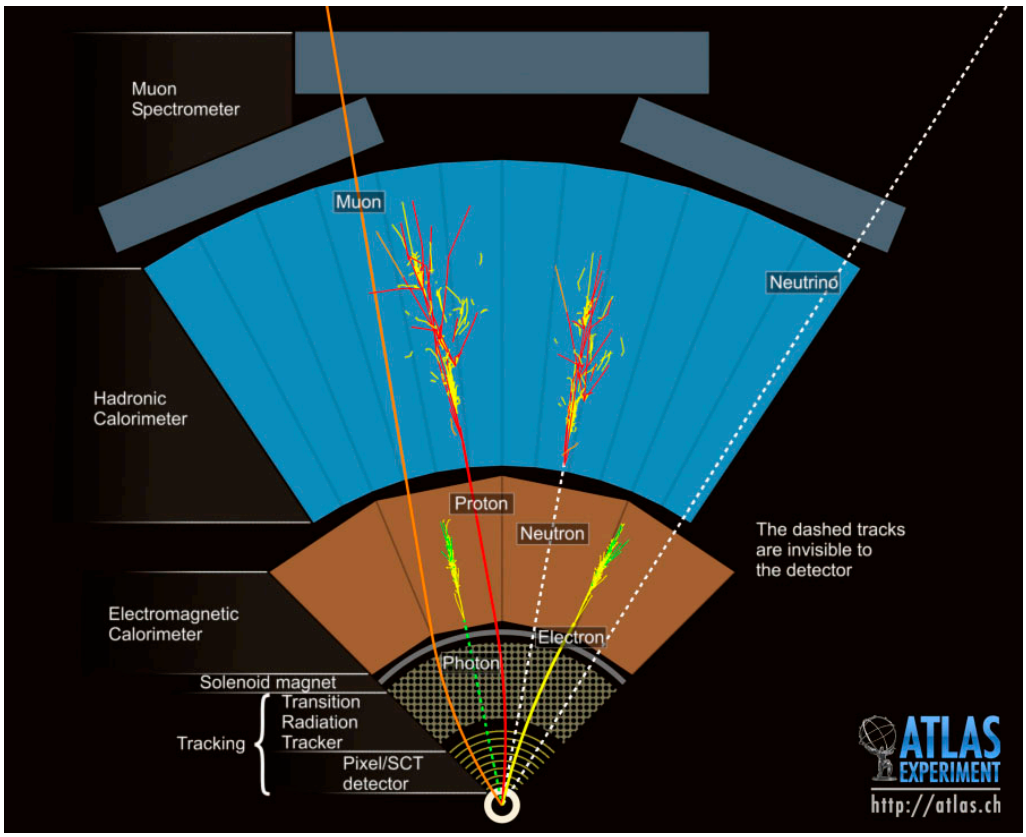


Figure 2.15: Illustration of particle Identification in ATLAS [41]

the track is used to determine the momentum of the particle and the sign of the electrical charge. Outside the tracking detectors are the electromagnetic and hadronic calorimeters. The calorimeters stop the particles, causing a cascade of new particles to be created, which again will interact in the calorimeter. From the amount of calorimeter activity, the energy of the incident particle is inferred. This information is combined with the tracking information to determine the electric charge of the particle e.g., the electron and photon are both stopped by the electromagnetic calorimeter, however only the electron has a charge. Thus, a track leading to an energy cluster would imply an electron while an energy cluster without an associated track would imply a photon. Muons are identified by matching hits in the muon spectrometer to tracks in the inner detector. As for the electrons, the curvature of the muon tracks are used to determine the momentum and sign of the electric charge of the muon. Neutrinos escape detection and are inferred based on missing (transverse) energy.

Chapter 3

Data Taking with ATLAS

The experimental conditions at the LHC are harsh, with a high interaction rate and signal-to-background-ratio of $1 : 10^{10}$ or worse. In order to bring the readout rate to an acceptable level and at the same time improve the signal quality of the recorded data set a sophisticated selective trigger system is needed.

This chapter motivates and describes ATLAS trigger system, the baseline trigger strategy and the data flow through the trigger system with particular emphasis on the first trigger level.

3.1. The Need for a Trigger

That the physics processes of interest are rare and that harsh experimental conditions are accepted to ensure an acceptable production rate of rare processes does not in itself imply the need for a trigger system with online data filtering capabilities: one could, at least in theory, record all the data and then later filter through the data without the worries of selecting the interesting events quickly, within the latency of the trigger system. However, the overall data rate produced by ATLAS, as a result of the high interaction rate, poses practical challenges that make it infeasible to store the full data set. The two main challenges are:

- Transferring data from the detector front-ends to the back-end and later permanent storage.
- Analysing and making use of the data after they have been recorded.

In this section, via a back-of-the-envelope calculation these points are made clear. The section further motivates the use of a two-level hybrid

trigger system consisting of a hardware based first level and a software based second level.

3.1.1. Data Volume Considerations

The following is a back-of-the-envelope calculation that aim at illustrating the storage and computational requirements needed if all of ATLAS data was to be stored.

With a mean number of events per-bunch crossing $\mu = 30$ and an estimated average event size of 1.5 MB after zero-suppression and compression[1], the ATLAS output data rate would be $r_{\text{ATLAS}} = 60\,\text{TB/s}$. From April 2018 to June 2018, ATLAS has recorded a data set corresponding to $21.9\,\text{fb}^{-1}$ or 25 days of constant uninterrupted data-taking at design luminosity. At the full data rate of 50 TB/s the size of the recorded data set would have exceeded 10^8 TB. Transferring, or simply reading a data set this size, once over an ideal 10 Gbit/s line would take a couple of thousands of years. Processing or analysing the data set of more than 10^{15} events, assuming that one could analyse 10,000 events per second, would similarly take a couple of thousands of years

While some of this could perhaps be compensated for by parallelisation of transfer and processing as well as by various technological improvements over time, given the very limited interest from an analysis point of view, in the majority of the data produced, it should be clear that this is not a feasible approach.

3.1.2. The Two-Level Trigger Model

Based on the considerations outlined above, it is not possible for a modern computer (network) to handle the raw amounts of data produced by ATLAS or to transfer the data from the detector front-ends. This implies that before data leaves the detector system the event and data rate need to be reduced. However, doing the full data reduction directly in hardware is not feasible. Many detectors do not provide readout or processing fast enough that their data can be used for triggering with 25 ns between collisions. While some detector technologies are inherently slower than others a much bigger problem arises where computationally heavy processing is needed

[1] In reality the event size dependant on the pileup.

to make sense of the detector data. One such example is the tracking detectors where (potentially millions of) "hits" need to be converted into tracks. While there are plans for a future hardware-based track finder for triggering [42], the example serves to illustrate the case in point: the rate reduction performed in hardware will, due to limited buffer sizes and large data rates, be bound to be made on imperfect grounds – for instance without any track information.

Instead of doing the full data reduction in hardware, "just-enough" data reduction and event filtering is done in hardware that a modern computer (network) can transfer and process the data in near real-time. A conventional server farm is used as the second trigger layer. The event is fully reconstructed on the farm using the full event data, after which the selection algorithms are run to make a final decision on what events to store.

3.2. ATLAS Trigger Overview

ATLAS' two level trigger system consists of the Level 1 trigger and the High Level Trigger (HLT). Events are accepted only if they are accepted by both of these levels. Each trigger level uses a set of selection criteria, applied in parallel, to determine if an event is accepted. For the Level 1 trigger, the selection criteria are referred to as *trigger items* and at the HLT the selection criteria are referred to as *trigger algorithms*. At either level an event is accepted if one of the selection criteria are met. A flow chart is shown in Figure 3.1. The Level 1 trigger provides a factor 400 rejection, from 40 MHz to around 100 kHz. The HLT provides rejection of a factor 100, resulting in a total accept rate of ~ 1 kHz.

From the detector data, Physics Object Candidates (POC) are produced. Here a POC can be either a candidate particle, e.g., a muon, an assembly of particles, e.g., a jet, or an event level quantity, e.g., the total energy. The selection criteria of each trigger item or algorithm are created as the logical combination of one or more requirements on the multiplicity (where applicable) of POCs passing adjustable momentum and energy thresholds outlined in Section 3.3.5. An example trigger item (or algorithm) could be created as the requirement of having "two photons with a transverse momentum larger than 8 GeV AND more than 20 GeV of missing energy". As there is limited time for processing at Level 1, and as only partial information is available, less (sophisticated) POCs can be constructed

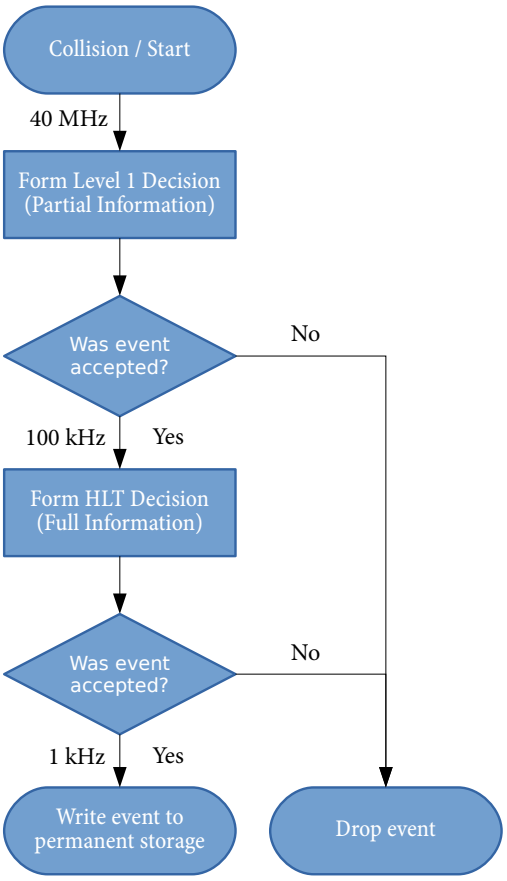


Figure 3.1: ATLAS trigger decision flowchart.

at Level 1 as compared to HLT. The available physics quantities at Level 1 and HLT are discussed further in Section 3.4.3 and 3.6.1, respectively.

The Level 1 trigger is pipelined and synchronous and operates at a fixed latency. That is, the processing of the Level 1 trigger is synchronous with the LHC bunch clock and at every clock tick the event data is pushed one processing step further. This allows the Level 1 trigger decision to be made in constant time and allows the trigger decision to be distributed to the sub-detectors at a fixed latency. This is important as the readout of the detector front-ends is based on timing and relies on a fixed latency, as will be discussed further in Section 3.5.1.

The HLT consists of a conventional server farm. The processing at the HLT is asynchronous, identifier based, and operates at varying latency. The HLT receives information about the Level 1 trigger and assign the HLT

processing to a node in the farm. The node requests event data from the Data Acquisition (DAQ) system based on an identifier it receives as part of the Level 1 trigger information. After receiving and reconstructing the event data, trigger algorithms are executed in parallel. The difference in running time and complexity of different algorithms, as well as the variable number of algorithms to be run in order to form the HLT decision results in a variable trigger latency at the HLT.

To reduce the processing time, the trigger algorithms run at the HLT are seeded by one or more Level 1 trigger items: instead of running all algorithms for all events a selected number of algorithms are run. I.e., if an event was accepted at Level 1 because it contained high p_{T} muons, a muon-based trigger algorithm is likely a better (first) choice at the HLT as compared to a calorimeter-based algorithm.

3.3. Trigger Strategy

The use of a trigger system calls for a trigger strategy. Only recording some collision events raises the question of utilisation of the available bandwidth. The trigger strategy firstly deals with a) how to select events of interest for physics analyses. Secondly, as only a small fraction of the collision events are stored, part of the trigger bandwidth must further be allocated to b) triggers that allow for a study of the inherent bias or skew of the recorded data sample that is introduced by the various selection criteria, and c) triggers that can be used to obtain the normalisation of the recorded data set to the full data set.

This section outlines the baseline trigger strategy for these three types of triggers before discussing how trigger rate and bandwidth is controlled and regulated in the trigger menu.

3.3.1. Triggering for Physics

From a physics perspective, the goal is to keep as many of the rare and interesting events as possible while rejecting as many of the common, and (hopefully) well understood, events. Recalling that particle spectra are well described, and well studied, in the energy range up to a couple of 100 GeV, new physics particles (if they exist) are expected in the mass scale > 100 GeV. The decay products of such particles are likely to carry

high momentum. As the total longitudinal momentum of the collision system is not known from from collision to collision – particles are selected based on their transverse momentum p_{T} as conservation of momentum can only be observed in the transverse plane. The baseline trigger strategy is to select collision events with *at least* one (or two) high-p_{T} lepton(s), photon(s) or jet(s). This selection provides a relatively high background rejection without making model dependant assumptions. The majority of the total bandwidth for physics triggers, $\sim 68\%$ in 2016 [43], is allocated to these types of triggers, with the majority ($\sim 90\%$) of the bandwidth being allocated to single lepton or jet triggers.

Many theories for new physics introduce new particles that avoid detection, resulting in an energy imbalance, commonly referred to as *missing energy*. Selecting events with missing energy above a certain threshold is a strategy for selecting such new physics events while effectively suppressing most SM events, except those containing neutrinos. As for the momentum, the longitudinal energy of the collision system from collision to collision can not be determined and the transverse energy E_{T} and the transverse missing energy $\not{E}_{\mathrm{T}}$ is used instead[2]. There is a caveat to using $\not{E}_{\mathrm{T}}$ however: $\not{E}_{\mathrm{T}}$ is an event level quantity calculated as the negative vector sum of the energy carried by each detected particle. This implies that ideally full event level information, particle identification, and calibration is needed to obtain a good estimate of $\not{E}_{\mathrm{T}}$. Further, as $\not{E}_{\mathrm{T}}$ is calculated as a sum, it is not possible to determine whether the observed missing energy was caused by one or more particles avoiding detection.

In 2016 $\sim$ 15% of the total bandwidth for physics triggers were $\not{E}_{\mathrm{T}}$ triggers. Around 12% of the bandwidth in 2016 was allocated to items for b-jets and b-physics as detailed in [43]. The selection criteria for these triggers are slightly more complicated and outside the scope of this discussion.

3.3.2. Support Triggers

The supporting triggers are similar in selection to the primary physics triggers and are used for studies of the selection efficiency of the primary triggers. The typical bandwidth allocation per support trigger is $\sim 0.5\,\mathrm{Hz}$ com-

[2]The two differ by more than a sign as E_{T} is calculated as the scalar sum of energy deposits while $\not{E}_{\mathrm{T}}$ is calculated as the (negative) vector sum of energy deposits.

pared to the typical allocation of $\sim 130\,\mathrm{Hz}/\,\mathrm{item}$ for the primary physics triggers [43].

The selection efficiency of a single lepton trigger, typically depends on the p_{T} of the candidate lepton. Further, due to varying p_{T} resolution, or detector geometry, the trigger efficiency usually has a dependence on (η, ϕ).

3.3.3. Normalisation, Minimum Bias and Zero Bias

Using one or more triggers to record a data sample implies that the recorded data sample is skewed or biased based on the selection criteria used. Understanding this bias is paramount for making sense of the recorded data. Part of the trigger bandwidth is dedicated to Min(imum)-Bias triggers and Zero-Bias triggers. Min-Bias triggers are typically based on the mere observation that a collision took place, but do not impose any event-level selection criteria. Zero-bias triggers are based on the knowledge that a collision *might* take place: exploiting that the bunch pattern is periodic, a zero bias trigger was created from the same signal that caused the min-bias trigger (or any other trigger), by delaying the signal with a full period. Another example is by combining the knowledge of the colliding bunches with a random number generator.

3.3.4. Triggering and Timing

As timing is of importance for later discussion, it should briefly be mentioned that triggers are also used as a method of achieving synchronisation between detectors: not all detectors have a response time that is fast enough that direct bunch crossing identification is possible, but by triggering using a fast detector and correlating data, the offset can be identified.

3.3.5. Thresholds, Prescale and Rate

As bandwidth is limited it is important to understand how trigger rate can be controlled and at what price. The following discussion focuses on the Level 1 trigger though most concepts are easily extended to the HLT.

The raw trigger rate R^T for a trigger T can be described as[3]:

$$R^T = \sigma^T \mathcal{L}\,, \tag{3.1}$$

[3]For triggers based on event level quantities, such as $\not{E}_{\mathrm{T}}$ triggers, this only holds to first order.

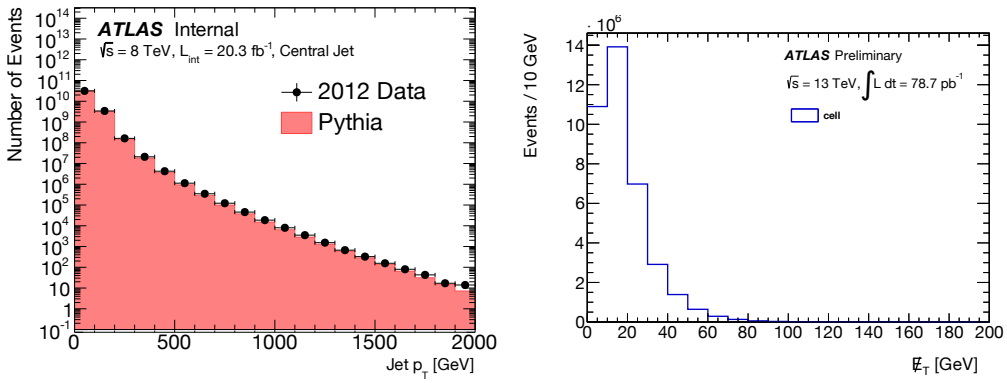


Figure 3.2: Inclusive distributions of transverse jet momentum [44] (left) and event level missing energy (right) using a calorimeter cell based algorithm [45].

where σ^T is the "trigger cross section" and $\mathcal{L}$ is the luminosity. The trigger cross section can be viewed as the convolution of a selection function with the cross section (times branching fraction) of any process producing particles that the trigger is sensitive to. The selection function is a function of overall detector acceptance as well as the trigger specific parameters such as object multiplicity, p_{T} or E_{T} thresholds, and detectors isolation requirements. In addition to thresholds and isolation requirements, prescaling can be applied to each trigger to control the rate. ATLAS uses pseudo-random prescaling to select M in N triggers. This results in an trigger rate after prescale, R^T_{TAP} of the trigger T, for a given prescale $p^T = \frac{M}{N}$:

$$R^T_{\mathrm{TAP}} = \frac{1}{p^T}\,\sigma^T\,\mathcal{L}\,. \tag{3.2}$$

A trigger is said to be unprescaled if $p^T = 1$.

The ATLAS trigger menu consist of several different triggers with different thresholds for each trigger type. Most of the Level 1 bandwidth is dedicated to unprescaled single object triggers with as low thresholds as possible. This is done to obtain the most inclusive sample suitable for analyses. The second biggest contribution are di-object triggers (e-e, e-μ, μ-$\not{E}_{\mathrm{T}}$ etc.) that provides additional rate reduction for certain signatures. Triggers of the same type but with different p_{T} or E_{T} thresholds will have different trigger rate and different signal purity. The rate reduction and the signal purity can be gauged from the inclusive distributions. Figure 2.2

shows the inclusive distribution for electrons and muons while the inclusive distribution for jets and $\not{E}_{\mathrm{T}}$ is shown in Figure 3.2. All the distributions can be seen to drop of rapidly with p_{T} or E_{T}, respectively. For the lepton distributions, a large background discrimination can be achieved as the QCD background does not abundantly produce particles mimicking those of electroweak decays. However, this does not hold for hadronic objects such as jet and τ candidates. For these types of objects, the threshold primarily provides an effective handle on the rate.

Suffix	**POC**	**Range** [GeV]	**Level 1 Thresholds** [GeV]	**Lowest Unprescaled** [GeV]
MU	μ	4-21	4,6,10,11,20,21	20
EM	e/γ	3-22	3,7,8,12,15,20,22,24	20
TAU	τ	8-100	8,12,20,30,40,60,100	100
J	jet	12-120	12,15,20,25,30,40,50,75,85,100,120	100
XE	$\not{E}_{\mathrm{T}}$	10-300	10,30,35,40,45,50,55,60,70,80,300	50

Table 3.1: Summary of trigger thresholds for common Level 1 trigger items for Run II as well as the lowest unprescaled trigger threshold for single object triggers (2016).

Table 3.1 shows examples of typical thresholds used by ATLAS Level 1 trigger. The lowest unprescaled trigger threshold is the lowest threshold that is used for a single object physics trigger: to keep the recorded data sample as inclusive as possible, it is desirable to use as low a threshold as possible while staying inside the bandwidth limitation. For multi-object triggers, lower thresholds can be used, reflecting the lower probability of having events with multiple objects each fulfilling a different threshold. An example is the (un-prescaled) J/Ψ b-physics trigger that requires at least three muon above the 4 GeV threshold, while the lowest acceptable threshold for a single muon trigger is 20 GeV. For both single and multi-object physics triggers, it is preferred to choose the threshold(s) such that the physics triggers can run unprescaled.

For the non-physics triggers, lower thresholds than are used for the physics triggers are often desired. At the same time, these triggers only receive a small fraction of the allocated bandwidth. Prescaling is commonly used with non-physics triggers for this purpose to reduce the rate while permitting lower thresholds to be used.

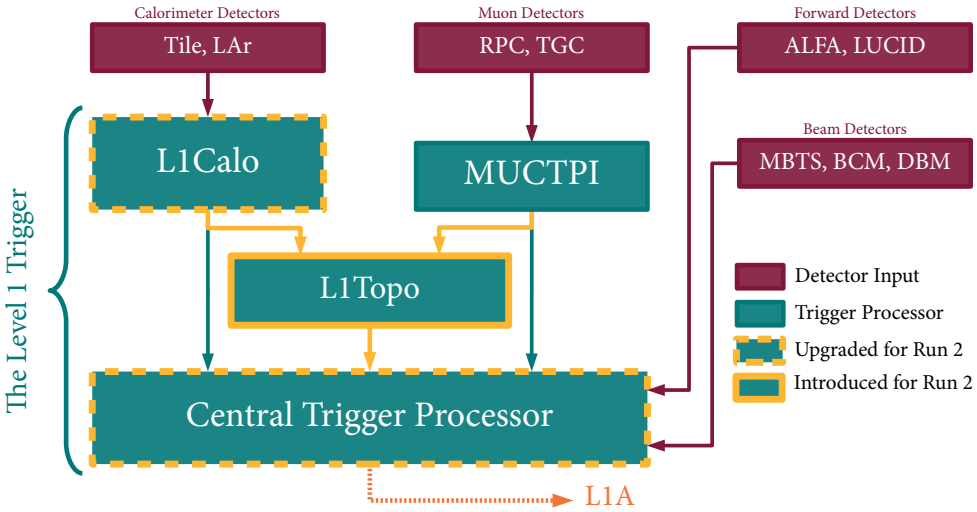


Figure 3.3: ATLAS Level 1 Trigger Processors

The lowest trigger threshold for unprescaled single object triggers shown in Table 3.1, can be seen to be lower for electromagnetic objects and muons than for the hadronic objects (taus and jets). Given that the background is primarily QCD, this primarily reflects the poor background rejection achievable for hadronic objects. Better discrimination against the SM background is usually achieved by combining the jet trigger requirement with other trigger requirements, such as $\not{E}_{\mathrm{T}}$ triggers or other jet trigger requirements, to form multi-jet triggers.

3.4. Level 1 Trigger

An overview of the Level 1 trigger is shown in Figure 3.3. The Level 1 trigger consists of four trigger processors. Two of these, the Muon-to-CTP Interface (MUCTPI) and L1Calo receives data from the trigger detectors, see below. The MUCTPI [46], and the calorimeter processor, L1Calo [47], aggregate the data from the muon and calorimeter trigger detectors, respectively, and produce POC summaries that are pushed to the CTP. The L1Topo, which was installed for Run II, provides input to the CTP based on geometric or kinematic association between the POC created by the L1Calo and/or the MUCTPI [48]. Based on the received inputs, the CTP applies its trigger logic and forms the trigger decision. If the event is accepted, the CTP issues a Level 1 Accept (L1A) and distributes it to all sub-detectors to initiate the

readout procedure. The trigger processors and the POCs will be discussed in more detail in Section 3.4.2.

At each bunch crossing, all the detectors of ATLAS sample their data and store it in a memory buffer on the detectors front-end electronics, awaiting the L1A. The depth of these buffers determine the time available for the Level 1 trigger to receive and process the data from the trigger detectors, form the trigger decision, and transmit the L1A to all detectors. The maximum latency of ATLAS Level 1 trigger, is 100 BCs or 2.5 μs.

3.4.1. The Trigger Detectors

Due to the latency constraint, only the detectors supporting a fast readout can be used for triggering. For the calorimeters, the data from the fast readout is a subset of the full readout.

The muon trigger detectors are the RPC in the barrel region and the TGC in the end-cap regions. Most of the signal processing of the muon trigger signals happen in the muon systems: the muon trigger system groups the muon trigger detectors into 208 trigger sectors - 64 barrel sectors and 144 for the forward and end-cap regions. Within a sector, coincidences of hits in different detectors are used to identify muon candidates. The muon trigger electronics classifies muon candidates according to six programmable p_{T} thresholds. Each sector provides information to the MUCTPI about up to two muon candidates, where preference is given to high-p_{T} candidates.

The calorimeter trigger detectors are the tile and LAr calorimeters. Unlike the muon trigger, most of the signal processing happens at the Level 1 Calorimeter trigger processor: L1Calo. The information provided by the tile and LAr calorimeter is the (raw) energy sums of trigger towers formed by analogue summation directly in the on-detector electronics. The trigger towers are 0.1×0.1 in $\Delta\eta, \Delta\phi$ for $|\eta| < 2.4$ and larger in the end-cap and forward region and cover the full depth of each of the detectors. For the Electromagnetic (EM) calorimeter trigger tower sums, the electronics used to produce the energy sum, converts the raw energy of the signal to transverse energy while for the hadronic trigger tower signals are simply summed and transmitted using the raw energy scale [49].

A number of other detectors provide trigger inputs to the CTP for more specialised triggers that are not used for the primary data-taking:

TRT : The TRT implements a high-occupancy trigger, referred to as "Fast OR". The trigger input is used for cosmic data-taking [50].

ALFA : ALFA provides a number of triggers for their forward physics program. The trigger inputs are used during special runs for forward physics and during van der Meer scan-scans.

BCM and BPTX : The BCM and BPTX provide beam(-timing) related input to the CTP. The inputs are particularly useful for the timing-in of ATLAS after longer shut-downs or major interventions as they provide sub-bunch-crossing timing resolution and sensitivity to even small bunch currents. The trigger inputs are generally used for minimum bias triggers.

MBTS : The MBTS is primarily used for triggering during special runs, e.g., beam splashes for timing, first attempts at new center of mass energies, van der Meer scan, low-μ runs, forward physics, etc. During these runs, ATLAS needs to trigger on collisions with the highest possible efficiency and with a minimum bias. In 2015 and 2016, the MBTS provided 2×12 trigger inputs to the CTP – one per channel per side. In the CTP, the inputs can be used individually and combined, allowing triggering on either side individually or on both in combination.

3.4.2. The Trigger Processors

The MUCTPI, L1Calo, and L1Topo are used to process the trigger detector data and provide the POC input to the CTP. Using the input from the other trigger processors the CTP applies its trigger logic to form the trigger decision.

Most of the processing of the muon data happens in the muon on-detector electronics. The MUCTPI receives information about the muon candidates from each sector – a tuple of $(\eta, \phi, p_{\mathrm{T}})$ for up to two candidates per sector – and compiles the total number of muons above each of the six p_{T}-thresholds. In the process, the (η, ϕ) information is used to perform a simple overlap removal to avoid double counting. The total multiplicity of muon candidates above each of the six programmable p_{T} thresholds are then forwarded to the CTP [51].

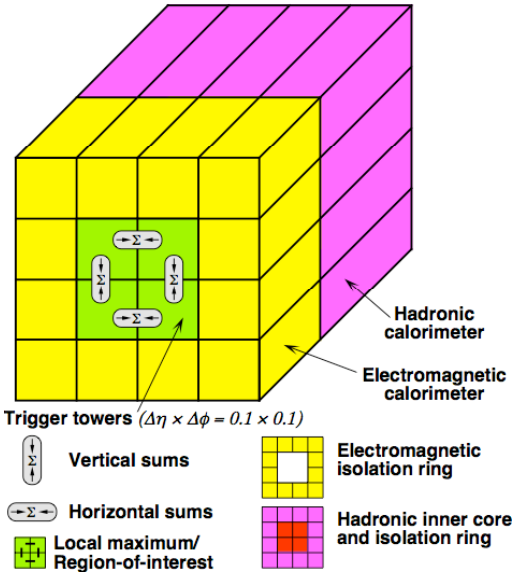


Figure 3.4: Visualisation of the elements used by CP algorithm for identification of e/γ and τ/hadron candidates. Source: [49]

A significant amount of signal processing happens in L1Calo [52] [47] in order to turn the raw analogue energy sums from Tile and LAr into the POCs provided to the CTP. Upon receiving the (raw) analogue energy sums, a Look Up Table (LUT) is used to perform several signal processing steps, such as pedestal subtraction, E_{T} calibration, noise suppression, and masking of problematic channels. This happens in the Pre-Processor Modules (PPM). There are 124 PPMs for receiving the calorimeter tower data from the front-end electronics.

The data is then transmitted to two algorithmic processors in parallel: the Cluster Processor (CP) and the Jet/Energy-sum Processor (JEP). The CP is used to identify e/γ and τ candidates above programmable E_{T} thresholds, optionally with an imposed restriction on isolation in the calorimeter. The CP algorithm considers all possible sets of overlapping 4×4 trigger tower windows, as illustrated in Figure 3.4. The e/γ algorithm looks for collimated high-E_{T} showers in the electromagnetic calorimeter, with a transverse isolation and no penetration to the hadronic calorimeters, to suppress the overwhelming hadronic background. The τ/hadronic algorithms similarly look for narrow energy deposits, permitting some level of isolation, but allowing the showers to penetrate into the hadronic calorimeters. The JEP is used to identify jet candidates as well as for determining global energy

sums, such as total transverse energy E_{T}, missing (transverse) energy $\not{E}_{\mathrm{T}}$, and (transverse) jet energy sum $\sum_j E_{\mathrm{T}}$. Both E_{T} and $\sum_j E_{\mathrm{T}}$ are calculated from the scalar sum of energy deposits while $\not{E}_{\mathrm{T}}$ is calculated as the (negative) vector sum. For jet identification, the JEP uses an algorithm similar to that of the CP, but with a somewhat coarser granularity of 0.2×0.2 in $\Delta\eta \times \Delta\phi$. The total multiplicity of identified candidates (e/γ, τ, j) above each of the programmable E_{T}-thresholds is transmitted to the CTP. For the energy sums (E_{T}, $\not{E}_{\mathrm{T}}$, and $\sum_j E_{\mathrm{T}}$) where multiplicity is not applicable, only the programmable E_{T}-threshold exceeded is transmitted.

L1Topo provides trigger input to the CTP based on geometric or kinematic association between the trigger objects produced by L1Calo and the MUCTPI. The quantities and their use will be discussed further in Section 3.4.3. L1Topo is subject to major latency constraints: the global upper limit of 2.5 μs combined with the processing time needed by L1Calo leaves $\sim 200\,\mathrm{ns} \approx 10\,\mathrm{BCs}$ for L1Topo to receive and process the data from the L1Calo and the MUCTPI and transmit the constructed POC to the CTP[4].

The CTP receives multiplicity information of candidates above threshold from the MUCTPI and L1Calo as well as event-level energy sums from L1Calo. The CTP further receives topological trigger information from L1Topo and inputs from a few other detectors as mentioned in Section 3.4.1. From the inputs, the CTP constructs *trigger items* as logical combinations of trigger inputs. The CTP allow for up to 512 trigger items to be defined. The CTP applies a per-item prescaling and global dead-time logic before forming the final trigger decision as the OR of the remaining triggers after prescale and dead-time veto has been applied. If the event is accepted, a Level 1 Accept signal is sent to all sub-detectors to initiate the readout. The trigger path and the CTP will be discussed in more detail in Chapter 4.

3.4.3. Available Quantities

The primary trigger inputs are the POCs produced by L1Calo, the MUCTPI and L1Topo. From L1Calo and the MUCTPI the CTP receives the multiplicity for e/γ, μ, τ, j candidates and exceeded threshold for event level E_{T}, $\not{E}_{\mathrm{T}}$, and $\sum_j E_{\mathrm{T}}$. As the candidates are constructed with partial event information the energy and momentum resolution of the Level 1 POCs are

[4]This constraint was the motivation for the direct inputs to the CTPCORE that will be discussed later: by omitting the CTPINs and the PIT bus 1-2 BCs could be saved.

poorer than what can be achieved with the full event information at HLT or in offline reconstruction. The programmable momentum and energy thresholds are chosen to get the best correspondence between Level 1 and the offline reconstruction. In order to compensate for the poorer resolution at Level 1, the Level 1 trigger is usually kept more inclusive than the HLT item it seeds: e.g., an HLT item requiring a muon with $p_{\mathrm{T}} \geq 25\,\mathrm{GeV}$ is likely to be seeded by a corresponding Level 1 item with a threshold of $p_{\mathrm{T}} \geq 20\,\mathrm{GeV}$ or lower.

L1Topo provides inputs to the CTP about the event topology. This allows for more selective trigger items at Level 1 to be created, based on kinematic or geometric relations between objects. This is for example useful for SM Higgs studies where spacial constraints on the τ-candidates in the $H \longrightarrow \tau\tau$ with one or both τs decaying hadronically, can significantly improve the signal-to-noise ratio in di-τ events. Like so, for B-physics, where di-muon triggers are commonly used, imposing geometric constraints can significantly improve the signal quality. Many searches for BSM physics look at events with multiple high energy jets and missing energy. The ability of L1Topo to set constraints on transverse- or invariant mass of a jet system or a spacial configuration, such as $\Delta R = \sqrt{(\Delta\eta)^2 + (\Delta\phi)^2}$ between muons, can improve the signal quality for these types of searches [53]. The better signal-to-noise ratio implies that the topological triggers can be run at a significantly lower rate than the corresponding (classic) multi-object trigger. Similarly, this rate reduction allows the trigger thresholds of the topological triggers to be lowered to make the trigger more inclusive.

3.5. From L1A to HLT

Upon receiving the L1A from the CTP, each detector push its event data from the memory on the detector front-ends to the detectors' Read-Out Driver (ROD) where the raw data is processed and formatted into the ATLAS data format. From the ROD, the data is pushed to the Read-Out System (ROS) where it is stored in a Read-Out Buffer (ROB) until the data is later pulled (and/or deleted) by the ROS. The CTP and the other Level 1 trigger processors additionally pushes their data fragment to the Region of Intrest Builder (ROIB), which assembles the Region of Interest (ROI)-fragment by summarising the trigger information from the trigger

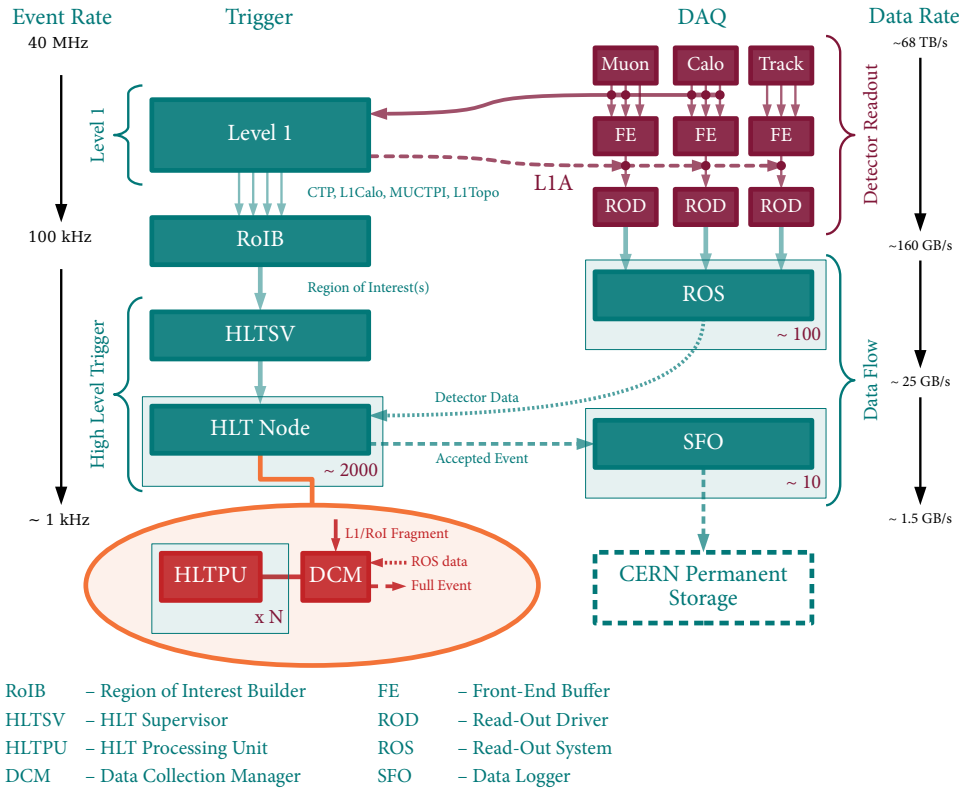


Figure 3.5: The ATLAS Trigger and Data Acquisition.

processors. The ROI-fragment is used to notify the HLT about the L1A. At the HLT, before the desired trigger algorithms can be executed, the HLT needs to reconstruct the event data from the detector event fragments. The HLT will request detector data from the DAQ system based on the CTP event information, that the HLT receives via the ROI fragment from the ROIB. The data flow in ATLAS Trigger and Data Acquisition (TDAQ) system is depicted in Figure 3.5.

3.5.1. An Archetype Detector

For the general discussion of ATLAS TDAQ, as well as the later discussion of preventive dead-time (Section 4.5.2), BUSY and dead-time (Section 4.5), and the CTP as a readout system (Section 4.6), it is instructive to define an archetype detector. The purpose of the archetype detector is to illustrate a general implementation pattern that allows for robust data-taking

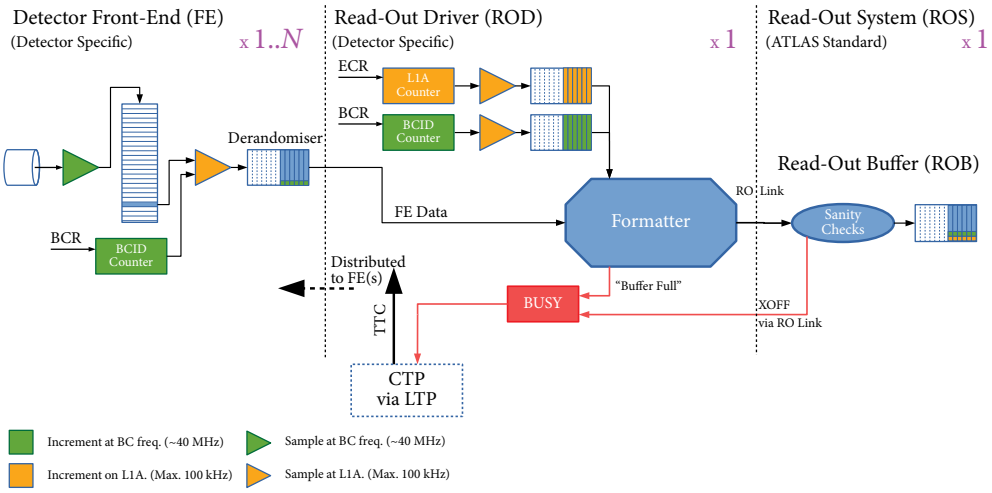


Figure 3.6: An archetype detector

at 40 MHz: each detector needs robust mechanisms for detecting and recovering from timing errors introduced by single event upset[5] or glitches in detector hardware or firmware. The discussion of the archetype detector is followed by a discussion of how the complete event data is reconstructed from the detector fragments.

Figure 3.6 shows a schematic of an archetype detector. The archetype detector consist of:

One or more front-ends (FE) these are typically tightly coupled to, if not directly on, the detectors in the cavern.

A Read-Out Driver (ROD) for merging and constructing the detector event fragment from the (pieces of) detector data provided by the front-end.

A Read-Out Buffer (ROB) for storing the detector event data until it is read out or deleted. The ROB is part of the ROS.

In relation to Figure 3.5, the archetype detector covers everything up until the ROB (and the ROS).

A derandomizer buffer is needed on the front-end because of the random arrival time of triggers as well as the finite transmission-time between

[5] A change of state caused by a single ionising particle striking a sensitive node in a piece of electronics.

the front-ends and the ROD. The transmission of detector data from the front-end to the ROD is sub-detector specific. The speed of the data stream from the front-ends to the ROD, as well as the size of the front-end derandomizer, must be sufficient to meet the ATLAS trigger rate requirements while keeping the dead-time at the derandomizer level at $< 1\%$ [54].

After the full detector fragment has been assembled in the ROD it is pushed to a ROS. The ROS is implemented as commodity servers hosting additional custom PCIe I/O boards for the physical implementation of the ROBs. The I/O boards are called RobinNP and are a firmware modified version of the C-RORC card originally developed by ALICE [55]. Each of these board have 12 optical link connectors, 8 GB DDR3 RAM, and are able to handle data rates of $\sim 1.5\,\frac{\text{GB}}{\text{s}}$. S-Link [56] is used for optical transmission between the ROD and the ROB. Communication between the ROB and the ROS happens via the PCI bus using Direct Memory Access (DMA). Standard gigabit Ethernet is used to connect the ROS to the DAQ network. The DAQ network is used for all communication between the ROSs and the HLT as well as for the Data Loggers that writes accepted events to the permanent storage.

For this discussion however, it suffices to think of the ROB as the output buffer for the event data constructed by the event formatter.

3.5.1.1. The Detector Front-end

At every bunch crossing, the front-ends typically sample data from the detector hardware and store the sampled data in a front-end buffer. The size of the front-end buffer must be enough to store 100 BC worth of data – the upper limit of the Level 1 latency. For the implementation in the archetype detector, the fixed latency of the Level 1 trigger is exploited: the event data corresponding to the bunch crossing that was triggered will always be at the same depth of the front-end buffer. Upon receiving a L1A, the front-end will transfer the corresponding detector data from the front-end buffer to a derandomizer buffer. In order to detect and resolve timing issues or glitches, the current Bunch Crossing Identifier (BCID) as seen by the front-end electronics, is stored in the derandomizer along with the detector data. The derandomizer is there to absorb the effects of trigger bursts: L1A triggers arrive randomly but follow a Poisson distribution with a mean close to the target Level 1 Accept rate of 100 kHz. The event

formatter of the ROD consumes the event data at an average rate close to the Level 1 Accept rate. The larger the derandomizers, the more of the statistical fluctuations can be absorbed. The derandomizers are discussed further in Section 3.5.2.1.

3.5.1.2. The Read-Out Driver

The ROD receives Trigger, Timing, and Control (TTC) signals from the CTP, as will be discussed in more detail in Section 4.2. In addition to the bunch clock signal and the L1A, each sub-detector receives, as part of the TTC, signals such as the Event Counter Reset (ECR) and the Bunch Counter Reset (BCR). The ECR and the BCR, respectively, are used to reset the L1A counters and the BCID counters. Each are issued periodically by the CTP to avoid counter overflow and to allow sub-detectors to quickly re-synchronise in case of timing errors. The ECR is typically issued every 5 s and at the beginning of each luminosity block, while the BCR is issued once every LHC turn.

A central element of the ROD is the event formatter. The event formatter assembles the data from the detector front-ends, performs sanity checks (such as checking that the BCID match) and performs sub-detector specific processing of the front-end data. The formatter then produces a sub-detector event fragment following the ATLAS byte-stream format [57]. Besides the detector specific data, the data must contain both the BCID and the value of the L1A counter. The final event fragment is then pushed to the ROB for buffering before consumption by the ROS.

Each sub-detector ROD has the possibility of signalling BUSY to the CTP in case it is unable to keep up with the Level 1 Accept rate. The main reasons for the ROD to signal BUSY are: a) saturation of the input bandwidth, due to high Level 1 Accept rate; b) long processing time due to, e.g., data corruption or large events; and c) back pressure from the DAQ system – in the form of XOFF from the ROB – i.e. saturation of the output bandwidth.

3.5.1.3. The Read-Out Buffer

The following is a brief overview of the ATLAS DAQ, which primarily serves the discussion of the archetype detector. A fuller description of can be found in [55].

The purpose of the ROB is to store the sub-detector event fragment before it is being read out by the ROS and passed on to the HLT. At this stage of the readout, the data format of the event fragment is no longer detector specific allowing the ROB design to be identical for all sub-detectors. After the event is accepted (and has been written to permanent storage) or rejected by the HLT, the ROS will request that the detector data is deleted from the ROB.

The ROD is connected to the ROB via an optical readout link implementing the S-Link protocol. The XON/XOFF flow control used allows the ROB to signal to the ROD if it is unable to receive more data. When receiving XOFF from the ROB, the ROD will apply the ROD BUSY, which is propagated back to the CTP. Consequently, upon receiving the ROD BUSY, the CTP will veto further triggers, allowing the ROB time to recover.

3.5.2. Event Assembly and Cross Checks

At two points in the readout process data from multiple sources are merged:

- The event formatter merges and processes data from one or more detector front-ends.
- After the event fragment is read out, all fragments are recombined at the HLT to form the full ATLAS event.

In order to detect and mitigate possible data, corruption and/or loss of synchronisation an identifier based system is used. Each front-end maintains a BCID counter and includes the count in the data pushed to the event formatter. The formatter, too, maintains a BCID counter as well as a L1A counter. While the event assembly in the formatter is primarily based on the order of arrival, the BCID allows for crosschecks. If the identifier is off, the front-end in question might have lost synchronisation or become corrupted, e.g., due to single event upset. The formatter, in a similar manner, includes an identifier in the event fragment header, containing the run number, the luminosity block number, the BCID, and the Level 1 counter. These numbers uniquely identify the event fragment within ATLAS and allow the full event to be correctly assembled.

There are additional header words in the event fragment, most significantly for signalling data corruption: If a ROD is unable to process an event (e.g., because of missing or corrupted front-end data) the resulting event fragment is flagged as corrupted.

3.5.2.1. Implications of Archetype Detector Design

Two aspects of the archetype detector design deserves further attention as it has implications for the CTP and for ATLAS data-taking in general.

Preventive Dead-time: Each detector ROD can signal BUSY to the CTP and thereby throttle the trigger rate in order to prevent data loss, e.g., due to buffer overflow. However, due to the distance between the detector front-ends and the RODs – and between the RODs and the CTP – relying solely on the ROD BUSY to protect the front-end buffers is not feasible: the signal propagation would take hundreds of ns during which time several new triggers could still arrive. This forms the basis for the complex dead-time in the CTP, discussed in more detail in Section 4.5.2: The complex dead-time algorithm in the CTP emulates the sub-detectors derandomizer buffers and applies a preventive dead-time should any (emulated) buffer get close to overflow.

