Dissertation
submitted to the
Combined Faculties for the Natural Sciences and for Mathematics
of the Ruperto-Carola University of Heidelberg, Germany,
for the degree of
Doctor of Natural Sciences

presented by

Dipl.Phys.: Florian Föhlisch
born in: Heidelberg

Oral examination: December 19, 2007

Calibration for the ATLAS Level-1 Calorimeter-Trigger

Referees: Prof. Dr. Karlheinz Meier
Prof. Dr. Ulrich Uwer

"Always keep things
as simple as possible,
but not simpler."

A. Einstein

Kalibrierung des ATLAS Level-1 Kalorimeter Triggers

Diese Arbeit beschreibt Entwicklungen und Tests die nötig sind, um den Prä-Prozessor des ATLAS Level-1 Kalorimeter Triggers zur Datennahme zu betreiben. Die Aufgaben des Prä-Prozessors sind vor allem die Digitalisierung, der zeitliche Abgleich und die Kalibrierung von Signalen aus dem ATLAS Kalorimeter. Eigens für diese Zwecke entwickelte Hardware muss zur Erfüllung dieser Aufgaben konfiguriert werden.

Software wurde entwickelt, die es erlaubt die Prä-Prozessor Hardware aufzusetzen wozu das Register-Model der Prä-Prozessor Module implementiert wurde. Eine Methode zur Konfiguration des Prä-Prozessors im Rahmen der ATLAS Datennahme verwendet vom Benutzer getroffene Einstellungen und Ergebnisse von Kalibrierungsmessungen um geeignete Werte für die Register des Prä-Prozessors zu finden. Verfahren, die es erlauben die erforderlichen Messungen vorzunehmen und in einer Datenbank zu speichern, werden präsentiert. Desweiteren werden Tests die mit der Installation des ATLAS-Experiments einhergehen vorgestellt und Ergebnisse gezeigt.

Calibration for the ATLAS Level-1 Calorimeter-Trigger

This thesis describes developments and tests that are necessary to operate the Pre-Processor of the ATLAS Level-1 Calorimeter Trigger for data acquisition. The major tasks of Pre-Processor comprise the digitizing, time-alignment and the calibration of signals that come from the ATLAS calorimeter. Dedicated hardware has been developed that must be configured in order to fulfill these tasks.

Software has been developed that implements the register-model of the Pre-Processor Modules and allows to set up the Pre-Processor. In order to configure the Pre-Processor in the context of an ATLAS run, user-settings and the results of calibration measurements are used to derive adequate settings for registers of the Pre-Processor. The procedures that allow to perform the required measurements and store the results into a database are demonstrated. Furthermore, tests that go along with the ATLAS installation are presented and results are shown.

Contents

Chapter 1

Introduction

In the Large Hadron Collider (LHC) at the European Center for Nuclear Research bunches of $\sim 10^{11}$ protons will collide with a center-of-mass energy of $\sqrt{s} = 14\,\text{TeV}$. This energy allows to search for the Higgs boson which is the only particle in the Standard Model that has not yet been discovered. Beside the search for the Higgs there are reasons to believe that physics beyond the Standard Model may occur at the energy achieved by the LHC. Together with CMS, ATLAS is one of the general purpose experiments at the LHC, whose main goals are to find the Higgs boson and to discover new physics if it appears. These searches, however, will have to cope with a dominant background of QCD events. The production of a Higgs boson, e.g., can be expected only in approximately every $4 \cdot 10^6$ th bunch collision. Therefore LHC is operated with a collision rate of 40 MHz. The amount of data a collision produces in the ATLAS detector, however, makes it impossible to store events with a higher rate than 200 Hz. Thus a fast and efficient trigger system is required that separates the desired events from the background.

The ATLAS trigger system is separated in three levels. The Level-1 Trigger reduces the event rate down to 75 kHz in 2.5 μs. The second trigger level achieves 1 kHz in 20 ms and the third level reduces the rate further down to the storage rate of less than 200 Hz in 1 s. The Level-1 Trigger is completely implemented in hardware and has a muon and a calorimeter part. This work addresses the calorimeter part.

The Level-1 Calorimeter Trigger works with $\sim$ 7200 analog input signals from the detector. The analog signals are prepared for digital processing in the Pre-Processor system. The tasks of the Pre-Processor comprise digitization, rate-metering, timing alignment, energy-calibration and identification of the bunch-crossing that is associated with a signal. The Pre-Processor is designed as an eight-crate VMEbus system. The signal processing is mainly implemented in ASICs that are mounted on VME Modules. Essentially all of the Calorimeter Trigger calibration is done by setting up these ASICs in the Pre-Processor system. In order to run the ATLAS experiment the Pre-Processor has to be configured in coherence with the rest of the system. Therefore adequate settings have to be determined in order to calibrate the Calorimeter Trigger. This work addresses the operation of the Pre-Processor system during an ATLAS run and the determination of parameters for an initial calibration of the ATLAS Level-1 Calorimeter Trigger.

In Chapter 2 the physics related to ATLAS is introduced. The LHC, the kine-

matics of a proton-proton collision and the ATLAS detector are described in Chapter 3. The calorimeters that are of special interest for this work are treated in greater detail than the rest of the detector. Chapter 4 shows how the Level-1 Calorimeter Trigger is embedded in the ATLAS Trigger and Data-AcQuisition (Trigger-DAQ) system. The Level-1 Calorimeter Trigger itself is explained in Chapter 6. Chapter 7 comprises everything that is associated to the calibration of the Calorimeter Trigger. First a detailed description of the calibration-related parameters is given, followed by an introduction of the setups the Pre-Processor can be, and has been, used in. After that the software used to operate ATLAS in a run is presented. This is necessary to understand the concepts and procedures used to gain the information needed to configure the Calorimeter Trigger during a run. Finally the possibilities for an initial Calibration of the Calorimeter Trigger are discussed. The next Chapter presents tests that are done in parallel to the commissioning. Finally a Summary and an Outlook of this work are compiled in Chapter 9.

Chapter 2

Physics Motivation

2.1 The Standard Model

The Standard Model of particle physics postulates the fundamental constituents of matter as point like particles of spin $\frac{1}{2}$: fermions. Fermions are divided into quarks and leptons, both of which exist in three generations, distinguishable by their mass. Up and down quarks as well as electrons and electron-neutrinos, are the lightest particles of their type and build up the first generation. The charged particles of the first generation are stable and form the visible matter that is surrounding us and that we are made of. The members of the second and third generation are unstable and have been discovered by experiments of particle physics, which was leading to the establishment of the Standard Model.

Fermions	**I**	**II**	**III**	**Charge**	**Bosons**
Quarks	*up*	*charme*	*top*	$+2/3$	g
	down	*strange*	*bottom*	$-1/3$	$W^{\pm}$
Leptons	ν_e	ν_μ	ν_τ	0	Z
	electron	*muon*	*tau*	-1	A

Table 2.1: Fundamental particles in the Standard Model

The Standard Model is a quantum field theory with the symmetry $SU(3) \times SU(2) \times U(1)$. Demanding invariance under local phase transformation for the fermion fields (gauge-symmetry) requires the existence of boson vector fields. These fields introduce interactions between the fermions, and their number is determined by the symmetry of the group [1]. The $SU(3)$ group introduces eight vector fields g that can physically be identified with the gluons as force-carrying particles of the strong interaction. The $SU(2)$ group introduces three $\mathbf{W}^\mu$ vector fields and the $U(1)$ group the the B^μ field.

Using the Pauli matrices as generators for the $SU(2)$ group the W_1 and W_2 component of the $\mathbf{W}^\mu$ fields have a physical representation in the $W^\pm = \frac{1}{\sqrt{2}} W_1^\mu \mp iW_2^\mu$ bosons. The third component W_3^μ, however, is mixing with the B field of the $U(1)$ group.

[1] It is the number of generators needed to generate a unitarity transformation for the group

One finds:

$$\begin{aligned} Z^\mu &= W_3^\mu \cos(\theta_w) - B^\mu \sin(\theta_w) \\ A^\mu &= W_3^\mu \sin(\theta_w) + B^\mu \cos(\theta_w) \end{aligned} \tag{2.1}$$

where θ_w is the so-called Weinberg angle or weak mixing angle, Z is the Z boson and A is the photon field.

The forces introduced by the boson fields differ in the charges they are coupling to. Strong interacting particles are triplets under the $SU(3)$-symmetry and carry a color charge. The theory describing the strong interaction is called the Quantum-Chromo-Dynamics (QCD). Except for the quarks only the gluons carry a color charge, which makes them self-interacting. As a consequence the strong interaction increases with the momentum transfer, which leads to the confinement of quarks in color neutral states. When trying to free a quark jets of color neutral hadrons are generated by quark-antiquark production. This process is called "hadronization".

Because the W_3^μ and the B^μ fields are mixing, one generally combines the forces introduced by the $SU(2) \times U(1)$ symmetry in the electroweak interaction [3]. One can, however, identify the photon A^μ as carrier of the electrical force, that couples to the electrical charge as described by Quantum-Electro-Dynamics (QED).

The $W^\pm$, the Z boson and the photon A^μ are the carriers of the electroweak interaction. The $SU(2)$ part of the electroweak interaction, carried by the $\mathbf{W}^\mu$ fields is coupling to the weak isospin T_3. Experiments show that left-handed fermions and right-handed anti-fermions have a weak isospin of $T_w = \frac{1}{2}$, resulting in two $(2T_w + 1)$ states the weak isospin doublets. Each component of a weak isospin doublet has a weak isospin quantum number $T_3 = \pm\frac{1}{2}$. Right-handed particles are singlets with respect to the weak isospin $T_w = T_3 = 0$.

Weak isospin doublets:

$$\begin{aligned} &\begin{pmatrix} u_L \\ d_L \end{pmatrix} \quad , \quad \begin{pmatrix} c_L \\ s_L \end{pmatrix} \quad , \quad \begin{pmatrix} t_L \\ b_L \end{pmatrix} \\ &\begin{pmatrix} \nu_e \\ e_L \end{pmatrix} \quad , \quad \begin{pmatrix} \nu_\mu \\ \mu_L \end{pmatrix} \quad , \quad \begin{pmatrix} \nu_\tau \\ \tau_L \end{pmatrix} \end{aligned} \tag{2.2}$$

With the coupling constant g and $\boldsymbol{\tau}$ the vector of the Pauli matrices (2.14), the interaction term of the $SU(2)$ symmetry reads:

$$\tfrac{g}{2} \bar{\Psi}_R \gamma^\mu \mathbf{W}_\mu \boldsymbol{\tau} \Psi_L \tag{2.3}$$

where:

$$\mathbf{W}^\mu \boldsymbol{\tau} = \begin{pmatrix} W_3^\mu & W_1^\mu - iW_2^\mu \\ W_1^\mu + iW_2^\mu & -W_3^\mu \end{pmatrix} \equiv \begin{pmatrix} W_3^\mu & \sqrt{2}\mathrm{W}^+ \\ \sqrt{2}\mathrm{W}^- & -W_3^\mu \end{pmatrix}$$

The coupling to a $W^\pm$ boson is flipping the weak isospin T_3, turning a ν_l into a lepton l and an *up*-type quark into a *down*-type quark or vice-versa.

The B^μ field is the gauge field of the $U(1)$ symmetry. With a coupling constant $g' \neq g$ the coupling of any fermion to B^μ is proportional to its weak hypercharge Y. As the W_3^μ-component of the $SU(2)$-symmetry and the B^μ field are mixing to the

left-handed	Q	T_3	Y	Q	T_3	Y	right-handed
ν_e,ν_μ,ν_τ	0	$\frac{1}{2}$	$-\frac{1}{2}$				
e_L,μ_L,τ_L	-1	$-\frac{1}{2}$	$-\frac{1}{2}$	-1	0	-1	e_R,μ_R,τ_R
u_L,c_L,t_L	$\frac{2}{3}$	$\frac{1}{2}$	$\frac{1}{6}$	$\frac{2}{3}$	0	$\frac{2}{3}$	u_R,c_R,t_R
d_L,s_L,b_L	$-\frac{1}{3}$	$-\frac{1}{2}$	$\frac{1}{6}$	$-\frac{1}{3}$	0	$-\frac{1}{3}$	d_R,s_R,b_R

Table 2.2: Electroweak quantum numbers.

photon A and the Z boson, Eqn.(2.1), the latter couple to both: the weak isospin and the weak hypercharge.

Interaction term of the neutral gauge bosons:

$$\begin{aligned} &-\bar{\Psi}\gamma^\mu[g\overbrace{(\cos\theta_w Z_\mu + \sin\theta_w A_\mu)}^{W_3} T_3 \\ &+g'\underbrace{(-\sin\theta_w Z_\mu + \cos\theta_w A_\mu)}_{B} Y]\Psi \end{aligned} \tag{2.4}$$

According to Eqn.(2.4) the coupling to the photon is $(g\sin\theta_w T_3 + g'\cos\theta_w Y)$. From QED we know that this must be equal to eQ where Q is the charge quantum number of the particle.

Therefore:

$$g\sin\theta_w = g'\cos\theta_w = e \tag{2.5}$$

and:

$$Q = Y + T_3 \tag{2.6}$$

Using these relations one can determine electroweak quantum numbers for left- and right-handed particles as shown in Table 2.2. One set of Q, T_3 and Y values is called flavor.

The $\mathbf{W}^\mu$ fields are coupling to the weak isospin doublets as listed in Eqn. (2.2). The coupling of left-handed particles to the $W^\pm$ bosons changes their weak isospin, e.g. transforming neutrinos ν_l to leptons l and *up*-type quarks to *down*-type quarks. These transformations are flavor changing, but only within one isospin doublet, i.e. they seem to be limited to one generation. For quarks, however, electroweak $W^\pm$-decays can be observed that change their flavor from one generation to another. This phenomenon can be explained with a mixture of quantum states: mass eigenstates of quarks do not exactly compare with the electroweak eigenstates known from the weak isospin doublets.

$$\begin{pmatrix} d_{EW} \\ s_{EW} \\ b_{EW} \end{pmatrix} = \begin{pmatrix} V_{ud} & V_{us} & V_{ub} \\ V_{cd} & V_{cs} & V_{cb} \\ V_{td} & V_{ts} & V_{tb} \end{pmatrix} \begin{pmatrix} d_M \\ s_M \\ b_M \end{pmatrix} \tag{2.7}$$

The Cabibbo-Kobayashi-Maskawa (CKM) matrix, Eqn (2.7), is a unitarity matrix, that quantizes how electroweak eigenstates are composed of mass eigenstates. The square Matrix elements $|V_{qq'}|^2$ can be interpreted as the probability of the transition of one quark q to another q' in an electroweak decay. The mixing is smallest between first and third generation.

The Standard Model can very successfully describe the strong, the weak and the electromagnetic interaction, but even in its simplest version, it has 19 free parameters:

- 3 coupling constants of the $SU(3) \times SU(2) \times U(1)$ symmetry,
  2 coupling constants + Weinberg angle θ_w, respectively.
- 3 angles in the CKM matrix.
- 1 phase in the CKM matrix that is responsible for CP-violation.
- 1 strong force CP-violating parameter.
- 9 fermion masses.
- 1 Z boson mass.
- 1 Higgs boson mass.

The Higgs mechanism explains the masses of quarks and leptons with the coupling to a scalar field, the Higgs field. Seven parameters, the fermion and Z masses can be described by this coupling. This does not reduce the number of open parameters, but solves problems simple mass terms would have.

2.2 The Higgs Mechanism

The theory as discussed so far cannot explain the mass of any fundamental particles. Fermion mass terms would transform left-handed particles in right-handed ones and vice-versa:

$$m_f \bar{f} f = m_f \bar{f}_R f_L + m_f \bar{f}_L f_R \tag{2.8}$$

which is forbidden, as left-handed fermions are isospin doublets and right-handed are isospin singlets. Simple gauge boson mass terms of the form $\frac{1}{2} m_\gamma^2 A^\mu A_\mu$ are forbidden, as they would break gauge symmetry. Terms like this would not be invariant under the transformation $A^\mu \rightarrow A^\mu + \partial^\mu \chi$ which is mandatory to allow local phase transformation for the fermion fields.

Introducing a scalar isospin doublet:

$$\mathbf{\Phi} = \begin{pmatrix} \Phi_1 \\ \Phi_2 \end{pmatrix} \tag{2.9}$$

the Higgs field, can solve the mass problem described above. It allows to explain fermion masses as Yukawa interactions with that field:

$$g_f \bar{f}_R \mathbf{\Phi} f_L + g_f \bar{f}_L \mathbf{\Phi} f_R \tag{2.10}$$

where g_f is the coupling constant of the fermion to the Higgs field, e.g. for electrons one finds:

$$g_e \left(\bar{e_R} \Phi_1 \nu_e + \bar{e_R} \Phi_2 e_L + \bar{\nu_e} \Phi_1 e_R + \bar{e_L} \Phi_2 e_R \right) \tag{2.11}$$

One can immediately see that the expectation value for the first isospin component of the Higgs field has to be zero in order for Eqn.(2.11) to have the form of mass terms. With:

$$\langle \mathbf{\Phi} \rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ v \end{pmatrix} \tag{2.12}$$

the electron mass m_e becomes $g_e v/\sqrt{2}$. Masses for the remaining fermions can be explained in a similar way. As Φ_2 must be neutral ($Q = 0$) and has a weak isospin of $T_3 = -\frac{1}{2}$, the weak hypercharge must be $Y = \frac{1}{2}$

The particular choice of $\langle \mathbf{\Phi} \rangle$ in Eqn.(2.12) breaks gauge symmetry. This happens as a spontaneous symmetry breaking of a global symmetry, which, according to the Goldstone theorem, evokes the appearance of a massless particle (Goldstone boson) for each broken symmetry.

In a gauge theory, however, one can always make the Goldstone bosons vanish by a gauge transformation, resulting in the so-called "unitarity gauge". One does not seem to have gained a lot. But, the appearance of Goldstone bosons that are forced to disappear again by gauge transformations is linked to another effect originating in the spontaneous symmetry breaking of the Higgs field. The non-zero expectation value of Higgs-field leads to terms in the kinetic energy part of the Lagrangian $|D_\mu \mathbf{\Phi}|^2$ that give masses to the gauge bosons. This effect is called Higgs mechanism.

The number of gauge bosons that can acquire masses via the Higgs mechanism equals the number of Goldstone bosons appearing or the number of broken symmetries, respectively. Of course the maximum number of symmetries that can be broken equals the number of generators required for the symmetry that is considered.

As the electroweak interaction is described by a $SU(2) \times U(1)$ symmetry, the complete electroweak gauge transformation is given by:

$$\mathbf{\Phi} \rightarrow \mathbf{\Phi'} = e^{i\omega_a \tau_a} e^{i\beta/2} \mathbf{\Phi} \quad ; \quad a = 1..3 \tag{2.13}$$

It has four generators altogether where the τ_a are defined via the Pauli matrices:

$$\tau_1 = \tfrac{1}{2} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad , \quad \tau_2 = \tfrac{1}{2} \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad , \quad \tau_3 = \tfrac{1}{2} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{2.14}$$

The expectation value $\langle \mathbf{\Phi} \rangle = \frac{1}{\sqrt{2}} \binom{0}{v}$ is not changed for:

$$\omega_1 = \omega_2 = 0 \quad , \quad \omega_3 = \beta \tag{2.15}$$

This means the theory has one unbroken symmetry represented by this specific choice of generators and coefficients; three of the four gauge bosons will acquire masses via the Higgs mechanism. As mentioned before the mass terms for the gauge bosons come from the kinetic energy part of the Higgs field:

$$|D_\mu \mathbf{\Phi}|^2 = \left| \left(\partial_\mu - igW_{a\mu}\tau_a - i\frac{1}{2} g' B_\mu \right) \mathbf{\Phi} \right|^2 \tag{2.16}$$

Using (2.12) they become:

$$\frac{v^2}{8} \left[g^2 (W_{1\mu})^2 + g^2 (W_{2\mu})^2 + (g' B_\mu - g W_{3\mu})^2 \right] \tag{2.17}$$