Timing In: Each detector makes heavy use of the various TTC signals. In order for the detector to properly take data all timing delays must be determined such that counter increments and resets arrive promptly and such that data corresponding to the correct bunch crossing is read from all the front-end buffers. The process of adjusting the delays is called timing-in. As will be discussed later, the CTP acts as both signal distributor (Section 4.2) and reference (Section 5.5.1) of the TTC signals and as such is involved with the timing-in of all detectors: each detector is responsible for ensuring internal consistency and when that is achieved, the detector is said to be timed-in relative to the CTP. The timing-in of detectors is a crucial step before data can be taken. However, during data-taking, timing-related parameters are not expected to change and the relevant procedures are only carried out between runs when changes are made that alter the timing, e.g., firmware changes, change of length of cable, etc.

3.5.3. Region of Interest Builder

The ROIB was part of ATLAS's original trigger system described in [21]. After the upgrade of the TDAQ for Run II, the primary function of the ROIB is to act as a bridge between the custom hardware trigger processors of the Level 1 trigger and the HLT server farm. Based on the event fragments from the trigger processors the ROIB produces a record for the HLT containing a summary of the trigger information that helps guide the selection and execution of the trigger algorithms.

The interface between the Level 1 trigger processors and the ROIB resemble the interaction between the ROD and the ROB. In many aspects they can be thought of as identical: each of the Level 1 trigger processors push their data to both the ROB and the ROIB via S-Link. Both the ROB and the ROIB can signal XOFF back to the trigger processor (ROD) if unable to receive the trigger processors data fragment. The trigger processor will assert the ROD BUSY if it receives an XOFF from either of the two.

In the same way that an XOFF from the ROB indicates problems with DAQ, an XOFF from the ROIB is an indication of problems with the HLT: if events are being accepted faster at Level 1 than they can be processed at the HLT the ROIB will eventually be unable to receive data from the Level 1 trigger processors and will assert XOFF causing first ROD BUSY to be asserted and subsequently the trigger rate to be throttled.

3.6. HIGH LEVEL TRIGGER

The HLT and its components is described in more detail in [58].

The HLT consists of two HLT Supervisors (HLTSVs) (one functional and one as hot-spare), and $\sim$ 2000 HLT Nodes, as indicated in Figure 3.5. Both the HLTSV and the HLT Nodes are implemented using commodity servers. The role of the HLTSV is to receive the ROIB data, schedule events for the HLT Nodes, on a one-node-per-event basis, clear the ROS/ROB buffers and handle possible timeouts. Each HLT Node consist of a Data Collection Manager (DCM) and a number of processing units – called HLT Processing Unit (HLTPU) – corresponding to the number of CPU cores available on the machine. The DCM handles all I/O for the node, including receiving the ROI information, requesting event data from the ROSs, and facilitating the event building. The HLTPU threads execute the event selection algorithms and can share reconstructed objects amongst each other.

If an event is accepted by the HLT it is sent via the data logger – the Sub-Farm Output (SFO) – to permanent storage.

3.6.1. Available Quantities

At the HLT the full detector data is available and the ability to scale the processing horizontally (by having more HLT Nodes) permit more time for event processing. The availability of the inner detector data allow tracks to be reconstructed, providing a means to distinguish electrons from photons at HLT. The tracks are used to impose isolation requirements. Most quantities from Level 1 are (re-)reconstructed at HLT, as the full event data allow for better determination of the candidate energy/momentum, and per object (type) calibrations can be applied. E.g, muon momenta are recalculated, leveraging the additional and more precise data from the CSC and MDT. Jets and $\not{E}_{\mathrm{T}}$ are the quantities that benefit the most from the HLT reconstruction: The algorithms used for jet reconstruction allow for smarter clustering and more efficient subtraction of the calorimeter contribution from nearby electrons and photons [59]. The $\not{E}_{\mathrm{T}}$ is reconstructed as a vector quantity from the (negative) vector sum of the calibrated energy of objects, instead of as a sum of (raw) calorimeter cells energies:

$$\boldsymbol{E_{\mathrm{T}}} = \boldsymbol{E_{\mathrm{T}}}^{e} + \boldsymbol{E_{\mathrm{T}}}^{\mu} + \boldsymbol{E_{\mathrm{T}}}^{\tau} + \boldsymbol{E_{\mathrm{T}}}^{\mathrm{jet}} + \boldsymbol{E_{\mathrm{T}}}^{\gamma} + \boldsymbol{E_{\mathrm{T}}}^{\mathrm{soft}} \tag{3.3}$$

where the terms, $\boldsymbol{E_{\mathrm{T}}}^{i}$, are the negative vector sum of the calibrated energy of all objects of type i, with exception of the last term, $\boldsymbol{E_{\mathrm{T}}}^{\mathrm{soft}}$, which is the contribution from objects that could not reliably be associated with any type of object [60].

The HLT further allows restrictions to be applied on the geometrical region and on the algorithm(s) used for particle identification, isolation, and reconstruction.

The selection criteria of the HLT algorithm is based on the same type of selection criteria as is used at Level 1, and follows the general trigger strategy outlined in Section 3.3.1. One example of a HLT trigger algorithm is `HLT_mu20_L1_MU15`: The algorithm requires one muon with a reconstructed momentum of $p_{\mathrm{T}} > 20\,\mathrm{GeV}$. The algorithm is seeded from the Level 1 item, `L1_MU15`, that requires at least one muon with momentum, as determined at Level 1, above $p_{\mathrm{T}} > 15\,\mathrm{GeV}$.

Chapter 4

The Central Trigger Processor

The Central Trigger Processor (CTP) serves several purposes for the operation of ATLAS. First and foremost, the CTP is responsible for receiving trigger inputs from the other trigger processors and combining these to form the Level 1 Accept: the CTP is at the heart of ATLAS trigger path. The CTP is also the interface between ATLAS and the LHC. The CTP receives the bunch clock and the orbit signal from the LHC and is responsible for relaying the timing and trigger information to all sub-detectors of ATLAS. The CTP is the only place in ATLAS where dead-time can be applied. Thus, as the root of the ATLAS busy-tree, the CTP is responsible for receiving the BUSY signals from the sub-detectors as well as accounting for, applying, and book-keeping the dead-time accordingly. The CTP is a sub-detector in its own respect with a ROD of its own. The CTP event fragment is a detailed record of what triggers fired, and serves as the reference data fragment for event reconstruction.

4.1. Anatomy of the CTP

The CTP, shown in Figure 4.1, consist of 12 custom-built VME modules of 6 different types housed in a 9 unit VME create [61][62]. A Single Board Computer (SBC) hosted in the same create is used to communicate with the boards. The SBC is a VP E24 with a 4 core Intel Atom processor and 4 GB RAM [63]. In addition to the VME bus the CTP has three custom made back-planes that allow effective communication between the boards. The CTP is located in the electronics cavern USA15 outside of

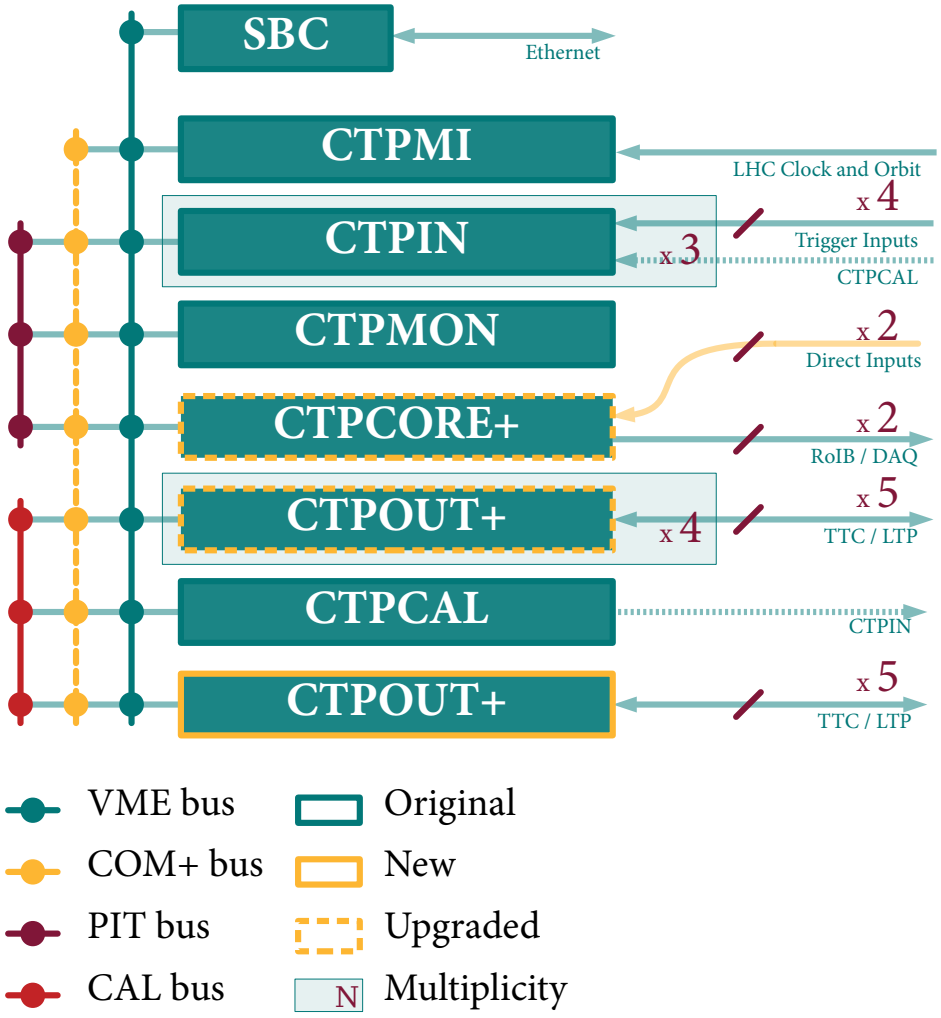


Figure 4.1: The boards and busses of the CTP. The CTP was upgraded for Run II and new parts and functionality introduced.

the experimental cavern. The CTP was upgraded during Long Shutdown I in preparation for Run II [1]. The CTP upgrade included a few hardware upgrades, as indicated in the figure, including a new COM bus backplane, the upgrade of the CTPCORE and CTPOUT module, and the introduction of an additional CTPOUT board. There are 6 different types of CTP boards [64]:

CTPMI - The board is the interface to the LHC machine. It receives the LHC clock and orbit signal. The board is also responsible for generating other timing-related signals such as the ECR.

CTPIN - The three boards receive the input from the other Level 1 trigger processors and aligns them in time. The time-aligned signals are then mapped onto the PIT bus where they are available to the CTPCORE and the CTPMON.

CTPCORE - The board is responsible for generating the L1A and applying dead-time. It is in the CTPCORE that the trigger inputs, received from the CTPIN – or from ALFA and the L1Topo via 2 dedicated inputs – are combined into trigger items, and where prescaling and bunch group masking is applied.

CTPOUT - The five boards are responsible for routing the various TTC signals *to* the sub-detectors as well as for routing the calibration requests and busy signals *from* the sub-detectors on to the COM bus (for busy) and the CAL bus (for calibration requests, see below). Each CTPOUT board has 5 front-panel connectors that are used to distribute the TTC signals as well as to receive the ROD BUSY and the calibration requests from the sub-detectors. The connection to sub-detectors is illustrated in Figure 4.3.

CTPCAL - The board is responsible for handling the incoming calibration requests as received via the CAL bus. Via a cable on the front panel the CTPCAL can provide a trigger input to the CTPIN, which then forms the basis for a calibration request trigger.

In addition to the VME bus, the CTP has three additional custom built busses that allow the boards to communicate with each other:

COM-bus used for common signals, in particular internal clock, busy, and L1A.

PIT-bus used for distributing the trigger inputs from the CTPIN to the CTPCORE as part of the real time trigger path and to the CTPMON for monitoring. As part of the CTP upgrade the PIT bus was changed to use Double Data Rate (DDR) allowing 320 signals to be transmitted over 160 physical lines at the cost of a latency penalty for multiplexing and demultiplexing the signals at the sending and receiving end.

CAL-bus used for transferring calibration requests received by the CTPOUT from the detectors to the CTPCAL that handles the orchestration.

4.2. Trigger, Timing and Control Signal Distribution

The CTP receives timing-related signals from the LHC and distributes these to the sub-detectors via the CTPOUTs along with the L1A and other control signals. These signals allow the detectors to stay synchronised in case of single event upset or similar. The sub-detectors rely on the CTP's distribution of timing signals for synchronisation with respect to the colliding bunches.

The Trigger, Timing, and Control (TTC) system [65] is a CERN wide standard for the synchronous distribution of timing signals, the level 1 trigger, as well as broadcasts and individually addressed control signals. The system uses two multiplexed bi-phase mark-encoded channels: the low-latency "A-channel" is dedicated to the distribution of the level 1 trigger. The "B-channel" is used for the transmission of short format synchronous data frames or long format asynchronous data frames. An example of a short format synchronous data frame is the transmission of the Bunch Counter Reset (BCR) to the detector front-ends for which exact synchronisation with the beam clock is paramount. Another example is the Event Counter Reset (ECR), used to periodically reset the Level 1 counter at the sub-detector side. In order to maintain synchronisation this is sent over the synchronous channel and is additionally protected by the introduction of a millisecond of dead-time on each side, to allow buffers to empty and detectors to reset before new L1As are issued.

Internally, sub-detectors of ATLAS use the B-channel for transmitting channel masks, calibration data, and other non-timing-critical information to detector front-ends.

4.2.1. From CTP to Sub-detector

ATLAS has developed the Local Trigger Processor (LTP) module [66] that is used as a TTC relay station for relaying TTC signals between the CTP and the sub-detectors. The module can receive and relay signals via Low-voltage Differential Signaling (LVDS) [67] as well as make the signals available via connectors on the front panel in the form of NIM-ECL levels [68][69]. The module is used by each sub-system as the interface to the CTP and acts as receiver (or generator) of TTC signals as will be discussed in Section 4.2.4. The LTP is an advanced switchboard that allows the user to choose the source and destination for the individual TTC signals such as clock, L1A and BUSY. The LTP also has a clock generator and a pattern generator. This allows the LTP to be used for operating a sub-detector in stand-alone setup by choosing the source of the TTC signals as the signals generated by the LTP itself rather than those received from the CTP. This functionality is used for detector commissioning and for parasitic running where a detector does not part-take in the data-taking.

All sub-detectors are connected to the CTP via a daisy chain of LTPs, as will be discussed in Section 4.2.4. The distribution of TTC signals between LTPs is electrical and only suitable for short ranges (less than 30 m). At the sub-system side, a TTC VME Interface (TTCvi) [70] module is used by each sub-systems to convert the received TTC signals, as well as sub-detector specific data, into the two data channels (the A- and B-channel) and a clock signal. The TTCvi feeds the clock as well as the two data channels to a TTC encoder and laser transmitter (TTCex) [71] module for encoding and electrical-to-optical conversion and transmission.

4.2.2. Timing Signals at the LHC

The bunch clock and orbit signals at the LHC are extracted from the RF cavities at Point 4 used to drive the two beams. There are separate RF cavities per beam, allowing for a per beam clock and orbit signal to be extracted: BC_1 and BC_2 as well as ORB_1 and ORB_2. The orbit signal is

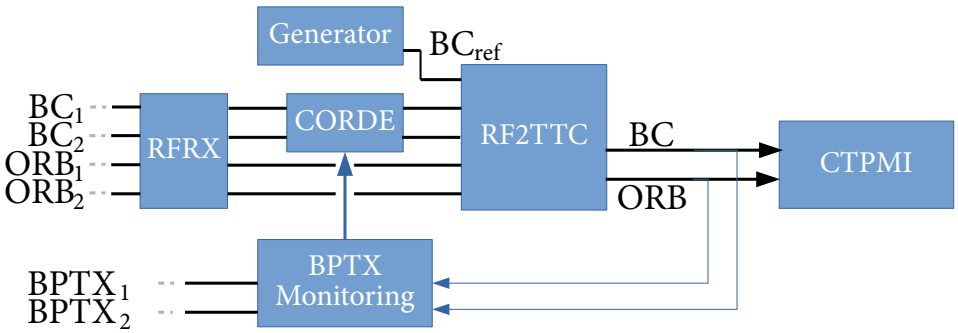


Figure 4.2: Correction of the LHC Clock.

a pulse emitted periodically at the LHC revolution frequency marking the $3564^{\underline{\text{th}}}$ bunch crossing in the LHC turn.

In most cases the timing signals are the same for the two beams and one of them can be ignored. There are two important consequences of extracting the clock signal directly from the beam: firstly, when the protons are accelerated, the RF frequency and thus the clock signal change. Secondly, when there is no beam in the machine the clock and orbit signals are undefined. ATLAS uses a reference clock $\mathrm{BC}_{\mathrm{ref}}$ extracted from a signal generator set to the injection frequency as a fall-back. Typically $\mathrm{BC}_{\mathrm{ref}}$ is used during setup and configuration of the detector after which the clock is switched to BC_1. This is normally done before the ramp of the beams and ATLAS thus experience the gradual increase in the frequency of the clock signal during acceleration.

From the RF cavities at Point 4, the clock signals are distributed to CERN Central Control Room (CCC) and CMS via TTC. From the CCC the clock signals are transmitted to ATLAS via optical cables. The optical fibers are drawn underground at a depth of $\mathcal{O}(1\,\mathrm{m})$. They are thus subject to variations in temperature, leading to a seasonal drift of $\mathcal{O}(7\,\mathrm{ns})$ between bunch clock and beam from winter to summer, which must be compensated for at the receiving end [72].

4.2.3. Synchronising to the Colliding Bunches

Figure 4.2 depicts how ATLAS extract a bunch clock (BC) and an orbit signal (ORB) from either the LHC provided timing signals (BC_i and ORB_i for $i \in 1, 2$) or the ATLAS reference clock ($\mathrm{BC}_{\mathrm{ref}}$).

ATLAS receives the timing signals from the LHC in the underground counting room via optical links. The RFRX module is used for optical-to-electronic conversion. The timing signals from the LHC arrive uncorrected for temperature drift. The temperature drift is corrected by a feed-back loop consisting of the beam pickups (BPTX) and the Clock ORbit DElay (CORDE) module [73]: the two clock signals pass through the CORDE module that delay the signals in steps of 10 ps. The BPTX are used to measure the drift, from which the proper delay is determined. The corrected timing signals are fed to the RF2TTC which produces an output bunch clock (BC) and orbit (ORB) signal. These are sent to the CTP via the CTPMI for distribution to ATLAS and to the BPTX monitoring station as part of the feed-back loop [51]. Corrections to the delay introduced by the CORDE are applied whenever a drift of more than 30 ps has been detected over a 10 min period.

4.2.4. TTC Distribution in ATLAS

In the CTP, additional TTC signals are generated, most notably the L1A and the ECR. The TTC signals, made available on the COM bus, are distributed to the remainder of ATLAS by the CTPOUTs. The CTPOUTs are connected to each sub-system in the counting room via daisy-chained LTPs as depicted in Figure 4.3. The Local Trigger Processor Interface (LTPi) modules, also shown in Figure 4.3, serve multiple roles. At the input of the LTPi, an equaliser reshapes the received signals to counter the attenuation caused by the electrical transmission from CTP to LTPi. The LTPi also contains a DELAY25 chip [74] that allow the output signals to be delayed for up to 25 ns in steps of 0.5 ns. This is useful for deskewing the received signals and for fine adjustment of the phases to facilitate the later clock latching. The LTPi module, in combination with an LTP, furthermore allow for a segment of sub-detectors to operate in a standalone mode. In standalone mode, one or more of the TTC signals normally received from the CTP are instead generated directly in an LTP, by signal generators. In this setup, the LTPi is configured to route the signals from the primary LTP to the LTPs of the sub-detectors. This feature allow a subset of detectors to be operated independently of the CTP and other detectors.

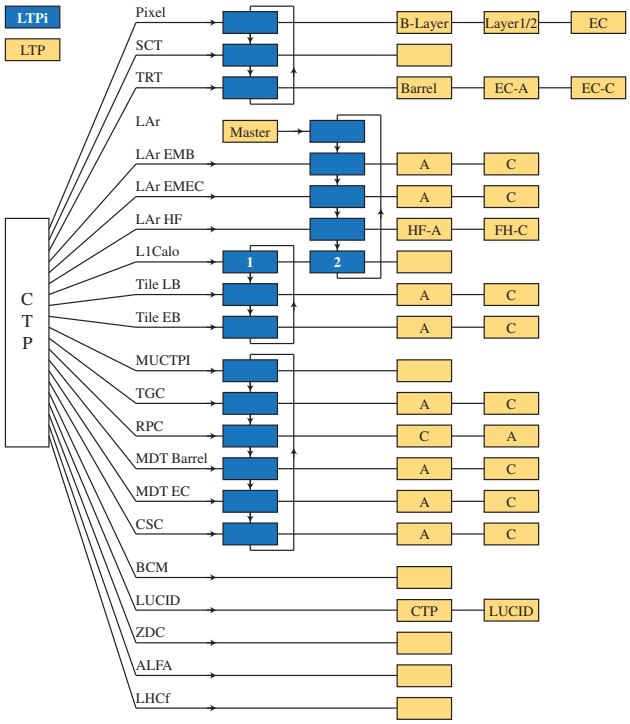


Figure 4.3: Distribution via LTPi of TTC from CTP to ATLAS sub-systems. The CTP is connected to all sub-systems via daisy-chained LTPs. Rings of LTPi modules are used to allow sub-detector segments to operate in a stand-alone mode where TTC signals are generated on a local LTP instead of taken from the CTP.

4.2.5. Receiving BUSY and Calibration Requests

In addition to TTC distribution, the LTPs are used to propagate BUSY signals and Calibration Requests from the sub-detector systems to the CTP. Upon receiving a Calibration Request, the CTP may issue a Calibration Trigger. Calibration triggers are special triggers for sub-detector calibration and are constrained in time to the LHC abort gap. The primary use-case is calibration of the tile calorimeter.

4.3. Common Monitoring Functionality

There are three recurring hardware implementation patterns used across all the CTP boards for monitoring of quantities that change from bunch crossing to bunch crossing, such as whether a trigger fired or not or if busy or dead-time was incurred. This section outline the functionality of these

implementation patterns, their strengths and weaknesses and their role in the monitoring of the CTP.

4.3.1. Counters and Counter Arrays

The counter is the simplest of the recurring implementation patterns and consist of at least one basic counter and a *turn counter*. The value of the basic counter is incremented at every bunch clock tick, in so forth the monitored signal is active. In most cases, the monitored signal would be whether a trigger input is active, a trigger item has fired, or if busy has been incurred. Where more than one signal is monitored, typically an array of counters is used.Each counter in the array is associated with a different signal. The turn counter is used for normalisation and is incremented on every LHC revolution.

Knowing the value n^i of a counter in the array, and the value of the turn counter, n_{turn}, the mean counter rate R^i can be determined as:

$$R^i = f_{\text{rev}} \frac{n^i}{n_{\text{turn}}} , \tag{4.1}$$

where f_{rev} is the revolution frequency. Assuming the counter n_i follows Poisson statistics, and that there is no note-worthy[1] uncertainty on n_{turn}, the uncertainty on the mean rate can be estimated as:

$$\sigma_{R^i} = f_{\text{rev}} \frac{\sqrt{n^i}}{n_{\text{turn}}} . \tag{4.2}$$

The readout procedure for a counter array (including the turn counter) follows the following four steps:

1. Stopping the counters ;
2. Copying the value of the counters to a buffer from where they can be read out ;
3. Resetting the counters ; and
4. Starting the counters again.

[1] The discretisation error on n_{turn}, of $\sigma_{n_{\text{turn}}} = \frac{1}{\sqrt{3}}$, is negligible in the common case where the integration time is $\mathcal{O}(1\,\text{s})$ and where the rate R is less than $\mathcal{O}(100\,\text{MHz})$.

There are several merits to this procedure: By stopping the counters, correspondence between the array counters and the turn counter can be ensured. By copying the counter values in the firmware to a buffer the counters can be reset and restarted without having to wait for the transfer of counter values over VME. This minimises the counter downtime. However, it is impossible to completely avoid a downtime which means that while the counters are inactive *monitoring dead-time* is incurred.

Most counters are 32 bit with one bit reserved for overflow. With an effective size of 31 bit and a theoretical maximal input rate of 40 MHz a counter will, in the worst case, overflow in $\sim 53\,\mathrm{s}$. Thus, all counter based monitoring needs to be read out at least once every 53 s to avoid overflow.

4.3.2. Sequencers

Sequencers are a (potentially) loss-less variant of the counter. These are solely used for monitoring of busy and dead-time, for which monitoring dead-time is undesirable. Unlike the counter (arrays) discussed above, the sequencers do not make use of a turn counter for normalisation and are never grouped into arrays.

A sequencer consist of a basic counter, a sampler, and a FIFO. The basic counter is used to count the number of bunch crossings in which a BUSY or dead-time signal is active. At a programmable frequency, once every N bunch crossings, the sampler samples and resets the basic counter. This procedure is gap-less and does not introduce monitoring dead-time. The sampled value is transferred to a FIFO, which can then be read out. If the FIFO does not overflow, the monitoring is both gap-less and loss-less.

Denoting the sampling frequency f, each FIFO sample corresponds to a time span of

$$\Delta t = \frac{1}{f} = N \cdot \frac{25\,\mathrm{ns}}{\mathrm{BC}}\,. \tag{4.3}$$

This serves as the normalisation for each FIFO entry. Thus, if M FIFO samples has been read out, the mean busy or dead-time fraction D can be calculated as:

$$D = \frac{\sum_{i=1}^{M} n^i}{M \cdot N}\,, \tag{4.4}$$

where n^i is the $i^{\underline{\text{th}}}$ FIFO entry read from the FIFO.

The above has two important implications for the readout:

- The sampler frequency and the FIFO size needs to be balanced with the frequency at which the FIFO is read out to avoid buffer overflow and to ensure that the FIFO is not empty.
- In order to determine the normalisation of the FIFO samples, the configured frequency of the sampler must be known.

The typical FIFO sizes are enough to hold $\mathcal{O}(1000)$ samples.

4.3.3. Per-bunch Counters

Per-bunch counters are used to monitor busy and dead-time as well as trigger input and item rates on a per-bunch basis. This is done using 3564 basic counters, a turn counter, and a multiplexer to choose which of the basic counters is incremented during the particular bunch crossing. The counter selected by the multiplexer is changed in a round-robin manner at each clock tick. The per-bunch counter thus differ from a counter array in that a) all basic counters share the same input signal b) only one counter can be incremented at each clock tick.

From the turn counter n_{turn} and the 3564 bunch counter values n_i, the per-bunch mean fraction can be obtained as:

$$f_i = \frac{n_i}{n_{\text{turn}}} \,. \tag{4.5}$$

This is used for per-bunch busy and dead-time monitoring, where f_i is the per-bunch busy or dead-time fraction.

For the per-bunch monitoring of the rate of trigger inputs and items, the mean fraction is converted into a mean rate, using the LHC revolution frequency:

$$R_i = f_{\text{rev}}\, f_i = f_{\text{rev}}\, \frac{n_i}{n_{\text{turn}}} \,. \tag{4.6}$$

The readout procedure for per-bunch monitoring is equivalent to the readout procedure for the counter array. Thus, the readout of a set of per-bunch counters introduces monitoring dead-time too.

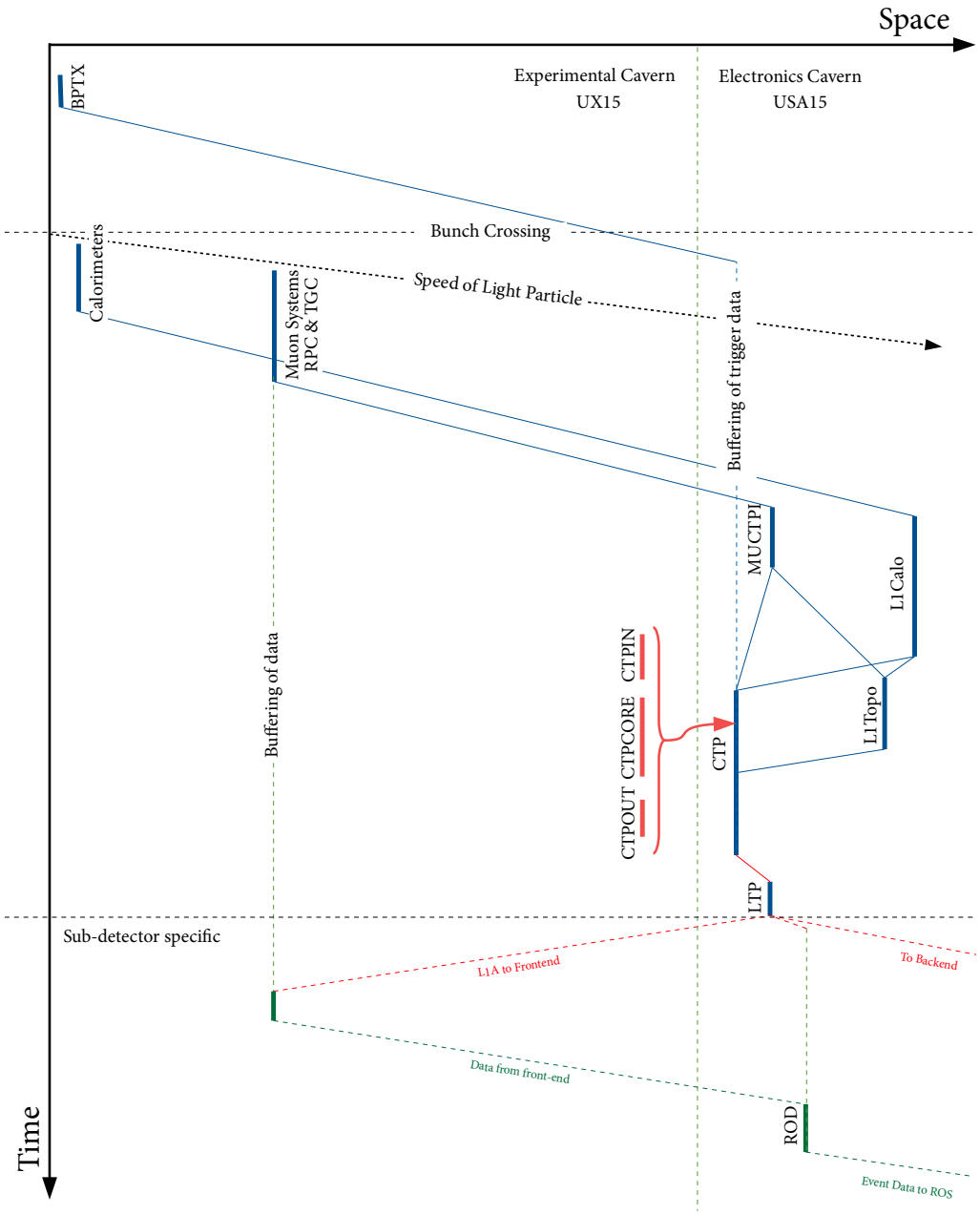


Figure 4.4: Space-time diagram of ATLAS trigger path. The depiction is not to scale. Processing and transmission of trigger data is indicated by blue lines. Distribution of the Level 1 Accept is indicated with dashed red lines. Data flow and processing after the Level 1 Accept is indicated in green.

4.4. The Trigger Path

The full trigger path, depicted in Figure 4.4 as a space-time diagram, is synchronous and operates at a fixed latency. Every 25 ns each sub-detector digitises and buffers the raw detector data. The muon detectors and calorimeters transmit their trigger data to the muon and calorimeter pre-processors, where the POCs are created and where cuts on programmable thresholds for transverse momentum and energy as well as on multiplicity, where applicable, are applied. The CTPIN receives the multiplicity of the POCs from the trigger pre-processors as well as event level calorimeter information about the total transverse and missing energy, from the L1Calo and aligns these in time. The trigger pre-processors also seed the topological trigger which provides input on event topology directly to the CTPCORE. The CTPCORE forms trigger items based on the trigger inputs and applies logic for rate reduction and dead-time. The L1A is formed as the logical OR of all trigger items after rate reduction and dead-time is applied. The L1A is distributed by the CTPOUTs via the LTP links to all sub-systems as part of the TTC distribution. From there, the L1A is distributed to the ROD and to the front-end(s) which prompts the readout and construction of the detector event-fragment, as discussed in Section 3.5.1.

4.4.1. The CTP Trigger Path

The *trigger path* describes the flow and processing of data from the time of collision up until the Level 1 decision is made. The following will cover the CTP's role in the trigger path from the trigger inputs arrive at the CTPIN, through the formation of the trigger decision in the CTPCORE, to the distribution of the L1A to ATLAS via the CTPOUTs. The trigger path operates in a pipelined synchronous manner, where the data processing is advanced with every clock step. As a result the L1A is emitted at fixed latency.

4.4.1.1. CTPIN

The candidate data from the MUCTPI and the L1Calo – e.g., muon and e/γ candidate multiplicities (see Section 3.4.3) – for a particular bunch crossing arrives at the CTPIN out of time and out of phase due to the unavoidable difference in processing time between the two trigger pre-processors and

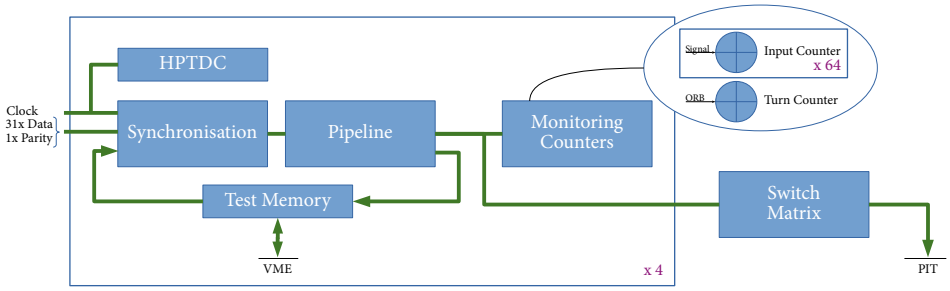


Figure 4.5: Diagram of a CTPIN board.

the difference in cable length between the pre-processors and the CTPIN. In order to process the data, it needs to be aligned in time.

Figure 4.5 shows a diagram of the CTPIN. Via the four identical front-panel connectors each CTPIN receive four groups of 34 signals: 31 data bits as well as parity, clock and ground. Most of the signal processing happens per cable, as indicated in Figure 4.5 by the blue box encompassing most of the functional blocks. The High Precision Time-To-Digital Converter (HPTDC) is used to measure the phase of all input signals with respect to the (CTP) internal clock. These phases are used to determine the per-signal clock-edge to use for latching. The inputs are latched to the internal clock on either rising or falling edge at the Synchronisation-step depending on a per-signal programmable configuration. The synchronised input data is then transferred to pipelines of programmable lengths to ensure that data are aligned in time, i.e. the earlier signals are delayed to match the signal arriving last. The pipeline delays the input data in steps of 25 ns.

In order to make the time aligned trigger inputs available to the CTP-CORE and the CTPMON they need to be made available on the PIT bus. The programmable Switch Matrix is used to select and route 320 of the $3 \times 4 \times 31 = 372$ trigger inputs.

For each connector (or similarly, group of input signals) a set of Monitoring Counters and a Test Memory exist. The Monitoring Counters block consist of a counter array with 64 counters. The counters are used to measure the rate of the trigger inputs to the CTP before processing. For a meaningful monitoring of the multiplicities of the POCs a decoding of the input signals is needed. This is done by a selector and a LUT not included in the figure. Having 64 counters available allow the monitoring of all individual signals as well as a subset of decoded inputs.

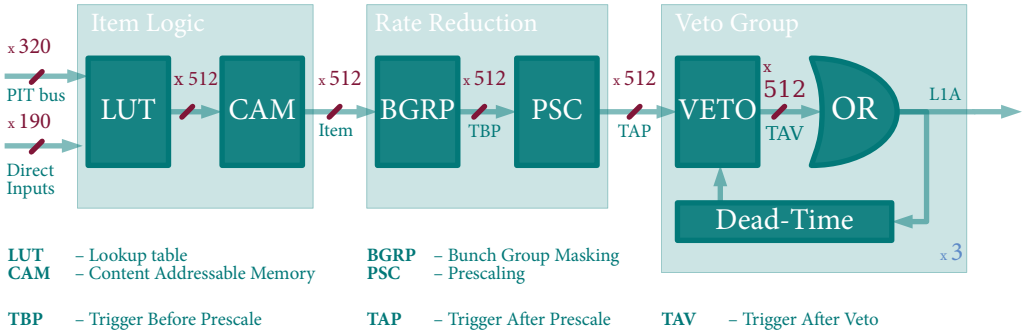


Figure 4.6: Depiction of the CTPCORE trigger path. The CTPCORE combines the received input of physics object candidates and their multiplicity into 512 trigger items using the LUT and the CAM. The rate of each item is then constrained by applying bunch group masking and prescaling. After prescaling the trigger items are fed to the veto logic. The L1A for each partition is formed as the OR of the triggers after applying the partitions veto logic.

The Test Memory can be used to record and replay the trigger inputs. Data can be loaded into the memory either via VME or by recording a snapshot sequence of inputs at the output of the Pipeline-step. The data in the memory can similarly be read out via VME or replayed as input to the Synchronisation-step. The primary use of the Test Memory is to debug timing issues, either at the input (by recording and analysing data) or downstream (by replaying data).

4.4.1.2. CTPCORE and CTPOUT

Figure 4.6 depict the trigger path of the CTPCORE. The CTPCORE receives 320 trigger input bits from the PIT bus, representing POC multiplicity and thresholds. Via three connectors on the front panel, referred to as direct inputs, the CTPCORE receive an additional 192 bits of trigger input from L1Topo and ALFA. The direct inputs reduce the latency by $2-3$ BCs by bypassing the CTPIN and the PIT bus. The reduced latency allows for longer processing time in L1Topo and is needed for ALFAs trigger signals to arrive early enough that they can be used as such. The CTPCORE uses a fully programmable array of parallel LUTs to form logical combinations of the trigger inputs. The LUT configuration is part of the trigger menu

definition and does not change during data-taking. The LUTs are static and the configuration of what signal goes where is implicitly done by the switch matrices of the CTPINs when routing the input onto the PIT bus. Typically the LUT logic is used to form ORs of the trigger inputs as well as apply cuts on the POC multiplicities. In the Content Adressable Memory (CAM) the inputs are combined into 512 trigger items as the (N)AND of one or more inputs.

The next step of the path through the CTPCORE is the bunch group masking. The CTP operates with 16 bunch groups where each bunch group is defined by a 3564 bit pattern, that identifies each bunch crossing within a turn as belonging to the bunch group or not. Each trigger item is assigned a 16-bit bunch group mask that confines the trigger item to a selection of the 16 bunch groups. The bunch groups are usually chosen to reflect the state of the beam in the machine. The two most commonly used are called ”Paired” and ”Empty” and have bit-masks that correspond to bunch crossings where there the two colliding bunches are both either filled or empty, respectively. Another important bunch group is ”CalReq” which is used to restrict calibration triggers to certain bunch crossings: it is typically desired that the calibration triggers are fired away from the paired bunch crossings to avoid losing collisions. The Calibration bunch group is typically placed in the abort gap of the LHC filling scheme. An example bunch group set is shown in Figure 7.4.

After bunch group masking, prescaling is applied. Each trigger item is assigned a prescale factor as part of the trigger configuration. The CTPCORE applies non-deterministic fractional prescaling: a pseudo-random number generator is used to ensure that on average M in N triggers are accepted. The prescale factor for individual trigger items can be changed throughout a run to reflect the change in instantaneous luminosity and thus trigger rate.

After prescaling, the 512 triggers are sent to the veto block where dead-time is applied and where the L1A is formed as the OR of all triggers after dead-time is applied. The resulting L1A is made available on the COM bus.

The CTPOUTs picks up the L1A from the COM bus and distributes it as part of the TTC distribution to the sub-detectors.

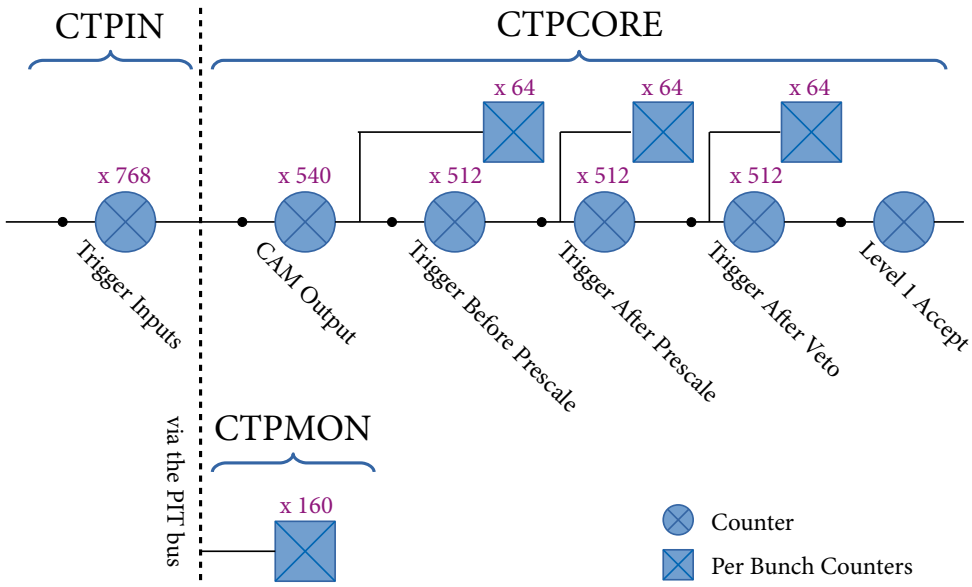


Figure 4.7: The monitoring along the trigger path. From the time-aligned inputs in the CTPINs to after the L1A formation in the CTPCORE the full trigger path is monitored using counters. In addition, the CTPCORE allows for per-bunch monitoring of selected trigger items before prescale, after prescale as well as after veto. The CTPMON provides per-bunch monitoring of the trigger inputs made available on the PIT bus.

4.4.2. Trigger Path Monitoring

4.4.2.1. Counter Based Monitoring

Throughout the trigger path, counter arrays and per-bunch counters are used for monitoring as summarised in Figure 4.7.

Each trigger input, after synchronisation and time alignment, is monitored using counters in the monitoring blocks of the CTPINs as depicted in Figure 4.5.

In addition to the counters in the CTPIN, the CTPMON provides detailed per-bunch monitoring of the trigger inputs on the PIT bus. The CTPMON implements 160 *per-bunch counters*. After the CTP upgrade, where DDR was introduced on the PIT bus, up to 320 signals are available on the original 160 data lines. Due to limitations in resources of the CTPMON, the board can only monitor one of the two signals per data line. When mapping input signals in the CTPIN onto the PIT bus, care must be

taken to not map two signals onto the same data line for which per-bunch monitoring is desired.

The CTPCORE implements monitoring of all trigger items at each step following the item formation using counter arrays. In Figure 4.6 this is corresponds to every step after the CAM. This allows for monitoring of the exact number of triggers and their average rate before bunch group masking, and as Trigger Before Prescale (TBP), Trigger After Prescale (TAP) and Trigger After Veto (TAV).

In addition, the CTPCORE implements 3×64 per-bunch counters, for the per-bunch monitoring of TBP, TAP and TAV counts/rates for a programmable subset of trigger items. For completeness it should be stressed, that the per-bunch TBP monitoring does not include the bunch group masking and thus is technically per-bunch *item* monitoring, in the language of Figure 4.6.

4.4.2.2. Event Monitoring

The CTPCORE allows for monitoring of the CTP event fragments produced by the CTP readout for the DAQ and HLT. The event fragment contains a detailed trigger record including the state of all trigger inputs and all trigger items, before prescale, after prescale and after veto. The record contains information about a programmable number of bunch crossings between 1 and 31 with a programmable offset with respect to the L1A. Figure 4.8 is a depiction of a part of the trigger record for a window size of 12 and a L1A cursor position of 4. Whether a trigger fired in a particular bunch crossing is here depicted as a logical high.

The ability to monitor the arrival time of individual triggers around the L1A is useful for the timing-in of triggers: if a physics trigger shows up in a bunch crossing with no collision the corresponding input at the CTPIN is either delayed too much or too little. By choosing a reasonable reference (e.g., the MBTS trigger during beam splashes) the relative offset can be determined. In this use-case the CTP can be thought of as a 512-channel oscilloscope with a sampling rate of 40 MHz.

The event monitoring can be operated in two ways: a single-shot mode and a burst mode. In both cases a VME write is used to start the process. In single-shot mode, the CTP event data of the next accepted event is written to the MON FIFO in addition to the DAQ and the ROI FIFOs – as

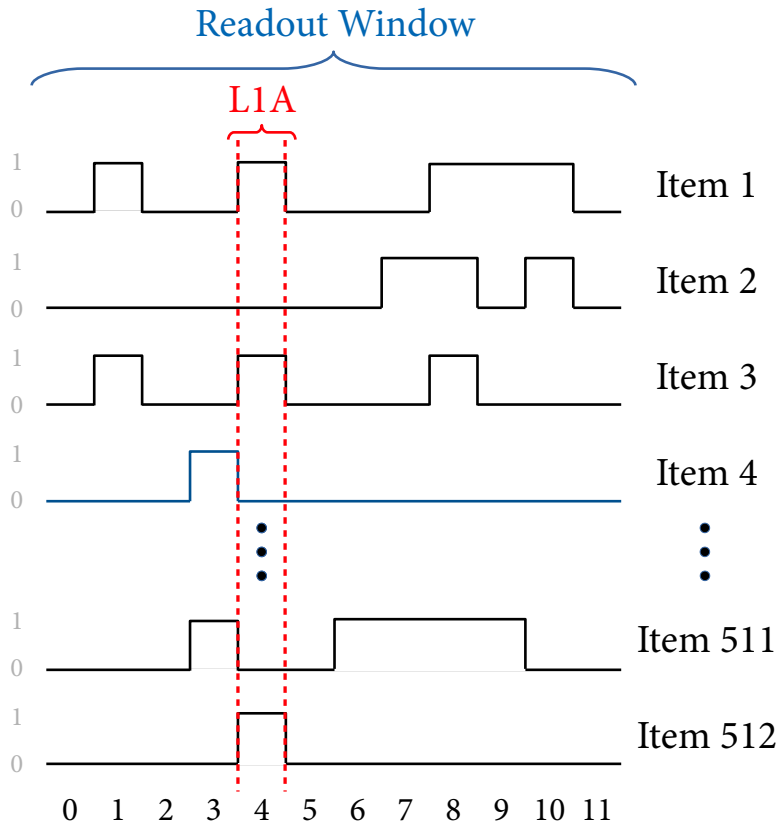


Figure 4.8: The trigger record included in the CTP event fragment.

indicated in Figure 4.9. In burst mode, all subsequent CTP event fragments are written to the MON FIFO until the watermark level is reached and the buffer is considered full. The size of the MON buffer corresponds to roughly 1000 events, depending on the event size. The VME bandwidth is $\sim 40\,\mathrm{MB/s}$[2] and the typical CTP event size is $\sim 1\,\mathrm{kB}$ which sets the theoretical upper limit for the average event monitoring rate at $\sim 40\,\mathrm{kHz}$. The event monitoring is however never used at such rates: during normal operation, event monitoring is used for periodic cross checks at typical rates of 1 event per minute (using single shot mode). The most extensive use of event monitoring is during special runs for timing where the window size, and thus the event size, is much larger than during normal data-taking. Consequently the average event monitoring rate is much lower.