This can be written as:

$$\Big(\frac{vg}{2}\Big)^2\Big(\frac{{W_1}^\mu - i{W_2}^\mu}{\sqrt{2}}\Big)\Big(\frac{W_{1\mu} + iW_{2\mu}}{\sqrt{2}}\Big) + \frac{1}{2}\Big(\frac{v\sqrt{g^2+g'^2}}{2}\Big)^2\Big(\frac{gW_{3\mu} - g'B_\mu}{\sqrt{g^2+g'^2}}\Big)^2 \quad (2.18)$$

which exactly is:

$$m_W^2 W^{+\mu} W_\mu^- + \frac{1}{2} m_Z^2 {Z_\mu}^2 \quad (2.19)$$

Note, that the photon, which is orthogonal to Z_μ:

$$A_\mu = \frac{g'W_{3\mu} + gB_\mu}{\sqrt{g^2+g'^2}} \quad (2.20)$$

does not have a mass term. Comparing (2.1) with (2.18) and (2.19) gives relations for the weak mixing angle:

$$\cos\theta_W = \tfrac{g}{\sqrt{g^2+g'^2}} \quad , \quad \sin\theta_W = \tfrac{g'}{\sqrt{g^2+g'^2}} \quad , \quad \sin^2\theta_W = 1 - \tfrac{m_W^2}{m_Z^2} \quad (2.21)$$

As an isospin doublet of two complex valued fields the general Higgs field has four degrees of freedom. In order to allow spontaneous symmetry breaking one has to assume a Lagrangian that leads to an vacuum expectation value for the Higgs field as Eqn.(2.12) shows. A suitable choice is the Linear-Sigma model. Once one has a Lagrangian the Higgs field can be parametrized using its vacuum expectation value. When using this parametrization in the Lagrangian three of the four degrees of freedom generate terms that can be interpreted as massless Goldstone bosons, and mass terms are generated for three of four gauge bosons due to the expectation value of the Higgs field. In a "unitarity gauge" the Goldstone bosons can be forced to vanish, which eliminates three degrees of freedom of the Higgs field. However these degrees of freedom are not lost: as the gauge bosons are vector fields they gain a polarization state in conjunction with their mass. Massless vector fields have two states of polarization, helicity ± 1, massive vector fields can also have a longitudinal polarization, helicity 0. To describe this effect it is common to say that the gauge fields have "eaten" the Goldstone bosons in order to create their extra polarization state.

There is still one degree of freedom left in the Higgs doublet, a field describing a real-valued deviation from the expectation value. This field leads to the prediction of a massive scalar particle, the Higgs boson. The discovery of the Higgs boson is mandatory for the justification of the electroweak theory and one of the announced goals of CMS and the ATLAS experiment. In order to illustrate the appearance of the Higgs boson we choose the Linear-Sigma Lagrangian:

$$\mathcal{L} = \underbrace{|D_\mu \mathbf{\Phi}|^2}_{\mathcal{L}_K} + \underbrace{\mu^2 \mathbf{\Phi}^* \mathbf{\Phi} - \lambda(\mathbf{\Phi}^* \mathbf{\Phi})^2}_{\mathcal{L}_V} \quad (2.22)$$

which is divided in a kinetic energy part $\mathcal{L}_K$ and a potential energy part $\mathcal{L}_V$. Choosing the coordinate system such that after symmetry breaking the expectation value v is in the Φ_2 component of the Higgs field, $\mathbf{\Phi}$ can be parametrized as:

$$\mathbf{\Phi}(\mathbf{x}) = \tfrac{1}{\sqrt{2}} \begin{pmatrix} a(x) + ib(x) \\ v + h(x) + ic(x) \end{pmatrix} \quad ; \quad v = \tfrac{\mu}{\sqrt{\lambda}} \quad (2.23)$$

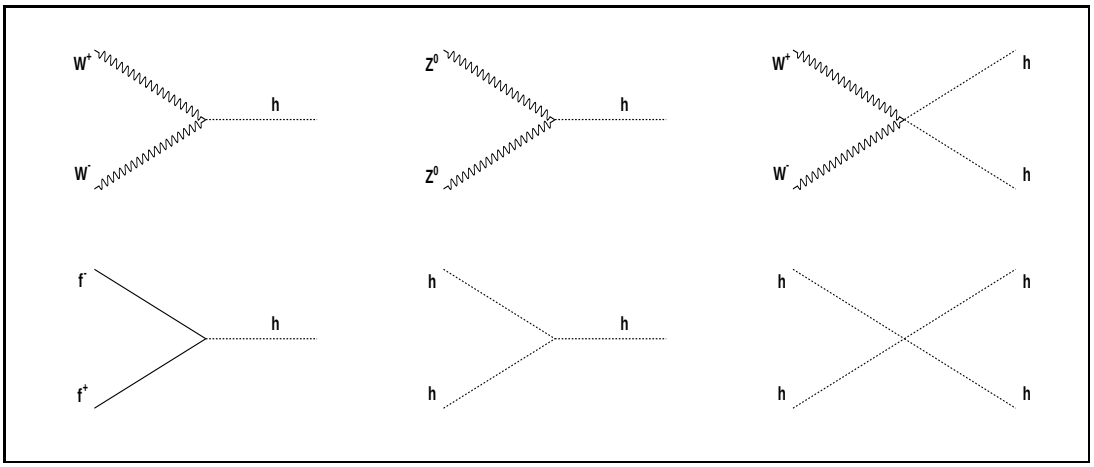

Figure 2.1: Sample Feynman graphs for the couplings of the Higgs boson.

In "unitarity gauge" this becomes:

$$\mathbf{\Phi}(\mathbf{x}) = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ v + h(x) \end{pmatrix} \tag{2.24}$$

with the real valued deviation $h(x)$ from the expectation value v.

Plugging (2.24) into the Lagrangian (2.22), the kinetic energy part becomes:

$$\mathcal{L}_K = \frac{1}{2}(\partial_\mu h)^2 + \left[m_W^2 W^{+\mu} W_\mu^- + \frac{1}{2} m_Z^2 Z^\mu Z_\mu \right] \left(1 + \frac{h}{v}\right)^2 \tag{2.25}$$

This gives extensions to (2.19), which was computed with the expectation value only. The first term is a kinetic energy term of the Higgs boson. The factor behind the $W^\pm$ and Z mass terms indicates that there are couplings between the Higgs boson and the massive gauge bosons of the electroweak theory (Fig. 2.1).

For the potential energy part one gets:

$$\mathcal{L}_V = -\mu^2 h^2 - \lambda v h^3 - \frac{1}{4}\lambda h^4 \equiv -\frac{m_h^2 h^2}{2} - \sqrt{\frac{\lambda}{2}} m_h h^3 - \frac{\lambda h^4}{4} \tag{2.26}$$

Beside self interactions of the third and fourth order there is a mass term for the Higgs boson. The Higgs mass is related to the vacuum expectation value $m_h = \sqrt{2}\mu = v\sqrt{2\lambda}$. Current data only give information on the combination of Higgs parameters $v = 246\,\mathrm{GeV}$. Once the Higgs mass is known, one can explicitly determine μ and λ.

N.B. Investigating the fermion mass terms (2.10) using the parametrization (2.24) for $\mathbf{\Phi}$ shows that the Higgs boson also couples to fermions:

$$\mathcal{L}_f = -m_f \bar{f} f \left(1 + \tfrac{h}{v}\right) \quad ; \quad m_f = g_f v/\sqrt{2} \tag{2.27}$$

2.3 Beyond the Standard Model

The Standard Model (SM) provides a very good description of the phenomena observed in particle physics. However there is evidence that it is an incomplete theory.

Even though the known particles and their interactions are nicely described, the SM does not give an explanation why the particles exist as they do. It does, e.g. not explain why there are three generations of fermions or why the particles have the masses they have.

Other problems with the current understanding of the world come from cosmology. There is evidence that the visible matter only makes up $\sim 5\%$ of the content of the universe. Another part of the matter in the universe is not electromagnetically interacting and referred to as Dark Matter. It is known that Dark Matter makes up $\sim 22\%$ of the energy-density in the universe but so far there is no understanding what it consists of. The remaining $\sim 73\%$ of the energy-density are not understood any better and referred to as Dark Energy. Dark Energy behaves fundamentally different from matter, e.g. it does clump but is spread smoothly everywhere.

A problem arising with the Higgs Model is the "Hierarchy Problem". Higher order calculations of the $W^{\pm}$ and top mass show that if there is a Higgs boson its mass must be about or smaller than 200 GeV. If there were no new physics up to the Great Unifying Theory (GUT) Scale (10^{16} GeV) one would expect large radiative corrections to the Higgs mass. These corrections would be of the order of the GUT Scale and should push the Higgs mass up to much higher values than 200 GeV. The Hierarchy Problem is the missing answer to the question why the Higgs mass is so small. A solution to the Hierarchy Problem that can do without introducing new physics is given by the possibility that the radiative corrections to the Higgs mass almost cancel and lead to a mass of $\lesssim 200$ GeV. This scenario, however, appears very un-natural and generally is referred to as fine-tuning (the Higgs mass). The much more natural way to solve the Hierarchy Problem is to introduce new physics that protects the Higgs mass from radiative corrections. One would expect any new physics that solves the Hierarchy Problem to manifest itself at a scale comparable to the Higgs mass, i.e. to be in the discoverage-range of LHC.

One example for an extension of the Standard Model, which solves the Hierarchy Problem is Supersymmetry. The general idea of Supersymmetry is to double the number of particles, giving each particle a supersymmetric partner. The simplest supersymmetric theory is called Minimal Supersymmetric Standard Model (MSSM). Particles and their supersymmetric partners have spins that differ by 1/2, i.e. partners of fermions are bosons and vice versa. If Supersymmetry exists and is responsible for the Higgs mass it is likely to be discovered at LHC.

Supersymmetry does not only solve the Hierarchy problem, but has also the feature to unify the coupling constants of the electroweak and the strong interactions at the GUT scale of $\sim 10^{16}$ GeV. This is not the case in the Standard Model. Moreover, Supersymmetry predicts massive neutral particles, the neutralinos. Under certain assumptions, the lightest neutralino is stable and a candidate for Dark Matter. If supersymmetric particles were produced at LHC, they would decay into the lightest neutralino that leaves the detector unseen. Therefore Supersymmetry can only be discovered by looking for unbalanced momentum in the detector.

Today nobody knows how the Standard Model has to be extended, and of course Supersymmetry is not the only theory that addresses the problems described above. However, it might be the most promising one. In any case LHC will do its contribution to give answers to the most fundamental questions in physics today.

Chapter 3

The ATLAS Experiment at the LHC

3.1 The Large Hadron Collider

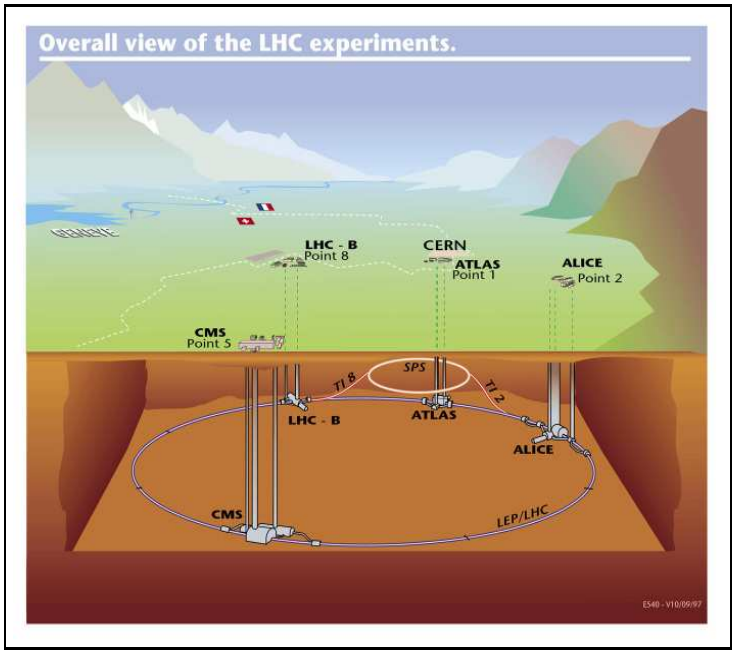

Figure 3.1: The Large Hadron Collider at the European Center for Nuclear Research (CERN) [4].

The Large Hadron Collider (LHC) is currently assembled at the European Center for Nuclear Research (CERN) near Geneva in the tunnel of the former Large Electron Positron Collider (LEP). With a center of mass energy of $\sqrt{s} = 14\,\text{TeV}$ for proton-proton collisions LHC will be the most powerful machine of this kind worldwide. Keeping the particles on their tracks requires a 8.3 T dipole field throughout the whole circumference of 27 km which is provided by superconducting dipoles working with superfluid helium at a temperature of 1.9 K. LHC will either be operated with heavy ions or with protons.

The LHC beam is subdivided into bunches, and the bunch structure determines the collision rate and the luminosity. In a proton-proton run the bunches are following each other at a minimum distance of $\sim 7.5\,\mathrm{m}$ which corresponds to an operating frequency of 40 MHz. With 2961 bunches altogether, each containing $\sim 10^{11}$ particles. The luminosity $\mathcal{L}$ is $10^{34}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$. The beam has a total energy of 360 MJ.

When operating with lead (Pb) ions only every fourth bunch is filled. With 592 bunches altogether, each consisting of $\sim 10^7$ particles, the luminosity during Pb runs becomes $\mathcal{L} = 10^{27}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$. An overview of the parameters for LHC operation with Pb ions can be found in [2]. The energy per nucleon is 7 TeV in proton runs and 2.76 TeV in Pb runs.

Four experiments, ALICE[1], LHCb[2], CMS[3], and ATLAS[3] are situated at different interaction points of the LHC. ALICE, located at Point 2, the only experiment dedicated to heavy ion physics, and designed to study strong interacting matter at high energy densities. The energies achieved with the LHC allow to recreate conditions as have been several microseconds after the big bang. At these conditions a new state of matter, the quark-gluon plasma, is expected. The existence and properties of the the quark gluon plasma are key issues for understanding the confinement of strong interacting particles.

Even though the cross-section for hadrons containing a bottom quark is about two orders of magnitude lower than the total cross section, LHC can be considered as a B-factory. The statistic for B-events is limited only by the rate at which data can be recorded. LHCb the experiment at Point 8, is dedicated to B-physics, particularly designed to measure the parameters of CP-violation. b-hadrons are predominantly produced in forward direction and hermeticity, which means a good capture of all particles coming out of an interaction, is not required for the investigation of b-decays. Thus the LHCb detector is designed as a single arm forward spectrometer.

The achieved energy at LHC is about an order of magnitude higher than that achieved at the Tevatron near Chicago, which is operated at $\sqrt{s} = 2\,\mathrm{TeV}$. It covers completely the considered mass-range of the the only particle that is predicted by the Standard Model and not yet discovered: the Higgs boson. Furthermore it allows the search for physics beyond the Standard Model. The two general purpose detectors CMS, at Point 5, and ATLAS, at Point 1 are dedicated to the the search for the Higgs boson and yet undiscovered physics.

3.2 Kinematics and Cross-Sections

In a proton-proton collider with a center of mass energy of $\sqrt{s} = 14\,\mathrm{TeV}$, the dominant process is an inelastic interaction involving two partons, one from an incoming proton of either direction. This process is illustrated in the sketch Fig. 3.2. Each parton carries the fraction x_i of the proton momentum p_i. Elementary particles like photons, $W^{\pm}$, Z, Higgs bosons, quarks or gluons can emerge from this interaction

[1] ALICE: A Large Ion Collider Experiment
[2] LHCb:Large Hadron Collider beauty
[3] CMS: Compact Muon Solenoid
[3] ATLAS: A Toroidal LHC Apparatus

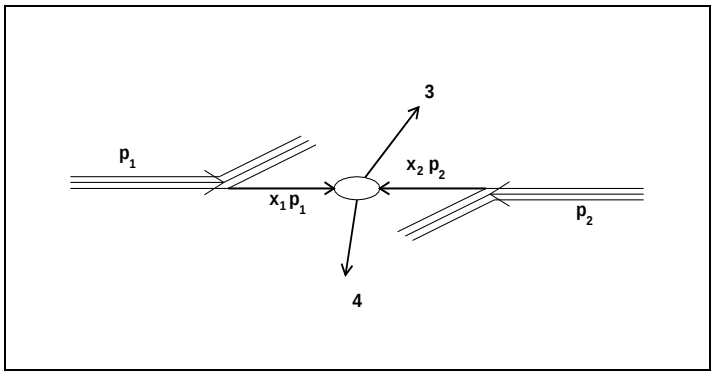

Figure 3.2: Sketch of an inelastic proton-proton interaction. It shows two incoming protons with the momenta p_1 and p_2, and two interacting partons with the momenta x_1p_1 and x_2p_2. Additionally two outgoing secondary particles, 3 and 4 and the rest of the protons is shown.

and propagate, decay or hadronize, respectively.

The distribution of the momentum fraction x_i a parton carries in a proton is given by so-called parton-density functions: $pdf(x_i)$. The pdfs are needed to make predictions on the cross-section for processes anticipated for the LHC. Therefore a good knowledge of the parton-density functions is mandatory for the design of the LHC and the LHC experiments. Parton distribution functions depend on the momentum transfer and have been measured by HERA and former fixed target experiments. Current data covers a kinematic area starting at low $x \approx 10^-5$ and low momentum transfer $Q^2 \approx 1\,\mathrm{GeV}^2$ up to high $x \approx 1$ at high $Q^2 \approx 4 \cdot 10^3$. Normally this is not sufficient in order to cover the dynamic range of LHC, but the distributions can be evolved to higher values of Q^2 using the so-called DGLAP[1] evolution. This allows the computation of cross-sections as shown in Fig. 3.4. Sample pdfs are shown in Fig. 3.3

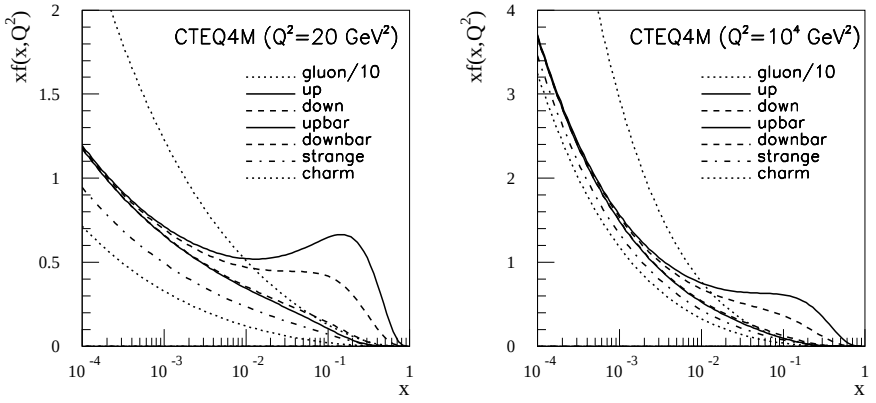

Figure 3.3: Parton density functions for protons according to the CTEQ4M distribution [14].