[2]The achievable bandwidth depends on the type of transfer as well as the driver. 40 MB/s is a commonly quoted theoretical value [75].

4.5. Dead-time and Busy

In order to recombine the event data from the sub-detectors, ATLAS uses event counter information from the header of each fragment, as discussed in Section 3.5.2. This ties each event data fragment to a unique L1A. In order to ensure the data integrity of all triggered events, dead-time needs to be applied globally. If dead-time would be applied locally by the sub-detectors it would be impossible to ensure that triggers were not issued at times where any detector is experiencing dead-time. This in turn would lead to loss of data fragments from the affected detectors and thus to incomplete events. The only place in ATLAS where dead-time is applied is in the CTP. When generating dead-time, the busy logic of the CTP takes into account input from the sub-detectors, the HLT, and operators in the control room as well as the internally generated preventive dead-time. The preventive dead-time as well as the busy logic and its monitoring is the main topic of this section.

For the following discussion it is instructive to make a distinction between the three related concepts: *BUSY signals*, *veto signals* and *dead-time*. BUSY signals are one-bit digital signals that serve as the inputs to the busy logic of the CTP. Most of the BUSY inputs originate from the sub-detector RODs. At the beginning of each run, the CTP is configured to mask out the BUSY from sub-detectors that are not part of the data-taking. From the BUSY signals received and the preventive dead-times, the CTP generates a veto signal[3]. The veto signal is formed as the OR of the BUSY, the simple dead-time, and a programmable selection of the complex dead-times. The veto signal is used to veto (or gate) the triggers left after prescale. With all triggers vetoed, no L1A is issued, and experimental dead-time is incurred.

4.5.1. From Busy to Dead-Time

Figure 4.9 shows a representation of the busy tree of the CTP. The CTP receives the BUSY signals from the sub-detectors via the LTP cables connected to the CTPOUT boards. The CTPOUT boards can be configured per cable to mask out signals coming from sub-detectors not taking part in the run. The BUSY signal per cable is monitored and directed to an on-board memory as well as to a partition demultiplexer. A per-partition

[3]In fact, for legacy reasons, two veto signals are created as shown. However, only one is used today.

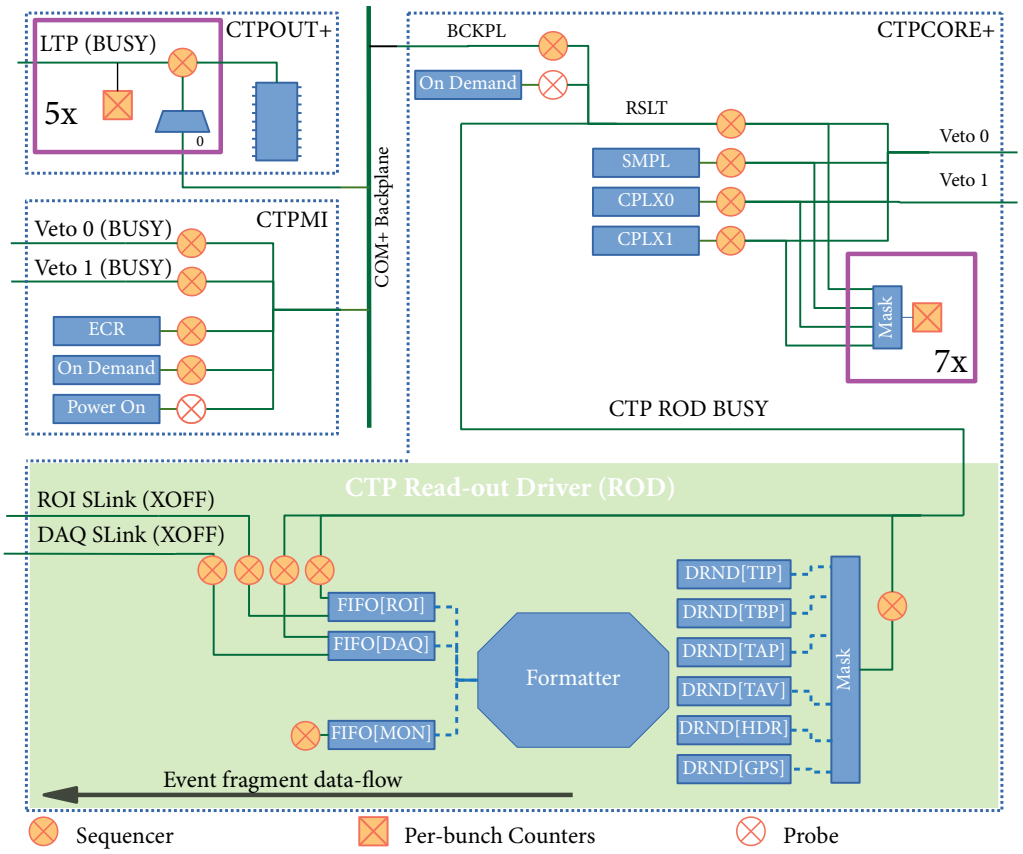


Figure 4.9: Diagram of the BUSY flow in the CTP. All intersections correspond to an OR of the incoming signals and all forks corresponds to fan-outs. The CTP receives BUSY from the sub-detectors at the CTPOUT. The BUSY is distributed to the CTPCORE via the COM backplane. The CTPMI, too, communicates its BUSY signals via the COM backplane to the CTPCORE. The CTPCORE combines the backplane BUSY with the internal busy from the CTP and the signals from the preventive dead-time to produce the final veto signals.

OR is created after the demultiplexer and the result is published on the COM bus.

Another source of BUSY on the COM bus is the CTPMI. As part of the ECR generation, a 1 ms BUSY is emitted on both sides of the ECR to allow enough time for event counters and buffers on the detector side to reset and empty before new L1As are issued. Another important source of BUSY from the CTPMI is the "On Demand" busy: a software programmable BUSY state which is used as a global busy. Its primary uses are

- the beginning and end of a run;
- during luminosity block transitions;
- during recovery procedures.

For the start and stop procedures, the busy On Demand is used to ensure a known stable state without any spurious triggers. The busy On Demand is also used for the recovery of sub-detectors, where the sub-detector is unable to send busy via the TTC system throughout the recovery procedure (e.g., power cycling of the electronics might cause the BUSY from the sub-detector to temporarily disappear). For the sake of completeness, the CTPMI has two specialised electronic inputs that can be used to inject a BUSY signal into the CTP. These inputs are not used during normal operations but provides extra flexibility. The CTPMI also generates a BUSY signal during Power On of the CTP.

The CTPCORE plays a double role in this setup: on the one hand it acts by routing the BUSY and forming the two veto signals, (the upper right part of Figure 4.9) and on the other hand it is a sub-detector ROD in its on respect, i.e., it can issue a ROD busy signal itself (the lower part of Figure 4.9). The CTPCORE combines the busy from the COM backplane in an OR with its own ROD busy and an On Demand busy to form the Result (RSLT). The Result can not be masked. In addition to the Result, three other dead-time signal are applied, namely those corresponding to the three preventive dead-times: Simple dead-time (SMPL) and the two complex dead-times (CPLX0, CPLX1). The On Demand busy of the CTPCORE is not utilised during data-taking as the same functionality, but with better monitoring, is available in the CTPMI. It is included in the figure for completeness.

4.5.2. Preventive Dead-Time

Preventive dead-time is introduced to protect the detectors' readout system including their derandomizer buffers. The CTP implements two types of preventive dead-time: simple dead-time and complex dead-time.

The dead-time settings are written to the CTP as part of the configuration and are not normally changed during operations.

4.5.2.1. Simple Dead-Time

The simple dead-time prevents L1As from being issued within the readout window of any sub-detector. This prevents possible data corruption at the detector front-end electronics.

The simple dead-time is defined by the minimum time between two L1As, to prevent new triggers from arriving while data are being read out. During normal operation this is set to 4 BCs or 100 ns, defined by the length of the LAr readout window. For dedicated test runs, a longer simple dead-time can be applied. As an example, in LAr test-runs, where the readout of all LAr samples are required, the simple dead-time is set to the duration of 2500 BCs.

4.5.2.2. Complex Dead-Time

The complex dead-time emulates the data buffers on the sub-detectors and protects against data loss due to buffer overflow in the derandomizers.

The CTP implements four *leaky bucket* algorithms for the complex dead-time, each representing a buffer with different properties. Each bucket i is defined by its size s_i and its leak rate $1/r_i$. At a L1A, a token is added to the bucket, and at every r_i bunch crossings, a token is removed. If the number of tokens in the bucket reaches s_i, the bucket is full and a BUSY signal is raised. A programmable mask can be used to select which of the dead-time algorithms are taken into account.

4.5.3. Monitoring Dead-Time and Busy

Sequencers are used for most of the busy and dead-time monitoring in the busy tree, as indicated in Figure 4.9, and in the CTP ROD. The important exceptions are the Power On busy in the CTPMI and the On Demand busy in the CTPCORE that can only be monitored using probes. A probe provides an instantaneous and unbuffered snapshot of the current busy status.

Each CTPOUT provide five per-bunch counters, allowing the BUSY from each connected sub-detector system to be monitored.

The CTPCORE provides seven per-bunch counters for monitoring of the dead-time. Each of these per-bunch-counters include a programmable mask for selecting the input signals to consider.

4.6. Read-Out Driver

The readout part of the CTP works in analogy of any other detector front-end: the raw trigger data of the CTP is buffered in derandomizer buffers from which the formatter reads and processes the data for each L1A. The formatted data is then put into three output FIFOs namely one for the DAQ, one for the ROIB, and one for the event monitoring as was discussed in Section 4.4.2.2.

If the L1 trigger rate is too high for the formatter to keep up, the derandomizers will fill up, potentially leading to data loss or corruption. Similarly, if the DAQ or the ROIB is unable to keep up and throttles the readout, the FIFOs will fill up, resulting in data loss. In order to avoid data loss, a BUSY is raised when the level of the FIFOs and derandomizers exceed the watermark a programmable threshold (typically 80 % of the total size) – a so-called watermark. The BUSY from the derandomizers have a programmable mask allowing the BUSY from one or more derandomizers to be ignored.

The CTP as a sub-detector is however not the cause of much dead-time: the main constraint is the trade-off between the CTP event size and the bandwidth of the S-Links to the DAQ and to the ROIB, that operate at 40 Mwords/s (32 bit). With the typical readout window of 3 bunch crossings during normal data-taking (Section 4.4.2.2), the CTP event size is 214 words, which sets the limit for the CTP trigger rate to $\sim$ 186 kHz, which is comfortably above the target Level 1 rate of 100 kHz.

4.7. Partitioning of Resources

As part of the Phase 1 upgrade, support for partitioning of the CTP's resources was added to better accommodate detector commissioning and calibration by allowing multiple concurrent users [58]. The most important functionality has been triplicated to support up to three simultaneous

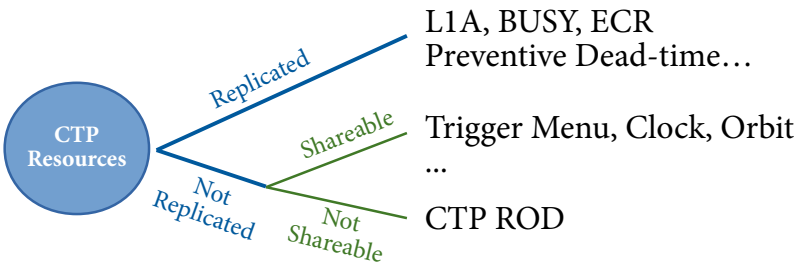


Figure 4.10: Depiction of classification of the CTP's resources and functionality for partitioning with examples, showing classification for common resources and functionality.

partitions, however not all resources are replicated and not all resources are shareable. In the following the details of the partitioning of the CTP resources as well as its implication for the use of the CTP's functionality will be discussed. A distinction is commonly made between the partition using the CTP readout and the other partitions. The partition using the CTP readout is called the *primary partition* while the others are called *secondary partitions*. As the primary partition is used for data-taking it is, from a design perspective, ascribed more importance than the secondary partitions. It is not possible to re-partition the CTP during data-taking: if a partition using all the resources of the CTP has been defined it is not possible to define secondary partitions.

For the discussion of partitioning, the CTPs resources can be divided into three groups as illustrated in Figure 4.10: replicated functionality, non-replicated functionality, with a sub-division of the latter into shareable resources and non-shareable resources.

Each partition in the CTP has its own busy logic and L1A generation. The description of the trigger path and the busy logic given in Section 4.4 and Section 4.5, respectively, holds true for the primary partition. However, for the secondary partitions, there are slight modifications. Many of the differences are caused by the lack of support in the CTPMI for partitioning of resources. Parts of the functionality provided by the CTPMI for the primary partition is instead emulated in the CTPCORE – primarily the ECR generation and the On Demand busy[4]. A depiction of the busy tree

[4]The On Demand busy of the CTPCORE is further used to emulate the ECR busy generated by the CTPMI on both sides of the ECR signal for the secondary partitions.

for the secondary partitions is shown in Figure A.1.1. The main differences, as compared to Figure 4.9, is the lack of the CTP ROD, the CTPMI. Most monitoring capabilities are however replicated as shown. For the primary partition, the On Demand busy functionality of the CTPMI is preferred over the On Demand in the CTPCORE, as the CTPMI provides sequencer based monitoring of the On Demand Busy, where the CTPCORE only provides a probe. Each partition implements its own simple- and complex dead-time.

The trigger menu is shared among all partitions. That is, all 512 trigger items after prescaling are triplicated in the CTP hardware and directed to the per-partition veto logic. In relation to Figure 4.6, this means that the veto group on the right most side is replicated three times. For each partition a programmable subset of the trigger items are considered before the per-partition dead-time is applied and the per-partition L1A is obtained.

In the description of the trigger path and the busy tree, given in Section 4.4 and Section 4.5, respectively, the role of the CTPOUT was simplified slightly to avoid the complications of multiple partitions. The COM bus carries the TTC signals as well as the received BUSY and calibration requests for each partitions. The CTPOUTs are additionally responsible for the routing of the TTC signals of each partition to the corresponding sub-systems as well as receiving BUSY and calibration requests for each sub-system and routing it back on the COM bus. That is, for each of the five cables on the front-panel, of each of the four CTPOUTs, one of the three L1A signals is chosen, depending on which partition the sub-detector at the other end of the cable is configured to belong to. Like so, the busy received from each sub-detector system is routed on to COM bus depending on the per-cable configuration of partition.

For both shareable and non-shareable resources and functionality of the CTP, the hardware support for enforcing the partitioning of the resources is generally limited: the only strict enforcement of partitioning in the CTP hardware relates to the routing of signal on the COM bus (e.g., the L1As and the BUSYs). Consequently, most of the partitioning is carried out by the CTP control software including enforcing partition boundaries and thereby preventing interference from other users.

Chapter 5

Monitoring in the CTP

Monitoring in the CTP is paramount for the operation of ATLAS, for the integrity of the recorded physics data, and for the later analysis of the recorded data. In particular the monitoring data provided by the CTP has the following use-cases:

Luminosity Block boundary determination used for any comparison between data recorded as part of ATLAS TDAQ and other experimental data, e.g., for matching and comparison of ATLAS data with LHC (beam) specific data.

Dead-time correction factors used by analyses for normalisation of a recorded data sample to the delivered integrated luminosity.

Busy monitoring to provide instantaneous dead-time information to the operators in the control room to prevent data loss due to ROD busy or back pressure and generally to ensure proper operation.

Luminosity corrections for providing an efficient online luminosity estimator and for providing an independent normalisation between ”dead-time aware” and ”dead-time blind” methods for luminosity determination.

Rate monitoring of the trigger bandwidth utilisation used for optimising the recorded physics output and by the operators in the control room to verify that operational conditions are stable.

Timing monitoring of trigger and timing signals to ensure that ATLAS sub-detectors trigger and record data from the desired bunch crossings.

State monitoring of the CTP hardware. This covers everything from configuration parameters and buffer occupancies to temperatures and voltages. The state monitoring data are primarily used to check the health and proper operation of the CTP hardware. The data is also used for debugging of CTP related issues.

This chapter covers the motivation for these types of monitoring as well as how the CTP hardware is being facilitated for monitoring in each case. Monitoring of dead-time is central to several of the above use-cases and this chapter begins with a general motivation of dead-time monitoring and the concept of luminosity blocks, before diving deeper into each of the above mentioned use-cases. The last section of the chapter deals with monitoring of the CTP internals as well as the possibilities for cross checks.

5.1. Dead-time Correction and Luminosity Blocks

In a data set corresponding to a certain integrated luminosity, L, the number N of events recorded belonging to a certain process is proportional to the product of the integrated luminosity and the production cross section σ for the particular process:

$$N = \sigma \int \xi(t)\, \mathcal{L}(t) dt \,. \tag{5.1}$$

where $\mathcal{L}(t)$ and $\xi(t)$ are the time-dependant luminosity and over-all efficiency function, respectively. The efficiency function can be expressed as the product of experiment and analysis specific efficiencies:

$$\xi(t) = \frac{A\,\epsilon\,(1-d)\,f}{p} \,, \tag{5.2}$$

where A is the acceptance of the analysis as imposed by geometric and kinematic cuts, ϵ is the fraction of events within the acceptance that pass the trigger selection and the offline selection, d is the experiments dead-time, p is the effective (product of Level 1 and HLT) prescale of the triggers used to record the data set, and f is a correction factor for failures or event loss during online data-taking or in offline processing. In reality most of these factors are time dependant and can not be moved outside the time integration of the luminosity. It is however possible to find short periods of

time for which the experimental conditions can be assumed constant. This in turn allows the time integral in equation (5.1) to be approximated as a sum over time intervals of approximately constant experimental conditions:

$$N = \sum_i N_i = A\,\sigma \sum_i \frac{\epsilon_i\,(1 - d_i)\,f_i\,L_i}{p_i} \tag{5.3}$$

where N_i and L_i are the number of events and the integrated luminosity, respectively, during the time period i.

The time intervals of approximate constant experimental conditions are called luminosity blocks. For the more common case of a cross section measurement, the above equation can be rearranged to read:

$$\sigma = \frac{\sum_i N_i}{A\ \sum_i \epsilon_i\,(1 - d_i)\,f_i\,L_i\,p_i^{-1}}\,, \tag{5.4}$$

where the sum, in both numerator and denominator, must be over *all analysed luminosity blocks* and not only those containing signal events.

The determination of d_i and p_i is based on the CTP monitoring and is a central part of this thesis. Section 5.3.2 details how the dead-time measurement is further used for luminosity corrections.

5.2. Luminosity Blocks and Boundaries

The luminosity block is the unit of experimental time in ATLAS. Each run is divided into several gap-less luminosity blocks of varying lengths. Each luminosity block within a run has a luminosity block number, starting at one and incrementally increased at each transition. During a luminosity block, configuration values and experimental conditions are roughly constant. The length of each luminosity block is variable: Luminosity blocks are issued periodically or per request. The typical duration of luminosity block is chosen to be 1 min and the shortest allowed time between transitions is 15 s.

5.2.1. Length of a Luminosity Block

There are several other experimental considerations, besides the requirement of stable experimental conditions, that affects the choice of the nominal length of luminosity block, the most important of which will be outlined here.

The upper bound on the length of a luminosity block is determined by the drop in luminosity and is $\mathcal{O}(1\,\mathrm{h})$. In one hour the luminosity typically drops $\mathcal{O}(1\%)$. However, as a luminosity block is the smallest unit of data that can be discarded in case an error occurs, luminosity blocks with a length of $\mathcal{O}(1\,\mathrm{h})$ could potentially results in large data loss.

There are further considerations that impact the length of a luminosity block: While there is a desire from a data quality perspective to keep luminosity blocks short to minimise data loss, a minimum amount of data needs to be recorded in order to assess quality of the recorded data.

Another more practical set of considerations arise from the fact that raw data is stored in one file per luminosity block. Thus there are file-system considerations such as the number and size of files to consider.

A duration of 1 min was acceptable to all parties and is used in ATLAS as the nominal length of a luminosity block.

5.2.2. Luminosity Block Transitions

Any changes to configuration, such as a change of prescales, can only happen in the transition from one luminosity block to the next, as the experimental conditions must be constant during a luminosity block, During the luminosity block transition ($\sim 5.3\,\mathrm{ms}$), no triggers can be issued leading to a small dead-time. In order to minimise the introduced dead-time it is important to make the luminosity block transition time as short as possible. During the luminosity block transition monitoring of dead-time related information as well as all trigger counters are read out in order to obtain exact per luminosity block measurements for dead-time corrections.

5.2.3. Luminosity Block Creation

The CTP is responsible for the creation of luminosity blocks. The boundaries of the luminosity block in time needs to be determined precisely for use with luminosity measurements that are independent of ATLAS TDAQ and thus do not operate with the concept of luminosity blocks. The CTP uses a GPS based timing module [61] to obtain a precise timestamp for the beginning and end of each luminosity block. By design, the luminosity blocks are gap-less: the end of one luminosity block is the beginning of the next luminosity block.

5.3. Dead-time Measurement

Along the trigger path, the CTP implements counter based monitoring that is used to obtain the number of triggers before prescale (TBP), after prescale (TAP), and after veto (TAV), for all trigger items. In addition, the CTP has a limited number of per-bunch counters that can be used for monitoring of selected trigger items. The trigger counts can be related to the dead-time correction factor used to normalise integrated luminosity of a recorded data set to the corresponding delivered luminosity. The following discussion prepares the grounds for a later discussion and cross check of the calculation of dead-time correction factors in ATLAS.

5.3.1. From Trigger Counts to Dead-time Fraction

Consider a luminosity block spanning the duration of n_{turn} LHC turns. The delivered luminosity during this period can be written as

$$L_{\text{delivered}} = t_{\text{BS}} \cdot \sum_{k=1}^{n_{\text{turn}}} \sum_{i=1}^{n_{\text{BC}}} \mathcal{L}_{ik} \tag{5.5}$$

where $t_{\text{BS}} = 25\,\text{ns}$ is the bunch spacing, $n_{\text{BC}} = 3564$ is the number of bunches in a turn, and $\mathcal{L}_{ik}$ is the instantaneous luminosity in the $i^{\underline{\text{th}}}$ bunch crossing of the $k^{\underline{\text{th}}}$ turn. Under the general assumption of constant experimental conditions during a luminosity block, it is meaningful to substitute $\mathcal{L}_{ik}$ with the turn-averaged instantaneous luminosity:

$$\mathcal{L}_i = \frac{1}{n_{\text{turn}}} \sum_{k=1}^{n_{\text{turn}}} \mathcal{L}_{ik} \,, \tag{5.6}$$

to obtain the following expression for the delivered integrated luminosity:

$$L_{\text{delivered}} = t_{\text{BS}} \cdot n_{\text{turn}} \sum_{i=1}^{n_{\text{BC}}} \mathcal{L}_i \,. \tag{5.7}$$

The luminosity seen by the trigger T can be written as:

$$L^T_{\text{bunch}} = t_{\text{BS}} \cdot n_{\text{turn}} \sum_{i=1}^{n_{\text{BC}}} l_i^T \, \mathcal{L}_i \,. \tag{5.8}$$

where l_i^T is the (turn-averaged) per-bunch live-time fraction seen by the trigger T. The live-time fraction l is generally related to the dead-time fraction d via the relation:

$$l = 1 - d\,. \tag{5.9}$$

Under the assumption that *the dead-time (or similar, live-time) is distributed uniformly on all filled bunches*, the factor including the per-bunch live-time in (5.8) can be taken outside the sum, and replaced with the luminosity block average live-time:

$$L^T_{\mathrm{LB}} = t_{\mathrm{BS}} \cdot n_{\mathrm{turn}} \cdot l^T \sum_{i=1}^{n_{\mathrm{BC}}} \mathcal{L}_i\,. \tag{5.10}$$

The average live-time l^T (per-bunch, per-luminosity block, or otherwise) seen by the trigger T can be written in terms of the observed trigger counts (in said period):

$$l^T = \frac{n^T_{\mathrm{TAV}}}{n^T_{\mathrm{TAP}}}\,, \tag{5.11}$$

where n^T_{TAV} and n^T_{TAP} are the TAV and TAP counts seen by the trigger T during the period considered. The dead-time correction factor, or live-time fraction, for the trigger T can also be defined as the fraction of the luminosity seen by the trigger:

$$l^T = \frac{L^T}{L_{\mathrm{delivered}}}\,. \tag{5.12}$$

Substituting the two different expressions for L^T given in (5.8) and (5.10) into (5.12) and using (5.11) to substitute the dead-time leads to two different expressions for the dead-time correction factors:

$$l^T_{\mathrm{bunch}} = \frac{\sum_{i=1}^{n_{\mathrm{BC}}} l^T_i\,\mathcal{L}_i}{\sum_{i=1}^{n_{\mathrm{BC}}} \mathcal{L}_i} = \frac{\sum_{i=1}^{n_{\mathrm{BC}}} \frac{n^T_{i,\mathrm{TAV}}}{n^T_{i,\mathrm{TAP}}}\,\mathcal{L}_i}{\sum_{i=1}^{n_{\mathrm{BC}}} \mathcal{L}_i}\,, \tag{5.13}$$

$$l^T_{\mathrm{LB}} = \frac{n^T_{\mathrm{TAV}}}{n^T_{\mathrm{TAP}}}\,. \tag{5.14}$$

where $n^T_{i,\mathrm{TAV}}$ and $n^T_{i,\mathrm{TAP}}$ are the TAV and TAP counts for trigger T in bunch crossing i. Equation (5.13) is a per-bunch luminosity weighted average live-time fraction while equation (5.14) is the simple average live-time fraction. The two equations will differ in case the per-bunch live-time fraction l^T_i (or similarly the per-bunch dead-time, d^T_i) and the per-bunch luminosity

$\mathcal{L}_i$ are correlated. Besides the difference in interpretation, the two expressions impose very different requirements for the level of information detail required: equation (5.13) requires knowledge of both luminosity and per-trigger dead-time at the bunch-to-bunch level, while equation (5.14) requires knowledge of per-trigger dead-time at the luminosity block level only. Thus, if the assumption, that the dead-time is uniformly distributed over all bunch-crossings, hold, $\mathcal{L}_i$ and d_i^T are uncorrelated and the calculation of the dead-time correction factors can be greatly simplified.

There are several effects that could lead to $\mathcal{L}_i$ and d_i^T being correlated and thus to violation of the assumption:

Shadowing Bunches with higher instantaneous bunch luminosity $\mathcal{L}_i$ are more likely to cause a trigger to fire. Due to the simple dead-time, the following bunch crossings will see more than average dead-time as a result.

Long Gaps The first bunch following a gap longer than the simple dead-time will never experience any simple dead-time. The first bunch following a long gap will also be less likely to experience complex dead-time or ROD busy: the extended time without triggers allows the buffers (real or emulated) time to empty.

Variations in instantaneous bunch luminosity $\mathcal{L}_i$ are to be expected as the per-bunch beam currents and emittance are not uniform.

The official way to calculate the dead-time correction factors makes use of the per-luminosity block counters, equation (5.14). The monitoring capabilities of the upgraded CTP allow testing the validity of these assumptions. Later, in Chapter 7, the validity of this and other assumptions made will be discussed.

5.3.2. Luminosity Corrections

Another related use of dead-time correction factors is the luminosity correction factor used to relate the delivered luminosity to the recorded luminosity. This is also important for how ATLAS determines luminosity online. The expression for the delivered luminosity in a luminosity block spanning n_{turn} is given in equation (5.7). The recorded luminosity can, in analogy with (5.8), be expressed as:

$$L_{\text{recorded}} = t_{\text{BS}} \cdot n_{\text{turn}} \sum_{i=1}^{n_{\text{BC}}} (1 - d_i)\, \mathcal{L}_i \,, \tag{5.15}$$

where d_i is the turn-averaged total experimental dead-time in the $i^{\underline{\text{th}}}$ bunch crossing. In relation to d_i^T, which describes the dead-time fraction per-bunch seen by a particular trigger T, d_i is the dead-time fraction per-bunch seen by (the OR of) all triggers.

The experimental dead-time can be separated into a bunch-dependent component $d_{i,\text{dep}}$ and a bunch-independent component d_{indep}. Equation (5.15) can then be written as:

$$L_{\text{recorded}} = (1 - d_{\text{indep}}) \cdot t_{\text{BS}} \cdot n_{\text{turn}} \sum_{i=1}^{n_{\text{BC}}} (1 - d_{i,\text{dep}})\, \mathcal{L}_i \,, \tag{5.16}$$

The luminosity correction factor D relating the delivered and recorded integrated luminosity can then be expressed as:

$$D = \frac{L_{\text{recorded}}}{L_{\text{delivered}}} = (1 - d_{\text{indep}}) \cdot \frac{\sum_{i=1}^{n_{\text{BC}}} (1 - d_{i,\text{dep}}) \cdot \mathcal{L}_i}{\sum_{i=1}^{n_{\text{BC}}} \mathcal{L}_i} \tag{5.17}$$

The total experimental dead-time is composed of the preventive (simple and complex) dead-times as well as the dead-time from sub-detector busy. The assumption that dead-time is equally distributed on all filled bunches implies that $d_{i,\text{dep}} \approx 0$. The affects of shadowing as well as the affect of long gaps, as discussed in Section 5.3.1, however implies that the preventive dead-times will have some bunch dependence. The monitoring capabilities of the upgraded CTP allow for per-bunch monitoring of the contribution from the individual sources of dead-time.

ATLAS utilises several ways of measuring luminosity. For the purpose of this discussion, they can be grouped into two types:

Dead-time aware methods rely on a data sample recorded by luminosity sub-detectors that are typically triggered by minimum bias triggers. As the data sample is obtained using ATLAS TDAQ the data are exposed to the full experimental dead-time. The data from recorded events can be used online (and with higher precision, offline) to estimate the recorded luminosity.

Dead-time blind methods rely on data that are not obtained via ATLAS TDAQ and thus do not experience the experimental dead-time. These methods are typically based on counters implemented directly on the luminosity monitors, such as BCM and LUCID, to count the number of hits per-bunch crossing. The hits provide a convenient handle on the delivered luminosity.

The minimum bias triggers, used for the dead-time aware methods, are typically heavily pre-scaled as only a fraction of the overall bandwidth is dedicated to minimum bias triggers. Thus, the dead-time aware methods are of limited use during data-taking as the precision of the luminosity estimates obtained from the recorded data will be limited by statistics. However, via the relation given in (5.17) an estimate of the recorded luminosity is obtained during data-taking, based on the dead-time blind methods and the experimental dead-time obtained from the CTP. Using the dead-time blind methods and the dead-time to obtain an estimate for the recorded luminosity has the added advantage of not requiring additional bandwidth to be allocated to minimum bias triggers. This method is further used offline for independent cross checks of the luminosity estimates obtained from the dead-time aware methods.

5.3.3. Dead-time Measurement Capabilities in the CTP

The live-time fraction, or conversely the dead-time fraction, of a trigger T can estimated from the number of triggers after veto n^T_{TAV} as well as after prescale n^T_{TAP}, as described in Section 5.3.1. In the CTP the number of triggers can be obtained from the counter based monitoring along the trigger path depicted in Figure 4.7.

Experimental dead-time is measured using the sequencers and per-bunch counters depicted in Figure 4.9, For the monitoring of experimental dead-time, the four sequencers on the top right (hand side) of Figure 4.9 are of particular interest, namely the ones measuring: the dead-time contribution from sub-detector ROD BUSY (RSLT), the simple dead-time (SMPL), and the two complex dead-times (CPLX0 and CPLX1). There are seven per-bunch monitoring blocks, each with a programmable mask, as indicated in Figure 4.9 on the top right of the figure, under the two final veto signals. The programmable mask is used to pick out the desired components of the

dead-time – the OR of which is then monitored. The default selection of dead-time sources used for the per-bunch dead-time monitoring is listed in Table 5.1. The per-bunch dead-time monitoring capabilities are replicated enough times that the contribution by the individual sources of dead-time, as well as all meaningful combinations of these, can be monitored at the per-bunch level. This offers several handles for measuring the experimental dead-time as discussed above and allow for cross check of the assumptions made.

	Result	*Simple*	*Complex 1*	*Complex 2*
Legend	**RSLT**	**SMPL**	**CPLX0**	**CPLX1**
Counter 1	✓	✓	✓	
Counter 2	✓	✓		✓
Counter 3	✓			
Counter 4		✓		
Counter 5			✓	
Counter 6				✓
Counter 7		✓	✓	

Table 5.1: Default selection of sources for the seven per-bunch dead-time counters. The logical OR of the selected sources is counted.

For the calculation of dead-time correction factors, the important values obtained from the counters and sequencers is the per luminosity block total values, i.e. the total number of bunch crossings in a luminosity block in which dead-time was incurred or similarly the total number of triggers (TAP and TAV) in a luminosity block. For the operators in the control room, the important numbers obtained from the sequencers are the instantaneous dead-time fractions. This is an example of what will later be discussed as the "Multi-use of Data".

5.4. Busy Monitoring

Monitoring of the busy tree is important for the operation of ATLAS. It provides health information about the state of ATLAS and its sub-detectors

and allows the operators in the control room to quickly identify the source of a problem: many operational issues will have the side effect of the affected system signalling BUSY to throttle the trigger rate.

5.4.1. The BUSY Tree

Like the dead-time, the ROD BUSY from the sub-detectors are gap-lessly monitored using sequencers, as indicated in Figure 4.9. The sequencers relevant for BUSY monitoring are distributed across three boards, namely the CTPOUT, the CTPMI, and the CTPCORE. The sequencers of the CTPOUT is used to monitor the ROD busy coming from ATLAS' sub-detectors. The sequencers in the CTPMI monitors BUSY introduced as part of the operation, primarily the BUSY around ECR generation discussed in Section 4.5.1 and the applied On Demand BUSY to be discussed in Section 5.4.2. The resulting BUSY is monitored using a sequencer in the CTPCORE.

For the operators in the control room, the sequencers are used to obtain the instantaneous busy fraction. For the busy fractions to be useful, all sequencers must be read out roughly at the same time, to provide a consistent picture, and at a frequency that allows the operators to react quickly in case of an operational issue. The sampling frequency of the sequencers is chosen to reflect this.

5.4.2. Busy On Demand

The busy and dead-time resulting from the busy On Demand in the CTPMI is monitored using sequencers.

However, the way the On Demand busy is used, here exemplified by its use for recovery of sub-systems, require additional bookkeeping and monitoring.

Any sub-detector may request the busy On Demand for recovery procedures. When the sub-detector no longer requires the busy to be raised it sends a request to the CTP to lift it. One could imagine a scenario where more than one detector system is needed to perform a recovery procedure that involves holding the trigger via the On Demand busy. In this case more than one system will request the On Demand busy. It is paramount for the successful recovery of all systems that the On Demand busy is only

released when all systems are done with their recovery procedure. Neither the CTPMI (nor the CTPCORE), however, has hardware support for handling and resolving requests from multiple sources for asserting the On Demand busy. Thus, handling the requests from the sub-detectors must be handled in software. The counting semaphore needed to obtain this synchronisation is implemented in the CTP control software and like so is the monitoring of what sub-system that has requested the On Demand and for what duration.

The CTP uses the Busy-on-demand in either the a) CTPMI (for the primary partition) or b) the CTPCORE (for the secondary partitions) to generate the BUSY as discussed in Section 4.7.

5.5. Monitoring of Timing Information

The operation of the Level 1 trigger processors and the readout of the sub-detector front-ends both rely on precise timing. The short spacing between bunches of 25 ns (or 7.5 m) require the timing of detector signals in the complete trigger and readout path to be well understood to within a few nanoseconds. The readout path begins with the distribution of the L1A to the sub-detectors. An important difference between the trigger and the readout path is that the trigger path operates at a fixed latency while the latency of the readout path will be different for all systems due to, amongst other things, the different length of cable connecting them to the CTP.

Given the size of ATLAS, a particle created at the interaction point, travelling at the speed of light, takes $\mathcal{O}$ (40 ns) to reach the outer parts of the detector. For a 25 ns bunch spacing this implies that particles from the next collision would already be travelling out through the detector, before all sub-detectors have finished the readout of the first collision. Precise timing information and monitoring is critical to ensure that each sub-detector is sampling data from the correct bunch crossing and at an optimal time within said bunch crossing.

Each sub-detector system, including the CTP, is responsible for the timing-in of their own system. For the CTP, this implies the timing-in of the CTP inputs, the CTP internal timing, as well as certain aspects of the global timing, such as the interface with the LHC and determination of the BCID. Due to the CTP's central role in clock distribution and triggering,

it is in a unique position to assist other systems in the timing-in of other systems. The detailed trigger monitoring capabilities of the CTP – e.g., the per-bunch monitoring – are particularly useful for timing-in of the trigger detectors.

Most of the timing-related monitoring and procedures described here are carried out once, whenever changes are made to ATLAS that affect the timing. Most commonly, these are delay changes along the trigger path due to local changes in firmware. Other more rare examples are cable changes or installation of new detectors, as was the case for IBL during Long Shutdown I. Once timed-in, the values are not expected to change from run to run. The relevant configuration parameters are however monitored frequently to ensure consistency, and to detect possible unexpected changes.

5.5.1. The CTP as Timing Reference

An important part of achieving synchronisation between all sub-detectors is the unanimously agreement upon the numbering of the BCIDs within a turn.

Each sub-detector system, including the CTP, is responsible for ensuring that their system is internally consistent, i.e. that all parts of the system internally agree on the numbering of bunch crossings within a turn. This is done at each system by injecting a trigger signal in a known arbitrary bunch crossing and ensuring that all parts of the system see the signal in the correct bunch crossing.

When internal consistency is achieved, the system is said to be timed-in with respect to a reference signal For the sub-detectors, the reference is the CTP. For the CTP, the reference is the LHC. For this, using a simple filling pattern, the LHC provides collisions in fixed BCIDs. By monitoring trigger rates per-bunch, the BCID offset can be determined.

5.5.2. Types of Timing and their Monitoring

Timing monitoring can conceptually be broken down into two types:

Input timing : the timing related to the CTP's inputs. Primarily used to ensure that the inputs to the CTPINs are aligned in time with respect to each other and latched properly with the CTP clock, as well as to check that inputs are aligned with the bunch pattern to ensure that triggers occur in the correct bunch crossing.

Round-trip timing : the sub-detector specific time from a particular bunch crossing to the arrival of a L1A. This amounts to the time up until the issuance of the L1A by the CTP plus a sub-detector specific fixed latency, mostly dependant on the cable length between the CTP and the sub-detector. The time up until the L1A is emitted depends on the latency introduced before and inside the CTPINs, as well as a fixed contribution, the processing time, within the CTP.

The main distinction between the two is that configuration of the input timing is an internal CTP issue, while the round-trip timing involves downstream sub-detectors. The monitoring functionality of the CTP used are the same for both cases, though the way they are leveraged differs slightly.

5.5.3. CTP Internal Timing

For the CTP internal timing, there are two primary concerns:

- Sampling each input at the right clock edge;
- Aligning all inputs in time using the input pipeline buffers.

These steps are necessary for correctly processing the input. In order to determine the correct settings for the pipeline lengths and the clock edges to be used for strobing the input signals, the trigger path monitoring, described in Section 4.4.2, is utilised – in particular the per-bunch monitoring: by monitoring the input per-bunch (in the CTPMON) or the resulting trigger item per-bunch (in the CTPCORE) the change to the pipeline length can be determined as the difference between the BCID where the input (or trigger) is observed and where it is expected. If the input is sampled at the wrong clock edge (in the CTPIN), close(r) to the meta-stable region, the sampled input signal may occasionally fall in the wrong bunch crossing (an off-by-one error). Using the per-bunch monitoring, this too can be detected. The event monitoring of the CTPCORE, discussed in Section 4.4.2.2, is also useful for the timing-in of the triggers. The CTP event data contains a full trigger record – a 512 bit mask of which triggers fired – for the bunch crossings within the readout window (typically 4 BC). This record is used to determine the relative offset between a particular trigger and a given reference.

Lastly, monitoring of the configured values (pipeline lengths and clock edges) is a part of the general configuration monitoring.

5.5.4. Clock Monitoring

The CTP receives the clock signal from the LHC as described in Section 4.3. As both source and frequency of the clock signal can change, close monitoring of the clock frequency is important for troubleshooting issues related to timing and clock distribution.

Monitoring of the clock frequency in the CTP happens in the CTP-CORE by counting clock edges. The implementation uses a sequencer-like setup: a counter counts the number of clock edges, at a programmable frequency a sampler samples the counter value, resets the counter and transfer the counter value to a FIFO for later readout. As for the sequencer, the sampling is gap-less. However, unlike the sequencer, the sampler is not tied to the bunch clock but rather to a clock derived from a GPS receiver. Thus, each FIFO sample corresponds to the number of clock edges during an integration time of $\Delta t = \frac{1}{f}$ where f is the sampler frequency. In order to obtain a sub-Hz resolution, an integration time of $\mathcal{O}(1\,\mathrm{s})$ is required.

5.6. Trigger Rates

The trigger rates are derived from the same counters along the trigger path as are used for monitoring the total number of triggers before and after prescale and veto as discussed in Section 5.3.

The instantaneous trigger rates per trigger input and item is an invaluable tool for managing the trigger bandwidth utilisation: the Level 1 trigger is designed to operate at a maximum accept rate of 100 kHz. During a run the luminosity of the beams will drop as the number of particles in each beam depletes, causing the production rate for physics processes – and thereby the various trigger rates – to decrease. In order to maximise the physics output, the bandwidth should be fully utilised and the trigger prescales adjusted to reflect both the data-taking goals and the decreasing trigger rates.

As discussed in Section 4.3, the average trigger rate can be estimated from the trigger counter(s) and the associated turn-counter. The upper bound for reading out the per input and per item counters to avoid overflow is $\mathcal{O}(1\,\mathrm{min})$. However, for most operational purposes a rate average over one minute provides too poor time resolution. For a proper time resolution on the trigger rates the counters should be read out much more often, i.e. at

a time scale comparable to human reaction time. The rate counters are typically read out roughly every 5 s.

Like the busy fractions, the trigger rates are a useful indicator for the health of ATLAS and its sub-detectors. Many operational issues results in a higher or lower than expected number of triggers. These could include:

- Hot cells of a calorimeter leading to additional spurious rate;
- Drop of magnetic field strength leading to increased rate of high p_{T} muons;
- Unbalanced gain on one side of the detector causing increase rate of missing energy triggers.

The trigger rates (per-bunch) are also used for timing-in detector subsystems as discussed in Section 5.5.

Chapter 6

The CTP Monitoring Software

The main focus of this chapter is the considerations behind the implemented architecture of the CTP software, with a strong emphasis on the monitoring software.

The design of the CTP software architecture and the implementation of particularly the monitoring-related applications and services has been an important part of my work.

6.1. The CTP Software Architecture

The monitoring software does not exist or operate in isolation. While some of the requirements and constraints to be discussed in this chapter are unique to the monitoring software, a lot of similar considerations went into the design of the other elements of the CTP software architecture. To better understand some of the later design choices, a brief introduction to the CTP software architecture will be provided here. A technical description of the full architecture is given in [1].

One of the main differences between the previous architecture and the new one, is the abandonment of one big monolithic application in favour of a microservice architecture. The two main arguments for this choice was i) added stability; and ii) facilitating multiple users of the CTP hardware. The new microservice architecture is depicted in Figure 6.1: in this architecture software either lives in the CTP-centric infrastructure domain (top) or in one of multiple (up to three) user-centric domains (bottom). A user domain with usage right to the CTP readout is used for data-taking with

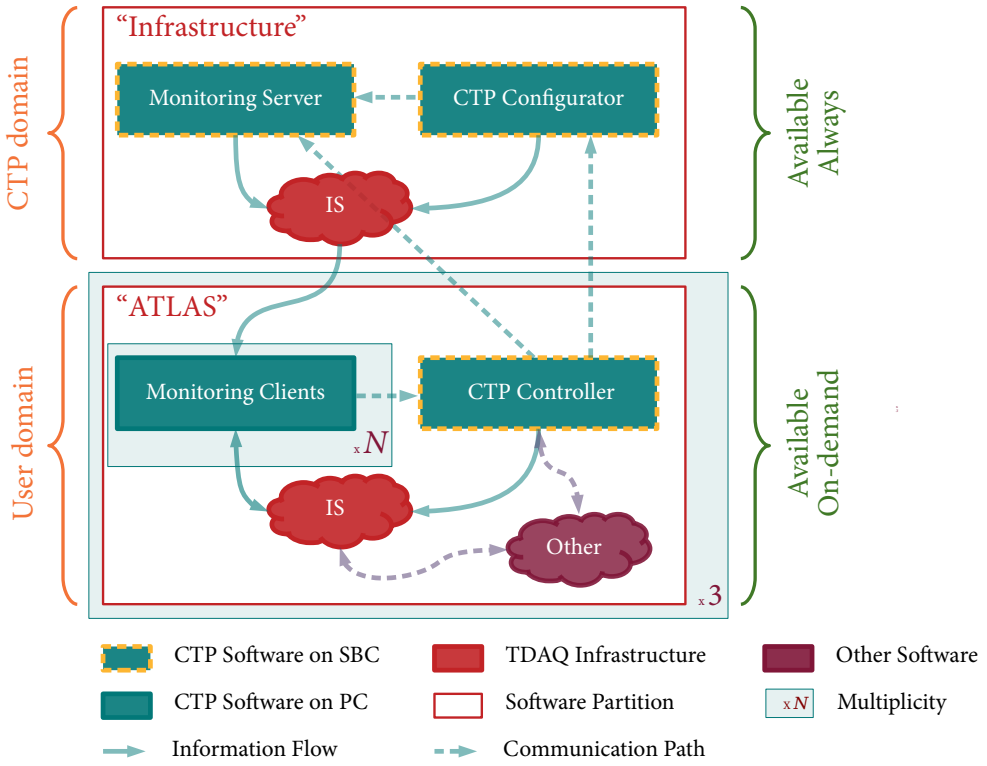


Figure 6.1: The full CTP software architecture. The architecture consists of a CTP-centric domain and up to three user-centric domains. With exception of the indicated communication and data flow, domains are unaware of each other.

ATLAS. The user domains are created and destroyed on demand and each only exists for as long as the user session. Consequently, if there are no users of the CTP no user domains exists. In contrast, the software of the infrastructure domain is running at all times. The CTP domain is generally domains and the User domains are agnostic to the existence of other user domains. In order to avoid conflicts between users, the CTP Controller of a User domain must request access and usage rights, to resources as well as configuration changes from the CTP Configurator in the CTP domain. The CTP Configurator maintains the internal resource reservation and applies configuration changes per request. The monitoring software is split in two parts:

A monitoring server which lives in the infrastructure context and is

(mostly) agnostic to the existence of the users of the CTP, the state of data-taking etc.;

A set of monitoring clients that lives in the user domain and "cherry-picks" the information, published by the monitoring server, relevant to the particular user.

The separation of monitoring software across domains allow for some elegant solutions to a number of usage and access problems as will be discussed in the following.

6.2. Overview of Requirements

The requirements for the monitoring software arise from the various use-cases of the monitoring data. For the purpose of this discussion, the use-cases can be roughly divided into two groups: the operational use-cases and the physics use-cases. The operational use-cases can be subdivided into the uses of monitoring data for the operators in the control room and the uses of monitoring data for effective postmortem analysis: in case an operational issue is discovered it is important to have enough information archived about the state of the CTP to be able to debug and resolve the issue offline, away from data-taking. Having the ability to do retrospective postmortem analysis is important as it allows one to "go back in time" and analyse and resolve issues that might either not have been discovered right away or do not critically impact data-taking (e.g., issues that can be treated symptomatically during data-taking but requires later follow-up).

The uses of the monitoring data that are important for running the experiment, and the requirements these impose, are:

Busy Rates The instantaneous busy fraction reported by detectors as well as the fractional dead-time (preventive or otherwise) of the experiment is a direct way of measuring the health of the experiment: if a detector is experiencing problems or can not keep up with the trigger rate it should signal BUSY to the CTP that then applies dead-time accordingly.

To aid the operators, the BUSY-state of the experiment, including the BUSY of all sub-systems and the over-all experimental dead-time, must be presented visually and made accessible to the operators in

ways that harmonise with the normal work-flow. The busy rates and dead-time readings should be updated frequently enough that the values reflect the current state of the experiment, preferably every few seconds.

Trigger Rates The instantaneous rates of the trigger inputs and of the trigger items along the trigger path are another important direct way to monitor the health of the experiment and the data-taking effort: increased dead-time leads to a drop in the trigger rates, certain detector problems can affect the trigger rate and even when everything is functioning, the trigger rates are a useful way of assessing the bandwidth utilisation for the various triggers and ensure that the recorded data match the data-taking goals.

To aid the operators in the control room, the trigger rates of the individual trigger items must be visualised, both as instantaneous rates (single numbers) and as function of time (graphs). Like the busy rates, the trigger rates should be read out often enough that the values reflect the current state of data-taking, which, also for the trigger rates, implies every few seconds.

Configuration Changes For the operators in the control room, to make sense of the dead-time fractions and trigger rates, they must also know the relevant configuration values. For the trigger rates, the relevant values would typically be the prescale values. For the dead-time fractions it would be the number of bunch crossings to be vetoed by the simple dead-time and the bucket size and leak rates for the complex dead-times.

Unlike the trigger rates and the busy rates the configuration is not expected to change very often during a run: the prescale values are only allowed to change between luminosity blocks and the dead-time settings are only allowed to change between runs. However, the read-out should be frequent enough to detect any change in configuration in reasonable time for the operators to act upon it: if the dead-time settings are changed by erroneous software or by human error the change should be detected quickly.

While slowly changing, the configuration parameters related to dead-time and triggering should be read out frequently enough to quickly

detect changes, at the very least once a minute (the typical length of a luminosity block), but preferably more often.

The operational use-cases of the monitoring data that is primarily of use for post-mortem analysis are:

State Monitoring The state of the CTP in the context of postmortem analysis encompasses both the operational state of the CTP (the readout and the configuration values for all programmable settings) and the physical state of the CTP (the voltage, temperature, cables connected where applicable).

Trigger Phase Monitoring The trigger inputs at the CTPIN are strobed to the CTP clock at either the rising or falling clock edge, depending on the phase of the signals with respect to the bunch clock, as described in Section 4.4.1.1. The phase monitoring is particularly important after periods where changes that alter the timing have taken place, such as Long Shutdowns and firmware updates. The phase of the inputs should be monitored to avoid issues with meta-stability where trigger inputs are strobed on the wrong edge.

Clock Monitoring As described in Section 4.2.2 the frequency of the LHC bunch clock changes during a ramp. For some systems the change of clock frequency can lead to spurious behaviour. In order to better understand and debug these issues, the clock frequency and its change should be monitored.

For most of these types of monitoring it is difficult to determine a minimum desired monitoring frequency as it would assume prior knowledge of the issues that could arise. One exception to this is the clock monitoring: with a clock frequency of 40 MHz and nanosecond timestamps from the GPS receiver, integration times of ~ 1 s is required for sub-Hz precision. The other types of monitoring should be read out on a "best effort" basis. Unlike the other operational requirements, the monitoring of the **State Monitoring**, **Trigger Phase Monitoring**, and **Clock Monitoring** must be carried out even when there is no run or colliding beams.

The use of the monitoring data for post-mortem analysis impose an additional set of requirements:

Archiving of Monitoring Data All monitoring data should be archived, preferably at the same time-wise granularity as it is being read out.

A Dead Man's Switch In case of a CTP related issue, a method to trigger a complete monitoring readout of the CTP should be in place to provide a snapshot picture of the state of the CTP.

The requirements imposed by the operational use-cases are soft requirements in the sense that even without appropriate monitoring in the control room the recorded data is (likely) still usable. The hard requirements are imposed by the need for correctly determining the dead-time, as discussed in Section 5.1, and by the potential data loss due to counter overflow, as discussed in Section 4.3. This leads to the following hard requirements:

Dead-time The total number of triggers fired before and after veto as well as the total number of bunch crossings with experimental dead-time per luminosity block need to be determined. This implies that all sequencers (for busy monitoring) and counters (for trigger monitoring) must be read out at the end of the luminosity block. Further, as high precision is needed, precision-loss due to counter or buffer overflow is unacceptable.

Counter overflow The counters used for monitoring the number of triggers fired are effectively 31 bit counters and will, at 40 MHz, overflow in ≈ 53 s. This implies that all counters must be read out at least once per 53 s and preferably much more often to avoid the risk of overflow.

As can be seen, these requirements are well within the operational requirements: the sequencers will be read out during busy monitoring and the counters will be read out for the rate monitoring.

That requirements can be divided into hard and soft requirements reflects that not all of the monitoring is of equal importance. The hard requirements constitutes monitoring requirements that are critical for running the experiment. Having a distinction between critical and non-critical monitoring implies a prioritisation of monitoring routines that can be formulated as an additional requirement:

Critical Monitoring Monitoring of critical components (i.e. counters and sequencers) important for determining the dead-time goes before any other monitoring: given the choice, a pending critical monitoring task must be chosen over a pending non-critical one.

To ease the ability of honouring this, and to improve the overall stability the critical monitoring tasks must be separated from the non-critical ones.

6.3. Overview of Constraints

In addition to the main requirements outlined in the previous section, an additional set of requirements arise from the constraints on the implementation, imposed mainly by the CTP hardware and the integration with the CTP into the ATLAS infrastructure.

There are seven primary constraints, as visualised in Figure 6.2, namely:

Single Point of Failure: The CTP is a single point of failure for ATLAS operations, due to its central role in aspects of trigger and timing and as there is no fail-over. Fault tolerance and stability is paramount for the monitoring routines. While a potential crash of a monitoring routine could locally impact the data quality of the affected luminosity block, it is not allowed to impact other critical functionality provided by the CTP. The best way of reducing the consequences of code crashes is by factoring out functionality into several smaller programs. However this is complicated by the constraint on the number of concurrent **VME accesses**.

Limited Computational Resources: The SBC is resources limited and runs processes critical for data-taking. Moving as much software as possible away from the SBC is paramount to avoid resource starvation, and with it, delays in critical operations that might result in experimental dead-time or data loss.

The importance of moving software from the SBC was emphasised by the requirement of **Multi-Partition Support** that naively would require running multiple software instances for each partition.

VME Access: There are a number of issues or challenges regarding the use of VME. These can be divided into two sub-classes: issues or

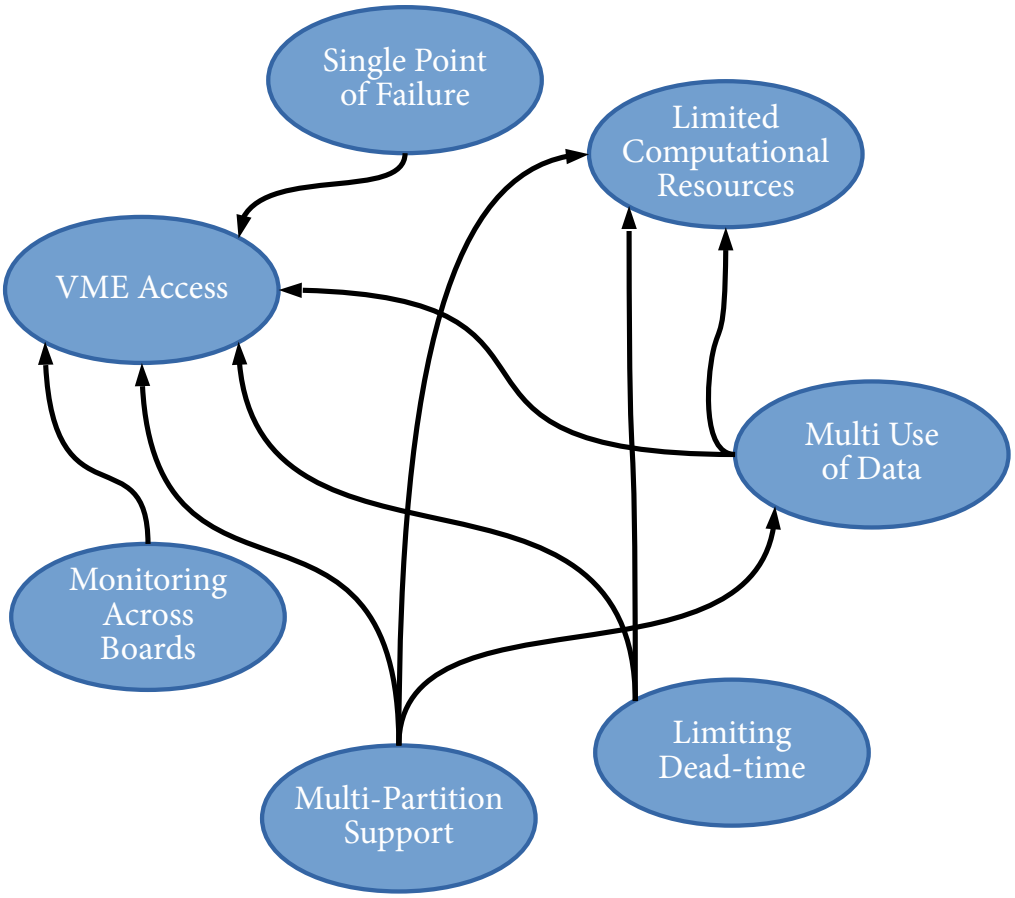


Figure 6.2: Diagram of software design constraints and their interdependence. The circles represent the various constraints. The arrows indicate a dependence on another constraint.

limitation with the VME technology itself and issues and limitations arising from the ATLAS-developed VME kernel driver. The VME technology has limited bandwidth. The VME bandwidth of roughly 40 MB/s implies that reading out a single per-bunch monitoring histogram[1] over VME, blocks the VME bus for more than an LHC beam revolution, potentially delaying any other operation or command between the boards of the CTP and the SBC.

The following issues with the ATLAS developed VME kernel driver imposed additional design constraints: 1) Only four programs can

[1]The CTPMON board has 160 per-bunch histograms.

simultaneously make use of the Linux kernel driver used for VME access. This is by far the strongest constraint on the design of the software. 2) Thread safety can not be guaranteed: the previous constraint can not be mitigated by spawning more threads. These limitations of the ATLAS VME driver code surfaced with the upgrade of all SBCs from 32 bit single-core processors to 64-bit multi-core processors.

Monitoring Across Boards: As detailed in Section 5.4 and Section 5.6 the sequencers and counters used for busy monitoring and trigger (input) rate monitoring are spread across multiple boards. In order to provide a consistent picture of rates or busy fractions across the full system the monitoring routine(s) need(s) to read out the same type of monitoring across all relevant boards in close succession: The operators in the control room are relying on all the values (trigger rates or busy fraction) as a whole to assess the current state of the experiment. If too much lag is introduced between the readout of values they no longer paint an accurate picture.

Multi Use of Data: In many cases the same monitoring data (e.g., counter values) are used for multiple purposes, either by the same user or by the multiple users of the CTP: The number of triggers for instance are used both for the instantaneous rates shown in the control room and integrated over the luminosity block for dead-time correction. As the trigger menu is a shared resource, multiple users will need the information about the instantaneous rates.

Multi use of Data impose an important constraint on the monitoring routines. The readout of both counters and sequencers is destructive in the sense that the counters are reset and the FIFOs are emptied as part of the readout procedure. This implies that if more than one entity is reading out the values, care must be taken such that the information is shared amongst all relevant parties to avoid wrong measurements. This is further complicated by **Multi Partition Support** as multiple users will need access to the same data.

Multi Partition Support: The CTP needs to facilitate multiple concurrent users and all of them need access to monitoring. Having multiple users introduce a number of constraints similar to those already

discussed, e.g., more users implies that more of the **Limited Computational Resources** will be used, more monitoring data and thus **VME access**es are needed, and multiple users monitoring the same (shared) resource constitute a variant of the **Multi Use of Data**.

Another implication of having multiple users is the allocation of the CTP's resources: safe-guards should be in place so that two users can not own the same CTP resource (trigger item, sub-detector, TTC partition, etc.) or accidentally steal a resource from another user.

These constraints were of key importance for the design of the architecture.

Limiting Dead-time: Read-out of trigger counters and busy sequencers must happen during luminosity block transition, as discussed in Section 5.2. The monitoring routines should be optimised for minimising the dead-time. This is complicated by the **Limited Computational Resources** and the restrictions on **VME Access**.

6.4. Significant Design Choices

This section summarises the main design choices made to meet the requirements and constraints laid out in the previous sections.

As can be seen from Figure 6.2, meeting the requirements are primarily limited by the **VME Access** and the **Limited Computational Resources**: most arrows are going to these two constraints and in fact no arrows are going from them. Resolving these two constraints first will make it easier to meet the remaining five constraints, removing most of the arrows from the diagram.

6.4.1. Limited Computational Resources

The issues with the limited resources of the SBC was mitigated by offloading all tasks that do not strictly need VME access to other more powerful computers. For the monitoring of the CTP, this creates a natural separation between the application that reads the raw data from the boards (and makes it available) and the applications that processes, formats, and display the monitoring data. It also allows the process that reads out the hardware to be completely agnostic towards the uses of the monitoring data.

The raw values read from the boards usually need some processing before they are useful to the consumer. E.g., before the instantaneous trigger rates can be displayed in the control room, the values of all the raw counters must first be normalised to the number of LHC beam revolutions before a trigger rate can be obtained. Each trigger rate must then be paired with the menu configuration to obtain a meaningful name for each item, i.e. the CTP hardware has no notion of the human-readable names, such as `L1_MU10`, ascribed to each of the trigger items. The only part of this that strictly needs to happen on the SBC is reading out the raw values: the arithmetic operations and formatting of data is better handled on a more powerful computer.

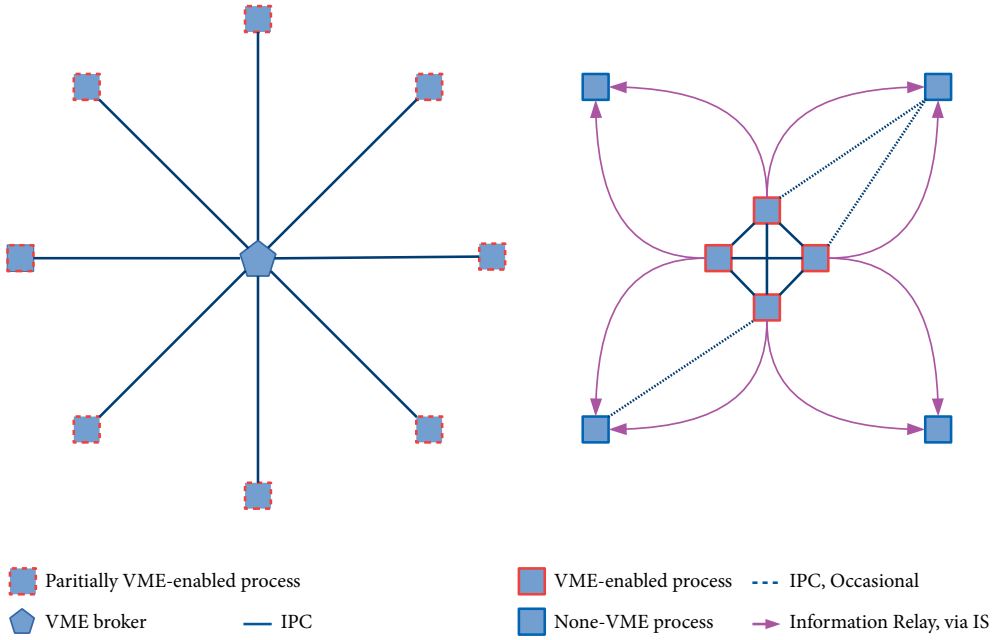
This leads to a setup with an agnostic monitoring server and a number of monitoring clients for processing the data. This setup also simplifies the **Multi Use of Data** as the process on the SBC responsible for reading out data is agnostic to the later use of the data. It also simplifies **Multi-Partition Support** as each partition (and user) will have its own set of monitoring clients. Further, it provides a simple solution for ensuring that e.g., **Clock Monitoring** is carried out, independently of whether there is a user of the system, and that buffers do not fill or overflow: as long as the monitoring server is running the monitoring of the hardware is running and available.

6.4.2. VME Access

There are two issues with the VME, namely the limited number of concurrent accesses and the finite bandwidth. There are two design patterns, depicted in Figure 6.3, each of which can be used to meet these constraints:

Broker Pattern: A centralised broker service maintains the access/use right to the VME and any service that needs access to the VME bus negotiates the acquisition, use, and freeing of the VME resource with the broker. In this setup, the various services remain agnostic of each other, and there is no communication between services besides the communication with broker.

Divide & Rule: A small number of services (below the critical number of VME accesses) maintains permanent access of the VME. These services then provide the remaining services with information and/or



(a) Broker Pattern (b) Divide & Rule Pattern

Figure 6.3: Diagram of VME access strategies.

executes commands on behalf of other services. The interaction pattern between the processes is less obvious and needs to be formalised on a case-to-case basis.

The Divide & Rule pattern was chosen over the (implementation wise) simpler Broker Pattern. One of downsides of the Broker Pattern is that all the applications that would request access to the VME bus still have to run on the SBC, which is in conflict with the **Limited Computational Resources** and the solution strategy outlined in Section 6.4.1. With the Divide & Rule pattern the processes that do not require VME access can be offloaded to more powerful computers.

The Divide & Rule method represents staying firmly within the boundaries of the available VME resources. The main downside to this pattern is the additional complexity of designing the Inter Process Communication (IPC) interfaces between the services: the protocols for requesting data, coordination between processes, as well as exception and error handling in the communication between individual processes. One example of such

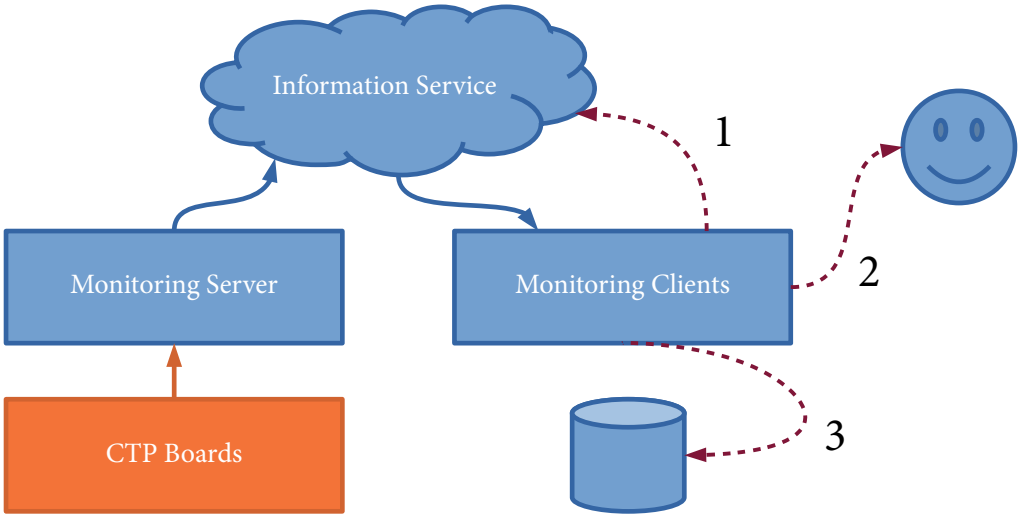


Figure 6.4: Conceptual depiction of the CTP monitoring. A server application reads data from the CTP boards and publishes the raw values to the Information Service (IS). Multiple monitoring clients exist for 1) processing and republishing; 2) present data in the control room; or 3) archive data.

IPC is where the service responsible for doing the luminosity block transaction needs to coordinate (and negotiate) with the monitoring service, that VME-heavy monitoring (such as per-bunch monitoring) is postponed until after the luminosity block transition. Drafting these protocols adds a layer of complexity to this design pattern but makes for more effective access patterns.

6.5. Overview of Final Design

Figure 6.4 illustrates the constituents of the final design for the CTP monitoring software. In the proposed and implemented design, only one monitoring program runs on the SBC, namely the monitoring server. The *monitoring server*, reads the raw values from the boards. The raw values are then published to the Information Service (IS) [76], an infrastructure service provided by ATLAS DAQ group for holding temporal data during a run. Since the beginning of Run 2, the DAQ group has been running the *pbeast* service [77] at Point 1, which, with the exception of the publications of histograms[2], archives the IS publications. CORBA [78] is used as

[2]The per-bunch histograms published by the monitoring server is archived by a monitoring client

the implementation for IPC. In CORBA, the interface between processes is defined in a contractual manner. The contract is written in IDL and can then be compiled into a number of language extensions, including C++ and Java.

A variety of *monitoring clients* consume the information published to IS and do either of three things

1. perform a calculation or process the data and republish it to IS.

   Example clients are services that take the raw counter values as published by the server, and calculates a rate or performs a time integration per luminosity block and publishes this value back to IS. Another example is the clients that take the trigger rates, as calculated and published by previous clients, pairs the trigger rates with the names used in the trigger menu, and re-publishes the combined trigger rate and trigger name;

2. display the value(s) to the operators in the control room.

   An example client is a program that displays the trigger rate (as published by a type (1) application) to the operators in the control room; or

3. write the values to persistent storage.

   An example of this type of client is a program that writes a time integrated value to a database for offline use.

6.5.1. Implications for Remaining Constraints

With the proposed solution to the two biggest constraints, several of the remaining constraints are resolved as well:

Single Point of Failure: The distributed nature of the design minimises the probability of any single mistake to jeopardise data-taking[3]. Most of the programs and services can be restarted independently of each other and without affecting data-taking.

[3]This is in contrast to the monolithic Run I architecture where a crash or un-handled exception in any subroutine would have led to an abrupt termination of the CTP control software and of the data-taking.

Multi Use of Data: As all (raw) values are published to and are available in IS, monitoring clients can pick, choose, and reuse the information they need at all times.

Multi-Partition Running: The monitoring server is agnostic to the use-case of the data it reads out and who needs it, only (some) clients need to be partition-aware. The monitoring server can read out and publish the observed busy from all sub-detectors and all partitions and publish the raw data. Partition-specific monitoring clients can then pick up the raw data, 'cherry pick' and re-format the data before (re)publishing the data relevant to the IS of the partition.

6.5.2. The Scheduling Problem

The two remaining constraints, **Minimising Dead-time** and **Readout Across Boards**, along with the requirement for frequent readout forms the basis for a scheduling problem, namely the problem of when to read out what functional part of what board. It is the monitoring server that does the actual monitoring, and thus also the monitoring server that needs to schedule and execute the individual monitoring routines to meet the requirements and constraints. This section details the design considerations and requirements for the scheduler while the implementation details are the topic of Section 6.6.4.

In order to minimise dead-time during luminosity block transitions the monitoring server implements and accepts two IPC calls from the CTP service responsible for generation of luminosity blocks, in order to dynamically alter the scheduling algorithm. Without this communication, in a worst-case scenario, the monitoring server might block the luminosity block transition for up to $\sim 600\,\text{ms}$ before the transition could proceed. This should be compared to a normal transition time of $\sim 5.3\,\text{ms}$. The two signals are:

Pre-transition Warning: The program responsible for issuing new luminosity blocks will call the monitoring server $\sim 10\,\text{s}$ before issuing a new luminosity block, if the current luminosity block is about to exceed the nominal length, warning the server that a luminosity block transition is imminent.

Upon receiving the warning, the monitoring server will delay any VME-heavy monitoring (per-bunch histograms) until after the luminosity block transition has completed.

If a configuration change has prompted the creation of a new luminosity block the warning can not be used. In the rare cases, where the configuration change causes a luminosity block transition to happen exactly when one or more bunch histograms are being read out, additional dead-time might be incurred. The chances of this happening is reduced by scheduling readout of per-bunch histograms shortly after the transition so that the minimum length of a luminosity block prevents the readout from clashing with a transition.

Transition in Progress During the luminosity block transition the monitoring server receives a callback to readout the values that are tightly coupled to the luminosity block transition. Any other monitoring is delayed.

To resolve **Readout Across Boards**, it is instructive to abandon a board-centric view of the monitoring routines and instead focus on functionality. That is, instead of thinking of "monitoring of the CTPCORE" it is better to think of "Busy Monitoring", say. Busy Monitoring then involves the CTPCORE, but also the CTPMI and the five CTPOUT modules (see Figure 4.9). The respective functional blocks (in this case sequencers, masks, etc.) of these boards all need to be read out at (roughly) the same time to provide a consistent picture. As the monitoring server has access to all boards, this does not require additional coordination with other processes. The scheduler operates with such functional groups and instead of scheduling individual monitoring routines the entire functional group of monitoring routines are scheduled for execution. When receiving the pre-transition warning, for instance, the groups containing bunch histogram monitoring tasks are (re-)scheduled for later execution. The size and complexity of these monitoring groups, as well as their execution time, however varies dramatically – from reading out a few bytes for the clock monitoring to reading out megabytes of per-bunch counters from the CTPMON.

Dividing the monitoring into functionality centric groups facilitates an elegant way of meeting the requirement of separation between the critical monitoring tasks and the non-critical ones, as will be shown in Section 6.6.2.

6.6. The Server Application

This section covers aspects of the implementation of the monitoring server. In particular how the individual monitoring tasks are grouped in task groups of functionally similar tasks. A summary of the data volumes of the monitoring task groups and the expected execution time per task group is also presented. In order to solve the scheduling problem, the monitoring server implements a custom scheduler, to dynamically assess the urgency of the various monitoring tasks and schedule them for execution. This is the topic of Section 6.6.4.

6.6.1. The Task Groups

		CTPCORE	CTPMI	CTPIN	CTPOUT	CTPCAL	CTPMON
	Multiplicity	1	1	3	5	1	1
Task Group	Status	✓	✓	✓	✓	✓	✓
	Busy *	✓	✓		✓		
	Event	✓					
	Dead-time/Bunch	✓			✓		
	Phase			✓			
	Clock	✓					
	Trigger Rate *	✓		✓			
	Item/Bunch	✓					✓

*) Part of the critical monitoring; must be carried out during luminosity block transition.

Table 6.1: Task groups of the monitoring server with the indication of board involvement and multiplicity of boards/tasks.

Table 6.1 shows a matrix of the task groups of the monitoring server, and the per-board type multiplicity of tasks. For all of the CTP boards, a status monitoring task exists, i.e., the configuration and general state of each board is read out periodically. However, only the CTPIN boards are capable of doing monitoring of the trigger input phase and thus only the trigger input phase monitoring tasks of the CTPIN boards are contained

in the task group for trigger input phase monitoring. Each board-specific task needs to be replicated and run on each of the board of the same type. Thus there are 3 trigger input phase monitoring tasks – one for each of the physical CTPIN boards in the group.

Figure 6.5 shows a visualisation of the task groups, indicating the per-board sub tasks included in each group. The underlying driver used to communicate with the boards requires one driver instance per physical board, thus this is the lower bound on the number of tasks.

6.6.2. Task Group Overview

The monitoring of any part of the CTP must be assigned a to a task group. In most cases, the choice of task group is obvious but in some cases, particular when a combination of counter/sequencer values, need paring with configuration values, the choice of task group(s) becomes less obvious.

A detailed breakdown of what is being monitored for each board as part of what task group can be found in the Appendix A.2.

6.6.2.1. Status Monitoring

The status monitoring tasks monitor aspects of the CTP that change slowly as compared to the bunch clock and data for which counter and buffer overflow is not an issue. This primarily includes configuration values, voltages, and temperatures. Configuration values are here interpreted broadly and also include some configuration values for functionality that belongs to different task groups. The philosophy is to offload the critical monitoring tasks and the data heavy monitoring tasks to allow faster execution without loss of information.

A summary of the amount of the data read out per board is given in Table 6.2. The largest contributors can be seen to be the CTPCORE and the CTPIN. As the CTPCORE is the most complex of the boards it is expected that the amount of configuration data generally exceeds the other boards. The primary contribution from the CTPINs is the configuration of BCID masks used by the input rate monitoring logic. The masks are used to pick out (or conversely mask) individual signals in particular bunch crossings. The BCID masks are slowly changing[4] compared to the underlying counters and it suffice to read them out at a much lower frequency.

[4] At most they change between runs.

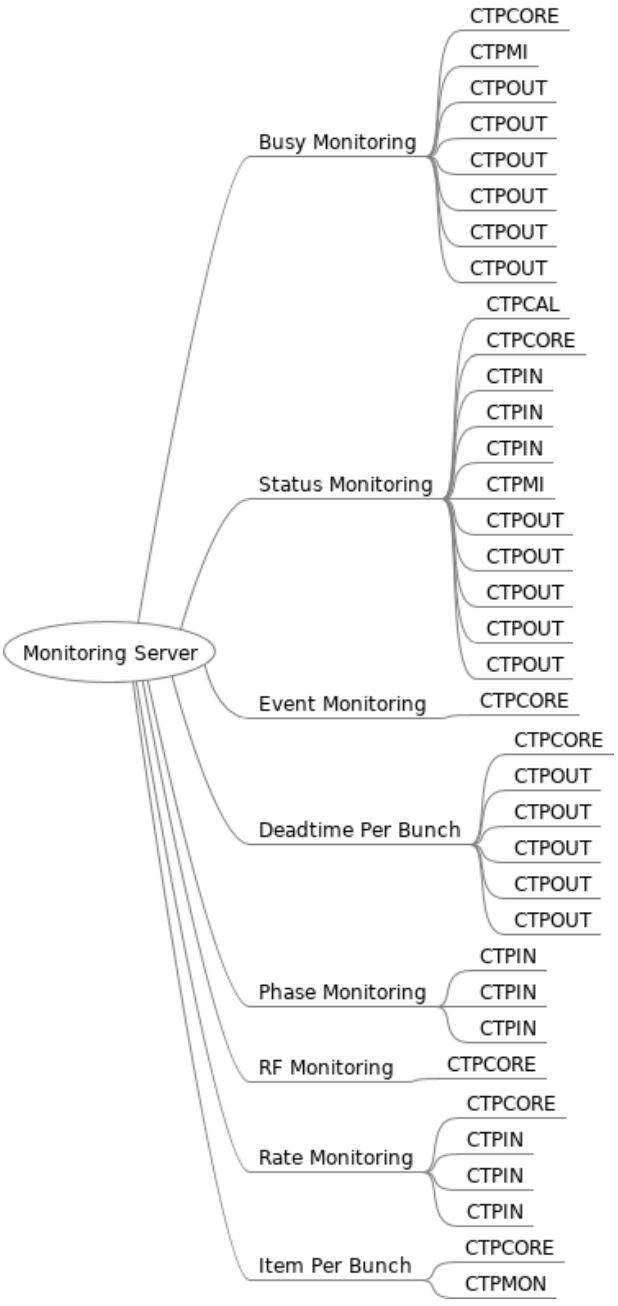


Figure 6.5: Task groups of the monitoring server with associated sub tasks per board.

Status Monitoring				
		Per Board		
Board	**Multiplicity**	**Calls**	**Size**	**Total Size**
CTPMI	1	35	52 kB	52 kB
CTPIN	3	85	66 kB	199 kB
CTPMON	1	252	183 B	183 B
CTPCORE	1	1889	8 kB	8 kB
CTPOUT	5	1903	44 B	220 B
CTPCAL	1	2185	551 B	551 B
Total across boards				259 kB

Table 6.2: Per board summary of Status Monitoring tasks.

6.6.2.2. Busy Monitoring

The busy monitoring tasks are responsible for reading all information relating to the busy sequencers (see Section 4.3), particularly the frequency of the sequencer and the FIFO entries containing the actual busy values, but also the enabled-ness of the sequencer and the FIFO level to avoid data loss due to miss configuration or buffer overflow. The busy monitoring tasks are a part of the critical tasks that must executed during the luminosity block transition in order to ensure that the total dead-time of the luminosity block can be correctly determined.

Busy Monitoring				
		Per Board		
Board	**Multiplicity**	**Calls**	**Size**	**Total Size**
CTPMI	1	16	117 B	117 B
CTPCORE	1	24	154 B	154 B
CTPOUT	5	31	102 B	510 B
Total across boards				781 B

Table 6.3: Per board summary of Busy Monitoring tasks.

A summary of the amount of data read out per board is given in Table 6.3.

6.6.2.3. Event Monitoring

The event monitoring is a CTPCORE specific task that allows for buffering and reading out the entire CTP event fragment, as described in Sec-

tion 4.4.2.2. The typical event size is 856 bytes. The parameters relating to the event monitoring, such as the critical level (or "watermark") of the FIFO holding the event fragment, as well as the current mode of operation are read out as part of the status monitoring, as the configuration is slowly changing. By delegating this monitoring to another task group, the event monitoring becomes faster and easier to schedule, which thereby increases the achievable rate, in line with the discussion in Section 4.4.2.2.

6.6.2.4. Dead-time per-Bunch Monitoring

Dead-time per-bunch monitoring is implemented slightly differently in the CTPOUTs as compared to the CTPCORE: for the CTPOUTs, it is the BUSY from the sub-detectors that is monitored, while for the CTPCORE it is a configurable selection of the resulting dead-time signals (the preventive dead-time and sub-detector dead-time) that is monitored. For each of the CTPOUTs, capability of monitoring busy per-bunch exist for each of the five cables connecting the sub-detectors to the CTPOUT (the busy tree is depicted in Figure 4.9). The CTPCORE implements seven per-bunch histograms. For each of these there is an associated bit mask to pick out the desired combination of sub-detector, simple and complex dead-time, the OR of which is counted, as described in Section 5.3.3. The bit masks are read out as part of the per-bunch dead-time monitoring, due to how the monitoring clients archive these histograms. The per-bunch dead-time from the CTPCORE is used in Section 7.3.3 for a cross-check of the calculation of dead-time correction factors.

Dead-time per Bunch Monitoring

		Per Board		
Board	**Multiplicity**	**Calls**	**Size**	**Total Size**
CTPCORE	1	9	100 kB	100 kB
CTPOUT	5	15	71 kB	356 kB
Total across boards				456 kB

Table 6.4: Per board summary of Dead-time per Bunch Monitoring tasks.

A summary of the amount of data read out per board is given in Table 6.4. Comparing Table 6.4 to Table 6.2 it can be seen that the Dead-time per Bunch Monitoring is twice the size of the entire Status Monitoring. The

main contribution in terms of data volume are the per-bunch counters them selves.

6.6.2.5. *Trigger Input Phase Monitoring*

For each CTPIN, the phase of each of input signal of each cable is determined by sampling the HPTDC (see Section 4.4.1.1) a number of times, typically $N = 100$, and then determining the mean and standard deviation of the phase as measured by the HPTDC. The monitoring should is carried often enough that the phase can be measured every couple of minutes.

6.6.2.6. *Clock Monitoring*

The clock monitoring is a CTPCORE-specific task for measuring the frequency of the LHC provided clock based on timing signals from a GPS receiver and the counting of clock edges. Data-wise the clock monitoring is extremely light weight.

6.6.2.7. *Rate Monitoring*

The rate monitoring covers the monitoring of all the counters along the trigger path depicted in Figure 4.7. The data read out as part of this task group are the counter values and the corresponding turn counters, as described in Section 4.3. The Rate Monitoring is part of the critical monitoring and must be carried out during the luminosity block change.

Rate Monitoring

		Per Board		
Board	**Multiplicity**	**Calls**	**Size**	**Total Size**
CTPIN	3	8	1 kB	3 kB
CTPCORE	1	25	27 kB	27 kB
Total across boards				30 kB

Table 6.5: Per board summary of Rate Monitoring tasks.

The data size summary is presented in Table 6.5. The data volume read from the CTPCORE can be seen to be a lot bigger than the CTPIN monitoring. This is primarily due to a) more signals being available and monitored in the CTPCORE ; and b) that the trigger counts and rates are monitored at several steps along the trigger chain, e.g., before and after prescale, as illustrated in Figure 4.7.

6.6.2.8. Trigger per Bunch Monitoring

The trigger per-bunch monitoring is similar to the rate monitoring and covers the monitoring of all the per-bunch counters along the trigger path as depicted in Figure 4.7. Unlike the rate monitoring, the trigger per-bunch monitoring is not considered critical. The values from the per-bunch counters are used in Section 7.3.3 to perform a cross-check of the calculation of dead-time correction factors, discussed in Section 5.3.1.

Two boards has capability to do monitoring of trigger rate per-bunch: the CTPCORE and the CTPMON. In both case only a subset of the signals is selected for per-bunch monitoring. The CTPMON can monitor 160 of the PIT bus signals at a time and the CTPCORE implements a total of 192 per-bunch counters as 3 sets of 64 per-bunch counters as depicted in Figure 4.7.

Item per Bunch Monitoring

		Per Board		
Board	**Multiplicity**	**Calls**	**Size**	**Total Size**
CTPMON	1	44	570 kB	570 kB
CTPCORE	1	239	3 MB	3 MB
Total across boards				3 MB

Table 6.6: Per board summary of Item per Bunch Monitoring tasks.

The data volume is summarised in Table 6.6. The trigger per-bunch monitoring task is the most data heavy of the monitoring task groups.

6.6.3. Task Group Summary

Table 6.7: Task group summary of the monitoring server.

Group	**Data Volume**	**Calls**	**Time Est.**
Status Monitoring	259 kB	2341	46.82 ms
Busy Monitoring	781 B	59	1.18 ms
Event Monitoring	7 kB	1	0.27 ms
Dead-time per Bunch Monitoring	456 kB	39	18.25 ms
Phase Monitoring	155 kB	1236	24.72 ms
Clock Monitoring	48 B	6	0.12 ms

Rate Monitoring	30 kB	41	0.82 ms
Item per Bunch Monitoring	3 MB	239	132.30 ms
Total	4 MB		

Table 6.7 presents the data summary for the different task groups. For each task group, estimates are given for the raw data volume as well as for a probable execution time for all the routines within the task group. The data volume as well as the expected execution time are important parameters for the design and scheduling of the task groups: it is important that all task groups can be executed frequently and efficiently enough to meet all requirements, in particularly the hard requirements on avoiding overflow and minimising experimental dead-time. This is particularly true for the critical monitoring.

The time estimates are obtained assuming that a normal VME-read of 4 bits takes 20 μs and that larger transfers, such as the per-bunch monitoring tasks and the event monitoring, can be carried out using DMA at a typical speed of 25 MB/s. The VME bandwidth is often quoted to be 40 MB/s. The achievable transfer speed will depend on the transfer methods and the latency introduced by software and hardware drivers [75].

Execution times for all task groups are well below the desired time between executions. This makes it easier for the scheduler to meet the scheduling deadlines and perform efficient scheduling of all monitoring tasks.

The data sizes presented, are the data sizes transferred over the VME bus. This is but one side of the story: after receiving the (binary) data from the boards, the SBC formats the data in a data structure compatible with the IS infrastructure. IS infrastructure uses a data structure based on compressed JSON blobs [79]. Bandwidth starvation on the network interface is however not an issue when publishing to IS, even when data is inflated, sometimes by several hundred percent: a bit used to encode a Boolean value might be encoded into a text string of length 8 (“disabled”) with an associated attributed name (e.g., “is_trigger_enabled”).

6.6.4. The Scheduler

The scheduler is a key component in the design of the monitoring server and is the part responsible for scheduling the execution of the task groups. The task groups are scheduled in a way that the requirements of each task group can be met and such that the VME bus is otherwise kept free for other operation-critical operations. The scheduler also solves the scheduling challenge, outlined

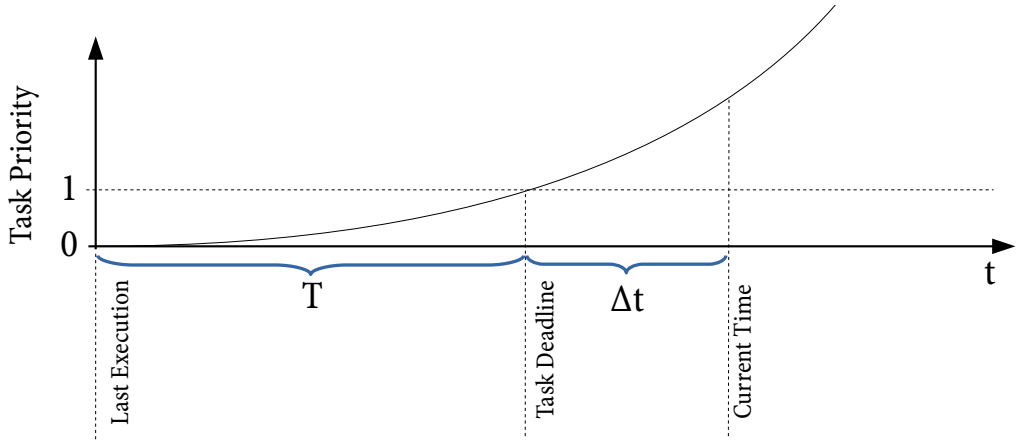


Figure 6.6: The relation between the task priority, p, as function of time, t, since last execution, for a configured time, T, between executions.

in Section 6.5.2, to minimise the dead-time introduced around the luminosity block transitions.

The scheduler is a dynamic scheduler with several operational modes. In order to keep luminosity block transitions as effective as possible, the scheduling algorithm is changed several times around the luminosity block transition.

The default mode of the scheduler is a variant of a Earliest Dead-line First (EDF) scheduler: by default each task (group) is dynamically assigned a priority, p, based on how close to its "dead-line" it is: with the configured time between executions T, and the time since last execution t, the task is due for execution in $\Delta t = t - T$ as illustrated in Figure 6.6. The relationship between priority and scheduling time is chosen to be

$$p = \exp[t - T] \,. \tag{6.1}$$

The task with highest urgency is scheduled for execution.

The scheduling and execution of each monitoring task takes place in parallel threads that share memory and handle to the VME driver. The underlying drivers for accessing the boards are however not thread safe imposing bounds on what tasks can be monitored in parallel: clock monitoring (CTPCORE) and trigger input phase monitoring (CTPIN) can be scheduled in parallel, but clock monitoring (CTPCORE) and event monitoring (CTPCORE) can not. This is handled dynamically by the scheduler as each task (group) adheres strictly to a Finite State Machine (FSM) that allows the state of driver and resource utilisation to be determined.

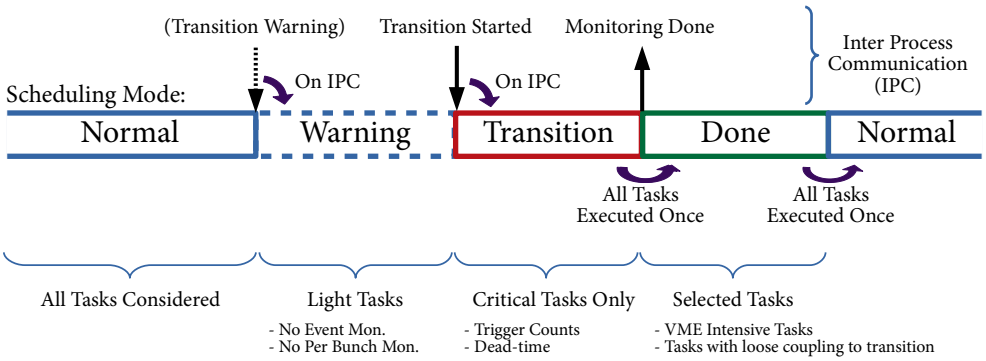


Figure 6.7: Diagram depicting the different scheduling modes used by the monitoring server around the luminosity block transition.

In order to make the luminosity block transition as effective as possible different scheduling modes are used by the scheduler around the transition. The modes are depicted in Figure 6.7. As part of its configuration, each task (group) is assigned a transition configuration, specifying the coupling between the task and the luminosity block transition. Around the luminosity block transition, this configuration is used by the scheduler to determine what tasks to consider for scheduling and what tasks to postpone.

The three special operating modes of the scheduler are:

Warning This mode is triggered by an IPC (see Section 6.5) and is used to prevent VME-intensive monitoring around the luminosity block transition.

Transition Is triggered via IPC as part of the luminosity block transition and is used to only schedule/execute the critical tasks that must executed during the transition, i.e., the tasks responsible for reading out the trigger counts and the dead-time.

As soon as these tasks are executed the scheduler changes to Transition Done.

Done This mode is temporary and used for reading out per-bunch histograms as well as information that should be read out at least once per luminosity block, such as the configuration parameters. As soon as these tasks are executed the scheduler changes to its normal operational mode.

An advantage of using the **Done** state for VME heavy tasks is that it is guaranteed to be relatively "quite" due to the constraint on shortest luminosity block.

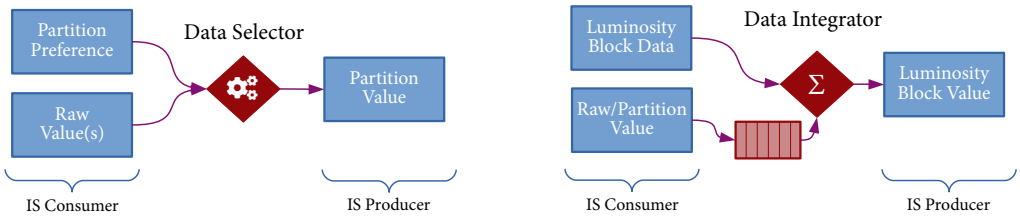


Figure 6.8: Conceptual depiction of data processing monitoring clients. The clients receive input from IS, process the information and publishes it back to IS. For data selectors (a) new values are typically produced at every update of the source publication. For data integrators (b) values are typically buffered until they are processed at the end of the luminosity block.