[1]DGLAP: Dokshitzer-Gribov-Lipatov-Altarelli-Parisi

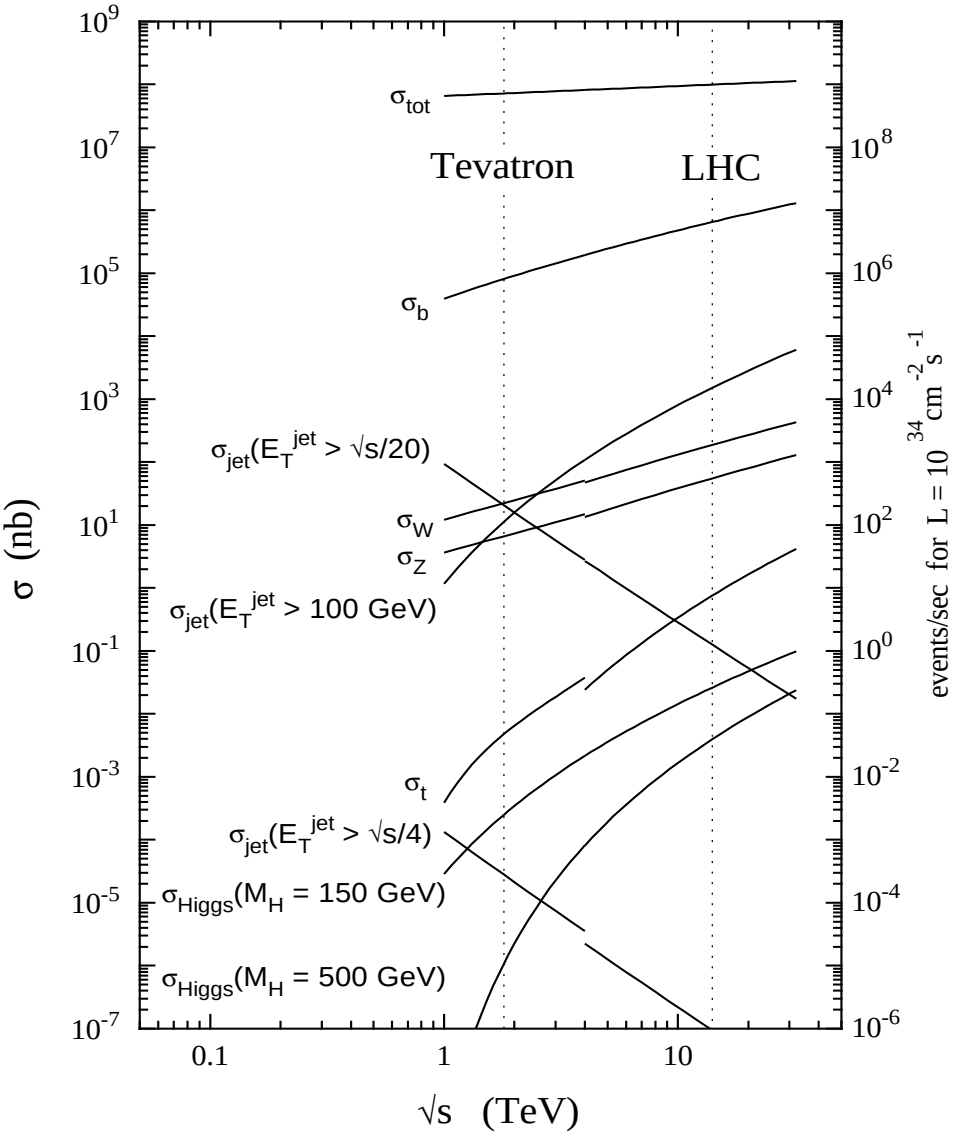

Figure 3.4: Cross-section of relevant processes at the Tevatron and LHC [7].

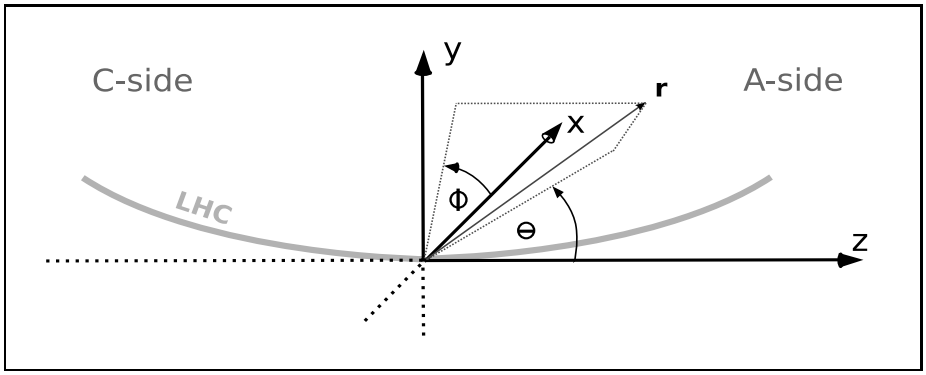

Figure 3.5: Coordinate system used in ATLAS.

A concept often used in relativistic kinematics is the concept of rapidity y. Rapidity is a parameter that describes how the four-momentum p_μ of a particle changes when it is boosted along a direction. In the coordinate system used in ATLAS the z-axis is in beam direction (Fig. 3.5). Therefore let's consider a particle that is boosted along the z-direction. One finds:

$$\begin{pmatrix} E' \\ p'_3 \end{pmatrix} = \begin{pmatrix} \cosh y & \sinh y \\ \sinh y & \cosh y \end{pmatrix} \begin{pmatrix} E \\ p_3 \end{pmatrix} \tag{3.1}$$

The p_1 and p_2 component are unaffected. As differences Δy are Lorentz-invariant, the rapidity is additive under successive boosts. Starting from the center of mass (cms) frame of the particle ($E = M, \mathbf{p} = \mathbf{0}$), one can obtain p_μ in any frame by boosting:

$$\begin{pmatrix} E \\ p_3 \end{pmatrix} = \begin{pmatrix} M \cosh y \\ M \sinh y \end{pmatrix} \tag{3.2}$$

On the other hand one can compute y, from a particle's momentum:

$$y = \frac{1}{2} \ln \frac{E + p_3}{E - p_3} \tag{3.3}$$

Fig 3.6 gives an overview of the kinematics needed to create a particle with mass M and rapidity y by parton-parton fusion as described in Appendix A. As both protons have a momentum of 7 TeV the choice of M and y fixes the momentum fractions x_1 and x_2 for the two partons involved. The corresponding formula is: $x_{1,2} = (M/14\,\mathrm{TeV}) \exp(\pm y)$. The more massive the particle of interest is, the higher the fractions x_i must become. This is why the potential y-range decreases with the mass of the particle.

When a particles momentum is much higher than its mass, the mass contribution to it's energy becomes negligible: $E \approx p$. Using this approximation (3.3) becomes:

$$y \approx \eta \equiv \frac{1}{2} \ln \frac{p + p_3}{p - p_3} \tag{3.4}$$

which defines the so-called pseudo-rapidity η. Since in spherical coordinates p_z is $p \cos\theta$, this can be written as:

$$\begin{aligned} \eta = & \frac{1}{2} \ln \frac{p(1 + \cos\theta)}{p(1 - \cos\theta)} \\ \Leftrightarrow \quad \eta = & -\ln \tan \frac{\theta}{2} \end{aligned} \tag{3.5}$$

The relation (3.5) gives a correlation between the geometrical parameter θ and the kinematic parameter η. Therefore η becomes important for the description of the detector geometry. Pseudo-rapidity distributions for the production of charged particles can be generated by means of Monte Carlo methods as shown in Fig. 3.7. The plateau for $|\eta| \leq 4$ in Fig. 3.7 a) exceeds the θ-range covered by the ATLAS-detector. It is because of this plateau that designing the detector-cells equidistant in η, results in approximately the same occupancy of cells with respect to θ.

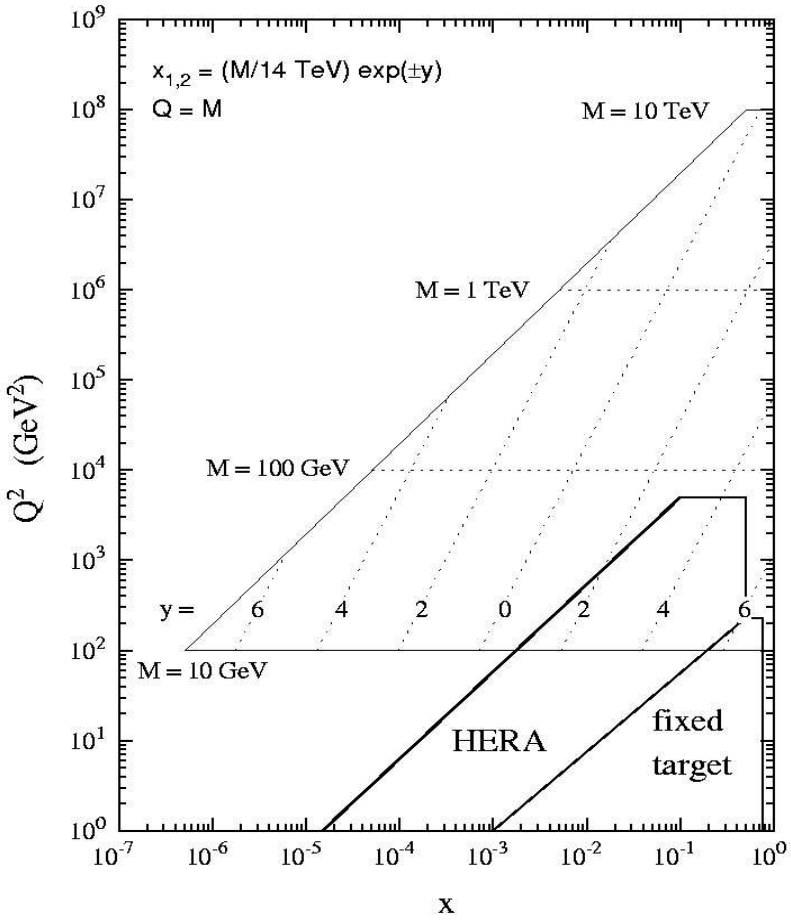

Figure 3.6: Kinematic plane at HERA, fixed target experiments and the LHC [9]. The dashed lines show at which rapidity y a particle of a given mass M is produced if one of the partons carries the momentum fraction x.

3.3 Detector

The ATLAS detector is located in UX15, a pit of about 100 m depth. It is designed as a general purpose detector dedicated to the discovery of the Higgs boson and the search for physics beyond the Standard Model. Table 3.1 shows requirements for some of the searches addressed by ATLAS. Similar to most modern accelerator-detectors it is designed in a general three-layer structure with a magnet system. Fig. 3.8 gives an overview. The inner detector is a high-resolution tracking system. It is enclosed by a solenoid magnet system which generates a magnetic field of approximately 2 Tesla and allows transverse momentum measurements for charged particles. The second layer contains the electromagnetic and hadronic calorimeters which are embedded in the third layer: the muon-spectrometer. The muon spectrometer uses the magnetic field generated by the toroid magnet-system and dominates the extensions of the ATLAS detector.