6.7. The Monitoring Clients in Detail

The CTP monitoring clients consume the monitoring data published by the monitoring server and by other monitoring systems in ATLAS, such as luminosity providers or beam status providers. This definition is very broad and spans everything from applications that do simple cross checks to application used at all times in the ATLAS Control Room. The monitoring clients can be divided into three categories, as discussed in Section 6.5 and illustrated in Figure 6.4.

My work on monitoring clients has been focused on the data presenters (type 2 clients). Thus the main focus will be on the monitoring clients that are of most importance for the data presenters and for the monitoring server.

6.7.1. Monitoring Processing Clients

The monitoring processing clients subscribe to data published to IS and are notified when information is updated. The monitoring processing clients can be divided into two types *data selectors* and *data integrators* as illustrated in Figure 6.8. The names are chosen for the common use-case of the two types:

Data Selectors are used to pick out (select) partition-specific monitoring data from the raw/complete data published by the monitoring server, and optionally pair it with information from other sources.

Data Integrators are used to produce per-luminosity block summaries or time integrations of frequently updated monitoring data.

Important data selectors include:

Busy Selector The Busy Selector of each partition-subscribe to the partition-specific information about what detectors are included in the partition (and their names) as well the busy data published by the monitoring server. The busy data is updated every couple of seconds and the busy selector picks out the partition-specific busy data from the publication and matches the busy information of each CTPOUT cable with the partition-specific selection and data to produce a partition-specific publication. This is the publication later used by e.g., the "Busy Panel" as described in Section 6.7.2.2.

Item Selector The Item Selector is similar to the Busy Selector and is used to select which of the 512 trigger items are relevant for the current partition and match the raw counter values from the monitoring server with meaningful names. The resulting publication with relevant trigger inputs and human readable names is later used by e.g., the Web Rates Presenter as described in Section 6.7.2.1.

Important data integrators include:

Busy Integrator The Busy Integrator is used to obtain the total busy or dead-time for a luminosity block by integrating the much more frequent publications of instantaneous busy and dead-time. The information is used both offline for dead-time corrections, as outlined in Section 5.1, and online in the control room as a down-sampled version of the instantaneous busy.

Item Integrator The Item Integrator integrates the number of triggers fired by each item over the period of a luminosity block. The values produced are used both online and offline.

It should be noted that the depictions in Figure 6.8 are conceptual: e.g., not all selectors subscribe to partition-specific data and not all integrators implements a buffer or sums data. It should also be noted that the integrators can be effectively pipelined with the selectors, e.g. the output of the Busy Selector serves as an excellent input for the Busy Integrator.

Outsourcing these data processing tasks to more powerful computers means less SBC CPU usage and more robustness, following the discussion in Section 6.3.

6.7.2. Data Presenters

When presenting data there is an inherent trade-off between detail and overview. For the presentation of the monitoring data this trade-off is typically reflected in whether the instantaneous value(s) of parameters are shown or if the time

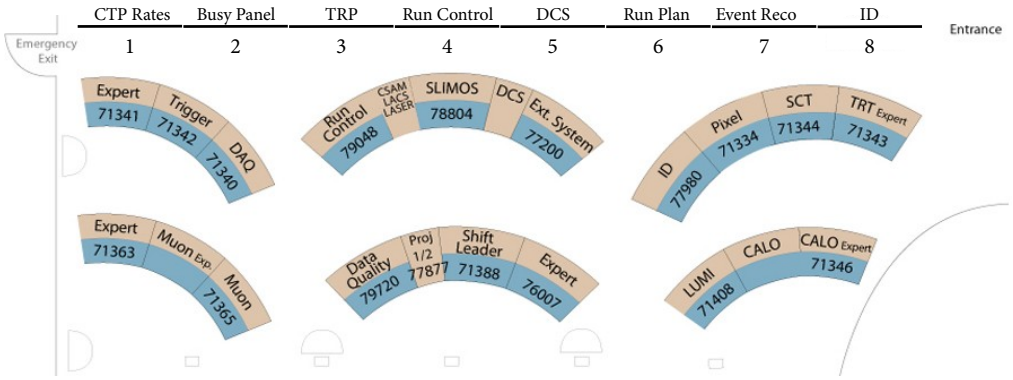


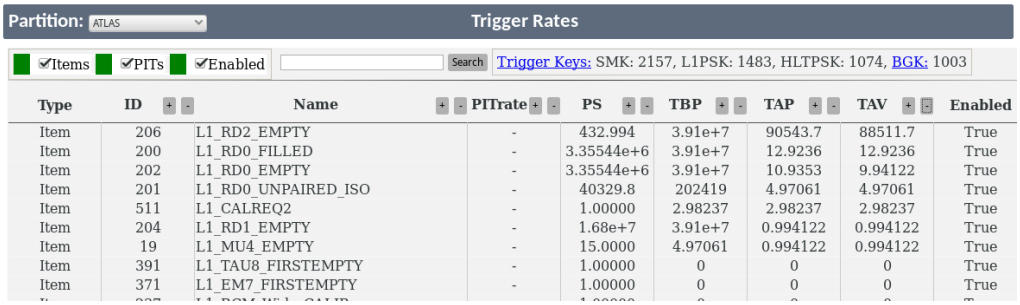
Figure 6.9: Layout of ATLAS control room with emphasis on the eight main screens indicated at the top.

evolution of a parameter is shown. When one representation is more suitable depends on the use-case. The following covers visualisation of the instantaneous trigger rates and of the instantaneous dead-time and busy – these are among the most important visual aids for the operators in the control room and to assess the state of ATLAS data-taking. There are eight main screens in ATLAS control room as illustrated in Figure 6.9. Two screens are allocated for the two data presenters described here.

6.7.2.1. The CTP Rate Presenter

For the trigger rates, a visualisation of both the instantaneous trigger rates (current values) as well as the evolution of the trigger rates over time (graph) is desired for the control: the instantaneous trigger rates provides detailed information about the current bandwidth allocation and in case of trouble potentially provide insights into the source of the problem. Tracking the trigger rates over time allows the operators to track the performance of the experiment and adjust the trigger bandwidth allocation if needed to meet the data-taking goals.

A tool for visualising the time evolution of both Level 1 and HLT trigger rates already exists. With the introduction of the new software infrastructure for the CTP, it became feasible to have a visualisation of the instantaneous Level 1 rates. The information published to IS is made available through a REST API and the web page polls the API to receive the data published by the Item Selectors (Section 6.7.1). The data is presented in a sort-able table with the possibility to filter on trigger items and trigger inputs. A screenshot of the web page is shown in Figure 6.10. The web page is projected on the first of the eight projected screens in the ATLAS Control Room.



Type	ID	Name	PITrate	PS	TBP	TAP	TAV	Enabled
Item	206	L1_RD2_EMPTY	-	432.994	3.91e+7	90543.7	88511.7	True
Item	200	L1_RD0_FILLED	-	3.35544e+6	3.91e+7	12.9236	12.9236	True
Item	202	L1_RD0_EMPTY	-	3.35544e+6	3.91e+7	10.9353	9.94122	True
Item	201	L1_RD0_UNPAIRED_ISO	-	40329.8	202419	4.97061	4.97061	True
Item	511	L1_CALREQ2	-	1.00000	2.98237	2.98237	2.98237	True
Item	204	L1_RD1_EMPTY	-	1.68e+7	3.91e+7	0.994122	0.994122	True
Item	19	L1_MU4_EMPTY	-	15.0000	4.97061	0.994122	0.994122	True
Item	391	L1_TAU8_FIRSTEMPTY	-	1.00000	0	0	0	True
Item	371	L1_EM7_FIRSTEMPTY	-	1.00000	0	0	0	True

Figure 6.10: Screenshot of the CTP web page presenting the instantaneous rates for trigger items and trigger inputs.

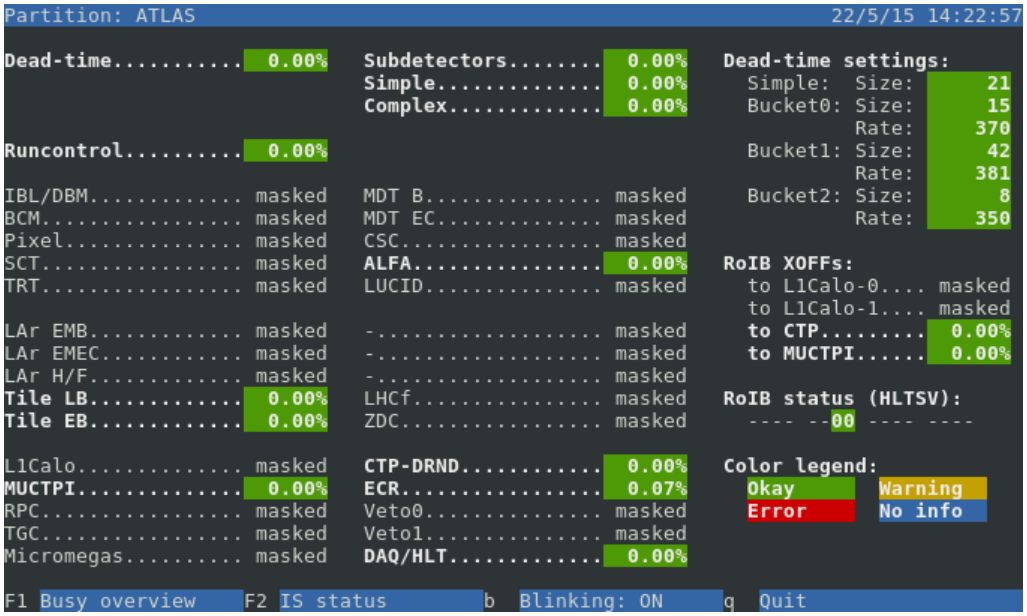


Figure 6.11: Screenshot of the Busy Panel summarising the instantaneous busy and dead-time of the experiment.

6.7.2.2. The Busy Panel

The instantaneous values of the busy rates are of great value to the operators in the control room: When everything is working smoothly, the busy fraction is typically very low ($\mathcal{O}(0.01)$). When things do not work, busy values are very typically high (~ 0.7 to ~ 1.0). As soon as an issue is resolved the busy values will return to normal. The instantaneous busy rates are thus useful for spotting and understanding issues as they arise, while the time evolution dimension is mostly useful for retrospective analyses or for summarising what happened during a run. This type of plot is used as a talking point at the daily run coordination meeting.

The Busy Panel, shown in Figure 6.11, is used to display the instantaneous BUSY as well as the dead-time related information in the ATLAS Control Room and is projected on one of the screen in the control room. The choice to make the Busy Panel a ncurses based terminal application [80] instead of a web page like the CTP Rate Presenter was based on considerations of ease of use and stability for detector experts not present in the control room: the only way of getting access to the otherwise isolated network used for data-taking is via an SSH gateway [81]. This implies that most experts logging in from remote to resolve an issue only have a terminal session available.

6.7.3. Data Persistors

Custom monitoring clients exist for persisting the per-bunch histograms and for writing critical CTP data to the conditions database. These generally follow the same general design pattern as outlined in Figure 6.8 with the major difference that the output is not an IS publication but rather a write to a file or to a database. Most of the data published to IS is automatically archived by the pbeast service [82], however the per-bunch histograms created from the per-bunch counters are not. Further, for other monitoring data, the way pbeast store and indexes archived data – as a time-series database using Geneva time as reference – is ineffective for the later use of the data: Most of the data describing the experimental conditions are stored in relational databases indexed by run number and luminosity block, as mentioned in Section 5.1.

Chapter 7

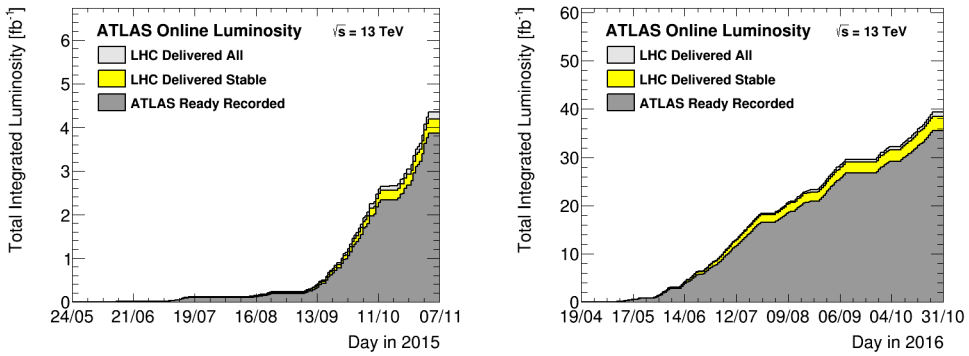
Operational Monitoring in Run II

7.1. Data Period Overview

The upgrade of the CTP took place during Long Shutdown I of the LHC run schedule. At the time of my mid-term report [1], ATLAS was celebrating the successful start of Run II. Since then, unprecedented amounts of physics data has been recorded and heaps of monitoring data have been collected.

The monitoring data presented here is collected as part of, and alongside, the data-taking between the beginning of Run II, in 2015, to the end of 2016.

Figure 7.1 shows the delivered and recorded integrated luminosity as function of time for the 2015 and 2016 proton-proton physics program. The small difference between the recorded and the delivered integrated luminosity are due primarily to experimental dead-time. In 2015, a total of $3.859\,\mathrm{fb}^{-1}$ of data was



(a) 2015 data summary [83] (b) 2016 data summary [84]

Figure 7.1: Data summary of the 2015 and 2016 data-taking, showing the delivered (yellow) and recorded (dark grey) integrated luminosity.

recorded out of the $4.193\,\text{fb}^{-1}$ delivered, corresponding to a data-taking efficiency of 92.0% [83]. In 2016, a total of $35.56\,\text{fb}^{-1}$ of data was recorded out of the $38.5\,\text{fb}^{-1}$ delivered, corresponding to a data-taking efficiency of 92.4% [84]. The increased rate in the delivered integrated luminosity was due to the gradual increase in instantaneous luminosity achieved by the LHC. - the design luminosity of $\mathcal{L} = 10^{34}\,\text{cm}^{-2}\text{s}^{-1}$ was reached in early 2015 and has since increased to $\mathcal{L} = 2.14 \times 10^{34}\,\text{cm}^{-2}\text{s}^{-1}$ as illustrated in Figure 2.7.

The CTP is monitored all the time, independently of whether ATLAS is recording data. Thus, the monitoring data volume is independent of the data-taking, except for a couple of extra publications which are only created during data-taking (see Section 6.7.1). However, the number and total size of these are negligible compared to the constant data volume produced and published by the monitoring server. In 2015 and 2016 an estimated 60 TB of CTP monitoring data was published to IS.

The archived monitoring data provides powerful insights that will be illustrated through a couple of examples of use-cases of the monitoring data for clock monitoring, trigger rate monitoring and dead-time monitoring. In particular, a cross check of the calculation of the dead-time correction factors as well as illustrative examples of the usefulness of trigger rates for detecting operational issues will be given. This chapter will start with a demonstration of some of the general observations and conclusions that can be made from the monitoring data before turning to specific examples and cross checks.

Run Number	Start Time (UTC)	End Time (UTC)	LBs	Peak Luminosity	Rec. Events
299055	Thu May 12 2016 17:24:10	Fri May 13, 05:20:39	731	$2.0 \times 10^{33}\,\text{cm}^{-2}\text{s}^{-1}$	149,888,972
302872	Sun Jun 26 2016 12:43:40	Tue Jun 28, 04:49:33	2613	$1.1 \times 10^{34}\,\text{cm}^{-2}\text{s}^{-1}$	71,392,082
310738	Sun Oct 16 2016 19:18:32	Mon Oct 17, 11:22:54	1095	$1.3 \times 10^{34}\,\text{cm}^{-2}\text{s}^{-1}$	71,392,082
311481	Wed Oct 26 2016 06:56:08	Wed Oct 26, 21:34:17	921	$1.2 \times 10^{34}\,\text{cm}^{-2}\text{s}^{-1}$	52,500,764
314199	Sun Dec 04 2016 06:48:18	Mon Dec 05, 05:06:53	1344	$3.2 \times 10^{27}\,\text{cm}^{-2}\text{s}^{-1}$	251,199,892

Table 7.1: Summary information of selected runs, showing start and end time, number of luminosity blocks (LBs), peak instantaneous luminosity and number of recorded collision events.

Out of all the data-taking runs of the 2015 and 2016 data-taking effort, five runs was selected and used for most of the following discussion. The runs are summarised in Table 7.1. The runs are selected to reflect the conditions during normal operation as well as to illustrate a couple of special cases where operational issues impact the trigger system and the data-taking.

The times given in Table 7.1, and in most of this section, are in UTC. Geneva follows CET (UTC + 1h) during winter time and CEST (UTC + 2h) during summer time.

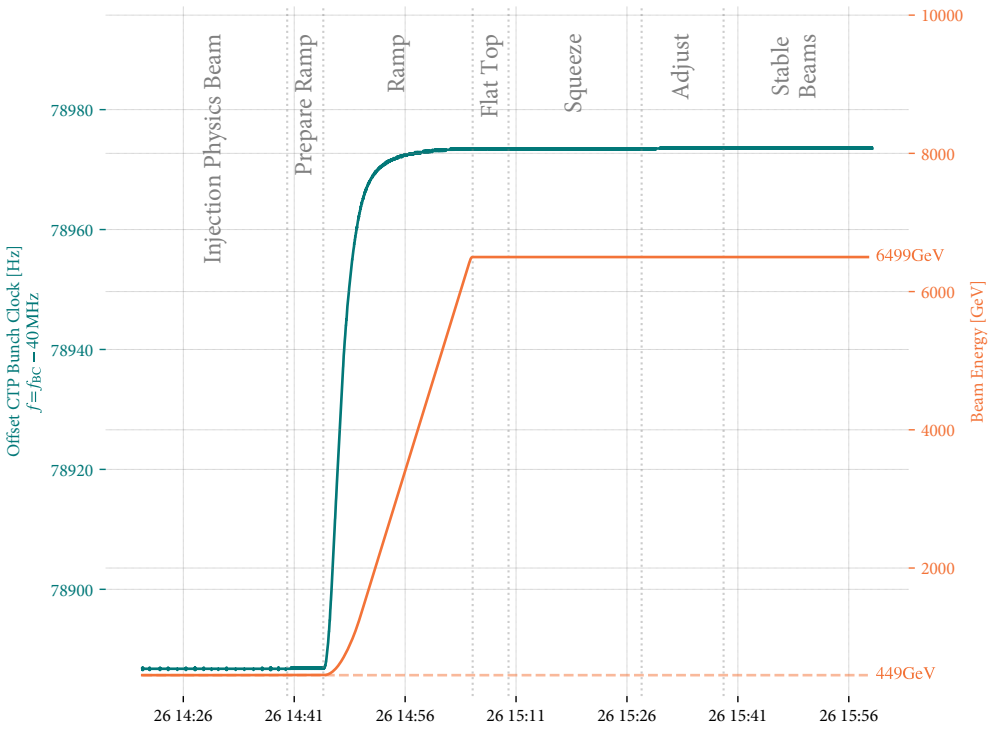


Figure 7.2: Monitoring of the CTP bunch clock frequency during injection of fill 5045 for run 302872.

7.2. General Observations

This section will make use of data from different runs, or parts or runs, to illustrate the power of the monitoring. The recorded monitoring data can be used to illustrate a few points made in Section 5 and Section 6 as well as to support some of the claims and assumptions made.

7.2.1. Clock Frequency Monitoring

Figure 7.2 shows the change in the bunch clock as measured by the CTP, as well as the beam energy, during the different stages of beam injection and acceleration. The change in the bunch clock frequency, from injection to stable beams is typically $\sim 86.5\,\mathrm{Hz}$ for *pp*-collisions.

Recalling that the clock signal provided by the LHC is derived from the RF driving the particles in each beam, the expected change in frequency with the increase in beam energy can be estimated by considering a relativistic particle moving in a ring: If $l = 26.7\,\mathrm{km}$ is the circumference of the ring, and the ring is

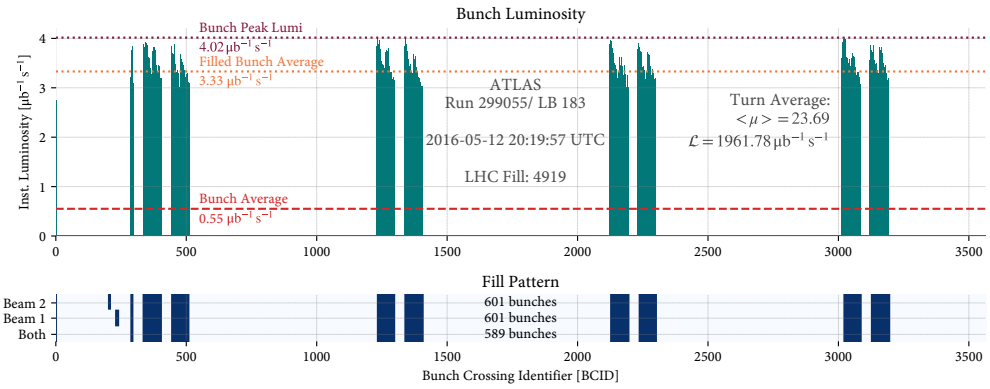


Figure 7.3: Per bunch instantaneous luminosities (top) and LHC fill pattern (bottom) during luminosity block 183 of run 299055.

divided into $N = 3564$ buckets or bunch crossings, a relativistic particle of mass m and energy E would naively give rise to a bunch clock frequency, f, of

$$f = N \frac{c}{l} \left(1 - \frac{m^2 c^2}{E^2}\right)^{\frac{1}{2}} . \tag{7.1}$$

Letting, E_i denote the injection energy and E_t denote the target energy, the change in frequency can be estimated as

$$\Delta f = N \frac{c}{l} \left[\left(1 - \frac{m^2 c^2}{E_t^2}\right)^{\frac{1}{2}} - \left(1 - \frac{m^2 c^2}{E_i^2}\right)^{\frac{1}{2}}\right] . \tag{7.2}$$

For a proton mass of $m = 0.938\,\text{GeV}$, an injection energy of $E_i = 449\,\text{GeV}$, and a target energy of $E_t = 6499\,\text{GeV}$, one obtains an estimated frequency shift of $\Delta f = 86.9\,\text{Hz}$, which is consistent with the observed frequency shift.

7.2.2. Fill Pattern and Bunch Groups

Figure 7.3 show the per-bunch instantaneous luminosity (top) and the LHC fill pattern (bottom) for luminosity block 183 of run 299055. The filling pattern contains 601 bunches per beam, with 589 paired bunches, most of which are arranged in 4×2 bunch trains. In addition to the paired bunches in the main bunch trains, the fill contains an isolated paired bunch in the first bunch crossing, unpaired bunches for both beams before the remaining bunches, and a short train before the main bunch trains.

From the fill pattern a bunch group mask is created. The bunch group mask used during run 299055 is shown in Figure 7.4. As mentioned in Section 4.4.1.2,

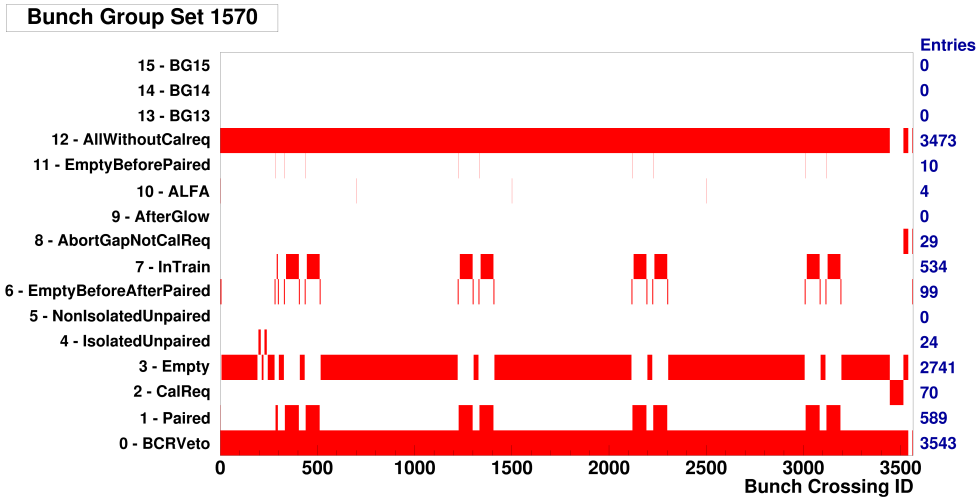


Figure 7.4: Bunch Group Mask used during run 299055. Source [85].

ATLAS operates with the notion of 16 bunch groups. Each trigger item is assigned a (16 bit) bunch group mask, which confines the trigger item to one or more bunch groups. Notable bunch groups are perhaps 1, 2, 3 corresponding to 1) the paired (colliding) bunches; 2) the bunch crossings reserved for calibration (requests) triggers; and 3) the empty bunches. Most physics triggers are assigned to bunch group 1. Calibrations triggers deserve a honorary mentioning as the CTP is handling calibration requests and creating calibration triggers. The empty bunches, as will be shown later, are useful for detecting detector related issues.

7.2.3. Luminosity and Trigger Rate

The trigger rate of the trigger T, before veto and prescaling, can be expressed as the product of a trigger cross section and the luminosity:

$$R^T_{\text{TBP}} = \sigma^T \mathcal{L} \tag{7.3}$$

where σ^T is the trigger cross section (see Section 3.3.5). The trigger cross section is normally constant but certain effects, such as pile-up or beam related backgrounds, might render the trigger-cross section dependant on instantaneous luminosity or bunch currents, respectively.

The upper plot in Figure 7.5 show the trigger rate before prescale, and thus before dead-time and veto, as function of luminosity for a muon trigger (`L1_MU10`), an EM trigger (`L1_EM10`), a τ trigger (`L1_TAU12`), a jet trigger (`L1_J50`), and a missing energy trigger (`L1_XE30`), while the lower plot shows the

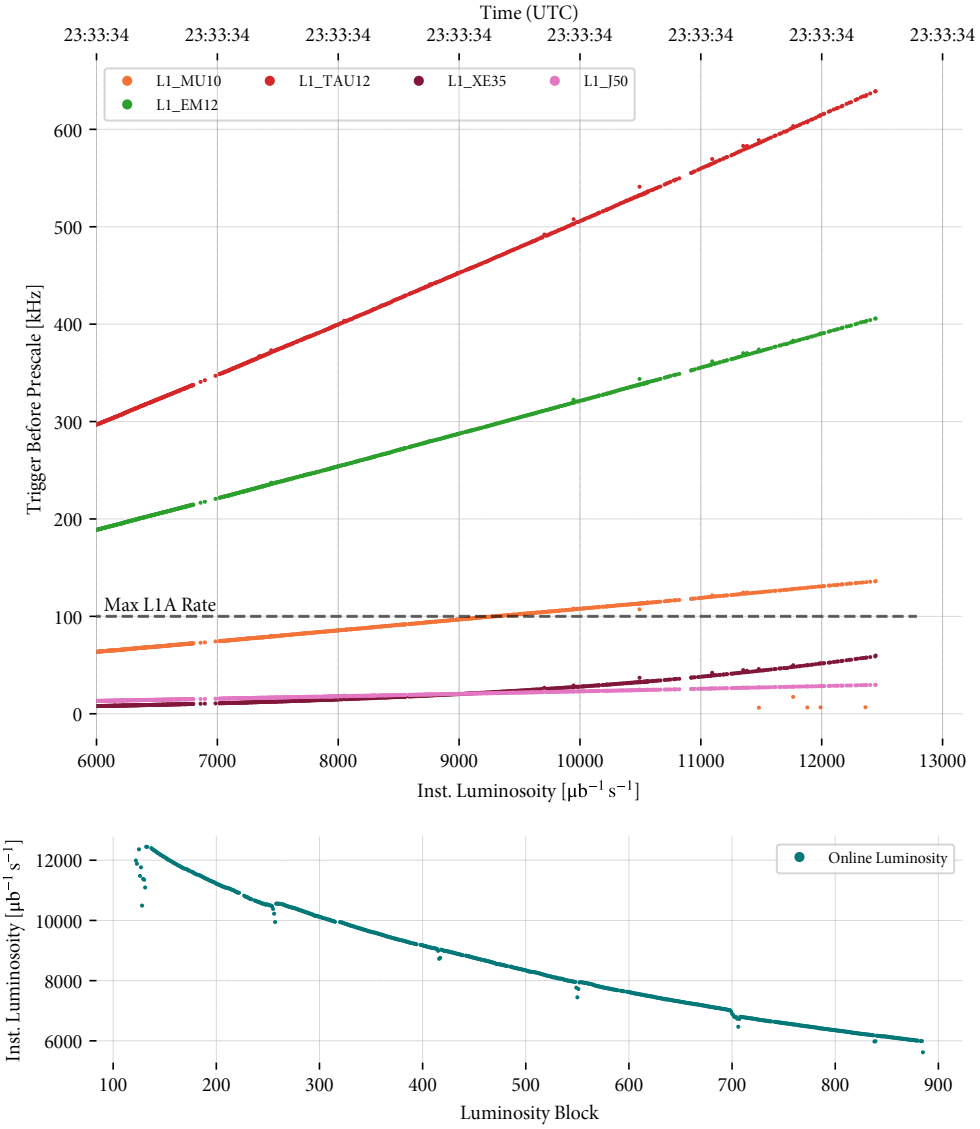


Figure 7.5: Level 1 trigger rates for different types of single object triggers as function of luminosity (top) and luminosity as function of luminosity block (bottom) for Run 311481.

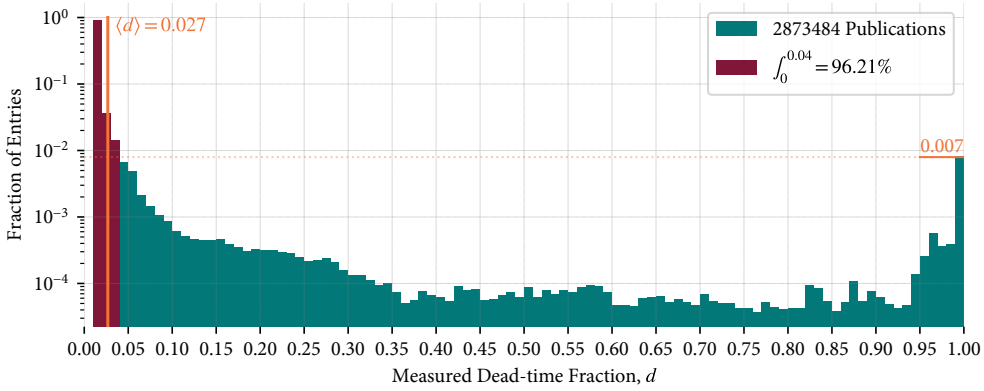


Figure 7.6: Dead-time distribution during data-taking in 2015 and 2016.

luminosity as a function of luminosity block in run 311481. The trigger types represent the most common Level 1 triggers while the exact trigger items were chosen for aesthetic reasons to allow all triggers to be plotted in the same plot without scaling. The plot illustrates the expected linear behaviour of the trigger rates for most trigger types as well as the non-linear behaviour of the missing energy trigger. The missing energy triggers has an implicit luminosity dependence as the calculation of missing energy at Level 1 is sensitive to in-time and out-of-time pileup: The missing energy at Level 1 is determined as the negative vector sum of energy deposits in the calorimeter detectors. In bunch crossings with high in-time pileup, the missing energy from each of the hard scattering events, caused by either produced neutrinos or mis-measurement of the energy, causes a pile-up dependant smearing of the total missing energy. This smearing of the missing energy increases the likelihood of exceeding the trigger threshold, leading the trigger rate to grow faster than linearly. The dependence on out-of-time pileup is caused by the signal pulse lengths in the calorimeters which are longer than one bunch crossing, causing a per-bunch crossing carry over. Some corrections can be – and are already – applied by the L1Calo but still a dependence which grows faster than linearly remains.

7.2.4. Global Dead-time Distribution

Figure 7.6 show the distribution of the total experimental dead-time in ATLAS during data-taking of 13 TeV *pp*-collisions in 2015 and 2016. Each dead-time measurement spans a time period of ~ 1 s. The mean experimental dead-time fraction of ATLAS is 2.7%. During more than 96% of the data-taking, the

experimental dead-time was less than 4%. During less than 1% of the data-taking ATLAS experienced periods of 100% dead-time.

One of the main causes of experimental dead-time is the preventive dead-time - particularly the simple dead-time, described in Section 4.5.2. The simple dead-time was configured to veto $N = 4$ bunch crossings following a L1A for the majority of the data-taking period. The simple dead-time is irreducible: with a target Level 1 rate of 100 kHz and a simple dead-time of 4 BC, the average contribution to the total dead-time from the simple dead-time is 1%, assuming that dead-time always falls onto colliding bunches[1]. There are other irreducible contributions to the total dead-time besides the preventive dead-time, e.g., the dead-time applied during the distribution of the ECR and the dead-time introduced during luminosity block transition. However, these can be estimated to be small: the ECR dead-time is applied for 1 ms on both side of the ECR signal, which in turn is emitted every ~ 5 s, corresponding roughly to $\sim 0.04\%$ of dead-time. Similarly, dead-time is applied for ~ 6 ms during the luminosity block transition. With a nominal luminosity block length of 1 min this corresponds to $\sim 0.04\%$ of dead-time. The irreducible dead-time adds up to $\sim 1.5\%$, thus the empty first bin in Figure 7.6.

7.2.5. Components of Experimental Dead-time

The total experimental dead-time is the logical OR of the sub-detector dead-time, the simple dead-time and the complex dead-time as discussed in Section 4.5.1. Via the per-bunch dead-time monitoring capabilities of the CTP, the OR of a programmable subset of these sources are monitored, allowing for detailed studies of per-bunch effects. Figure 7.7 shows a typical example of the average distribution of per-bunch dead-time during a luminosity block. The choice of run and luminosity block for this plot was aesthetic: the run (299055) was chosen for its simple filling pattern, and the luminosity block (184) was chosen to reflect the conditions during data-taking at peak instantaneous luminosity. The gap in the bunch pattern reserved for calibration requests is omitted from the plot of the total and detector dead-time to allow better visualisation of the distribution of dead-time in bunch crossings with colliding bunches.

The simple dead-time can be seen to be the primary cause for dead-time during bunch crossings with colliding bunches. The structure of the simple dead-time roughly follows that of the per-bunch luminosities shown in Figure 7.3. This is to be expected given the relationship between instantaneous luminosity

[1] With gaps in the filling patterns this will be less.

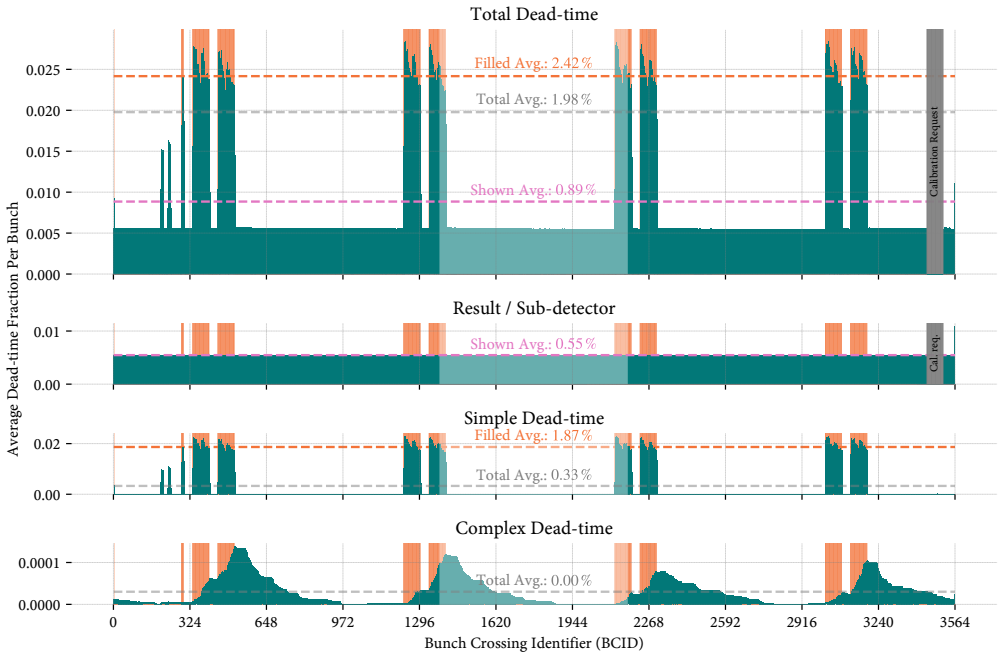


Figure 7.7: Dead-time per-bunch for run 299055, averaged over luminosity block 184. The total dead-time (top) is the OR of the detector dead-time (2nd from top) and the simple (3rd from top) and complex (4th from top) preventive dead-time. The bunch-crossings reserved for calibration requests are not shown. Where removed, "Shown Average" is used to indicate the average of the shown bunch crossings. A light orange background colour is used to indicate bunch crossings with colliding bunches. Dashed lines are used to indicated relevant bunch averages.

and trigger rate discussed in Section 3.3.5: the increased trigger rate in bunch crossings with higher than average instantaneous luminosity implies an increase in simple dead-time seen by the following bunch crossings.

The distribution of the complex dead-time is consistent with the description of the leaky bucket algorithm discussed in Section 4.5.2: during periods of high trigger rate (colliding bunches) the complex dead-time increases, reflecting the integrated probability of a number of triggers larger than the bucket size to occur. The leaky buckets emulate the front-end derandomizer buffers where the buffer occupancy grows during periods with high trigger rate and recovers during periods with no collisions and low trigger rate. In Figure 7.7, this can be seen as the rise in complex dead-time within each bunch train followed by a "cool down"

period after each bunch train. The uneven profile of the cool down period is caused by triggers in bunch crossings without collisions.[2]

The sub-detector contribution to the total dead-time can be seen to be roughly constant and, in bunch-crossings with colliding bunches, account for $\sim 1/4$ of the observed dead-time. At an accept rate of $\sim 100\,\text{kHz}$, each sub-detector ROD has around $10\,\mu\text{s}$ to process an event. Typically, the ROD BUSY is raised before all buffers are completely full. The typical ROD BUSY duration is therefore $\mathcal{O}(10\,\mu\text{s})$ and thus roughly constant on the time-scale of an LHC turn or similarly a per-BCID distribution.

7.3. Cross Checks Based on Monitoring Data

7.3.1. Expected and Actual Prescaling in the CTP

With the upgrade of the CTP, the prescaling mechanism was changed from a deterministic method to a probabilistic one. The monitoring data from the CTP proved useful for verifying, that the actual behaviour of the new prescaling mechanism corresponded to the expected behaviour. In the following, the total number of triggers per luminosity block before and after prescale, for some trigger item, will be used to obtain an estimator for the prescale applied by the CTP. Using this estimator, the expected and actual behaviour of the CTP prescaling mechanism is assessed for the missing energy trigger `L1_XE45` during run 311481. The choice of trigger item as well as run is purely illustrative: the prescale of this particular item was changed 15 times throughout the run to reflect the change in luminosity and trigger rate. The method for cross-checking the behaviour of the prescaling mechanism is however illustrative of the cross-checks carried out before the beginning of data-taking.

An estimate of the applied prescale can be obtained as the ratio of the number of triggers before and after prescale:

$$p = \frac{N_{\text{TBP}}}{N_{\text{TAP}}}\,, \tag{7.4}$$

where N_{TBP} and N_{TAP} are the number of triggers before and after prescale, respectively. Assuming p follows binomial statistics, the uncertainty σ_p on the estimator is

$$\sigma_p = \frac{1}{\sqrt{N_{\text{TBP}}}}\, p\, \sqrt{p-1}\,. \tag{7.5}$$

[2] The "empty" triggers are based on the same detector requirements as their "filled" counterparts, but are confined to bunch crossings without collisions. Empty triggers are primarily used for detecting detectors issues.

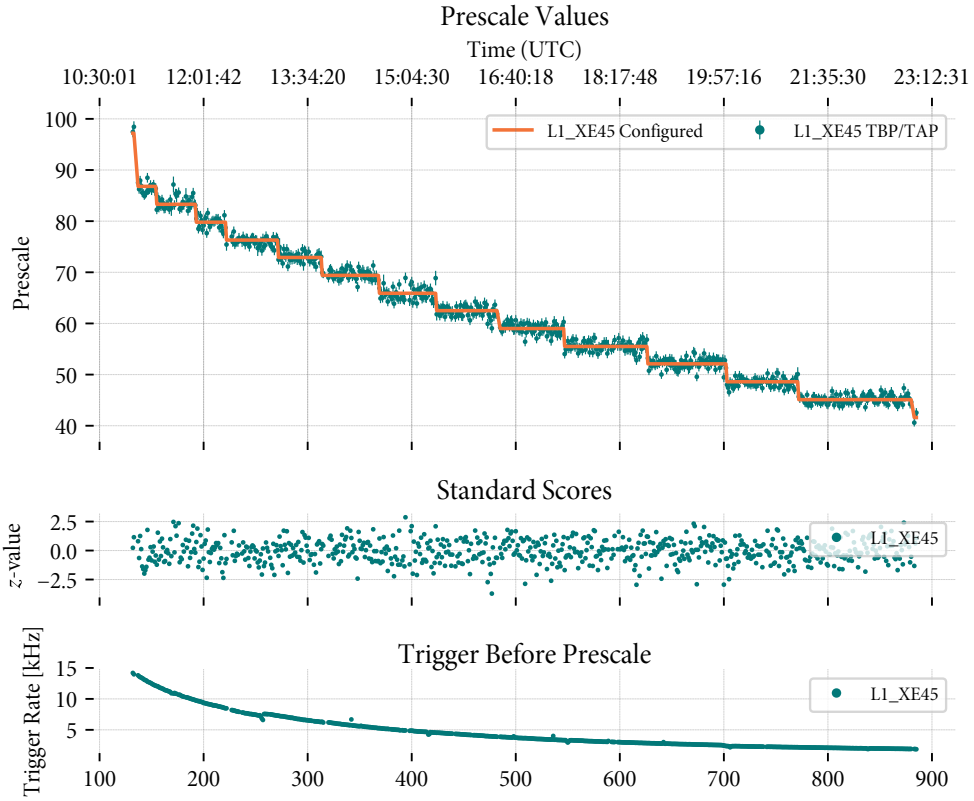


Figure 7.8: Observed and configured prescaled (top), z-score of observed prescale behaviour (middle), and trigger rate before prescale (bottom), for `L1_XE45` during run 311481,

The top plot of Figure 7.8 shows the prescale evolution for `L1_XE45` during Run 311481. The estimated prescale value per luminosity block as well as the configured values are plotted. The middle plot shows the z-score for each luminosity block, defined as

$$z = \frac{p_{\text{conf}} - p}{\sigma_p} \tag{7.6}$$

where p_{conf} is the configured prescale, p is the estimator, and σ_p is the uncertainty on the estimator. The reduced χ^2 of the goodness of fit between the estimated prescale and the configured prescale is

$$\chi^2 = 0.99987\,, \tag{7.7}$$

and similarly the p-value of $p = 1$. Both of these results are to be expected in the normal case where the CTP is applying the correct prescaling. Consequently, for

any analysis that needs the prescale value, the *configured* value is preferred over the *estimated* prescale value.

7.3.2. Luminosity Block Stability

Stable experimental conditions are a key requirement for the introduction of luminosity blocks, discussed in Section 5.2. ATLAS is able to enforce this at the level of configuration and conditions of the sub-detectors, but has no way of doing so for the conditions of the LHC: the LHC may perform slight changes to the beam optics during a fill that change the luminosity. Even without such external changes, the luminosity is expected to drop over time as the bunch population depletes. The intra-luminosity block publications of the trigger rates can be used to quantify the stability of the luminosity during the luminosity block, as the rate is proportional to the luminosity, as discussed in Section 7.2.3. This is an important cross-check as only the mean trigger rate (and the uncertainty on the mean) can be obtained from the monitoring data normally archived for later analysis: only the total number of triggers during a luminosity block, for all triggers, along with the corresponding turn counters, are stored in the trigger database.

The trigger rate monitoring was carried out once roughly every second during run 311481, yielding ~ 60 rate measurements per luminosity block for all triggers at all stages of the trigger path. For this cross-check, only the TBP values will be considered as they provide the clearest picture of the luminosity evolution. Each publication of the trigger rates contains the number n_i^T of triggers for all triggers T as well as the value of the relevant turn counter $n_{i,\text{turn}}$, since last publication. Using (4.1) and (4.2), a rate estimate for the $i^{\underline{\text{th}}}$ publication can be obtained as:

$$R_i^T = f\,\frac{n_i^T}{n_{i,\text{turn}}} \pm f\,\frac{\sqrt{n_i^T}}{n_{i,\text{turn}}}\,. \tag{7.8}$$

Similarly, by summing the value of all publications, an estimate of the mean trigger rate over a luminosity block can be obtained as

$$\left\langle R^T \right\rangle = f\,\frac{n^T}{n_{\text{turn}}} \pm f\,\frac{\sqrt{n^T}}{n_{\text{turn}}}\,, \tag{7.9}$$

where $n^T = \sum_i n_i^T$ and $n_{\text{turn}} = \sum_i n_{i,\text{turn}}$. [3]

[3] n^T and n_{turn} are exactly the values stored in the trigger database; the intermediate values, n_i^T and $n_{i,\text{turn}}$, need to be extracted from the archived CTP specific monitoring data.

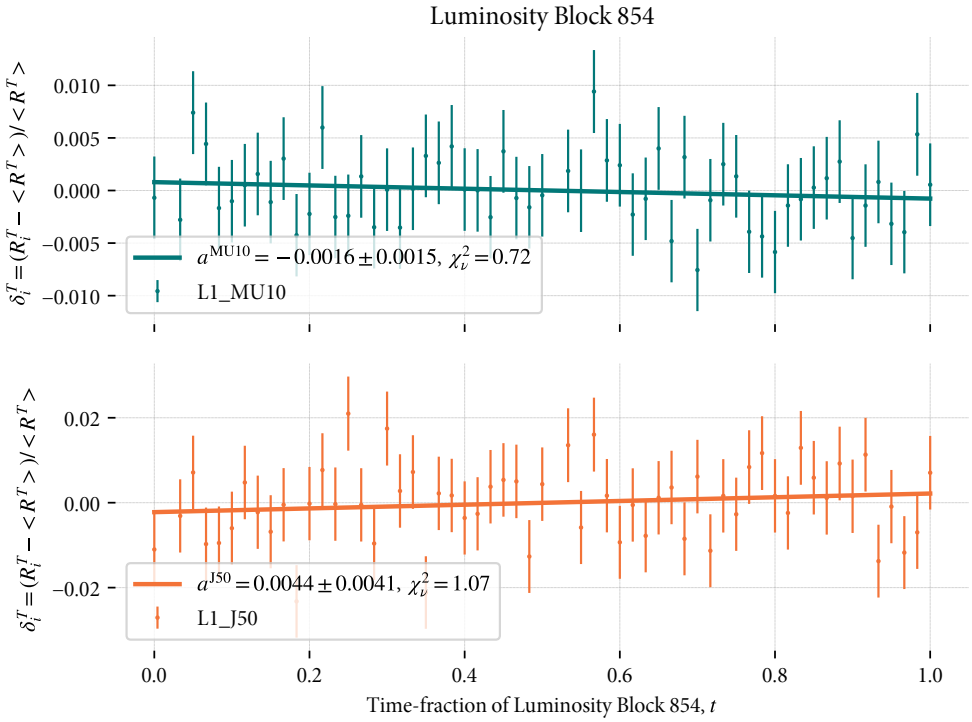


Figure 7.9: Relative trigger rate as function of the luminosity block duration for `L1_MU10` (top) and `L1_J50` (bottom). For both, a straight line has been fitted.

As a measure of the stability of the trigger rate and luminosity, it is instructive to look at the relative difference δ_i^T between a rate publication R_i^T for some trigger T, and the mean trigger rate $\langle R^T \rangle$ of the same trigger:

$$\delta_i^T = \frac{R_i^T - \langle R^T \rangle}{\langle R^T \rangle} = \frac{\mathcal{L}_i - \langle \mathcal{L} \rangle}{\langle \mathcal{L} \rangle} \,. \tag{7.10}$$

Figure 7.9 shows plots of δ_i^T for two orthogonal triggers, `L1_MU10` and `L1_J50`, as function of the relative time within the luminosity block, $t = 0$ being the start of the luminosity block and $t = 1$ being the end of the luminosity block. The triggers are orthogonal in the sense that they are based on information coming from different detector types (muon spectrometer and calorimeter, respectively) and have no data or processing overlap. This becomes important in later discussion.

A straight line is fitted to the data points shown in the figure, and the slope, a_T, its uncertainty, and the reduced χ^2 of the fit is given in the legend. The small value of the slopes, $\mathcal{O}\left(10^{-3}\right)$, and a reduced χ^2 of ~ 1 already provides some

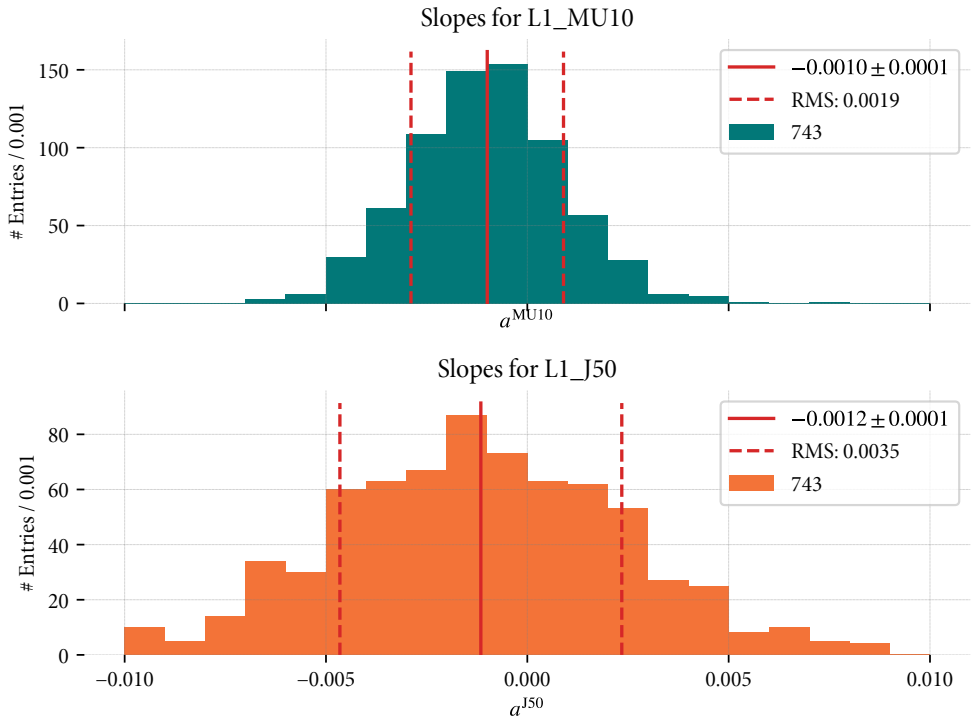


Figure 7.10: Distribution of fitted slopes of relative trigger rate, run 311481, for `L1_MU10` (top) and `L1_J50` (bottom).

evidence that the conditions are close to constant. However, as illustrated by the small difference between the slope in the top and bottom plot of Figure 7.9, statistical fluctuations within the luminosity block affects the slope obtained.

Figure 7.10 shows the distribution of slopes obtained by fitting a straight line to every luminosity block with colliding beam during the run. The mean slope and RMS obtained for `L1_MU10` and `L1_J50` are

$$\left\langle a^{\mathrm{MU10}} \right\rangle = -0.0010 \pm 0.0001\,, \;\; \mathrm{RMS}\left(\left\langle a^{\mathrm{MU10}} \right\rangle\right) = -0.0019\,; \tag{7.11}$$

$$\left\langle a^{\mathrm{J50}} \right\rangle = -0.0012 \pm 0.0001\,, \;\; \mathrm{RMS}\left(\left\langle a^{\mathrm{J50}} \right\rangle\right) = -0.0035\,. \tag{7.12}$$

The broader distribution obtained for `L1_J50` is a reflection of the lower trigger rate of this item compared to `L1_MU10` (see Figure 7.5) and thus the higher statistical uncertainty on the rate measurements. For both, however, a value of $\langle a^T \rangle \approx -0.001$ is obtained reflecting an average relative drop in luminosity (and trigger rate) of $\approx 0.1\%$ during a luminosity block.

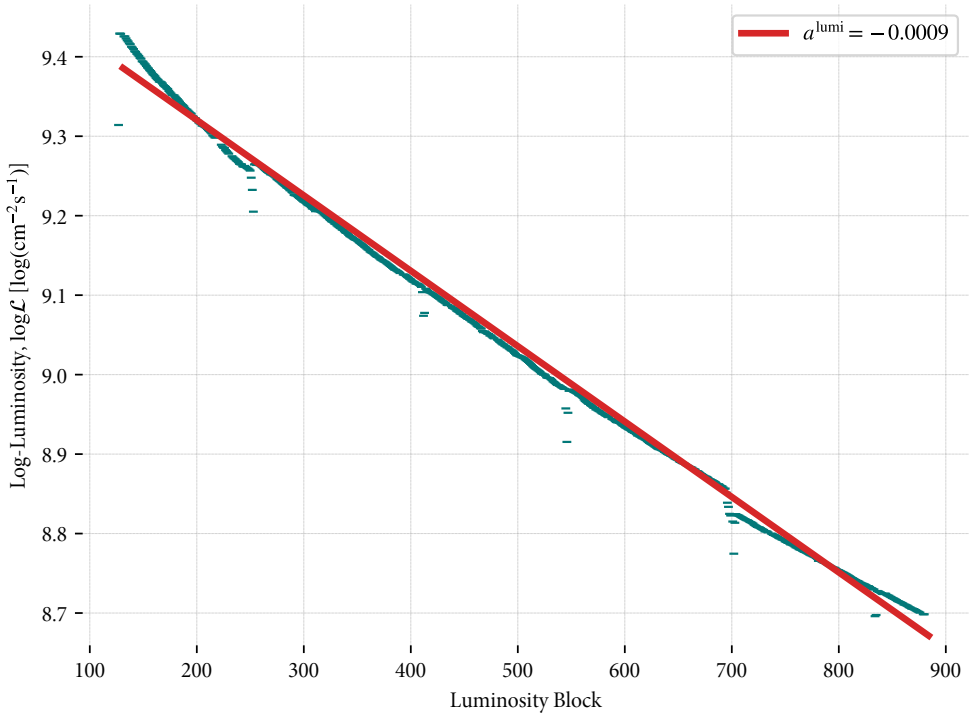


Figure 7.11: Log-luminosity as function of luminosity block during run 311481 and an exponential function (red) with time constant taken from a linear fit.

The luminosity profile of run 311481 as function of luminosity block is shown in the bottom plot of Figure 7.5. The luminosity profile can be approximated as an exponential function, and an expected relative drop per luminosity block can be obtained from the characteristic time of the exponential function fitted to the profile.

Figure 7.11 shows the luminosity as function of luminosity block on a log-plot, overlaid with the result of the fit. The slope of the straight line is

$$a^{\text{lumi}} = -0.0009 \pm 2 \times 10^{-6}\,, \tag{7.13}$$

which is consistent with the two values for $\langle a^T \rangle$ obtained above.

A few luminosity blocks were found to exhibit strange behaviour, not consistent with statistical fluctuations. Two such luminosity blocks are shown in Figure 7.12. In both cases, a relative drop of up to $\sim 10\%$ can be observed for up to ~ 0.25 of the duration of the luminosity block (or roughly 15 s assuming a 60 s luminosity block). Given that this phenomenon is observed for two orthogonal

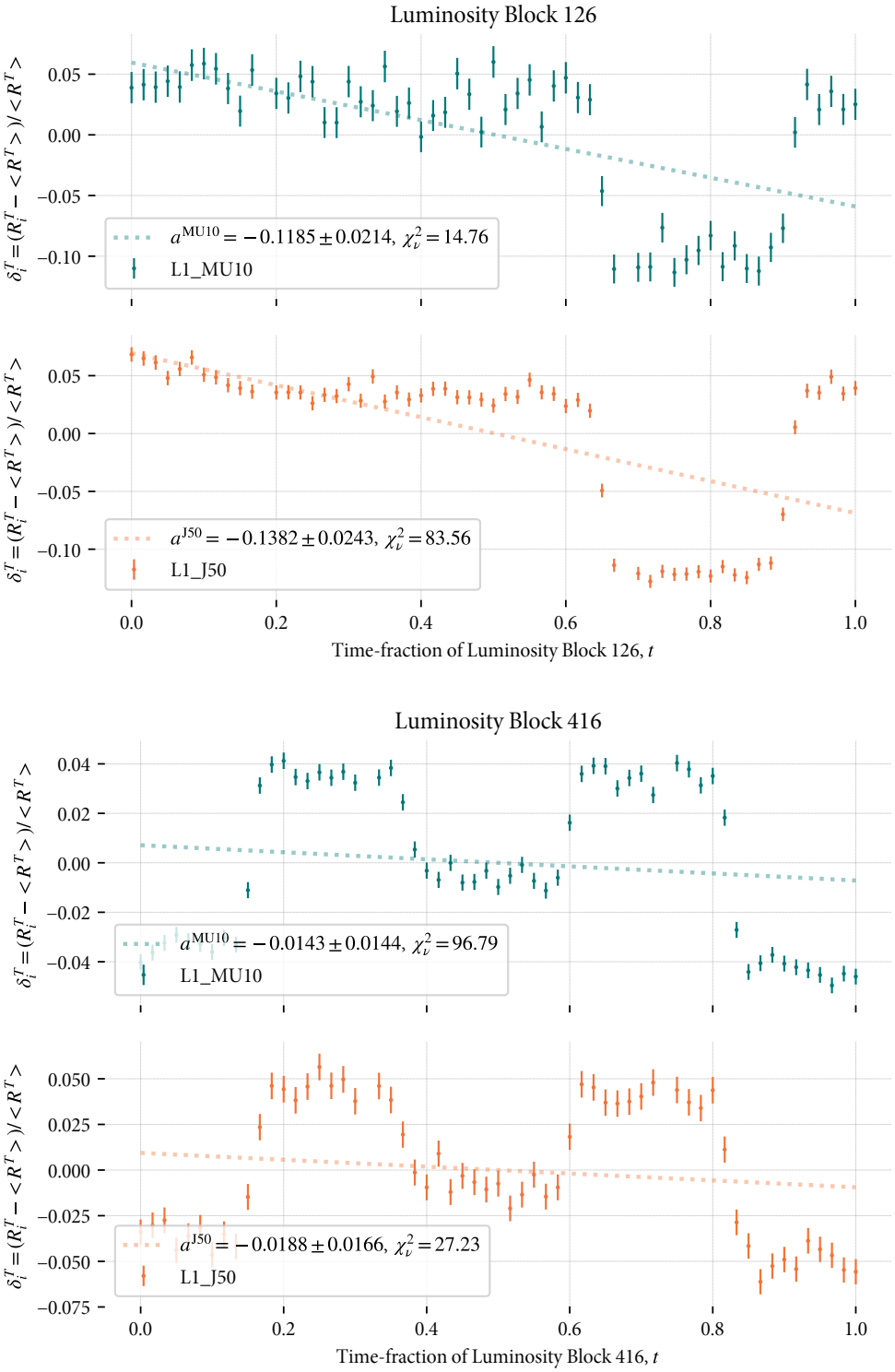


Figure 7.12: Relative trigger rate as function of the luminosity block duration for luminosity block 126 (top) and 416 (bottom), run 311481, for `L1_MU10` (sub-plot top) and `L1_J50` (sub-plot bottom)

triggers, it is reasonable to assume that the drop is caused by an LHC adjustment to the beam parameters, causing the instantaneous luminosity to change. As the LHC is unaware of ATLAS luminosity blocks, there are typically no prior warning. In the analysed run, 24 out of 739 (3.2%) luminosity blocks exhibited such behaviour.

7.3.3. Dead-time Corrections Factors

The dead-time correction factors are needed for any cross section measurement in order to correct for the data not recorded during dead-time and thus relate the recorded luminosity to the delivered luminosity. In the databases used for analysis, one typically has the delivered luminosity, while for an analysis, one needs the recorded luminosity, which is the delivered corrected by the dead-time correction factor. In Section 5.3.1 two expressions for the dead-time correction factors for a data-set recorded with a trigger T were given. The first and most general expression, given in (5.13), makes use of per-bunch luminosity and per-bunch TAP and TAV counters. Under the assumption that dead-time is equally distributed on all filled bunches, this expression simplifies to the second expression (5.14), which only relies on the per-luminosity block integrated TAP and TAV counters. As detailed in Section 5.3.1, several affects might challenge this assumption. ATLAS relies on the second method for determining the dead-time correction factors and thus a cross check of the two expressions – (5.13) and (5.14) – and of the underlying assumption is warranted. The added per-bunch monitoring capabilities of the CTP allow for two such cross checks to be made: by using two different estimates for the per-bunch dead-time fraction (or conversely live-time fraction), two different dead-time correction factors can be obtained using (5.13). These can then be compared directly to the official value.

The ATLAS recommended method for obtaining the dead-time correction factors, uses the second expression, (5.14), with a well-understood high rate physics trigger T' as a 'drop-in' replacement for the trigger T:

$$l^{T}_{\mathrm{ATLAS}} = \frac{n^{T'}_{\mathrm{TAV}}}{n^{T'}_{\mathrm{TAP}}} \,. \tag{7.14}$$

The motivation for using a 'drop-in' replacement is to avoid being limited by statistics for low rate triggers where n^{T}_{TAV} and n^{T}_{TAP} might be small. The trigger item T' used by ATLAS as well as in the following cross-checks, is `L1_EM24VHI`[4].

[4] `VHI` stands for Veto Hadronic and Isolation: it is an `EM` trigger item with an additional isolation requirement and a strict veto on hadronic activity in the ROI. See Figure 3.4.

The two cross-checks are carried out by comparing the dead-time correction factors obtained from the per-bunch calculation (5.13) to the official value. There are two ways of obtaining an estimator of the per-bunch live-time fraction l_i^T seen by the trigger T using the CTPs per-bunch monitoring capabilities:

1. By direct calculation from the per-bunch TAP and TAV counters:

$$l^T_{i,\text{direct}} = \frac{n^T_{i,\text{TAV}}}{n^T_{i,\text{TAP}}} \,. \tag{7.15}$$

2. By substituting the per-bunch dead-time seen by the trigger T with the per-bunch total experimental dead-time:

$$l^T_{i,\text{exp}} = 1 - d_i \,. \tag{7.16}$$

For (7.15) the statistical uncertainty, assuming binomial statistics, is

$$\sigma^T_{i,\text{direct}} = \frac{1}{\sqrt{n^T_{i,\text{TAP}}}} \sqrt{l_i^T \left(1 - l_i^T\right)} \,. \tag{7.17}$$

For (7.16), the per-bunch experimental dead-time is determined from the per-bunch dead-time counters as the fraction of bunch-crossings with experimental dead-time n_i^d to the number of turns n^d_{turn}. Thus, again assuming binomial statistics, the statistical uncertainty on the estimator obtained from (7.16) is

$$\sigma^T_{i,\text{exp}} = \frac{d_i^T}{\sqrt{n^d_{\text{turn}}}} \,. \tag{7.18}$$

To calculate the dead-time correction factor based (5.13), the per-bunch luminosity $\mathcal{L}_i$ is needed too. The relative uncertainty on the online $\mathcal{L}_i$ is taken to be 5% based on [86].

The expressions for the statistical uncertainties given in (7.17) and (7.18) will underestimate the error in the limit $l_i^T = 1 - d_i^T \approx 1$. However, as will be discussed in the following, the statistical uncertainty on the estimators in these cross-checks will be dominated by a systematic uncertainty on the per-bunch counter values, and thus a more elaborate treatment of the statistical error, such as the one outlined in [87], is well beyond the scope of this discussion.

The way that the per-bunch monitoring capabilities of the CTP is being utilised and read out causes a systematic error on the per-bunch counter values due to a "spill-over" effect: to reduce the dead-time during luminosity block

transition, none of the per-bunch monitoring routines are tightly coupled to the luminosity block transition, and are only carried out *after* the transition is done (see Figure 6.7). This leads to a spill-over effect in case the counters were started in one luminosity block and read out in the next. It should be stressed, that this effect affects all per-bunch monitoring but not the per-luminosity block monitoring: the total number of triggers per luminosity block, for all triggers T, at all stages along the trigger path (TBP, TAP, and TAV) are read out as part of the critical monitoring during the luminosity block transition as discussed in Section 5.6 and Section 6.3. This ensures that the dead-time correction factors based on (5.13), can be determined correctly for every luminosity block.

The affects of this spill-over on the cross-check calculations depends on a) how different, in terms of trigger rate and dead-time, the two luminosity block are, and b) the length of the luminosity blocks. There is unfortunately no trivial way of correcting for the spill-over. In order to suppress the effect, the luminosity blocks considered in the following are required to

- be at least 55 s long – to suppress operational issues (manual intervention) and minimise the chance and relative size of a spill-over ;
- have a discrepancy of maximum 10% in integration time – to suppress spill-over.

The integration time is determined from the turn counters present for both the per-luminosity block counters and for the per-bunch counters: the length of the luminosity block can be determined to within a couple of LHC revolutions, based on the luminosity block integrated turn counters. Comparing this turn count to the value of the turn counter of the per-bunch monitoring provides a strong handle on the size of the spill-over. The requirement of a maximum 10% discrepancy between the two integration times was chosen as a trade-off between sample size and sample purity: in most cases – where the two neighbouring luminosity blocks are comparable in terms of dead-time and trigger rate – the effect of the spill-over on the calculation will be small, which favours a looser selection criteria. The biggest differences caused by the spill-over effect, is expected for luminosity blocks where the experimental conditions differ greatly in terms of trigger rate and dead-time. The worst-case spill-over can be estimated to be $\sim 4.8\%$ for a luminosity block of length 60 s, 100% dead-time, and a spill-over of 3 s into a luminosity block with 0% dead-time during the spill-over. This situation is symmetric and it is expected that transition from a "good" luminosity block to a "bad" luminosity block happens as often as the other way around[5]. In the more common case, with

[5]Selecting luminosity blocks based on a requirement of a minimum or maximum dead-time would break this symmetry and consequently introduces a shift in either direction.

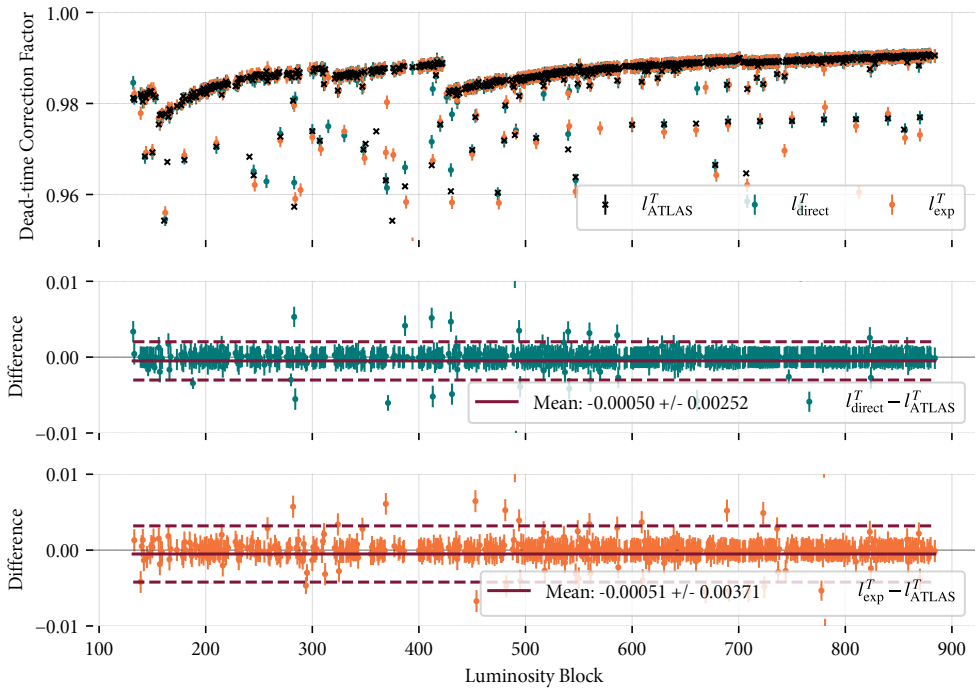


Figure 7.13: Dead-time correction factors for each luminosity block of 311481 passing basic selection criteria, calculated in three different ways (top). The difference between the official method l^T_{ATLAS} and each of the two cross check methods are shown in the middle and bottom plot.

nearly identical experimental conditions between neighbouring luminosity blocks, the effect of the spill-over is much smaller: assuming more optimistically, near homogeneous conditions and no notable dead-time in the two luminosity blocks, and a $\sim 10\,\text{ms}$ period of 100% dead-time in the beginning of the next luminosity block – introduced by the luminosity block transition and captured in the spill-over – the expected difference between the cross-check and the recommended method is expected to be $\mathcal{O}\,(0.01\%)$.

For the cross-checks presented here, luminosity blocks of run 311481 passing the above selection was used. Out of 701 luminosity blocks for which there was both collisions and luminosity information available, 643 luminosity blocks pass the above selection criteria. The top plot of Figure 7.13 shows the three different dead-time correction factors as function of luminosity block: l^T_{ATLAS} obtained from (7.14) as well as l^T_{direct} and l^T_{exp}, obtained from (5.13) by using $l^T_{i,\text{direct}}$ and $l^T_{i,\text{exp}}$, respectively, as estimator for the per-bunch live-time fraction. The middle and bottom plot of Figure 7.13 shows the the difference between the

value obtained via the two cross-checks, l^T_{direct} and l^T_{exp}, and the official value, l^T_{ATLAS}. The uncertainties shown in the plots are the statistical uncertainties. As expected, the statistical uncertainty can be seen to underestimate the total uncertainty: for both cross checks a core distribution with a smaller scattering, consistent with the statistical error, can be observed, but outliers, likely caused by the spill-over effect, drives up the uncertainty on the mean. For completeness a plot of the distribution of the differences is included in Appendix A.3.

The observed mean differences between the two cross checks and the official value are:

$$\Delta_{\text{direct}} = \left\langle l^T_{\text{direct}} - l^T_{\text{ATLAS}} \right\rangle = -0.00050 \pm 0.00252 \tag{7.19}$$

$$\Delta_{\text{exp}} = \left\langle l^T_{\text{exp}} - l^T_{\text{ATLAS}} \right\rangle = -0.00051 \pm 0.00371\,. \tag{7.20}$$

The relative differences, as compared to l^T_{ATLAS}, are in both cases $\mathcal{O}(0.1\%)$. This is consistent with the expectation of a small to zero difference between the two ways of calculating the dead-time correction factors. While it is difficult to conclude how much of this is an affect of the method or of the data, the difference between the cross-checks and the reference is so small that it is unlikely to have had any impact on any ATLAS result. For the cross section calculation, discussed in Section 5.1, the relative difference of $\mathcal{O}(0.1\%)$ is to be compared with the relative uncertainty on the luminosity, which is 3.2% for the combined 2015-2016 data [88] and 2.0% for the combined 2015-2017 data [89]. Given the much larger uncertainty on the luminosity, the impact of using the simplified method for the calculation of dead-time correction factors can be neglected.

7.4. Monitoring Effects of Sub-system Failure

The discussion so far has dealt with the most common case where there are no significant operational issues. Sometimes, however, things do not go according to plan and this is where the monitoring data is most important for the operation of the experiment – both during the run but also after the run is over. The monitoring data is used during the run for identifying the issue and doing damage control, and is studied again after the issue is resolved as part of a post-mortem analysis, from which one can hopefully learn how to avoid or how to detect and respond to such incidents in the future.

This section gives two examples of incidents and illustrates how they impact the trigger system. Lastly, a couple of run summary plots, produced using the CTP data, is presented – these are similar to the ones presented every morning at the ATLAS run meeting as the basis for discussion of operational issues.

7.4.1. Magnet Outage

During run 299055, at around 7 o'clock in the morning, Geneva time, on the 13$^{\underline{\text{th}}}$ of May 2015, the toroid magnets where shut down. The magnet dump was a controlled "slow dump" and the decision to dump the magnets was postponed as long as possible towards the end of the ongoing fill, in order to minimise the data loss. It was decided to keep the run going for as long as LHC provided collisions: while the ATLAS detector is running, it provides LHC with an luminosity measurement.

Ramping down the magnetic field during a run gives rise to a myriad of side effects, many of which were reflected in the trigger system.

The most pronounced effect is that on the muon triggers. The muon detectors rely on the toroidal field for determining the momentum of muon candidates. The decrease in magnetic field strength translates directly into an apparent increase in muon candidate p_{T}. As a direct result, the trigger rate for all muon items sky-rocketed during the dump.

Another set of issues arose as current was induced in some of the detector electronics of nearby detectors – both the muon detectors and the calorimeters were affected. The implication on the trigger system was an added noise on almost all triggers caused by the erroneous signals. However, compared to the effect on the muon triggers, these were more of a curiosity.

The top plot of Figure 7.14 shows the TBP rate for a number of muon triggers. From when the magnet dump is started, around luminosity block 650, the rate can be seen to increase dramatically as the apparent muon candidate p_{T} increases. Not surprisingly, the low threshold muon triggers are most affected. The middle plot of Figure 7.14 shows the rate after prescaling for the same four muon items, and the bottom plots shows the experimental dead-time. Due to the heavy prescaling on the low threshold muon items, the increased rate could be sustained without introducing significant amounts of dead-time.

7.4.2. Calorimeter Hot Cell

On occasion, parts of the LAr calorimeter are prone to generating noise bursts when detector-specific parameters, such as temperature, pressure, and high voltage, for some reason are not in tune[6]. Such an issue quickly propagates through the trigger system, as the cell energy value is transferred to L1Calo, where it is

[6] Unfortunately, at present it is not really understood what causes these noise bursts. Their rate is luminosity dependant, and it is known that one can quiet it down by lowering the voltage by some percent. [90]

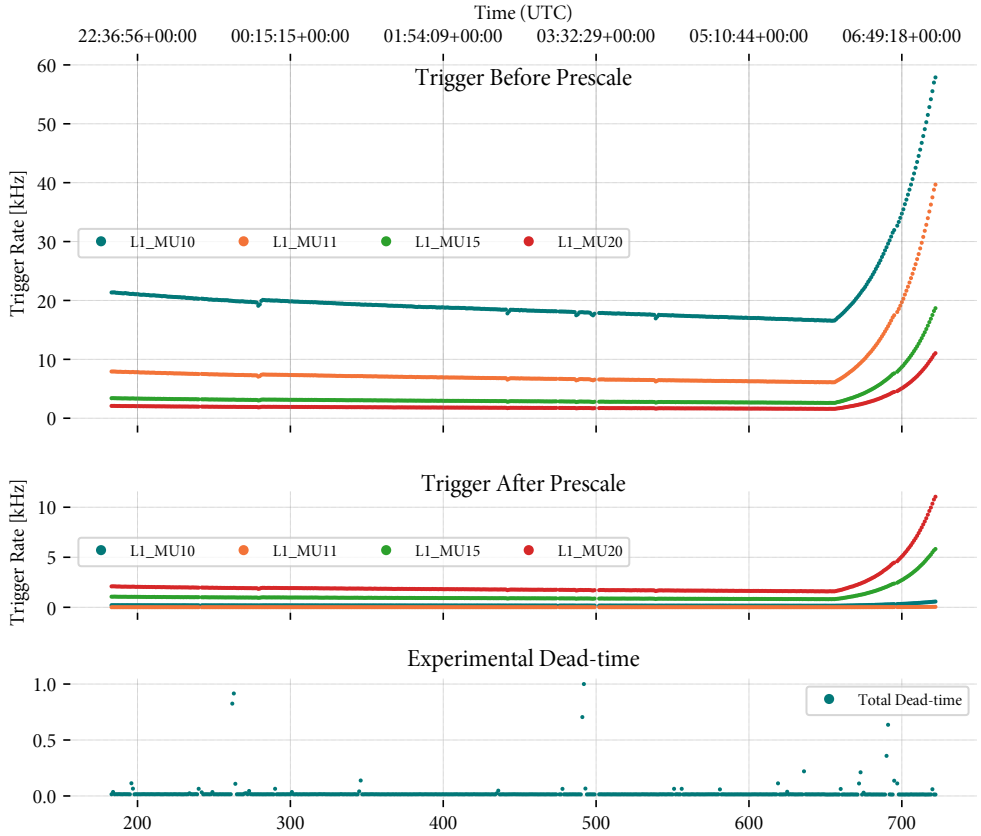


Figure 7.14: Plot of the trigger rate before prescale (top) and after prescale (middle) for various muon triggers, as well as the experimental dead-time (bottom) during run 299055, as a function of luminosity block and time.

calibrated, clustered, and used to determine the energy of what the pre-processor might perceive as a physics object.

Often equilibrium is restored quickly within seconds, but sometimes it takes longer and might require manual intervention by a detector expert.

From an operational point of view, discrimination against this type of operational issue can be achieved by observing that calorimeter noise is generally uncorrelated with the bunch instantaneous luminosity, i.e. independent on whether there are colliding bunches or not. Recalling from Section 4.4.1.2 that a trigger item is a combination of a trigger input and the applied bunch group mask, it is possible to create a trigger item, using the same trigger input but a different bunch group, for instance the bunch group of bunch crossings without collisions. That is, the trigger `L1_TAU8_EMPTY` is based on the same trigger requirement as

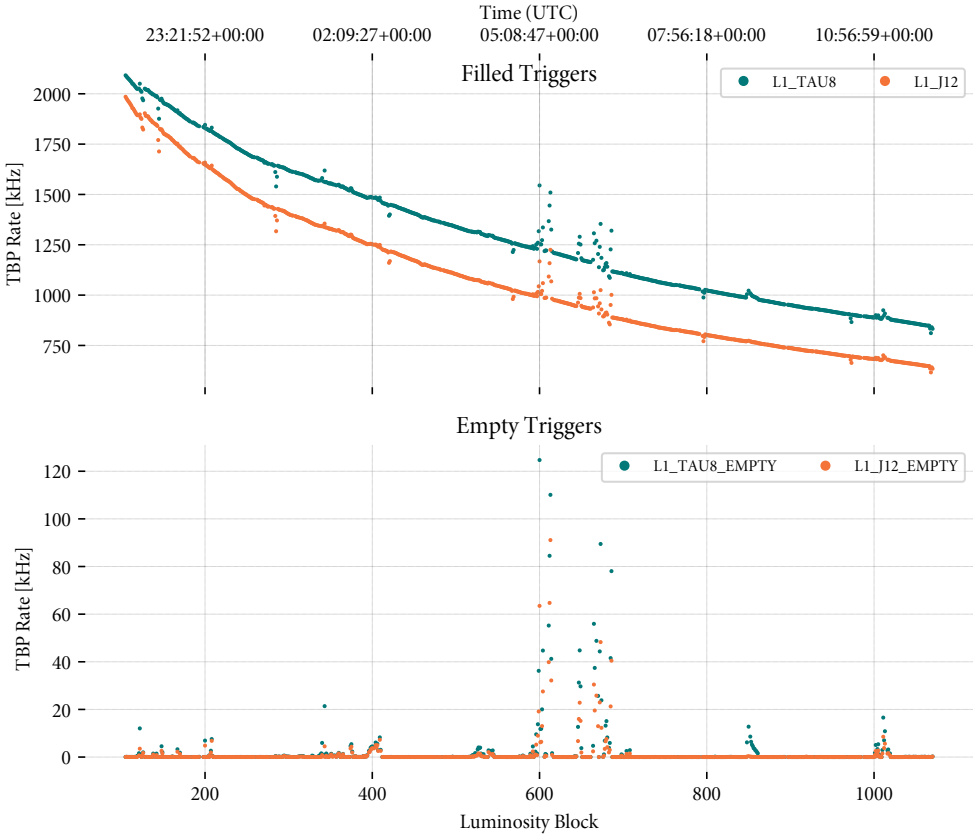


Figure 7.15: The plot shows the TBP rates for two calorimeter based triggers during run 310738. The top plot shows the trigger rate in bunch crossings with colliding beams while the bottom plot shows the trigger rate for items based on the same trigger condition but confined to bunch crossings without collisions.

`L1_TAU8` but is confined to bunch crossings without beam-beam collisions. The ”empty” triggers serve as an excellent indicator of calorimeter noise.

Figure 7.15 shows the trigger rate of `L1_TAU8` and `L1_J12` (top) and the corresponding empty triggers, `L1_TAU8_EMPTY` and `L1_J12_EMPTY` (bottom) during run 310738. Run 310738, as it turned out, was a particularly bad run, with more issues than those caused by the calorimeter, as will be discussed later. As can be seen from the plot, the run contains several prolonged periods with calorimeter noise, affecting calorimeter-based triggers. Two major periods with noise bursts can be seen around luminosity block 600 and slightly before luminosity block 700, however several smaller periods can also be seen around luminosity block 200, 400, 550, and 1000. The time-wise correlation between empty and filled

triggers as well as the noise being visible across several trigger types strongly support an operational issue in the calorimeter. As can be seen from the plot, the rate increase caused by a noisy period, varies. In the cases depicted, the rate increase varies from $\sim 5\,\text{kHz}$ to $\sim 100\,\text{kHz}$. Noise bursts causing rate increases of $\mathcal{O}\,(\text{MHz})$ are not unheard of. With the rate exceeding 100 kHz, ATLAS will experience increased dead-time from sub-detectors struggling to keep up with the rate.

The noise bursts are problematic for the trigger system for a number of reasons. Firstly, triggering on corrupt data decreases the trigger efficiency and impacts the quality and utility of the recorded data. In fact, the luminosity blocks affected by noise bursts are likely going to be flagged as corrupted/useless by data quality. Secondly, noise bursts makes it difficult to effectively control the trigger rates and as will be shown later, the consequences of exceeding – or attempting to exceed – the maximum Level 1 trigger rate, quickly leads to a degraded performance with high experimental dead-time. In order to suppress the effects of the noise bursts, it is the strategy to apply a very large prescale to the affected trigger items until the issue is resolved and normal operation can be resumed.

Using the monitoring of the individual trigger items as well as the detailed monitoring of the experimental BUSY and dead-time typically allow for the responsible sub-system to be identified, in this case the calorimeter system. These typically have more detailed information and tools to mitigate the problem, in this case the ability to mask and re-configure the tower in question. Until the issue is resolved, the trigger expert can treat the issue symptomatically by increasing the prescale of affected items.

7.4.3. Post-mortem Analysis

So far, the discussion of the dead-time has focused on the over-all experimental dead-time and on the preventive dead-time. However, the dead-time monitoring capabilities of the CTP extend beyond that. A detailed break-down of the experimental dead-time as function of time, is commonly used after a run to better understand what happened, in particular in case of any operational issues.

Figure 7.16 shows a busy summary for run 314199. The plot is produced using archived monitoring data from the CTP with a time granularity of one value per luminosity block. For more extensive troubleshooting or analysis, a similar plot can be produced with a time granularity of the typical publishing frequency of once per 3-5 seconds. While the instantaneous busy monitoring data is constantly displayed in the control room (Section 6.7.2.2) and used to identify

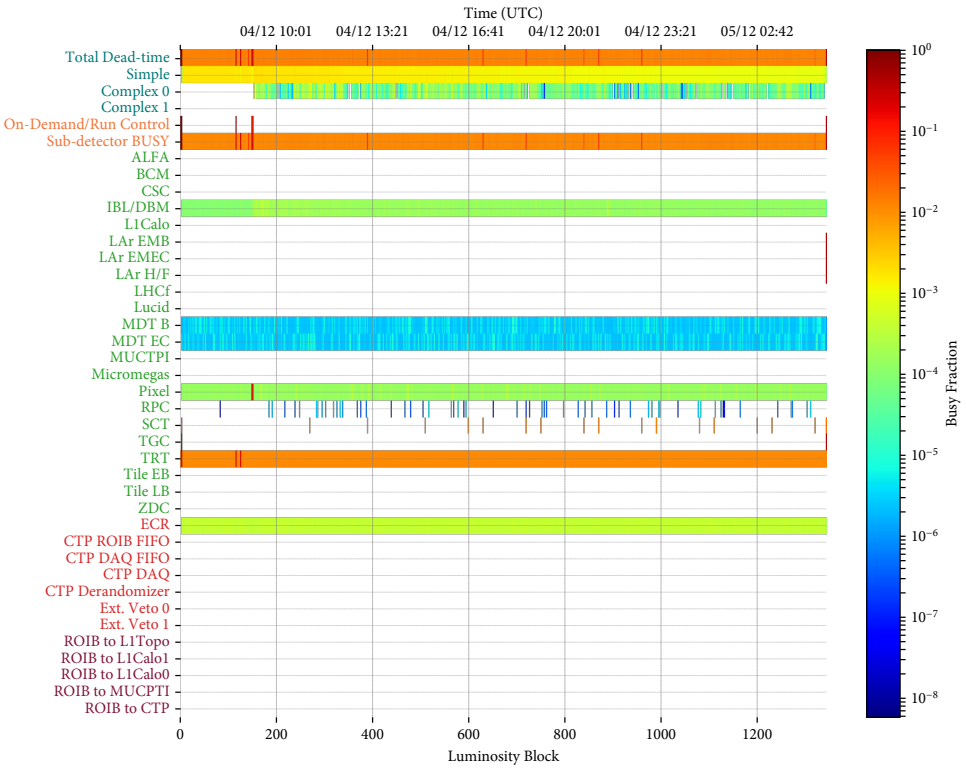


Figure 7.16: Busy summary for run 314199, showing the busy and dead-time fraction for each luminosity block for the various busy sources.

and troubleshoot issues as they occur, plots like Figure 7.16 are used as a talking point during daily and weekly run-coordination meetings where the issues are discussed with the sub-detector communities, and for error statistics, where they help to quantify the number of errors per sub-system. Another important use-case of the archived dead-time monitoring data is post-mortem analysis, where the busy information, in particular the fine-grained busy information, can point to the exact time.

The top rows of Figure 7.16, labelled in dark green, show total experimental dead-time and the preventive dead-times. As can be seen, the complex dead-time only becomes pronounced around luminosity block 130, when stable beams are declared. At that point, the prescale sets are changed from a set designed for background studies to a set for physics data-taking, and the Level 1 trigger rate goes from $\sim 1\,\text{kHz}$ to around $80\,\text{kHz}$.

The total experimental dead-time is the logical OR of the preventive dead-time, and the sub-detector BUSY. The BUSY from the sub-detectors is, in turn,

the logical OR of the BUSY from each sub-detector (labelled in light green) and the Busy-On-Demand, exercised by Run-Control.

The primary contribution to the dead-time can be seen to come from the TRT. The source of the TRT busy is a front-end polling procedure during which the detector can not receive any triggers. The amount of busy from the front-end polling amounts to an average of 1.15%. This polling is however only carried out during the calibration request sequence (see Figure 7.7 for BCID-wise location of the calibration request gap) and thus does not negatively impact data-taking.

Besides the TRT, the main source of sub-detector busy is the busy from the pixel detector, and the IBL. The high busy fraction of the two detector systems was an issue in the first months of operation at high rates close to 100 kHz. The two detectors were operated in tandem and the busy was primarily caused by (yet) slow firmware in the IBL readout. The dead-time coming from MDT and RPC is measurable but negligible.

In case of a detector issue or a loss of synchronisation, (automatic) recovery procedures are carried out, during which the On Demand busy is used to prevent more triggers from being issued, as described in Section 4.5.1. A couple of such procedures can be identified as a coincidence between a sub-detector busy and the on-demand busy. Short periods of high BUSY coming from other detector systems suggest temporary issues with detector readout.

The CTP itself can also be the source of busy, labelled in red in Figure 7.16, either in its capacity as a sub-detector with a ROD and a ROB or most commonly from the dead-time introduced around the generation of the ECR. Lastly, the HLT, via the ROIB, can exert back-pressure XOFF to any of the Level 1 trigger processors if it can't keep up with the L1A rate. This causes the local buffers in the trigger processors to fill up, and eventually forces the trigger processor to assert BUSY to avoid data loss. The busy caused by HLT back pressure is labelled in dark red.

In stark contrast to run 314199 is run 310738 with its many issues, including the recurrent calorimeter issues, that were discussed in Section 7.4.2.

A profile of the total L1A rate and the TAV rate of selected triggers are shown as function of time in Figure 7.17 along with the dead-time. The calorimeter issues discussed in Section 7.4.2 are visible in TAV rates and overall L1A rate too: the bursts shown in Figure 7.15 – e.g., around luminosity block 600 and 650 – can see to correspond to dips in the trigger rate after veto. The increased TBP rate shown in Figure 7.15 causes an increased dead-time (primarily coming from ROD busy) which leads to triggers being collectively vetoed. As can be seen from the dead-time profile in the lower plot of Figure 7.17, there are several

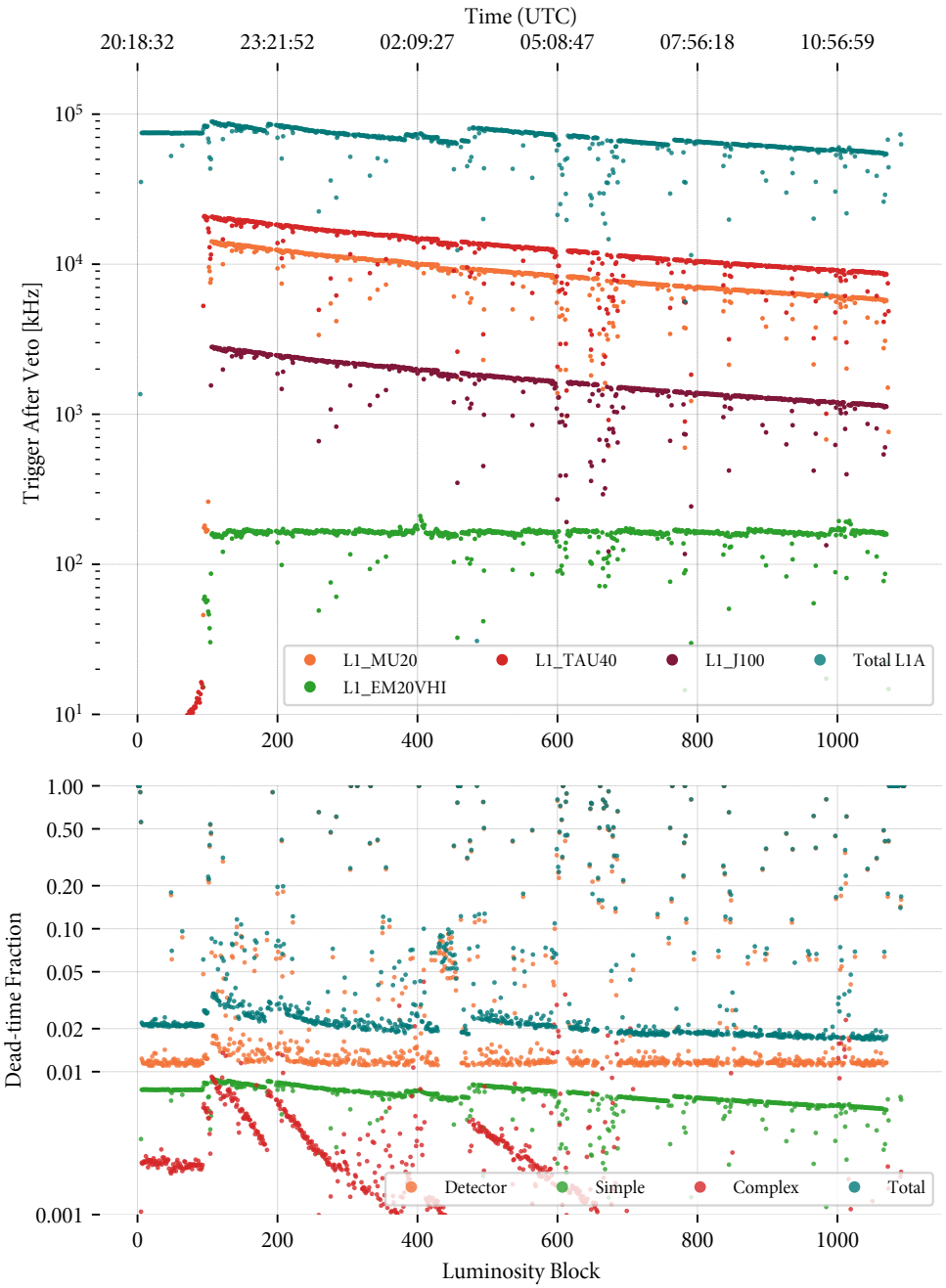


Figure 7.17: The Level 1 TAV rate (top) and dead-time (bottom) for each luminosity block in run 310738 as function of time.

Figure 7.18: Busy summary for run 310738, showing the busy and dead-time fraction for each luminosity block for the various busy sources.

luminosity blocks with dead-time well above the normal levels of $\sim 2\%$. Most of these are short bursts of a duration of order of one luminosity block, and a couple of these bursts are visible before the collision begins at around luminosity block 120. Around luminosity block 450 a prolonged period with increased dead-time caused by the sub-detectors can further be seen. From this level of detail, it is however not possible to say more.