The high luminosity of the LHC results in an average of $\sim$ 23 proton-proton interactions per bunch collision. The need to separate these overlaying events poses high demands on the tracking system and the granularity of the detector. As LHC

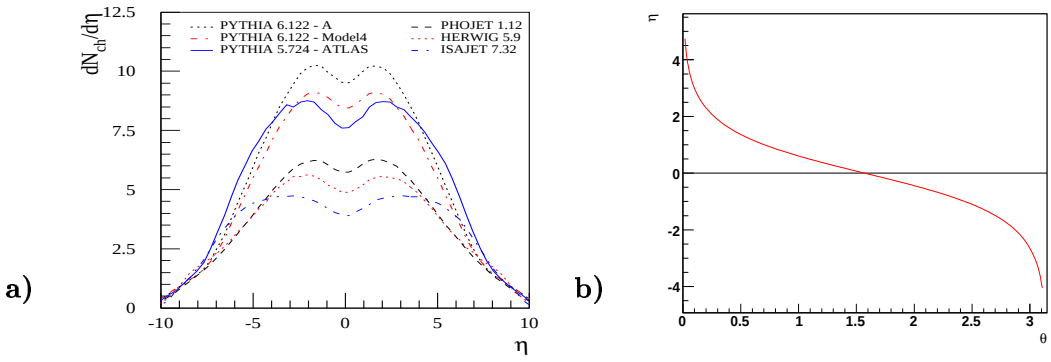

Figure 3.7: η distribution for inclusive charged particles production for different Monte Carlo generators [5] and θ-η relation. The requirements come from the dominant decay modes the particles of interest have

Search		**Detector requirements**
Higgs	$106 < m_H/\text{GeV} < 130$: $H \to \gamma\gamma$	Reconstruction of photon energy and angle, reduction of $\gamma\pi^0$, Jet-Jet and Jet-γ.
	$130 < m_H/\text{GeV} < 600$: $H \to ZZ^* \to llll$	μ and e identification, lepton energy reconstruction, μ-momentum measurement.
	$600 < m_H/\text{GeV} < 1000$: $H \to WW \to l\nu$ Jet-Jet	Jet reconstruction
Supersymmetry		E_T^{MISS}-measurement, b-tagging, hermeticity

Table 3.1: Detector requirements for different searches.

is a proton-proton collider, the energy of the parton-parton interaction has to be reconstructed. Energy reconstruction and E_T^{MISS}-measurements are major challenges for the ATLAS calorimetry. Detailed physics studies have been taken into account when defining the basic design of the ATLAS-detector.

Inner Detector

The two inner detector measures data needed to reconstruct tracks, vertices and transverse momenta of charged particles. It consists of three sub-detectors. The innermost, the Pixel Detector and the Semiconductor Tracker, are based on semiconductor technology. The third one, the Transition Radiation Tracker, is a "straw" detector measuring Transition Radiation (TR). The Transition Radiation is generated by polyethylene-polypropylene stacks or polyethylene foils in the barrel or end-caps, respectively. The intensity of the TR is increasing with the Lorentz factor γ. As for highly relativistic particles $\gamma \approx \beta\gamma = p/m$ the Transition Radiation Tracker can be used to distinguish particles with different mass, once their momentum is known; the Transition Radiation Tracker is used for particle identification. With $\sim$ 147 million channels altogether, 140 million of which come from the Pixel Detector, the Inner Detector delivers by far most of the ATLAS readout data; it is, however, not used by the ATLAS trigger. The Inner Detector has a diameter of 2.3 m, a length of 7 m and covers the pseudorapidity range $|\eta| < 2.5$.

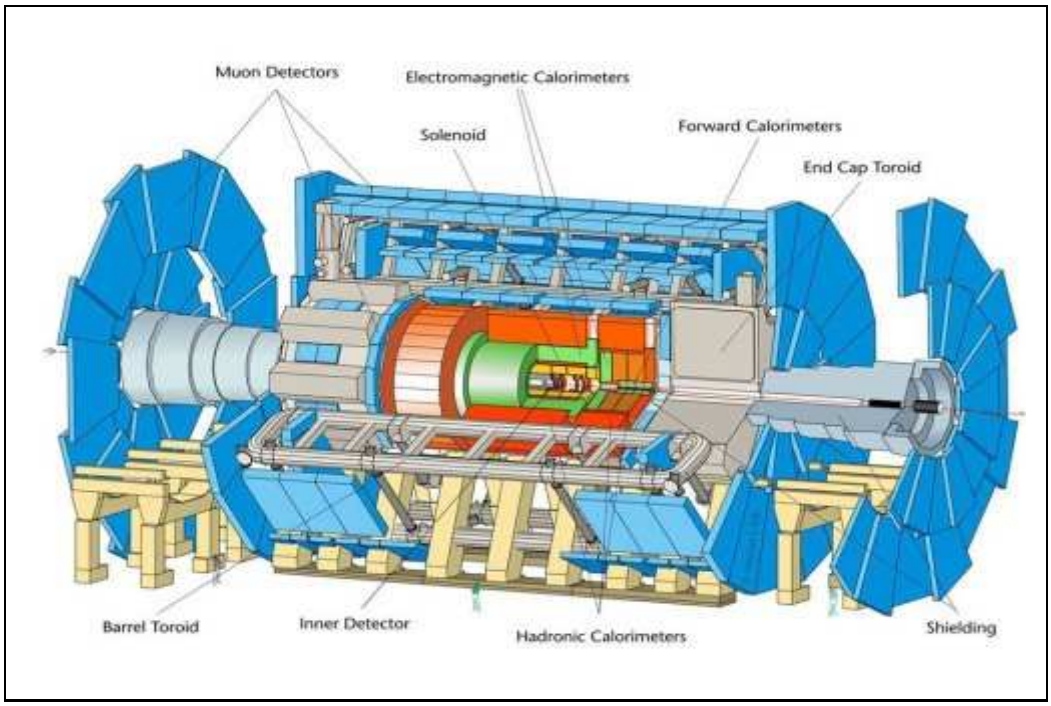

Figure 3.8: Overview of the ATLAS Detector [5].

Calorimeters

The ATLAS calorimeter is designed to reconstruct the energy of high p_T particles coming the interaction point with a good coverage in η for hermeticity's sake. It consists of an inner, electromagnetic (EM), and an outer hadronic layer dedicated to electromagnetically and strong interacting particles, respectively. The energy resolution of the electromagnetic calorimeter has been measured with testbeams. In case of the electromagnetic calorimeter the result was approximately $11\,\%/\sqrt{\mathrm{E}} \oplus 0.26\,\%$ and in case of the hadronic calorimeter the acievement of the ATLAS requirenment of $50\,\%/\sqrt{\mathrm{E}} \oplus 3\,\%$ could be confirmed; E is the measured energy in GeV. In terms of detector geometry, both the electromagnetic and the hadronic calorimeter are divided into three sub-systems. In the low η range $|\eta| < 1.4$ for the electromagnetic and $|\eta| < 1.7$ for the hadronic part the calorimeter is called "barrel" calorimeter. In the η range $1.4 \leq |\eta| < 3.2$ for the electromagnetic and $1.5 \leq |\eta| < 3.2$ for the hadronic part it is referred to as "end-cap" and in the η range $3.2 \leq |\eta| < 4.9$ as "forward" calorimeter.

Two different sampling calorimeter techniques have been chosen for different parts. The hadronic barrel calorimeter is an iron-scintillator calorimeter. Because scintillator and iron are arranged like tiles, it is commonly referred to as "Tile" Calorimeter. It consists of one central and two extended barrels as illustrated in Fig. 3.9. The electromagnetic barrel, the hadronic end-cap and forward calorimeter, where higher radiation hardness is required, have been realized using the Liquid-Argon technology. The different parts of the Liquid Argon system differ in the chosen absorber material and geometry. Table 3.2 gives an overview of the ATLAS calorimeter components. One finds that the granularity is decreasing with increasing

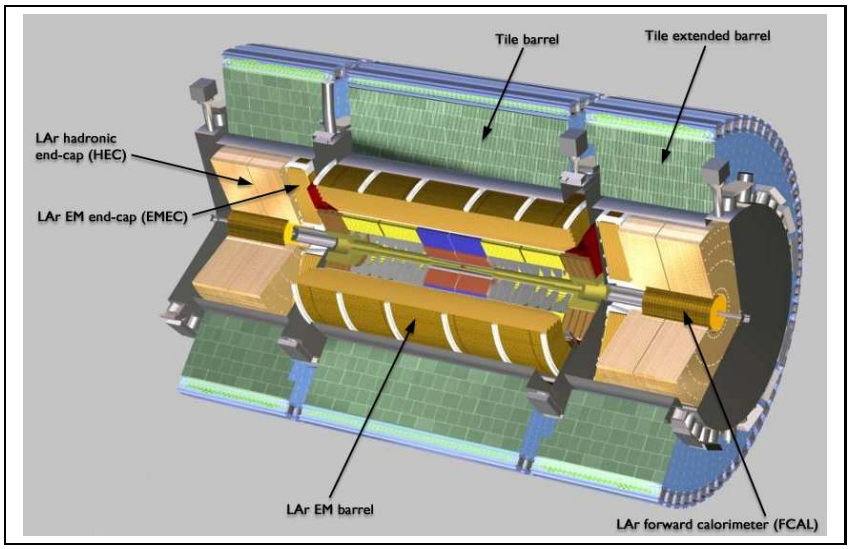

Figure 3.9: Calorimeter overview [13].

η, even though it was mentioned in Chapter 3.2 that it would be desirable to arrange the detector cells equidistant in η. The reason for the increasing granularity is the η-θ relation which is illustrated in Fig. 3.7 b). Cells have to become very small, in order to keep them equidistant in η for the very high $|\eta|$-regime. Therefore a trade-off between technical effort and physical requests is made.