However, the busy summary for run 310738 shown in Figure 7.18 provides further insights. The prolonged busy around luminosity block 450 was caused by the TGCs. Due to faulty software, the detectors would lose synchronisation, and when they do, they would be unable to automatically recover, using the standard (quick) re-synchronisation method [91] [92]. Additionally, the TGC suffered problems with high voltage trips notable in Figure 7.18 as short periods with full sub-detector busy [92]. An overall high trigger rate, combined with occasional bursts, resulted in an overall degraded performance of most systems.

An interesting observation is, that despite the noise bursts from LAr, very little dead-time is coming from any of the LAr sub-systems, indicating that the

system was otherwise functioning fine, and that the method of recovery did not require holding the trigger. This is often the case when the procedure consists of setting new configuration parameters (masking out problematic trigger towers) that are applied during the luminosity block transition. Staying with the calorimeters, the Tile calorimeters can be seen to generate dead-time at the (sub-)percent level. However, rather than an operational issue with Tile, this is ROD busy due to a too high Level 1 rate.

Lastly, the HLT can be seen to suffer. This is evident from the back-pressure on the various trigger processors at Level 1 from the ROIB. Additional processing time due to corrupted data fragments from sub-detectors and periods of high trigger rates were found to be the primary cause for the observed back-pressure [93]. The BUSY from the "CTP ROIB FIFO" is raised to avoid that the local buffers on the CTP overflow and corresponds to the "ROIB to CTP": as the ROIB stops accepting events the event data starts accumulating at the CTP (as well as at the other Level 1 trigger processors) until the CTP (or one or more of the other trigger processors) is forced to raise a BUSY to avoid data loss due to buffer overflow.

The trigger and dead-time monitoring data from the CTP is an invaluable operational aid for all of ATLAS for understanding state of the experiment and the data-taking. Through a couple of illustrative cases the monitoring data was used to show, identify, and document the source of operational issues and illustrate how one issue (with the calorimeter) can lead to a myriad of other problems down the line (here, in particular, the back pressure from the HLT). In the following, the monitoring data from the CTP is further used to automatically detect and correct for operational issues based on the characteristic signature in the trigger system of these issues.

Chapter 8

Automatic Trigger Rate Control

As part of my doctorate work I have designed, and supervised the implementation of, a system for automatic trigger rate control. The system is rule based and creates a feed-back loop where the monitoring data from the experiment, and particularly the trigger system, is used to alter the configuration of the trigger system. The system is currently in use during ATLAS data-taking and has operated without problems since its inception. This chapter outlines the motivation for creating such a system, as well the design of the Automatic Prescaler and its constituents. This discussion is followed by a discussion of possible extensions – not least how this system could be extended to include the HLT.

8.1. Motivation and Background

8.1.1. Manual Prescale Change

A large part of the day-to-day trigger operations revolves around adjusting the prescales, and thus trigger rate, throughout each run as the luminosity changes or problems arise, to minimise the experimental dead-time and ensure that the bandwidth allocation best matches the data-taking goals of ATLAS. The prescale of each trigger item and algorithm can be chosen individually. In order to make the management of prescales easier, prescale values are arranged into uniquely numbered prescale sets for the Level 1 prescales and the HLT prescales. Currently a number of prescale sets (roughly 10 to 20 sets), for both Level 1 and HLT, are generated in advance, based on the expected rate at a target luminosity and bandwidth allocations according to the trigger menu. The prescale sets are stored in a database, with keys, that can be used to download the prescale set to the hardware. It is the responsibility of the trigger operator in the control room to monitor the instantaneous luminosity during each run and apply the

Level 1 PSK	HLT PSK	Lower Limit, $< \mathcal{L}$	Upper Limit, $\geq \mathcal{L}$
21168	15603	$10500\,\mu b^{-1}s^{-1}$	$11000\,\mu b^{-1}s^{-1}$
21167	15602	$10000\,\mu b^{-1}s^{-1}$	$10500\,\mu b^{-1}s^{-1}$
21166	15349	$9500\,\mu b^{-1}s^{-1}$	$10000\,\mu b^{-1}s^{-1}$
21165	15348	$9000\,\mu b^{-1}s^{-1}$	$9500\,\mu b^{-1}s^{-1}$
21164	15347	$8500\,\mu b^{-1}s^{-1}$	$9000\,\mu b^{-1}s^{-1}$
21163	15346	$8000\,\mu b^{-1}s^{-1}$	$8500\,\mu b^{-1}s^{-1}$
21162	15345	$7500\,\mu b^{-1}s^{-1}$	$8000\,\mu b^{-1}s^{-1}$
21161	15344	$7000\,\mu b^{-1}s^{-1}$	$7500\,\mu b^{-1}s^{-1}$
21160	15343	$6500\,\mu b^{-1}s^{-1}$	$7000\,\mu b^{-1}s^{-1}$
21159	15342	$6000\,\mu b^{-1}s^{-1}$	$6500\,\mu b^{-1}s^{-1}$
21158	15341	$5500\,\mu b^{-1}s^{-1}$	$6000\,\mu b^{-1}s^{-1}$
21157	15340	$5000\,\mu b^{-1}s^{-1}$	$5500\,\mu b^{-1}s^{-1}$
21156	15339	$4500\,\mu b^{-1}s^{-1}$	$5000\,\mu b^{-1}s^{-1}$
21155	15338	$4000\,\mu b^{-1}s^{-1}$	$4500\,\mu b^{-1}s^{-1}$
21154	15337	$3500\,\mu b^{-1}s^{-1}$	$4000\,\mu b^{-1}s^{-1}$
21153	15336	$3000\,\mu b^{-1}s^{-1}$	$3500\,\mu b^{-1}s^{-1}$
21152	15335	$2500\,\mu b^{-1}s^{-1}$	$3000\,\mu b^{-1}s^{-1}$
21151	15334	$2000\,\mu b^{-1}s^{-1}$	$2500\,\mu b^{-1}s^{-1}$
21150	15333	$1500\,\mu b^{-1}s^{-1}$	$2000\,\mu b^{-1}s^{-1}$
21149	15332	$1000\,\mu b^{-1}s^{-1}$	$1500\,\mu b^{-1}s^{-1}$
21148	15331	$800\,\mu b^{-1}s^{-1}$	$1000\,\mu b^{-1}s^{-1}$

Table 8.1: Example table of Level 1 and HLT prescale-set keys for different luminosity levels.

correct combination of Level 1 and HLT prescale sets, using a provided table like Table 8.1, when the luminosity has dropped sufficiently. Each Prescale Key (PSK) is a unique identifier for a complete set of Level 1 or HLT prescales. The Level 1 prescales and HLT prescales are always applied pairwise as the effective prescale of any item is the product of the Level 1 prescale and the HLT prescale.

In case of an operational issue, the trigger operator will contact a trigger expert who will create an ad-hoc prescale set, often based on the currently applied one, which allows data-taking to continue while the issue is being resolved. After an issue has been resolved, the set of prescales are reverted to the ones for normal operation.

The luminosity blocks during periods with operational issues are likely to be marked as invalid by data-quality, rendering the recorded data useless. Adapting the prescales on the fly is thus more about damage control than about recording

good data – it is about ensuring that as soon as an issue is resolved, the recording of interesting physics data can resume without a huge backlog of unbalanced and potentially corrupted events to process or write to disk.

8.1.2. Automatic Prescale Change

The above description illustrates two types of action that can be automated, reflected in the roles of the trigger operator and the role of the trigger expert:

Automation of Operation The trigger operator's task of changing prescale keys based on the drop in luminosity can be automated.

Automatic Error Correction The trigger expert's alterations can be automated where the underlying issue is well understood.

The signature in the trigger monitoring of certain detector issues can, too, be understood, detected, and corrected for, allowing for Automatic Error Correction. As shown in the previous chapter the expected and actual behaviour of the trigger system can be well described based on the available monitoring data: the trigger rates of most items scale linearly with luminosity (Figure 7.5), and the variation in the output rate can be understood in terms of the input rate and the applied prescale value, as discussed in Section 7.2.3. This allows for a high degree of Automation of Operation.

Combining the detailed monitoring data from the trigger system with the online luminosity information and information from the LHC about beam states, a framework was constructed that automatically generates and applies prescale sets on the fly. The generation of new prescale sets is based on the observed behaviour and an expressed desired behaviour or bandwidth utilisation. This system constitutes a feed-back loop where the monitoring data, reflecting the experiments behaviour, is used to automatically correct and steer the data-taking to ensure a better correspondence with the data-taking goals. By implementing a robust rule-based layer on top of this feed-back loop, all existing cases of Automation of Operation and Automatic Error Correction can be handled in a robust a reproducible manner.

As most changes can be applied faster and more effectively by a computer than by a trigger operator, the implications are a more dynamic trigger menu, a more efficient bandwidth utilisation, and a more stable operation.

Trigger operators are still needed to monitor the behaviour of the automated system and trigger experts are still needed to cover cases where the automation fails, but such an automation of the trigger system reduces the amount of effort required, and the risk of human error, in the day-to-day operation of ATLAS.

The introduction of new software for automation of operational procedures is not completely without risk. There is the risk of bugs in the software and in the configuration of the software – both of which might lead to unexpected behaviour. Apart from actual bugs, unforeseen conditions can also lead to the wrong automated action being undertaken, and, since it is automated, this might not be noticed immediately. For the safety and stability of the experiment, it is important that changes are rolled out slowly and that procedures are established for overriding and/or reverting automated changes. However, this provides an excellent starting point and learning opportunity where the benefits can be reaped in a relatively isolated context.

8.1.3. Limitation of Scope

The mandate for implementing the trigger automation only extended to automation of changes to the Level 1 prescale sets. Due to this constraint and the tight coupling between the Level 1 prescale and the HLT prescale, the use of the implemented system is currently limited to handling cases that only require changes to the Level 1 prescales. However, as will be discussed later, relatively few adjustments are needed for the current implementation to accommodate automation of both Level 1 and HLT.

8.2. Overview of the Automatic Prescaler

As the automatic feed-back system has a direct impact on data-taking, stability and traceability is a key requirement. Like so, the system needs to handle different types of automations robustly in a variety of operational conditions – for Automatic Error Correction this implies during situations with sub-system failure. The only way to achieve this is with a sound system and software design. This section will give an overview of the automatic prescaler while Section 8.3 and Section 8.4 will cover the details of the two main parts of the system, the Rule Checker and the Orchestrator, respectively.

The automatic prescaler consists of two parts, as illustrated in Figure 8.1:

A Rule Checker: Responsible for evaluating the automation rules against the available monitoring data and requesting any prescale changes.

An Orchestrator: Responsible for processing the prescale changes, generating a new Level 1 prescale set from the requested prescale changes if needed, and request of run control that the new prescale set is applied.

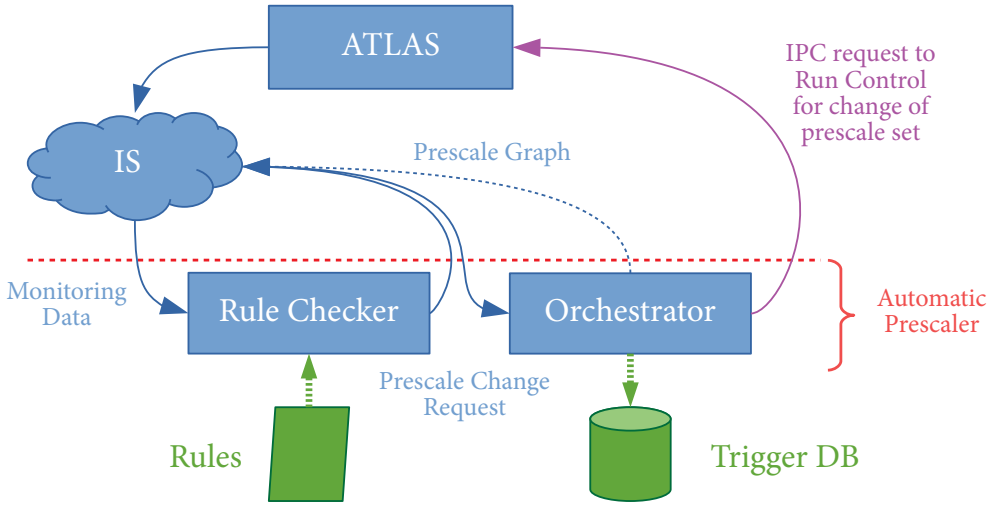


Figure 8.1: The design of the Automatic Prescaler, its two main constituents, the Rule Checker and the Orchestrator, and their interaction.

Together, these two applications constitute a feed-back loop where monitoring data from IS is used to generate and apply new prescale sets.

Throughout data-taking, the CTP and other parts of ATLAS publish monitoring data to the IS. Any online application can subscribe to, and receive updates upon change of any information published to the IS. The Rule Checker and the Orchestrator make extensive use of this mechanism: the Rule Checker subscribes to information relevant for evaluating the automation rules. This typically includes trigger rate (as published by the CTP monitoring clients discussed in Section 6) and the current beam state (as announced by the LHC). It should be stressed that *any* information published to the IS can be used by the rule checker, including luminosity or sub-detector-specific data.

For requesting a change of prescales, the Rule Checker uses an IS publication, too. The Orchestrator subscribes to this publication and is notified upon change. The Orchestrator then determines, a prescale set that complies with the requested changes, and requests of the ATLAS Run Control application that the prescale set is changed appropriately. If no appropriate prescale set can be found, the Orchestrator creates one and writes it to the trigger database, before requesting the change.

8.3. The Rule Checker

The following discussion outlines the operation of the rule checker. The discussion is kept general and it is instructive to refer to Section 8.6 for more concrete

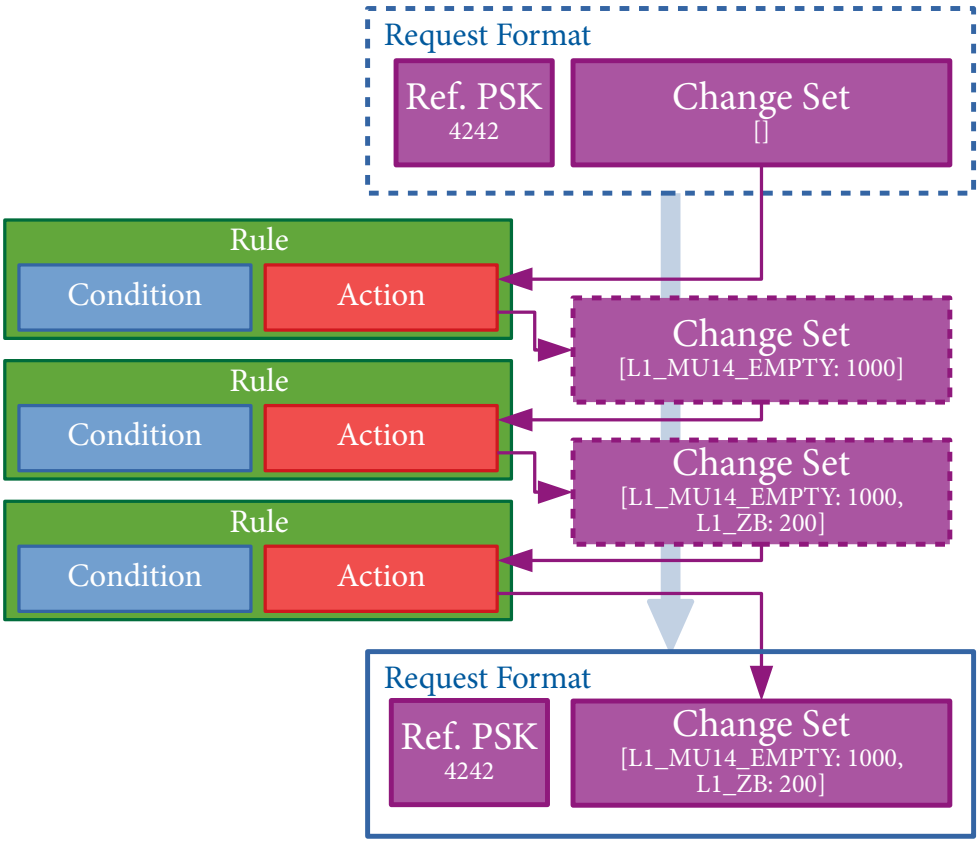


Figure 8.2: Example of Rule execution flow. The Change Set is built by sequential evaluation of Rules, during which the Action associated with a Rule acts on the Change Set. The final Change Set is included in the request for prescale change.

examples of rules, conditions and actions.

8.3.1. From IS-update to Prescale Request

The Rule Checker receives an update whenever an IS publication it subscribes to is updated. When this happens, the Rule Checker will re-evaluate all so-called Conditions that relies on this information. The Conditions can be thought of as simple logical expressions based on IS information. Examples could be "TBP rate of `L1_MU10` is above 100 kHz" or "ATLAS is currently taking data".

After updating all affected conditions, the Rule Checker turns to a list of automation rules. Each Rule ties an Action to a Condition. The purpose of evaluating the rules is to construct the prescale Change Set. Figure 8.2 illustrates the interplay between Rules, Condition, Actions and the Change Set. The Change

Set specifies the prescale changes to the currently applied prescale set required by the automation rules. Each Rule is evaluated in turn and can freely act on the change set.

After all rules are evaluated, the change set is compiled and published to IS if it is not empty.

8.3.2. Rules, Conditions and Actions

A Rule is primarily a matching of an Action to a Condition. This implies that the logic and complexity of the automation rules need to be captured by the Conditions and Actions.

Conditions are by design passive. This choice is made to allow re-use of Conditions between automation rules and thereby minimise resource overhead. The Rule object is used to track the state of the Condition and pass the state information on to the Action. This is done as, most often, it is upon a change in the Conditions value that a change in prescale is required. The Actions are, via the Rules, tied to the Conditions and constitute the active part: based on the state (change) of the Condition, and available IS information, the Action acts on the Change Set.

Each Condition has an associated logical value (its state), the value (True/False) of which is determined by the evaluation logic of the particular Condition. The evaluation logic of a Condition makes use of

1. The information retrieved from IS – allowing conditions like "LHC beam state is Adjust" or "The TAP rate of `L1_TAU8_EMPTY` is $> 100\,\text{kHz}$ (and has been for the last 5 s)" to be formed.
2. The logical value of other conditions, allowing logical operations (such as, but not limited to, ANDs and ORs) to be formed. This is also useful for more advanced rules where the condition of the current rule depend on the condition of other rules (previous or future).

This flexibility is reflected in Figure 8.3: on the left hand side of the figure are two conditions based on the (IS) information of the current beam state and the trigger rate of some item. The logic applied by the conditions could be "LHC beam state is 'STABLE BEAMS'" and "The TBP rate of `L1_EM12` is above 50 kHz". These two conditions can be logically combined using an AND-Condition, say, to create more complex rules. Another example of a logical operator shown in Figure 8.3 is the NOT-Condition.

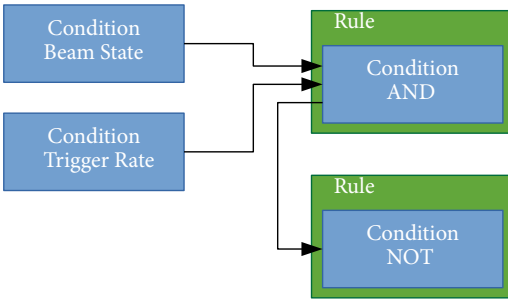


Figure 8.3: Examples of how Conditions can be combined to form new Conditions, and how this can be used to form dependant Rules.

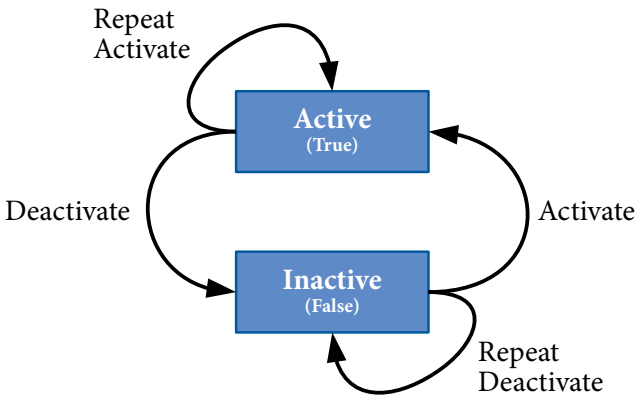


Figure 8.4: Finite state machine maintained by a Rule for the internal state of the associated Condition.

Each Rule implements a Finite State Machine (FSM) based on the current and past truth value of its Condition as illustrated in Figure 8.4. This gives rise to four possible state transitions:

- Activate/Deactivate – when the previous state is different from the new state.
- Repeat-Activate/Repeat-Deactivate – when the previous state is the same as the new state.

In terms of this FSM, the Action object defines the action undertaken during state transition. The Action receives information about the transition from the Rule, allowing the Action to act differently on the Change Set depending on whether, say, the Condition has just become active (”Activate”) or if it is remaining active (”Repeat Activate”).

Besides the Change Set, the Action has access to the following information during rule evaluation:

1. The IS data – to allow an appropriate prescale to be calculated based on the current rate, luminosity, etc.
2. The state of the Condition – to allow Actions to define their own FSM if needed[1].
3. Configuration parameters – to allow thresholds, timeouts, or other configuration parameters to be passed to the Action.

Each Action is allowed to maintain internal state, and thereby preserve values between evaluations. Examples of this could be an internal history of changes used for later reverting requests, a state history of the Condition for implementing a more sophisticated state machine than the one provided by the framework. Another example of state that could be maintained by an Action is where critical values are "learned", such as the average rate over the past 30 s.

8.3.3. Rule Priority and the Change Set

The result of evaluating the automation rules is the Change Set. By design, the execution of rules is sequential and the Action of each Rule is allowed to modify the Change Set freely. This design implies that later rules may override or delete changes requested by previously evaluated rules. It also implies that the priority of each rule is determined by the order of evaluation – later rules have higher priority.

As illustrated in Figure 8.2, the final set of prescale changes, the Change Set after evaluation of all Rules, is published along with the PSK of the currently applied prescale set, as the request to the Orchestrator, if the change set is not empty.

8.4. The Orchestrator

The Orchestrator processes the requests for prescale changes published by the Rule Checker. If the request is found to be valid, the applied prescale set will be changed to a new set that reflects the changes requested. The main reason for separating the functionality of the Orchestrator and the Rule Checker, and to operate with the Change Set (and reference PSK) as intermediary, is to avoid

[1] This is not to be confused with the FSM of the Rule.

a number of conflicts arising in a system with many potentially overlapping automation rules. If each automation rule is able to apply its own prescale changes directly, a large number of prescale sets would be generated and many more luminosity block transitions would be needed, resulting in an unacceptable amount of dead-time and thus data-loss. It is possible to substitute this problem with a race condition, by allowing automation rules to act independently and in parallel. However, this compromises the deterministic behaviour of the automation system. Clearly, neither of these are acceptable solutions. The request-object and the Orchestrator elegantly solves all of these issues in a robust manner that further handles manual intervention.

8.4.1. From Request to Prescale Set

The Orchestrator subscribes to the request publication of the Rule Checker. The publication contains a set of prescale changes and a reference PSK. A flow-chart for the request handling is shown in Figure 8.5.

If the reference PSK of the request does not match the PSK of the currently applied prescale set, the request is ignored after an error message has been issued. If the PSK of the request *does* match the PSK of the currently applied prescale set, the processing of the request continues.

Firstly, the Orchestrators will determine whether to create a new prescale set. This is done using a local cache mechanism, based on an internal graph structure (Section 8.4.2). If a suitable prescale set does not yet exist, the Orchestrator creates one from the reference prescale set, based on the requested changes, and writes it to the trigger database to obtain the new PSK.

After the PSK has been obtained the Orchestrator first updates its internal prescale set graph and then make an IPC request to the ATLAS Run Control application to have the prescales changed.

8.4.2. The Graph of Prescale Sets

The Orchestrator maintains an internal graph of automated prescale changes. The graph primarily serves as a sophisticated cache for the generated prescale sets and offers the ability to determine if the current prescale was applied automatically or by a user, the ability to safely roll back, and the ability to reuse automatic prescale sets.

Each node in the graph is a prescale set (key) and each edge in the graph is the difference between the two prescale sets, as illustrated in Figure 8.6. The graph is reset to a single (root) node if a manual prescale change is detected and the graph is grown as automatic prescale changes are applied.

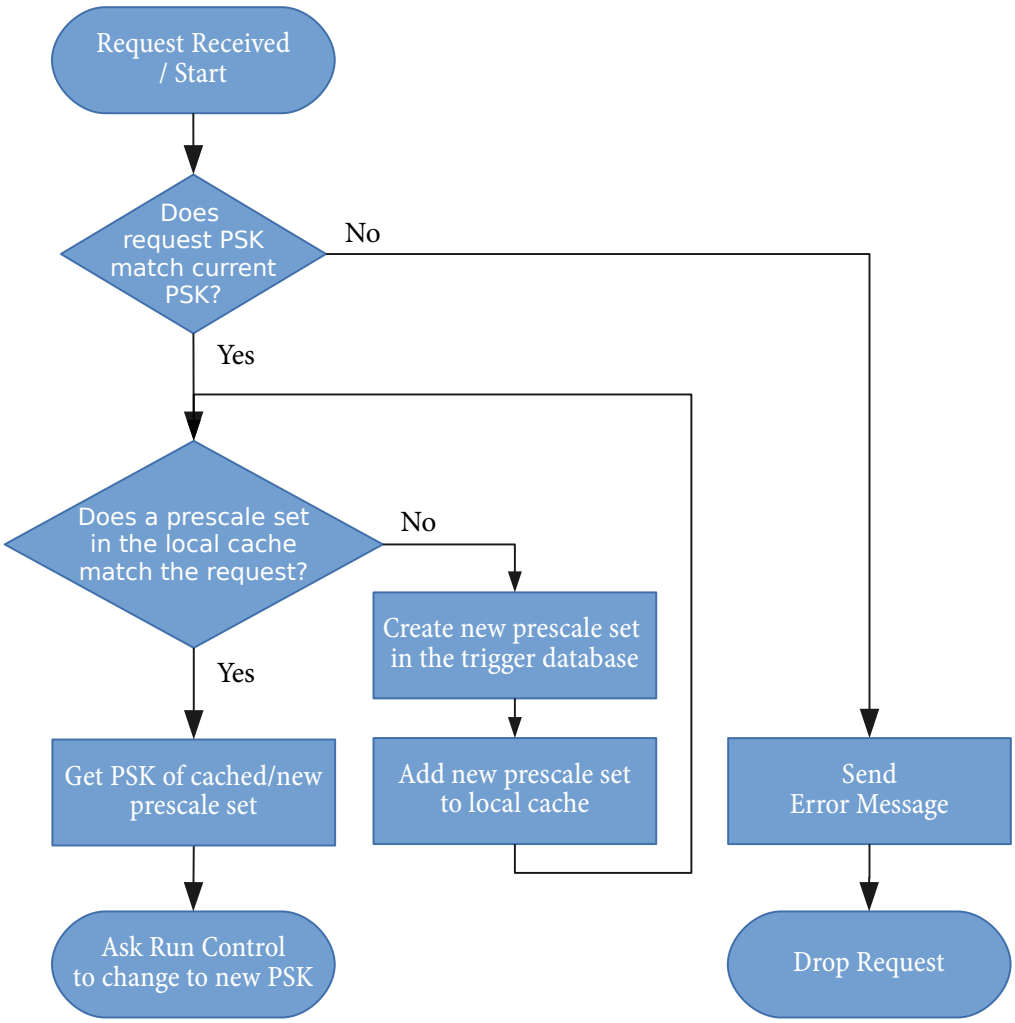


Figure 8.5: Orchestrator flow-chart for handling requests for prescale changes.

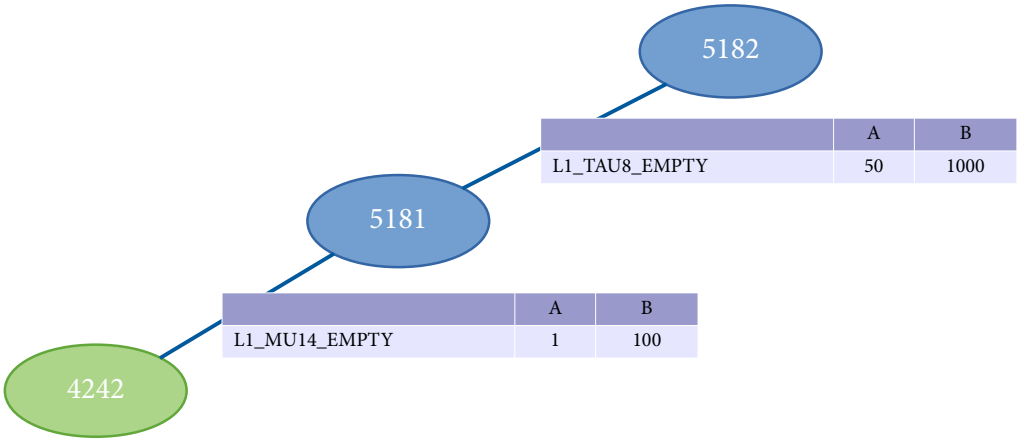


Figure 8.6: Orchestrator internal graph. Each node is a prescale set, represented by the Prescale Set Key (PSK), and the link between two nodes is the prescale difference set, displayed here as tables with A and B representing the two nodes.

A manual prescale change can be detected by checking that no node exists in the graph corresponding to the PSK of the currently applied prescale set. Note that a manual change to an automatically generated PSK derived is still seen as an automated change, if the PSK is already present in the graph. Like so, one can determine if the current PSK is the user-defined key or not: if the PSK corresponds to the root node of the graph, the PSK can be assumed to be the user or target prescale.

Automatic prescale changes can be undone effectively by traversing the graph: each request specifies a set of prescale changes and reference PSK. The PSK is used to identify the starting node and the graph is traversed to find a path matching the desired set of prescale changes.

The nodes of the graph are published to IS upon creation and are thus available to all online applications, including the Rule Checker.

8.5. Manual Intervention

There are two aspects to consider for the discussion of manual intervention: Firstly, as the automatic prescaler needs to operate and co-exist with the manual prescale changes, mechanisms and safe guards are needed to ensure that the automatic prescaler does not counter-act the manual changes made. Secondly, there needs to be a safe and easy way to manually break the feed-back loop and thus prevent the automatic prescaler from applying changes.

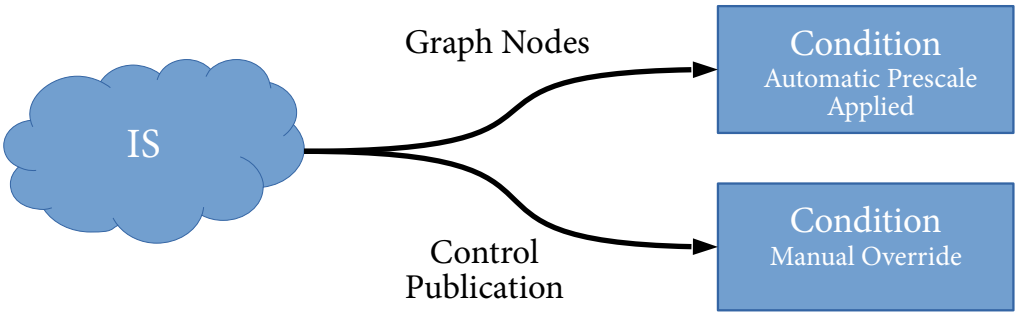


Figure 8.7: Examples of how IS data can be used to detect and react to manual intervention.

The automatic prescaler already contains some safe-guards that prevents it from counteracting the user. The reference PSK provided by the Rule Checker provides a first layer of robustness against manual intervention: should the user change the prescale set, the reference PSK of the request would no longer match that of the applied prescale set and the request would be ignored. Further, IS publication can be leveraged to create special conditions that can be used to steer the rule evaluation, as illustrated in Figure 8.7. As the Orchestrator publishes information about its internal graph structure to IS, Conditions can be created in the Rule Checker, that are based on whether the current prescale set is a manually applied one or one applied by the Orchestrator. Similarly, one can use one or more boolean publication in IS as the source for "veto Conditions" in the Rule Checker. The veto Condition can then be combined with other Conditions (e.g., via an AND Condition) and used to toggle the enabled-ness of a set of Rules.

Having the Rule Checker and the Orchestrator separated allows for a lot of flexibility for breaking the feed-back loop: as the loop is not complete without both parts, an effective way of breaking it is to stop either of the two. The recommended way is to stop the Orchestrator. In this scenario the Rule Checker keeps running and making requests for prescale changes but the request for changes are not picked up. This, too, provides a good way of testing new rules: a extra instance of the Rule Checker, configured with new rules, is started, but instructed to publish requests to a destination within IS where the Orchestrator is not listening. In this way, the behaviour of the new rules can be assessed, by inspecting the un-handled requests.

Figure 8.8: Example of a hot item rule.

8.6. SETTING UP RULES

Currently, 39 rules are implemented and in use in ATLAS. Most of these fall into two categories:

Hot Item Prescaling where detector effects cause periods with a high number of triggers coming from certain trigger items.

Beam State Rules where certain trigger items (typically background, and zero-bias triggers) need to be enabled or disabled depending on the current beam state, e.g., minimum bias items for background studies that only should be enabled in the beginning of each fill.

8.6.1. Hot Item Prescaling

Hot item prescaling is primarily applied to muon and calorimeter triggers during detector issues that causes the trigger rate for the affected items to dramatically increase. One typical example of how a detector issue might increase the trigger rate was illustrated and discussed in Section 7.4.2. Hot item prescaling works by temporarily applying a very large prescale to the affected trigger item(s). When the detector issue is resolved and the rate restored, the original prescale is reverted.

As an example, the Hot Item Prescaling rule for the trigger item `L1_TAU8_EMPTY` will be considered. The rule, illustrated in Figure 8.8, could be formulated as

- The maximum allowed rate after prescale is 500 Hz.
- The time window to consider is 15 s.
- The target rate after prescale is 200 Hz.
- Round any new prescale to the nearest 5.

The first two bullets defines the Condition: if the observed trigger rate after prescale exceeds 500 Hz for a time period of at least 15 s, the rate is too high and the Condition activates.

The Action defines three of the four transitions as illustrated in Figure 8.9, namely:

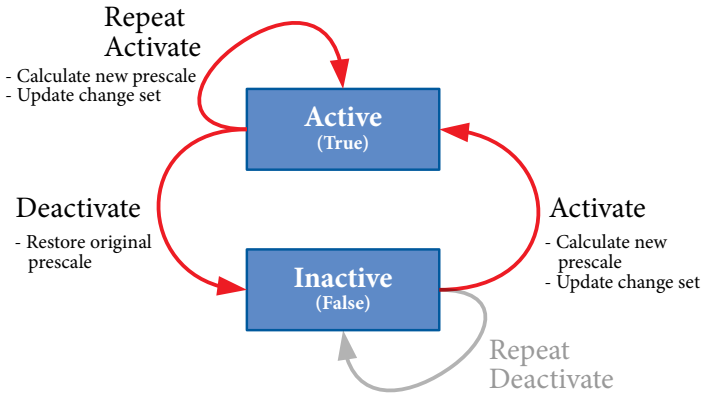


Figure 8.9: State transition actions executed by the Action of a hot item Rule.

Activate – A new prescale is calculated based on the mean TBP rate during the previous 15 s and the desired TAP rate of 200 Hz. The new prescale value (e.g., 1042.424) is rounded – in this case to the nearest 5 (1040). The prescale change is added to the change set.

Deactivate - The original prescale is restored.

Repeat-activate – Despite having already requested a prescale change, the trigger rate is still too high and must be prescaled again. The procedure of Activate is repeated.

Repeat-deactivate has no special meaning for hot item prescaling, and no action is undertaken.

The rounding of the applied prescale can be considered a hysteresis to avoid generating and applying several prescale sets with roughly the same value. If needed in the future, instead of using hard-coded values, the size or order of magnitude of this hysteresis could be estimated dynamically based on the expected TAP rate given the TBP rate and the prescale.

8.6.2. Beam State Rules

The beam state rules represent an example of Automation of Operation. The rules are used to control the trigger rate of certain zero bias and minimum bias triggers during particular LHC beam states – typically "Adjust".

A beam state rule exists for the zero bias item `L1_ZB`. The rule, illustrated in Figure 8.10, can be formulated as:

- The rule should apply only during the beam state "Adjust".

Figure 8.10: Example of a Beam State rule.

- The target rate after prescale is 500 Hz.
- The time window to consider is 5 s.
- Round the new prescale to the nearest 2.

The Condition of this type of rule only relies on the beam state information. The behaviour of the Action is the same as for the hot item prescaler. The implementation of Repeat-activate ensures that the applied prescales, and thus output rate, always matches the target rate, within the desired rounding of the prescale.

8.7. Possible Extensions

The following is a collection of ideas and considerations for extending the functionality of the automatic prescaling or bridging gaps between an automated setup and the current operation of the trigger.

8.7.1. Luminosity based Conditions and Actions

While the beam condition rule from before could form the basis for automatic prescaling of most items, it might be desirable to base the calculation of the desired prescale on the known *expected* behaviour of a trigger item. That is, instead of forcing a particular output rate based on the current input rate it might be beneficial to calculate and use the expected input rate, based on the current luminosity, and use that as the basis for calculating the new prescale. This implies using luminosity information from IS, either to form a Condition for an automation Rule, or better, to use with the Action of the Rule.

The difference between using the current input rate and a calculated expected input rate is important. In most cases, when there is no operational issues, the two will yield the same result. However, using the current input rate for the calculation makes the Rule (strictly the Action) blind to the expected behaviour: no matter what the reason for the high rate is, the Action will choose a prescale that is sure to bring the output rate down to the target level. This is ideal for

automation rules of Automatic Error Correction but not suited for Automation of Operation.

By using a luminosity-based estimate of the input rate to derive an appropriate prescale, potential underlying detector problems would still be visible. For instance, a misconfigured gain on one side of the calorimeter system would cause any $\not{E}_{\mathrm{T}}$ trigger to fire at increased rate. The current setup would quench this increased rate by simply increasing the prescale until the output rate matches the target, while a prescaling based on the expected behaviour would fail to meet the target rate and hint to the underlying problem. This is the approach needed for Automation of Operation.

The luminosity based Actions and Conditions have not yet been implemented as the use of the automatic prescaler for Automation of Operation was outside the initial scope of the application. However, no changes to the design of the automatic prescaler is needed to allow for this functionality to be added in the future.

8.7.2. Rule Checker Watch Dog

The Rule Checker implements a flexible framework for retrieving and calculating quantities from IS publications. The functionality of the Rule Checker could be extended by allowing the Actions to do more than requesting prescale changes. For instance,

- create IS publications of their own; and/or
- use IPC to communicate directly with other programs and services.

This would allow rules to be set up to notify operators or experts of both the trigger system but also of other ATLAS systems. An example could be, to notify trigger or sub-system experts if the observed (TBP) rate deviates sufficiently from the rate expected at the current luminosity. This would allow alarms to be created: if such a Condition was satisfied the Action could notify run control, the trigger expert, and the expert of the affected system e.g., via email or text message.

This functionality was not implemented primarily due to a lack of time and partial overlap in functionality with notification system created by the DAQ community called CHIP [94].

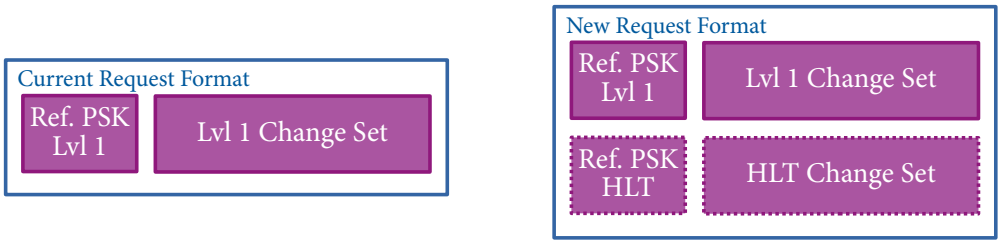


Figure 8.11: Extension of the request format to allow extension of the Automatic Prescaler to the HLT. The extended format additionally includes a HLT Change Set and a HLT reference PSK.

8.7.3. Extension to HLT

Extension to the HLT would allow for more flexible Automatic Error Correction and would open the door for a full automation of the trigger operation. The extension to the HLT was not implemented a) because it was outside the initial scope; b) due to concerns about automation and robustness from the trigger community. The following is an account of the relatively few changes that would be required in order to extend the framework to HLT.

In order to extend the functionality of automatic prescaling to the HLT, the request format should be extended with an HLT reference PSK and an HLT prescale change set, as illustrated by Figure 8.11. This is to encompass the tight coupling between HLT and Level 1 prescales. This extension has almost no implications for the implementation of the Rule Checker, but a few for the Orchestrator.

The Orchestrator would need to be updated to operate with a tuple consisting of the Level 1 and HLT PSKs instead of just the Level 1 PSK:

$$(\mathrm{PSK_{L1}}) \longrightarrow (\mathrm{PSK_{L1}}, \mathrm{PSK_{HLT}})\,. \tag{8.1}$$

This implies small changes to the operational flow, such as discarding request where either the Level 1 PSK OR the HLT PSK do not match those of the currently applied prescale sets, but also slightly larger ones, such as how to manage the internal graph.

The naive approach would be to use a graph of nodes consisting of tuples of $(\mathrm{PSK_{L1}}, \mathrm{PSK_{HLT}})$, and to make the edges be a tuple of the 2×2 prescale difference sets. However, a better approach might be to operate with two graphs, one for each of the two trigger levels, and each similar to the current one. This would ease graph traversal and allow greater flexibility in combining Level 1 and HLT prescale sets.

These are the only changes needed to the framework to allow the automation to be extended to the HLT. Further, as this new request format does not alter the way changes to the Level 1 prescales are handled, the existing automation rules would continue to work with little or no modification needed.

8.7.4. Full Automation of Operation

There are two direct approaches for achieving full Automation of Operation, depending on whether the use of a prescale table, like the one in Table 8.1, is continued or abandoned. Both can be achieved within the framework, but there are subtle complications and drawbacks to consider if such a prescale table is to be used. In either case, luminosity based Conditions would be a prerequisite and so would the extension to the HLT of the automatic prescaler, due to the tight coupling between the Level 1 and HLT prescale.

If the pre-computed prescale table is to be used, one would create a Rule with an Action for changing the prescale set, pending a Condition based on the luminosity – in essence, doing the work of the trigger operator. However, currently the framework does not allow an Action to replace the current prescale set with another prescale set: an Action can only request changes to the current prescale set. While one could implement such functionality for an Action, it complicates the execution flow of the Rule Checker, i.e. it is unclear how the sequential evaluation of rules should be handled if one rule suddenly changes the entire prescale set. Having Actions capable of actually changing the applied prescale set also breaks the separation between the Rule Checker and the Orchestrator, and one would have to be careful to avoid scenarios where the two get in the way of each other. Even if the the practical issues are resolved and automation rules exist to change the entire prescale set based on luminosity, the automation will suffer many of the same limitations as the current setup, as the prescale sets are pre-computed, in particular:

Limited granularity: The use of a pre-computed prescale table implies that prescale changes will happen in steps instead of being adjusted gradually. Admittedly, one might be able to generate a table much longer than the one shown in Table 8.1, but the solution is not elegant.

Blind to online conditions: By using pre-calculated prescale sets it is impossible to factor in online conditions or learned behaviour. This will lead to a setup where any error-correcting Rule will be fighting against the automations that changes the entire prescale set: as the error-correction is not included in the pre-calculated prescale set, the automation rule will

need to detect the change, recalculate the correction, and apply for the new correction to be applied. This is inefficient and introduces unnecessary overhead. This problem grows if the step size in the pre-calculated table is made small.

Alternatively, the pre-calculated table of prescale sets could be abandoned for normal operations[2]. This would imply a transition to a declarative trigger system, where instead of specifying the exact prescale, the desired behaviour or bandwidth allocation is specified. This information already exist and is formalised, both in writing, but also in the configuration of the program that is used to generate the prescale sets in the tables. It is thus a matter of porting the configuration from the old system to the new. Such a declarative system would be easier to maintain (it is generally easier to formulate what you want rather than how to achieve it), completely dynamic, in the sense of not relying on pre-computed configurations, and would allow the bandwidth allocating and policy bearing rules to co-exist with the error-correcting ones without introducing overhead.

[2]It might be useful to have in a transition period, if for a while it is necessary to revert to manual prescale changes.

Chapter 9

Conclusion and Future Perspectives

Summation

This thesis covered the motivation for the ATLAS trigger system with a particular emphasis on the upgrade of the Central Trigger Processor (CTP) and its monitoring for Run II. The ATLAS trigger system is a two layer trigger system consisting of a hardware-based Level 1 trigger and a software-based High Level Trigger (HLT). A baseline trigger strategy of selecting charged leptons with high p_{T}, energetic photons and jets, as well as events with an observed energy imbalance ($\not{E}_{\mathrm{T}}$), is used to reduce the event rate from 40 MHz to 100 kHz at Level 1 and 100 kHz to 1 kHz at the HLT while achieving efficient rejection against the QCD background. The CTP is responsible for making the final trigger decision of the Level 1 trigger, and distribute the decision to the sub-detectors as part of the Trigger, Timing, and Control (TTC) distribution. The upgrade of the CTP was motivated by a desire of increasing the number of trigger items at Level 1, to support more sub-detectors as well as several simultaneous users of the CTP's resources, and to improve the monitoring capabilities of the CTP.

The design and implementation of a new monitoring infrastructure, for the upgraded CTP has been covered: the new infrastructure is a microservice architecture with a server application reading out values from the CTP hardware and other smaller services for processing or displaying the monitoring data. The CTP is monitored at all times, to allow detailed post-mortem analysis of any operational issue. During data taking, the monitoring data is used to provide the operators in the ATLAS Control Room with information about bandwidth utilisation as well as general health information about the data taking effort. Further, some of the monitoring data provided by the CTP is critical for the later utility of the recorded data: in order to normalise the recorded data set

to the delivered data set, prescale information as well as dead-time correction factors are needed.

Using monitoring data from the improved per-bunch monitoring capabilities of the upgraded CTP two cross checks of the calculation of the dead-time correction factors were carried out. The observed relative differences between the cross-checks and the ATLAS recommended (simplified) method for calculating the dead-time correction factor, are in both cases $\mathcal{O}(0.1\%)$. However the uncertainty on both estimates are too large to conclude if there is a systematic effect or not. A large part of the uncertainty can be ascribed to the per-bunch monitoring routines (deliberately) not being synchronised to the luminosity block transitions. That the observed relative differences are small and consistent with zero, and much smaller than the relative uncertainty on the luminosity ($\sim 2\%$), gives confidence that the simplification in the recommended method is unlikely to have had an impact on any analysis results.

Using the intra-luminosity block monitoring data describing the trigger rates, which is not normally available after data-taking, the stability of the experimental conditions, quantified by the stability of the luminosity, was determined. A relative drop in the trigger rate (and luminosity) of -0.0010 ± 0.0001 and -0.0012 ± 0.0001 was observed for a muon trigger and a jet trigger, respectively, consistent with the observed rate at which the luminosity drops over time. A small fraction (3.2%) of the analysed luminosity blocks exhibited behaviour that can not be described as statistical fluctuations. As the behaviour is seen for orthogonal triggers, it is likely due to changes to the beam parameters made by the LHC.

The effects of a magnet dump or a noisy calorimeter tower on the trigger system was also shown. Operational issues commonly have particular signatures in the trigger system or introduce notable experimental dead-time, that can then be easily detected in the CTP monitoring data.

Lastly a rule-based framework was described for automation of the Level 1 trigger. The framework relies heavily on monitoring data from all of ATLAS and the LHC – in particular on the monitoring data from the CTP. The framework consist of two parts: a Rule Checker for evaluating the automation rules and requesting prescales changes and an Orchestrator for processing such requests. As of now, 39 different rules are implemented for two general categories of automated prescaling: ”hot item” prescaling, where a trigger item is prescaled when the trigger rate exceeds a configurable threshold, and ”beam state” prescaling, where trigger items are prescaled (or enabled) based on the current beam state. The framework is extremely flexible and could be extended beyond these two

specialised cases. The modification needed in order to generalise the framework to the HLT as well as other possible extensions were also discussed.

Looking Forward

With the high luminosity upgrade of the LHC (HL-LHC), the experimental conditions will only get more challenging as the luminosity is expected to increase to $\mathcal{L} = 7.5 \times 10^{34}\,\mathrm{cm}^{-1}\mathrm{s}^{-1}$ in 2026, corresponding to an in-time pileup of roughly $\langle\mu\rangle = 200$ [95]. The aim is to provide up to $4000\,\mathrm{fb}^{-1}$ of additional data to the experiments. Meeting these harsher conditions poses a challenge well reflected in the number of upgrades and replacements scheduled for the TDAQ systems and for the ATLAS sub-detectors. The current trigger system will, with few modifications, be operational until 2023, where a major upgrade of the entire TDAQ system is planned [95]. The latency of the hardware trigger will be increased from 100 bunch crossings (BCs) to ~ 2400 BCs and the accept rate will be increased from 100 kHz to 1 MHz. The upgrades to the hardware trigger are substantial and are expected to provide enough background rejection that the current trigger thresholds can be retained, despite the higher overall interaction rate.

The upgrade of the hardware trigger includes a complete redesign of the existing trigger processors and the introduction of a Global Event Processor, for improved calorimeter clustering algorithms replacing the L1Topo for event topological calculations. The upgraded CTP-system will likely grow from 512 to 1024 inputs and items and new FPGA technology will likely allow for much more monitoring information with more granularity in channels and time (i.e. more per-bunch monitoring).

Due to difficulties with projecting the trigger rate coming from primarily hadronic objects at $\langle\mu\rangle \approx 200$ and uncertainties regarding the occupancy of the planned replacement for ATLAS tracker, the ITk, the technical design report [95] foresees the potential necessity of splitting the hardware trigger in two (Level 0 and Level 1) each with their own CTP.

Such a split would dramatically increase the complexity of the hardware trigger, especially for the configuration with 2 CTP-like systems: having a two layer model implies two trigger decisions, two trigger menus, and with it, two set of prescales for the individual items, as well as the necessity for rethinking preventive dead-time. Even without the split of the hardware trigger into two, the upgraded CTP will be a more complex system than the existing CTP.

Operating a more complex trigger system implies the need for additional monitoring on all fronts: for data integrity, operational monitoring in the control room, and for post-mortem and timing purposes. In order to achieve efficient

monitoring in a complex system with many autonomous parts, it is paramount that a distributed software architecture, like the one adopted here, is fully embraced: relying on a monolithic architecture is no longer a viable option. While an automation of the operation of the upgraded hardware system could possibly be avoided, it is my belief that a rule-based system like the one designed for the current trigger system, relying on the available information about the state of the experiment, will increasingly outperform a human operator as the complexity of the trigger systems and as its operation increases. Any such system should, of course, be tested and allow for manual override, as is true for any automation.

Thanks to a robust design, the CTP has operated stably with no mentionable issues for the duration of Run II. The experience gained, in terms of monitoring, automation, and micro service architecture, should be carried forward for the upgraded system, that will likely operate under even more challenging conditions.

Appendix A

Appendix

A.1. Busy Path of Secondary Partitions

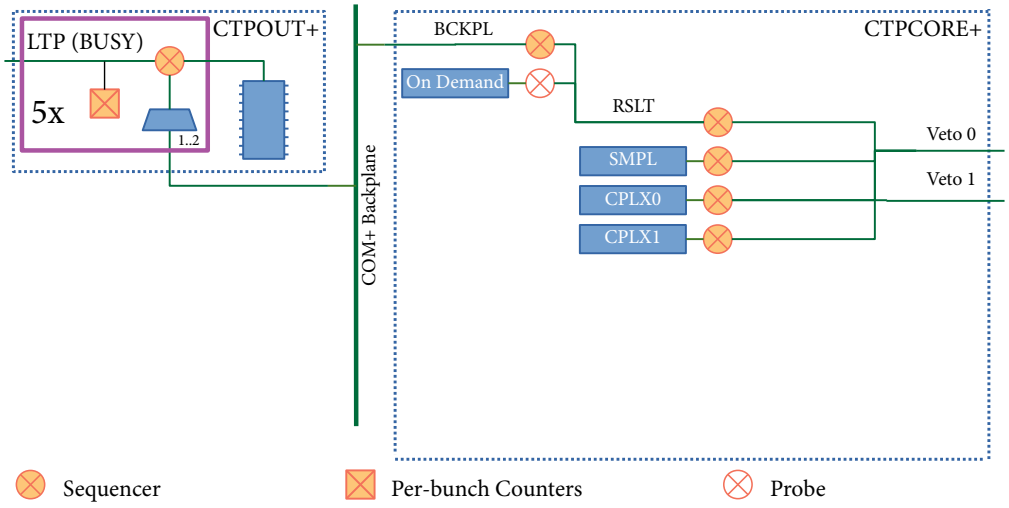


Figure A.1.1: Diagram of the BUSY flow in the CTP for secondary partitions. All intersections correspond to an OR of the incoming signals and all forks corresponds to fan-outs. The CTP receives BUSY from the sub-detectors at the CTPOUT, and routes these to the COM backplane. The CTPCORE combines the backplane BUSY with the internal busy from the CTP and the signals from the preventive dead-time to produce the final veto signals.

A.2. Detailed Breakdown of Monitoring Routines

A.2.1. Status Monitoring

Table 1.1: Per board summary of Status Monitoring tasks.

Status Monitoring

	Description	Multiplicity	Unit Size	Total Size
CTPMI	Input Signal Selection	6	2 B	12 B
	SYN Generation	1	2 B	2 B
	ECR Generation	1	100 B	100 B
	HPTDC Setup	1	84 B	84 B
	HPTDC Control	1	8 B	8 B
	HPTDC Status	1	16 B	16 B
	HPTDC Data	4	13 kB	51 kB
	Temperature Monitoring Configuration	10	14 B	140 B
	Temperature Monitoring Readings	10	4 B	40 B
	Board multiplicity and total	1	52 kB	52 kB
CTPIN	Channel Delays	5	10 B	50 B
	DELAY 25 Config	1	32 B	32 B
	Signal Enabled	4	32 B	128 B
	Pipeline lengths	4	32 B	128 B
	Input Clock Edge	4	32 B	128 B
	Parity Error Configuration	4	4 B	16 B
	MBTS Control Word	4	2 B	8 B
	Pipeline Stretch	4	2 B	8 B
	Memory Configuration	4	4 B	16 B
	Monitoring Counter Configuration	4	4 B	16 B
	Monitoring Counter Selection	4	31 B	124 B
	Monitoring BCID Mask	4	16 kB	66 kB
	Monitoring BCID Offset	4	2 B	8 B
	Board multiplicity and total	3	66 kB	199 kB
	PIT Clock Edge	160	1 B	160 B
	Power Supply Monitoring	1	4 B	4 B

Table continues on next page.

Status Monitoring *(continued)*

	Description	Multiplicity	Unit Size	Total Size
CTPMON	Input Selection	1	2 B	2 B
	Window Size	1	2 B	2 B
	BCID Offset	1	2 B	2 B
	FIFO Config	1	5 B	5 B
	Configuration	1	4 B	4 B
	Readout Status	1	4 B	4 B
	Board multiplicity and total	1	183 B	183 B
CTPCORE	CDC Status	1	44 B	44 B
	CDC Output Phase	1	4 B	4 B
	Link Status	2	8 B	16 B
	Direct Input Clock Edge	3	96 B	288 B
	Direct Input Phase Alignment	3	32 B	96 B
	Direct Input Fine Delay	3	32 B	96 B
	Direct Input Signal Enabled	3	64 B	192 B
	Pipeline Length	3	7 B	21 B
	TIP Selection	2	1 B	2 B
	BGRP Definition	1	3 kB	3 kB
	Item BGRP Mask	512	2 B	1 kB
	Item Prescale	512	2 B	1 kB
	Random Generators	4	4 B	16 B
	Item Counter Mask	4	512 B	2 kB
	Item Counter Config	4	4 B	16 B
	Readout Pipeline Lengths	4	1 B	4 B
	Readout Window Size	4	1 B	4 B
	Readout Derandomizer Config	6	4 B	24 B
	Readout Derandomizer Status	6	5 B	30 B
	Readout Busy Config	1	2 B	2 B
	Readout TAV Enabled	512	1 B	512 B
	Readout Configurable Header	1	14 B	14 B
	Readout Turn Counter BCID Offset	1	2 B	2 B
	Event Formatter Configuration	1	4 B	4 B
	Event Formatter Busy Config	1	2 B	2 B
	Readout FIFO Configuration	3	4 B	12 B

Table continues on next page.

Status Monitoring *(continued)*

	Description	Multiplicity	Unit Size	Total Size
	Readout FIFO Status	3	4 B	12 B
	Event Monitoring Configuration	1	5 B	5 B
	Event Monitoring Statistics	1	8 B	8 B
	GPS Receiver Status	1	3 B	3 B
	GPS Calibration	1	4 B	4 B
	Busy Mask, Primary Partition	1	2 B	2 B
	Busy Mask, Secondary Partition	2	3 B	6 B
	Dead-time Monitoring Configuration	3	2 B	6 B
	Dead-time Monitoring FIFO Configuration	3	2 B	6 B
	Dead-time Monitoring FIFO Status	3	2 B	6 B
	Dead-time Mask	3	6 B	18 B
	Dead-time Enabled	3	8 B	24 B
	Dead-time Per Bunch BCID Offset	1	2 B	2 B
	Dead-time Per Bunch Status	1	1 B	1 B
	ECR Generation Configuration	2	17 B	34 B
	Temperature	10	4 B	40 B
	Board multiplicity and total	1	8 kB	8 kB
CTPOUT	Clock Source	1	1 B	1 B
	Clock Delay	4	1 B	4 B
	Signal Routing	5	4 B	20 B
	Pattern Generator Config	1	8 B	8 B
	Memory Config	1	8 B	8 B
	Dead-time Per Bunch BCID Offset	1	2 B	2 B
	Dead-time Per Bunch Status	1	1 B	1 B
	Board multiplicity and total	5	44 B	220 B
CTPCAL	Turn Selection	20	1 B	20 B
	Turn Number	1	1 B	1 B
	BCID Offset	1	2 B	2 B
	Sync Selection	4	4 B	16 B
	BCID Mask	256	2 B	512 B
	Board multiplicity and total	1	551 B	551 B

Table continues on next page.

Status Monitoring *(continued)*

Description	Multiplicity	Unit Size	Total Size
Total across boards considering board multiplicity			259 kB

A.2.2. Busy Monitoring

Table 1.2: Per board summary of Busy Monitoring tasks.

Busy Monitoring				
	Description	**Multiplicity**	**Unit Size**	**Total Size**
CTPMI	Busy-at-power-on Status	1	1 B	1 B
	Busy-on-demand Status	1	1 B	1 B
	Sequencer Control	5	5 B	25 B
	Sequencer FIFO Status	5	2 B	10 B
	Sequencer Data[1]	4	20 B	80 B
	Board multiplicity and total	1	117 B	117 B
CTPCORE	Busy-on-demand Status	1	1 B	1 B
	Sequencer Selection	1	2 B	2 B
	Sequencer Control	1	5 B	5 B
	Sequencer FIFO Status	1	2 B	2 B
	Sequencer Data[2]	4	36 B	144 B
	Board multiplicity and total	1	154 B	154 B
CTPOUT	Sequencer Selection	1	2 B	2 B
	Sequencer Control	1	2 B	2 B
	Sequencer FIFO Status	1	2 B	2 B
	Sequencer Data[3]	4	24 B	96 B
	Board multiplicity and total	5	102 B	510 B
Total across boards considering board multiplicity				781 B

[1] Assuming 4 samples in FIFO/readout.
[2] Assuming 4 samples in FIFO/readout.
[3] Assuming 4 samples in FIFO/readout.

A.2.3. Per-Bunch Dead-time Monitoring

Table 1.3: Per board summary of Dead-time per Bunch Monitoring tasks.

Dead-time per Bunch Monitoring

	Description	Multiplicity	Unit Size	Total Size
CTPCORE	Dead-time Per Bunch Mask	1	1 B	1 B
	Dead-time Per Bunch Data	7	14 kB	100 kB
	Turn Counter	1	4 B	4 B
	Board multiplicity and total	1	100 kB	100 kB
CTPOUT	Dead-time/Bunch Data	5	14 kB	71 kB
	Turn Counter	1	4 B	4 B
	Board multiplicity and total	5	71 kB	356 kB
Total across boards considering board multiplicity				456 kB

A.2.4. Phase Monitoring

Table 1.4: Per board summary of Phase Monitoring tasks.

Phase Monitoring

	Description	Multiplicity	Unit Size	Total Size
CTPIN	HPTDC Setup	4	84 B	336 B
	HPTDC Control	4	8 B	32 B
	HPTDC Status	4	16 B	64 B
	HPTDC Data	400	128 B	51 kB
	Board multiplicity and total	3	52 kB	155 kB
Total across boards considering board multiplicity				155 kB

A.2.5. Rate Monitoring

Table 1.5: Per board summary of Rate Monitoring tasks.

Rate Monitoring				
	Description	**Multiplicity**	**Unit Size**	**Total Size**
CTPIN	Input Rates	4	256 B	1 kB
	Turn Counter	4	4 B	16 B
	Board multiplicity and total	3	1 kB	3 kB
CTPCORE	LUT-Out Counts	1	2 kB	2 kB
	Item Counts, TBP	4	2 kB	8 kB
	Item Counts, TAV	4	2 kB	8 kB
	Item Counts, TBP	4	2 kB	8 kB
	Turn Counters	4	4 B	16 B
	Board multiplicity and total	1	27 kB	27 kB
Total across boards considering board multiplicity				30 kB

A.2.6. Per-bunch Item Monitoring

Table 1.6: Per board summary of Item per Bunch Monitoring tasks.

Item per Bunch Monitoring

	Description	Multiplicity	Unit Size	Total Size
CTPMON	Per Bunch Data	40	14 kB	570 kB
	Turn Counter	4	4 B	16 B
	Board multiplicity and total	1	570 kB	570 kB
CTPCORE	Per Bunch Data	192	14 kB	3 MB
	Turn Counter	3	4 B	12 B
	Board multiplicity and total	1	3 MB	3 MB
Total across boards considering board multiplicity				3 MB

A.3. Distribution of Differences

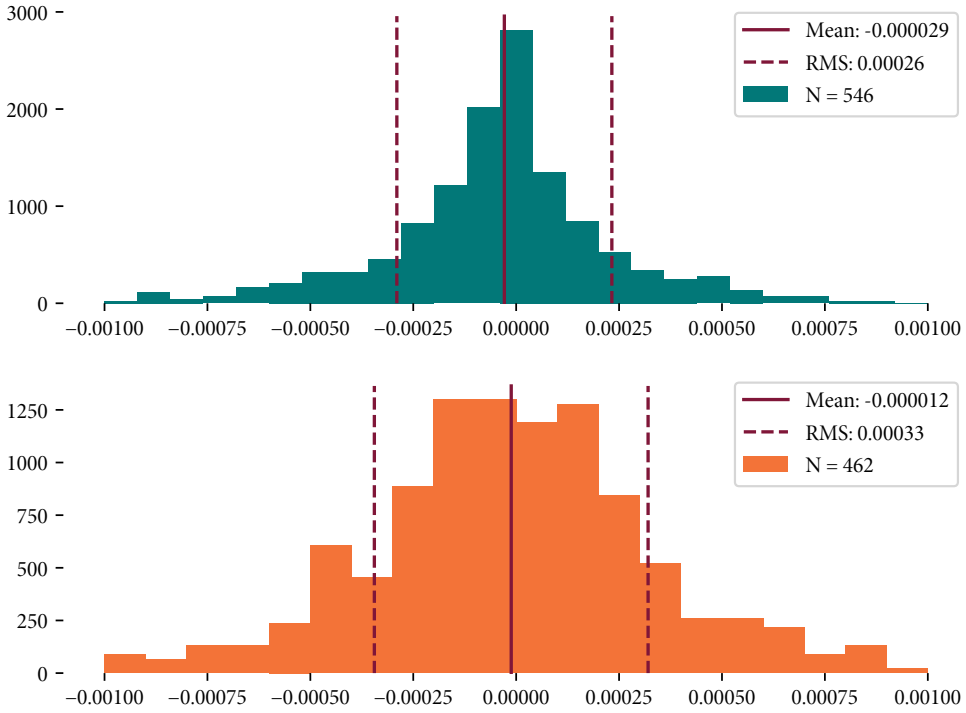


Figure A.3.2: Distribution of differences between the two cross-checks and the reference method for calculating the dead-time correction factor.

Bibliography

[1] G. Galster. "ATLAS Trigger: Preparations for Run II". The Niels Bohr Institute, 2015.

[2] Frederick "Bordry and LHC Research board". *"LHC long term schedule"*. [Online; accessed 29-November-2018]. URL: https://lhc-commissioning.web.cern.ch/lhc-commissioning/schedule/LHC-long-term.htm.

[3] John Ellis. "Outstanding questions: physics beyond the Standard Model". In: *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 370.1961 (2012), pp. 818–830. ISSN: 1364-503X. DOI: 10.1098/rsta.2011.0452. eprint: http://rsta.royalsocietypublishing.org/content/370/1961/818.full.pdf. URL: http://rsta.royalsocietypublishing.org/content/370/1961/818.

[4] Joseph D. Lykken. "Beyond the Standard Model". In: *CERN Yellow Report CERN-2010-002, 101-109*. 2010. arXiv: 1005.1676 [hep-ph]. URL: http://lss.fnal.gov/cgi-bin/find_paper.pl?conf-10-103.

[5] Morad Aaboud et al. "Observation of Higgs boson production in association with a top quark pair at the LHC with the ATLAS detector". In: (2018). arXiv: 1806.00425 [hep-ex].

[6] Pierluigi Campana, Markus Klute, and Pippa Wells. "Physics Goals and Experimental Challenges of the Proton–Proton High-Luminosity Operation of the LHC". In: *Ann. Rev. Nucl. Part. Sci.* 66 (2016), pp. 273–295. DOI: 10.1146/annurev-nucl-102115-044812. arXiv: 1603.09549 [hep-ex].

[7] Stefano Di Vita et al. "A global view on the Higgs self-coupling". In: *JHEP* 09 (2017), p. 069. DOI: 10.1007/JHEP09(2017)069. arXiv: 1704.01953 [hep-ph].

[8] Alexander L. Kagan et al. "Exclusive Window onto Higgs Yukawa Couplings". In: *Phys. Rev. Lett.* 114.10 (2015), p. 101802. DOI: 10.1103/PhysRevLett.114.101802. arXiv: 1406.1722 [hep-ph].

[9] M. Tanabashi et al. "Review of Particle Physics". In: *Phys. Rev.* D98.3 (2018), p. 030001. DOI: 10.1103/PhysRevD.98.030001.

[10] James Stirling. *W.J. Stirling, private communication*. URL: http://www.hep.ph.ic.ac.uk/~wstirlin/plots/plots.html.

[11] G. Aad et al. "Measurements of the electron and muon inclusive cross-sections in proton–proton collisions at s=7 TeV with the ATLAS detector". In: *Physics Letters B* 707.5 (2012), pp. 438–458. ISSN: 0370-2693. DOI: https://doi.org/10.1016/j.physletb.2011.12.054. URL: http://www.sciencedirect.com/science/article/pii/S037026931101522X.

[12] A. Denner et al. "Standard Model Higgs-Boson Branching Ratios with Uncertainties". In: *Eur. Phys. J.* C71 (2011), p. 1753. DOI: 10.1140/epjc/s10052-011-1753-8. arXiv: 1107.5909 [hep-ph].

[13] *Combined measurements of Higgs boson production and decay using up to 80 fb^{-1} of proton–proton collision data at $\sqrt{s}$ = 13 TeV collected with the ATLAS experiment*. Tech. rep. ATLAS-CONF-2018-031. Geneva: CERN, July 2018. URL: https://cds.cern.ch/record/2629412.

[14] Lyndon Evans. "The Large Hadron Collider". In: *New Journal of Physics* 9.9 (2007), p. 335. URL: http://stacks.iop.org/1367-2630/9/i=9/a=335.

[15] Oliver Sim Brüning et al. *LHC Design Report*. CERN Yellow Reports: Monographs. Geneva: CERN, 2004. URL: https://cds.cern.ch/record/782076.

[16] Esma Mobs. "The CERN accelerator complex. Complexe des accélérateurs du CERN". In: (July 2016). General Photo. URL: https://cds.cern.ch/record/2197559.

[17] R Bailey and Paul Collier. *Standard Filling Schemes for Various LHC Operation Modes*. Tech. rep. LHC-PROJECT-NOTE-323. Geneva: CERN, Sept. 2003. URL: https://cds.cern.ch/record/691782.

[18] Lyndon Evans and Philip Bryant. "LHC Machine". In: *Journal of Instrumentation* 3.08 (2008), S08001. URL: http://stacks.iop.org/1748-0221/3/i=08/a=S08001.

[19] ATLAS Collaboration ; Contact Eric Torrence. *ATLAS Data Summary 2018*. Nov. 13, 2018. URL: https://atlas.web.cern.ch/Atlas/GROUPS/DATAPREPARATION/DataSummary/2018/.

[20] *Expected pileup values at the HL-LHC*. Tech. rep. ATL-UPGRADE-PUB-2013-014. Geneva: CERN, Sept. 2013. URL: https://cds.cern.ch/record/1604492.

[21] The ATLAS Collaboration et al. "The ATLAS Experiment at the CERN Large Hadron Collider". In: *Journal of Instrumentation* 3.08 (2008), S08003. URL: http://stacks.iop.org/1748-0221/3/i=08/a=S08003.

[22] Georges Aad et al. *Technical Design Report for the Phase-I Upgrade of the ATLAS TDAQ System*. Tech. rep. CERN-LHCC-2013-018. ATLAS-TDR-023. Final version presented to December 2013 LHCC. Sept. 2013. URL: https://cds.cern.ch/record/1602235.

[23] J.J. Goodson. "Search for Supersymmetry in States with Large Missing Transverse Momentum and Three Leptons including a Z-Boson". Presented 17 Apr 2012. PhD thesis. Stony Brook University, May 2012.

[24] G. Aad et al. *Expected performance of the ATLAS experiment: detector, trigger and physics*. Geneva: CERN, 2009. URL: https://cds.cern.ch/record/1125884.

[25] Alessandro La Rosa. "The ATLAS Insertable B-Layer: from construction to operation". In: *JINST* 11.12 (2016), p. C12036. DOI: 10.1088/1748-0221/11/12/C12036. arXiv: 1610.01994 [physics.ins-det].

[26] M. Tanabashi et al. "Review of Particle Physics". In: *Phys. Rev. D* 98 (3 Aug. 2018), p. 030001. DOI: 10.1103/PhysRevD.98.030001. URL: https://link.aps.org/doi/10.1103/PhysRevD.98.030001.

[27] J R Pater. "The ATLAS SemiConductor Tracker operation and performance". In: *Journal of Instrumentation* 7.04 (2012), p. C04001. URL: http://stacks.iop.org/1748-0221/7/i=04/a=C04001.

[28] E. Hines. "Performance of Particle Identification with the ATLAS Transition Radiation Tracker". In: *Particles and fields. Proceedings, Meeting of the Division of the American Physical Society, DPF 2011, Providence, USA, August 9-13, 2011*. 2011. arXiv: 1109.5925 [physics.ins-det].

[29] Bartosz Mindur. *ATLAS Transition Radiation Tracker (TRT): Straw tubes for tracking and particle identification at the Large Hadron Collider*. Tech. rep. ATL-INDET-PROC-2016-001. Geneva: CERN, Mar. 2016. URL: https://cds.cern.ch/record/2139567.

[30] Peter Hansen. private communication. 2018.

[31] Nikiforos 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. DOI: 10.1109/ANIMMA.2013.6728060. arXiv: 1306.6756 [physics.ins-det]. URL: https://inspirehep.net/record/1240499/files/arXiv:1306.6756.pdf.

[32] Martin Wessels. "Calibration and Performance of the ATLAS Level-1 Calorimeter Trigger with LHC Collision Data". In: *Physics Procedia* 37 (2012). Proceedings of the 2nd International Conference on Technology and Instrumentation in Particle Physics (TIPP 2011), pp. 1841–1848. ISSN: 1875-3892. DOI: https://doi.org/10.1016/j.phpro.2012.02.505. URL: http://www.sciencedirect.com/science/article/pii/S1875389212018962.

[33] Giordano Cattani and the RPC group. "The Resistive Plate Chambers of the ATLAS experiment: performance studies". In: *Journal of Physics: Conference Series* 280.1 (2011), p. 012001. URL: http://stacks.iop.org/1742-6596/280/i=1/a=012001.

[34] Bernd Stelzer. "The New Small Wheel Upgrade Project of the ATLAS Experiment". In: *Nuclear and Particle Physics Proceedings* 273-275 (2016). 37th International Conference on High Energy Physics (ICHEP), pp. 1160–1165. ISSN: 2405-6014. DOI: https://doi.org/10.1016/j.nuclphysbps.2015.09.182. URL: http://www.sciencedirect.com/science/article/pii/S2405601415006719.

[35] C Ohm and T Pauly. "The ATLAS beam pick-up based timing system". In: *Nucl. Instrum. Methods Phys. Res., A* 623.arXiv:0905.3648. ATL-DAQ-PROC-2009-005. ATL-COM-DAQ-2009-022 (May 2009). Comments: 3 pages. Submitted to NIM A for the proceedings of TIPP09 (Tsukuba, Japan), 558–560. 3 p. URL: https://cds.cern.ch/record/1172170.

[36] Thilo Pauly. private communication. 2018.

[37] D Dobos and H Pernegger. *Diamond Pixel Modules and the ATLAS Beam Conditions Monitor*. Tech. rep. ATL-INDET-PROC-2010-007. Geneva: CERN, Mar. 2010. URL: https://cds.cern.ch/record/1248212.

[38] V Cindro et al. "The ATLAS Beam Conditions Monitor". In: *Journal of Instrumentation* 3.02 (2008), P02004. URL: http://stacks.iop.org/1748-0221/3/i=02/a=P02004.

[39] Georges Aad et al. "Measurement of the total cross section from elastic scattering in pp collisions at $\sqrt{s} = 7$ TeV with the ATLAS detector". In: *Nucl. Phys.* B889 (2014), pp. 486–548. DOI: 10.1016/j.nuclphysb.2014.10.019. arXiv: 1408.5778 [hep-ex].

[40] S. Abdel Khalek et al. "The ALFA Roman Pot detectors of ATLAS". In: *Journal of Instrumentation* 11.11 (2016), P11013. URL: http://stacks.iop.org/1748-0221/11/i=11/a=P11013.

[41] ATLAS. *ATLAS Experiment Public Results*. URL: https://twiki.cern.ch/twiki/bin/view/AtlasPublic.

[42] Mikael Martensson. *Fast pattern recognition with the ATLAS L1Track trigger for HL-LHC*. Tech. rep. ATL-DAQ-PROC-2016-034. Geneva: CERN, Nov. 2016. URL: https://cds.cern.ch/record/2234837.

[43] *Trigger Menu in 2016*. Tech. rep. ATL-DAQ-PUB-2017-001. Geneva: CERN, Jan. 2017. URL: https://cds.cern.ch/record/2242069.

[44] Benjamin Nachman. "Investigating the Quantum Properties of Jets and the Search for a Supersymmetric Top Quark Partner with the ATLAS Detector". PhD thesis. Stanford U., Phys. Dept., 2016. arXiv: 1609.03242 [hep-ex].

[45] ATLAS. *ATLAS Experiment Public Results - Missing Energy Trigger Public Results*. URL: https://twiki.cern.ch/twiki/bin/view/AtlasPublic/MissingEtTriggerPublicResults.

[46] S Ask et al. "The ATLAS central level-1 trigger logic and TTC system". In: *Journal of Instrumentation* 3.08 (2008), P08002. URL: http://stacks.iop.org/1748-0221/3/i=08/a=P08002.

[47] R. Achenbach et al. "The ATLAS level-1 calorimeter trigger". In: *JINST* 3 (2008), P03001. DOI: 10.1088/1748-0221/3/03/P03001.

[48] S. Artz et al. "The ATLAS Level-1 Muon Topological Trigger Information for Run 2 of the LHC". In: *Journal of Instrumentation* 10.02 (2015), p. C02027. URL: http://stacks.iop.org/1748-0221/10/i=02/a=C02027.

[49] R Achenbach et al. *The ATLAS Level-1 Calorimeter Trigger*. Tech. rep. ATL-DAQ-PUB-2008-001. ATL-COM-DAQ-2008-002. Geneva: CERN, Jan. 2008. URL: https://cds.cern.ch/record/1080560.

[50] The ATLAS TRT collaboration et al. "The ATLAS TRT electronics". In: *Journal of Instrumentation* 3.06 (2008), P06007. URL: http://stacks.iop.org/1748-0221/3/i=06/a=P06007.

[51] S Ask et al. "The ATLAS central level-1 trigger logic and TTC system". In: *Journal of Instrumentation* 3.08 (2008), P08002. URL: http://stacks.iop.org/1748-0221/3/i=08/a=P08002.

[52] R Achenbach et al. "The ATLAS Level-1 Calorimeter Trigger". In: *Journal of Instrumentation* 3.03 (2008), P03001. URL: http://stacks.iop.org/1748-0221/3/i=03/a=P03001.

[53] Sebastian Artz. *Physics performances with the new ATLAS Level-1 Topological trigger in Run 2*. Tech. rep. ATL-DAQ-PROC-2016-012. Geneva: CERN, Aug. 2016. URL: https://cds.cern.ch/record/2210103.

[54] Atlas Trigger-DAQ Steering Group. "Trigger & DAQ Interfaces with Front-End Systems: Requirement Document. Version 2.0." In: (June 9, 1998). URL: https://atlas.web.cern.ch/Atlas/GROUPS/FRONTEND/FEreq980310.pdf.

[55] William Panduro Vazquez and ATLAS Collaboration. *The ATLAS Data Acquisition system in LHC Run 2*. Tech. rep. ATL-DAQ-PROC-2017-007. 3. Geneva: CERN, Feb. 2017. URL: https://cds.cern.ch/record/2244345.

[56] Erik Van der Bij, R A McLaren, and Z Meggyesi. "S-LINK: A prototype of the ATLAS read-out link". In: (1998). URL: https://cds.cern.ch/record/405075.

[57] C.P. Bee et al. "The raw event format in the ATLAS Trigger & DAQ". In: (1998).

[58] R (SLAC) Bartoldus et al. *Technical Design Report for the Phase-I Upgrade of the ATLAS TDAQ System*. Tech. rep. CERN-LHCC-2013-018. ATLAS-TDR-023. Final version presented to December 2013 LHCC. Geneva: CERN, Sept. 2013. URL: https://cds.cern.ch/record/1602235.

[59] Jovan Mitrevski. "Preparing ATLAS reconstruction software for LHC's Run 2". In: *Journal of Physics: Conference Series* 664.7 (2015), p. 072034. URL: http://stacks.iop.org/1742-6596/664/i=7/a=072034.

[60] *Expected performance of missing transverse momentum reconstruction for the ATLAS detector at* $\sqrt{s} = 13$ *TeV*. Tech. rep. ATL-PHYS-PUB-2015-023. Geneva: CERN, July 2015. URL: https://cds.cern.ch/record/2037700.

[61] Ralf Spiwoks et al. *The ATLAS Level-1 Central Trigger Processor (CTP)*. Tech. rep. ATL-DAQ-CONF-2005-030. CERN-ATL-DAQ-CONF-2005-030. 2005. URL: https://cds.cern.ch/record/889547.

[62] *Letter of Intent for the Phase-I Upgrade of the ATLAS Experiment*. Tech. rep. CERN-LHCC-2011-012. LHCC-I-020. Geneva: CERN, Nov. 2011. URL: https://cds.cern.ch/record/1402470.

[63] Concurrent Tehcnologies. *VP E2x/msd – VME Processor*. Nov. 13, 2018. URL: https://www.gocct.com/product/vp-e2xmsd-vme-processor/.

[64] H. Bertelsen et al. "Operation of the upgraded ATLAS Central Trigger Processor during the LHC Run 2". In: *Journal of Instrumentation* 11.02 (2016), p. C02020. URL: http://stacks.iop.org/1748-0221/11/i=02/a=C02020.

[65] S. Baron. *Timing, Trigger and Control (TTC) Systems for the LHC*. Nov. 13, 2018. URL: https://ttc.web.cern.ch/TTC/.

[66] P Borrego-Amaral et al. *The ATLAS Local Trigger Processor (LTP) 018*. Tech. rep. ATL-DAQ-2004-018. ATL-COM-DAQ-2004-017. CERN-ATL-COM-DAQ-2004-017. Geneva: CERN, 2004. URL: https://cds.cern.ch/record/795917.

[67] Wikipedia contributors. *Low-voltage differential signaling — Wikipedia, The Free Encyclopedia*. [Online; accessed 29-November-2018]. 2018. URL: https://en.wikipedia.org/wiki/Low-voltage_differential_signaling.

[68] Wikipedia contributors. *Nuclear Instrumentation Module — Wikipedia, The Free Encyclopedia*. [Online; accessed 29-November-2018]. 2018. URL: https://en.wikipedia.org/wiki/Nuclear_Instrumentation_Module.

[69] Wikipedia contributors. *Emitter-coupled logic — Wikipedia, The Free Encyclopedia*. [Online; accessed 29-November-2018]. 2018. URL: https://en.wikipedia.org/wiki/Emitter-coupled_logic.

[70] Ph. Farthouat and P. Gälln. *TTC-VMEbus INTERFACE. TTCvi - MkII*. 2000. URL: https://ttc.web.cern.ch/TTC/TTCviSpec.pdf.

[71] B.G. Taylor. *TTC laser transmitter (TTCex, TTCtx, TTCmx) User Manual*. URL: https://ttc.web.cern.ch/TTC/TTCtxManual.pdf.

[72] S Baron et al. "Jitter impact on clock distribution in LHC experiments". In: *JINST* 7 (2012), p. C12023. URL: https://cds.cern.ch/record/1608681.

[73] C Gabaldon. "Performance of the ATLAS Trigger System". In: *Journal of Instrumentation* 7.01 (2012), p. C01092. URL: http://stacks.iop.org/1748-0221/7/i=01/a=C01092.

[74] A. Marchioro H. Furtado J. Schrader and P. Moreira. *Delay 25 an ASIC for timing adjustment in LHC*. Ed. by Geneva Switzerland CERN-EP/MIC. URL: http://proj-delay25.web.cern.ch/proj-delay25/papers/article_lecc2005_final.pdf.

[75] VITA John Rynearson Technical Director. *How fast is the VMEbus?* Nov. 13, 2018. URL: https://www.vita.com/page-1855188.

[76] Peter Jenni et al. *ATLAS high-level trigger, data-acquisition and controls: Technical Design Report*. Technical Design Report ATLAS. Geneva: CERN, 2003. URL: https://cds.cern.ch/record/616089.

[77] A Sicoe et al. *A Persistent Back-End for the ATLAS TDAQ Online Information Service (P-BEAST)*. Tech. rep. ATL-DAQ-PROC-2011-047. Geneva: CERN, Nov. 2011. URL: https://cds.cern.ch/record/1402973.

[78] Object Management Group. *CORBA*. Nov. 13, 2018. URL: http://www.corba.org/.

[79] I Soloviev and A Sicoe. "New Persistent Back-End for the ATLAS Online Information Service". In: (May 2014). URL: https://cds.cern.ch/record/1702937.

[80] E. Raymond. *ncurses: Portable Screen-Handling for Linux*. Nov. 13, 2018. URL: https://www.linuxjournal.com/article/1124.

[81] Cisco Systems Network Working Group and T. Ylonen. *The Secure Shell (SSH) Protocol Architecture*. Nov. 13, 2018. URL: https://tools.ietf.org/html/rfc4251.html.

[82] Alexandru D Sicoe et al. "A persistent back-end for the ATLAS TDAQ online information service (P-BEAST)". In: *Journal of Physics: Conference Series* 368.1 (2012), p. 012002. URL: http://stacks.iop.org/1742-6596/368/i=1/a=012002.

[83] ATLAS Collaboration ; Contact Eric Torrence. *ATLAS Data Summary 2015*. Nov. 13, 2018. URL: https://atlas.web.cern.ch/Atlas/GROUPS/DATAPREPARATION/DataSummary/2015/.

[84] ATLAS Collaboration ; Contact Eric Torrence. *ATLAS Data Summary 2016*. Nov. 13, 2018. URL: https://atlas.web.cern.ch/Atlas/GROUPS/DATAPREPARATION/DataSummary/2016/.

[85] Joerg Steltzer. *'Information on Bunch Group Set 1570'*. [Online; accessed 29-November-2018]. URL: https://atlas-trigconf.cern.ch/bunchgroups?key=1570.

[86] Alessandro Polini. private communication. 2018.

[87] M Paterno. "Calculating Efficiencies and Their Uncertainties". In: (2003). URL: http://home.fnal.gov/~paterno/images/effic.pdf.

[88] Morad Aaboud et al. "Measurement of inclusive and differential cross sections in the $H \to ZZ^* \to 4\ell$ decay channel in pp collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector". In: *JHEP* 10 (2017), p. 132. DOI: 10.1007/JHEP10(2017)132. arXiv: 1708.02810 [hep-ex].

[89] *Search for a new heavy gauge boson resonance decaying into a lepton and missing transverse momentum in 79.8 fb^{-1} of pp collisions at $\sqrt{s} = 13$ TeV with the ATLAS experiment*. Tech. rep. ATLAS-CONF-2018-017. Geneva: CERN, June 2018. URL: https://cds.cern.ch/record/2621303.

[90] Hass AbouZeid. private communication. 2018.

[91] Silvia Biondi. *ATLAS e-Log 325486: Constant busy status of TGC partition*. [Online; accessed 29-November-2018]. Oct. 17, 2018. URL: https://atlasop.cern.ch/elisa/display/325486.

[92] Silvia Biondi. *ATLAS e-Log 325507: Shift Summary for Muons desk*. [Online; accessed 29-November-2018]. Oct. 17, 2018. URL: https://atlasop.cern.ch/elisa/display/325507.

[93] Klaudia Burka. *ATLAS e-Log 325567: Shift Summary for Trigger desk*. [Online; accessed 29-November-2018]. Nov. 27, 2018. URL: https://atlasop.cern.ch/elisa/display/325567.

[94] G Anders et al. "Intelligent operations of the data acquisition system of the ATLAS experiment at LHC". In: *Journal of Physics: Conference Series* 608.1 (2015), p. 012007. URL: http://stacks.iop.org/1742-6596/608/i=1/a=012007.

[95] ATLAS Collaboration. *Technical Design Report for the Phase-II Upgrade of the ATLAS TDAQ System*. Tech. rep. CERN-LHCC-2017-020. ATLAS-TDR-029. Geneva: CERN, Sept. 2017. URL: https://cds.cern.ch/record/2285584.

Glossary

ALFA Absolute Luminosity For ATLAS. 29, 30, 44, 59, 71

ALICE A Large Ion Collider Experiment. 13, 50

ATLAS A Toroidal LHC ApparatuS. 1–5, 7, 8, 13, 19–21, 23–30, 33, 34, 40, 41, 43, 44, 47, 48, 50–54, 57, 61–63, 69, 76, 83, 85–87, 89–95, 105–107, 125, 127, 129, 131, 132, 134, 137, 142, 145, 147, 151, 152, 155, 160, 161, 163, 165, 170, 174, 177, 181–183

BCID Bunch Crossing Identifier. 50–52, 94, 95, 116, 140, 157

BCM Beam Conditions Monitor. 29, 44, 91

BCR Bunch Counter Reset. 51, 60

BPTX Beam Pickup. 28, 29, 44, 63

BSM Beyond the Standard Model. 2, 7, 8, 47

CAM Content Adressable Memory. 72, 74

CCC CERN Central Control Room. 62

CERN European Organization for Nuclear Research. 12, 13, 15, 60

CEST Central European Summer Time. 132

CET Central European Time. 132

CMS Compact Muon Solenoid. 7, 8, 13, 62

CORDE Clock ORbit DElay. 63

CP Cluster Processor. 45, 46

cross section σ. 9, 11, 12, 40, 84, 85, 135, 147, 151

CSC Cathode Strip Chambers. 26, 55

CTP Central Trigger Processor. 1–5, 28, 42–48, 51–53, 57, 59–61, 63–65, 69, 73–76, 78–87, 89–91, 93–97, 99–101, 103–109, 111–113, 115, 116, 118, 125, 127, 129, 131–133, 135, 138, 140–142, 147, 148, 151, 155, 157, 160, 165, 181–184

DAQ Data Acquisition. 37, 48, 50, 51, 54, 74, 80, 177

DCM Data Collection Manager. 54

DDR Double Data Rate. 60, 73

DMA Direct Memory Access. 50, 122

ECR Event Counter Reset. 51, 59, 60, 63, 78, 81, 93, 138, 157

EDF Earliest Dead-line First. 123

EM Electromagnetic. 43, 135

FIFO First In First Out. 66, 67, 74, 75, 80, 97, 107, 118, 119

FPGA Field Programmable Gate Array. 183

FSM Finite State Machine. 123, 168, 169

GPS Global Positioning System. 97, 103, 120

HLT High Level Trigger. 1, 35–37, 39, 47, 48, 50, 52, 54, 55, 74, 76, 84, 127, 157, 160–162, 164, 178, 179, 181, 183

HLTPU HLT Processing Unit. 54

HLTSV HLT Supervisor. 54

HPTDC High Precision Time-To-Digital Converter. 70, 120

IBL Insertable B-Layer. 5, 22, 95, 157

IP Interaction Points. 13

IPC Inter Process Communication. 110–113, 124, 170, 177

IS Information Service. 111–113, 122, 125, 127, 129, 132, 165–167, 169, 172, 173, 176, 177

ITk ATLAS new Inner Tracker. 183

JEP Jet/Energy-sum Processor. 45, 46

LAr Liquid Argon. 23–25, 43, 45, 79, 152, 159

LEP Large Electron Positron collider. 13

LHC Large Hadron Collider. 1, 4, 8, 9, 11–17, 19–21, 28–30, 33, 36, 57, 60–65, 67, 83, 94, 95, 97, 106, 120, 131–134, 140, 142, 145, 149, 152, 163, 165, 175, 182, 183

LHCb Large Hadron Collider beauty, experiment. 13

LINAC LINear ACcelerator. 15

Long Shutdown Periods between run periods for planned upgrades.. 103

Long Shutdown I First Long Shutdown, from Q1 2013 to Q4 2014. 1, 22, 27, 59, 95, 131

LSP Lightest Super-symmetric Particle. 8

LTP Local Trigger Processor. 61, 63, 64, 69, 76

LTPi Local Trigger Processor Interface. 63

LUCID luminosity detector. 91

LUT Look Up Table. 45, 70–72

LVDS Low-voltage Differential Signaling. 61

MBTS Minimum Bias Trigger Scintillators. 29, 44, 74

MDT Monitored Drift Tubes. 25, 26, 55, 157

MicroMegas Micro-Mesh Gaseous Structure. 5, 27

MUCTPI Muon-to-CTP Interface. 42–44, 46, 69

pbeast Persistent Back-End for the ATLAS TDAQ On-line Information Service. 111, 129

PMT Photo Multiplier Tube. 25, 29, 30

POC Physics Object Candidates. 35, 42–46, 69–72

PPM Pre-Processor Modules. 45

PS Proton Synchrotron. 15, 16

PSK Prescale Key. 162, 169, 170, 172, 173, 178

QCD Quantum Chromo Dynamics. 9, 11, 12, 41, 42, 181

RF Radio Frequency. 13, 16, 61, 62, 133

RMS Root mean square. 143

ROB Read-Out Buffer. 47, 49–52, 54, 157

ROD Read-Out Driver. 47, 50–54, 57, 59, 69, 76, 78, 79, 82, 83, 89, 91, 93, 140, 157, 160

ROI Region of Interest. 47, 48, 54, 74, 147

ROIB Region of Intrest Builder. 47, 48, 54, 80, 157, 160

ROS Read-Out System. 47, 49–52, 54

RPC Resistive Plate Chambers. 26, 27, 43, 157

Run I The first run period of the LHC run schedule, ended Q1 2013. 9, 112

Run II The second run period of the LHC run schedule, from Q1 2015 to Q4 2018. 1, 2, 4, 9, 41, 42, 54, 58, 59, 131, 181, 184

Run III The second run period of the LHC run schedule, scheduled from Q1 2021 to Q4 2023. 14

SBC Single Board Computer. 57, 105–110, 122, 126

SCT Semiconductor Tracker. 21–23

SFO Sub-Farm Output. 55

SM Standard Model of Particle Physics. 7–9, 12, 38, 42, 47

SPS Super Proton Synchrotron. 15, 16

SUSY Super Symmetry. 8

TAP Trigger After Prescale. 74, 87, 88, 92, 147, 149, 167, 175

TAV Trigger After Veto. 74, 87, 88, 92, 147, 149, 157, 158

TBP Trigger Before Prescale. 74, 87, 142, 149, 152, 157, 166, 167, 175, 177

TDAQ Trigger and Data Acquisition. 48, 54, 83, 86, 90, 91, 183

TGC Thin Gas Chambers. 26, 27, 43, 159

TRT Transition Radiation Tracker. 21, 23, 44, 156, 157

TTC Trigger, Timing, and Control. 51, 53, 59–64, 72, 78, 82, 108, 181

TTCex TTC encoder and laser transmitter. 61

TTCvi TTC VME Interface. 61

UTC Coordinated Universal Time. 132

van der Meer scan Beam scan method for luminosity determination named after the inventor and Nobel laureate Simon van der Meer.. 44

VME Versa Module Europa, bus standard. 57, 59, 66, 71, 74, 75, 105–111, 114, 122–124