	EM barrel	**EM end-cap**	**Had end-cap**
Size: $(\varnothing \times \mathrm{l})/\mathrm{m}^2$	4.5×6.8	$4.5 \times (2 \times 3.17)$	
η-range: $\lvert\eta\rvert$	< 1.475	$1.4 - 3.2$	$1.5 - 3.2$
Type	LAr Accordion		LAr Plate
Nr. of channels	$\sim 220\ 000$	$\sim 2 \times 63\ 700$	$\sim 2 \times 2\ 200$
Granularity/$\sqrt{\Delta\eta\ \Delta\phi}$	~ 0.025	$0.025 - 0.1$	$0.1 - 0.2$
Absorber/($Mat.$: mm)	Pb: $1.5 - 1.1$	Pb: 2.2, 1.7	Cu: 25, 50
LAr-gap/mm	~ 2.1	$0.9 - 3.1$	1.95
	Forward		**Tile**
Size: $(\varnothing \times \mathrm{l})/\mathrm{m}^2$	$0.9 \times (2 \times 0.45)$		$8.4 \times (5.6 + 2 \times 2.9)$
η-range: $\lvert\eta\rvert$	$3.2 - 4.9$		< 1.7
Type	LAr rods/matrix		Iron Scintillator
Nr. of channels	$2 \times 1\ 411$		$5980 + 2 \times 1820 + 390$
Granularity/$\sqrt{\Delta\eta\ \Delta\phi}$	0.2		0.025-0.2
Absorber/($Mat.$: mm)	Cu/W : $\varnothing$ rods=4.5 mm		-
LAr-gap/mm	$0.25 - 0.5$		-

Table 3.2: Technical calorimeter overview. Details in the text

The electromagnetic barrel and end-cap calorimeter have accordion-shaped Pb absorber plates with three layer Cu-Kapton readout electrodes that are separated from the absorber by placeholder layers, the "honeycomb" layers. To improve the energy-resolution, the lead thickness decreases with η in the electromagnetic barrel. The electromagnetic end-caps are composed of an inner and an outer wheel, with

absorber material thicker in the inner wheel than in the outer one. The Liquid-Argon (LAr) gaps of the electromagnetic end-caps are getting wider from inside to outside.

The hadronic end-caps have a conventional plate design with readout electrodes made of Cu-Kapton boards covered with a high resistive layer. Readout electrodes and absorber are separated by honeycomb layers. The hadronic end-cap calorimeters are composed of a front and a back wheel. The absorber material has 25 mm in the front wheel and 50 mm in the back one.

The forward calorimeter covers the high η-range, ensuring hermeticity and allowing to measure jets in the very forward direction. Due to its location in the detector it is exposed to an annual flux of 10^{16} neutrons cm^{-2} and a dose of $2 \cdot 10^6$ Gy. It consists of three modules, the electromagnetic FCAL1, and the hadronic FCAL2 and FCAL3 on either side of the detector. The modules are arranged along the beam line and consist of an absorber matrix carrying tube electrodes. The tubes are arranged in a hexagonal grid, and oriented parallel to the beam line. Electrode rods are centered in the tubes by means of a spiral of radiation-hard plastic (Polyetheretherketones or PEEK). The PEEK is wound onto the rods ensuring a very small LAr-gap that provides the ionization region. Due to the thermal load of ~ 100 W, rods and matrix are made of copper in the FCAL1, whereas tungsten has been chosen for FCAL2 and FCAL3. All modules have a length of 451.5 mm and an outer radius of 456.4 mm. The thickness of the LAr-gap is increasing from 0.25 mm in FCAL1 up to 0.4 mm in FCAL3.

The Tile Calorimeter consists basically of one central and two extended barrels. The gaps between central and extended barrels are needed to route cables to the inner detector components. Parts of the gap are covered by the Intermediate Tile Calorimeter. The Tile Calorimeter barrels and extended barrels are composed out of 64 modules in ϕ. One module is made out of trapezoidal wave length shifting scintillating Tiles, alternating with iron spacers mounted on iron carriers. Double clad wavelength shifting fibers are routed to photomultiplier tubes (PMT) that are located on top the modules. Readout cells are defined by grouping sets of fibers to one PMT.

Muon Detector

In the range $|\eta| < 4.9$ the total thickness of active calorimeters is not significantly smaller than 10 absorption-lengths. Beside very high energetic jets the only particles that can pass this material leaving tracks in all detector components are muons. Muon spectroscopy is performed based on magnetic deflection of the muon tracks. The required magnetic field is generated by a dedicated Toroid Magnet, Fig. 3.8. All muon chambers are gas detectors. There are trigger and precision chambers. Both subsystems measure each track in at least three points for curvature, i.e. momentum, estimation.

Most of the precision chambers are so-called Monitored Drift Tubes (MDT) that consist of two times $3-4$ monolayers of drift tubes. In the forward region however, where finer granularity is required, multi-wire proportional chambers with cathode strip readout, Cathode Strip Chambers (CSC), are used. The muon precision chambers are arranged such that the precision of the measurement in the track's bending-

plane is optimized. The whole of the muon precision chambers is also referred to as Muon Spectrometer and covers the range: $|\eta| < 2.7$.

The main purpose of the muon trigger chambers is Bunch Crossing Identification (BCID), i.e. to tag the clock cycle associated with an event for high granularity detector-readout. The trigger chambers are arranged such that the measurement of the coordinate orthogonal to the one measured by the precision chambers is optimized. Resistive Plate Chambers (RPC) and Thin Gap Chambers (TGC) are used for the trigger system in the barrel and end-cap region, respectively.

3.4 Calorimeter Front End Electronics

The electronics that is located on the detector are called Front End Electronics (FE). The FE is differentiated from the Back End Electronics that are situated in caverns close to the detector pit. All ATLAS sub-detectors buffer the event data in their FE until the trigger has made the decision whether to further address an event or not. This allows to reduce significantly the bandwidth from the detector and the Back End electronics. The ATLAS calorimeter-FE is different for the two adopted techniques, iron-scintillator and liquid-argon ionization. Both the FE used by the LAr detectors and the one used by the Tile Calorimeter allow to inject calibration pulses on a cell level.

The Calorimeter Trigger is working on so-called Trigger Towers (TT), each an aggregation of detector cells (Section 5.1). Summing readout cell information to build TT signals is one of the tasks addressed by the calorimeter's FE.

LAr Calorimeters

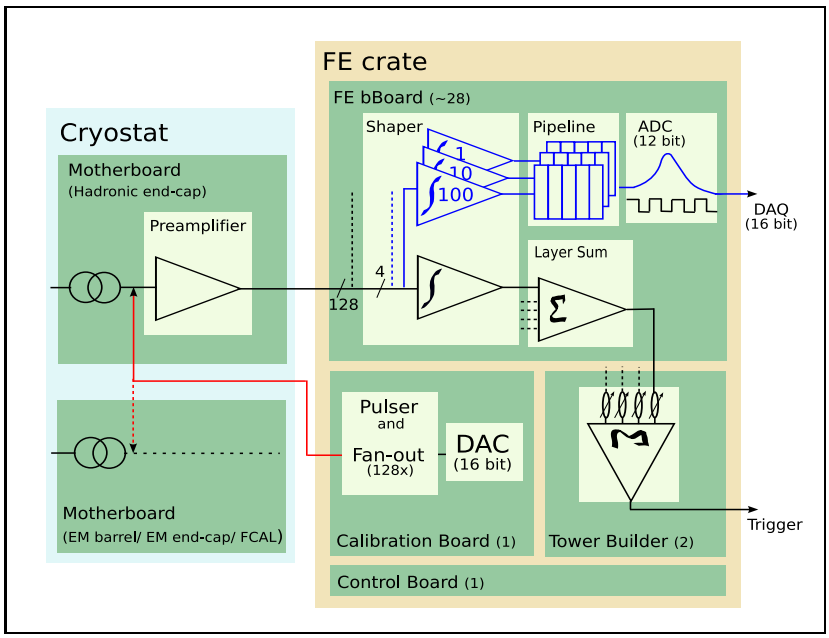

Figure 3.10: Overview of the Liquid Argon Front End electronics. In case of the EM barrel, EM end-cap and the FCAL the preamplifiers are located on the FE board.

Fig. 3.10 gives an overview of the LAr Front End Electronics. The cell signal is received on a so-called Motherboard in the cryostat, is galvanically separated from the detector and routed to the Front End crates that are installed at the detector. Each FE crate holds about 28 Front End Boards (FEB), two Tower Builder Boards (TBB), one Calibration Board (CB) and one Control Board.

Preamplifiers are required to raise the cell-signal above the noise of the downstream stages. "Cold" preamplifiers, i.e. preamplifiers that are installed in the cryostat, have the advantage to get the signal right from the electrode and reduce pick-up noise and cross-talk. Therefore cold amplifiers are used in the hadronic end-cap, installed on the Motherboards. In the FCAL, the EM end-cap the high radiation background is the reason for not having the preamplifiers in the cryostat. They have therefore been installed on the FE boards in the FE crates, an area with reduced radiation. In case of the EM barrel the preamplifiers are installed in the FE crates in order to avoid losing accessibility and to minimize the dead material in the cryostat.

The preamplifiers outputs are routed to the Shaper chips. Each Shaper chip processes four signals and transforms them to match the 40 MHz sampling frequency. The Shapers are followed by an analog pipeline and a 12 bit Analog to Digital Converter (ADC) digitizing requested data. For the (Data AcQuisition) DAQ path the desired dynamic range is 16 bits. It is, however, considered impossible to provide this dynamic range in the pipeline-ADC chain. In order to cover the full 16 bit dynamic range the Shaper produces three output signals differing in the gain factors: 1, 10 and 100. Using the 12 bit ADC a gain selection mechanism assures a final 16 bit dynamic range by selecting the proper Shaper output.

For the trigger path, signals from different cells have to be summed. The summation starts in the Shaper chips where all four channels are summed and amplified. This is done in order to minimize the effect of downstream noise and generates an extra output dedicated to the trigger path. Channels from one layer of the detector are summed in a Layer Sum plug-in board on the FEB using the Shaper output. The result of half a crate is linked to one of the two Tower Builder Boards where the final Trigger Tower signals are built. The TTB allows individual adjustment of gain, shape and timing before summing its inputs. The analog output of the TTB is used by the Level-1 Calorimeter Trigger.

Fig. 3.11 a) shows the detector signal and the Shaper result. The Shaper output is bipolar showing a fast peak with an area that is proportional to the measured energy followed by an undershoot. The Calibration Board allows to generate a signal of the shape $I(t) = I_0 e^{(t-t_0)/\tau}$, where I_0 is programmable via a Digital to Analog Converter (DAC), and τ is fixed by the parameters of the pulse generating circuit. The signal is injected on the Motherboard, before the preamplifiers. Fig. 3.11 c) shows a comparison of the Shaper output for a physics pulse and a pulse generated by the Calibration Board.

Tile Calorimeter

The Tile Calorimeter FE are located on drawers in girders on top of the modules. The drawers can be extracted from the girders in order to grant access to the FE.

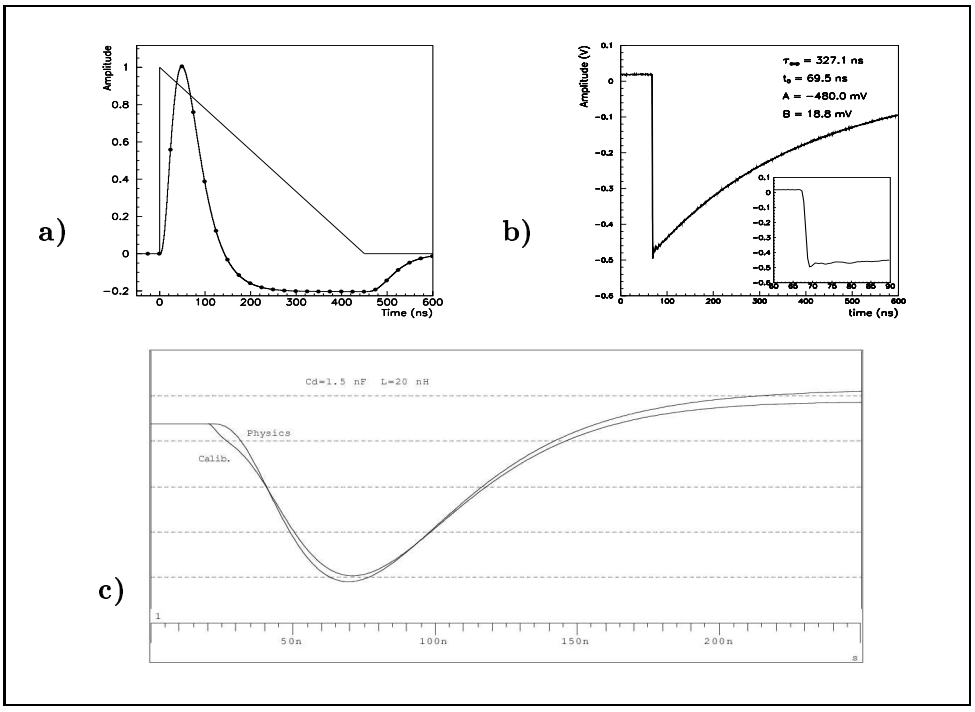

Figure 3.11: a) Signal shape from the detector and result of the Shaper (graph with dots), b) Calibration Board output output for a 500 mV DAC setting and c) comparison between physics and calibration signal from simulation [8]. Note that b) and c) have polarity different from a).

Two drawers build an electrical unit: the Super-drawer. There is one Super-drawer in the extended barrel modules, the barrel modules contain two Super-drawes.

The FE that are dedicated to an individual calorimeter channel is shown in Fig. 3.12. Optical fibers are grouped to cells and routed to a PMT (Photo Multiplier Tube) Block. Up to 24 PMT Blocks are installed in one drawer. In the PMT Block the light passes a Mixer and is then captured by a PMT. There are variations in the response of the Photomultiplier cathode w.r.t. the location of its surface. Therefore the Mixer decorrelates the position of a fiber and the location on the Photomultiplier that receives its light. The PMT output is fed into a 3-in-1 board which has the following tasks:

- Integrate the signal in order to measure the current from minimum bias events.
- Shape PMT output into a unipolar signal.
- Inject charges into the signal chain using the Charge Injection System (CIS) in order to generate Calibration pulses.

The shaping of the signal is done by a Shaper circuit. The output of the shaper is duplicated and amplified with two different gains that have a ratio of 1 to 64.

The drawers provide a three layer structure that is illustrated in Fig. 3.13. The first layer consists of Motherboards that carry various "Mezzanine" daughter boards for Control, the Trigger and an Integrator. The second layer consists of the

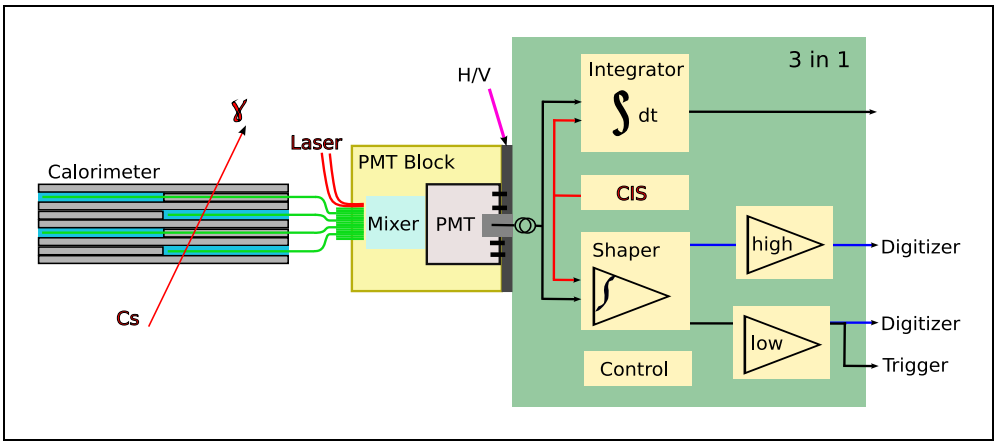

Figure 3.12: Overview of the Tile Calorimeter Front End electronics up to the 3-in-1 board. Three calibration systems are illustrated in red.

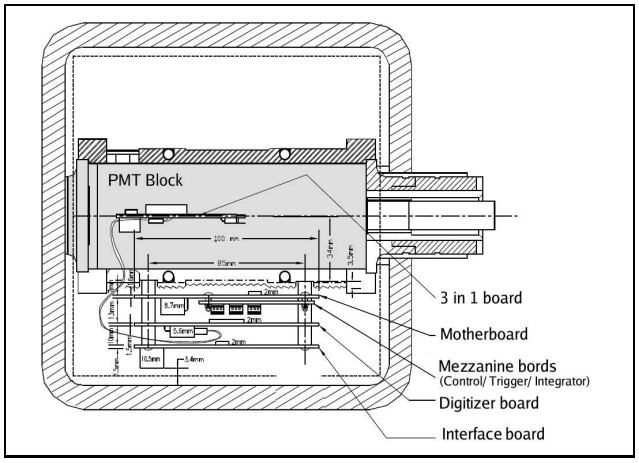

Figure 3.13: Cross section of a drawer [12].

"Digitizer" boards to which the low and the high gain signal are routed. On the Digitizer boards both signals are digitized using 10 bit ADCs. The digital data are pipelined for readout. A selection mechanism chooses which gain to use for a given pulse, assuring the required dynamic range 16 bits. A dynamic range of 16 bits is necessary in order to cover minimum energy deposits from muons and maximum energies up to 2 TeV from the very high energetic jets. The optical readout of the Digitizer boards is realized by means of "Interface" boards, that are installed in the third layer of the three layer structure.

The trigger path is separated from the readout path at the level of the low gain amplifiers. The low gain signal is duplicated and routed to dedicated Trigger Summation cards, installed on the Motherboards. The Trigger Summation card performs an analog sum of the low gain signals belonging to one Trigger Tower. The summation result is sent to the end of the drawer and from there to the Trigger cavern.

The Tile Calorimeter provides three calibration systems, that will be introduced now in the order they induce signals into the chain.

The Caesium Cs system allows to bring three movable 0.0662 MeV ^{137}Cs γ-sources, one in the barrel and two in the extended barrels, into the detector. The source capsules are inside a tube system and moved to individual modules pumping a driving liquid through the tubes. The Cs-system allows to check the optical response of the scintillating tiles and to equalize it adjusting the High Voltage (HV) of the PMTs.

A Laser system can be used to inject light pulses into the PMT block. The laser beam is created in the electronics cavern and routed to the Barrel modules by ~ 100 m and to the extended barrel by ~ 110 m clear fibers. From the modules the fibers are fanned out to the PMT blocks of a module using a 1-to-50 fiber connector. The Laser system fibers are bundled together with the readout fibers from the detector.

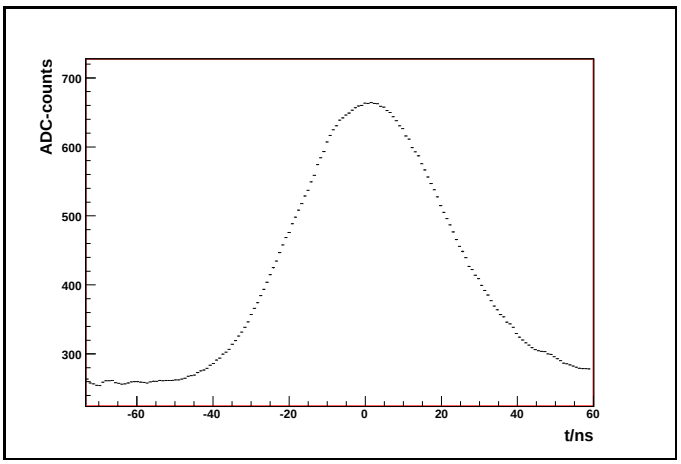

Figure 3.14: Tile Calorimeter pulse generated with the Charge Injection System (CIS) using 64 pC. The signal was sampled with the Tile Calorimeter Readout system.

A purely electrical approach is followed by the Charge Injection System (CIS). A charge is loaded on capacitors by means of DACs on the 3 in 1 boards. It is then induced into the Shaper and the Integrator. Fig. 3.14 shows a typical CIS pulse as read out by the Tile Calorimeter Readout system, Fig. 3.15 gives an overview the CIS input and output of a 3 in 1 board.

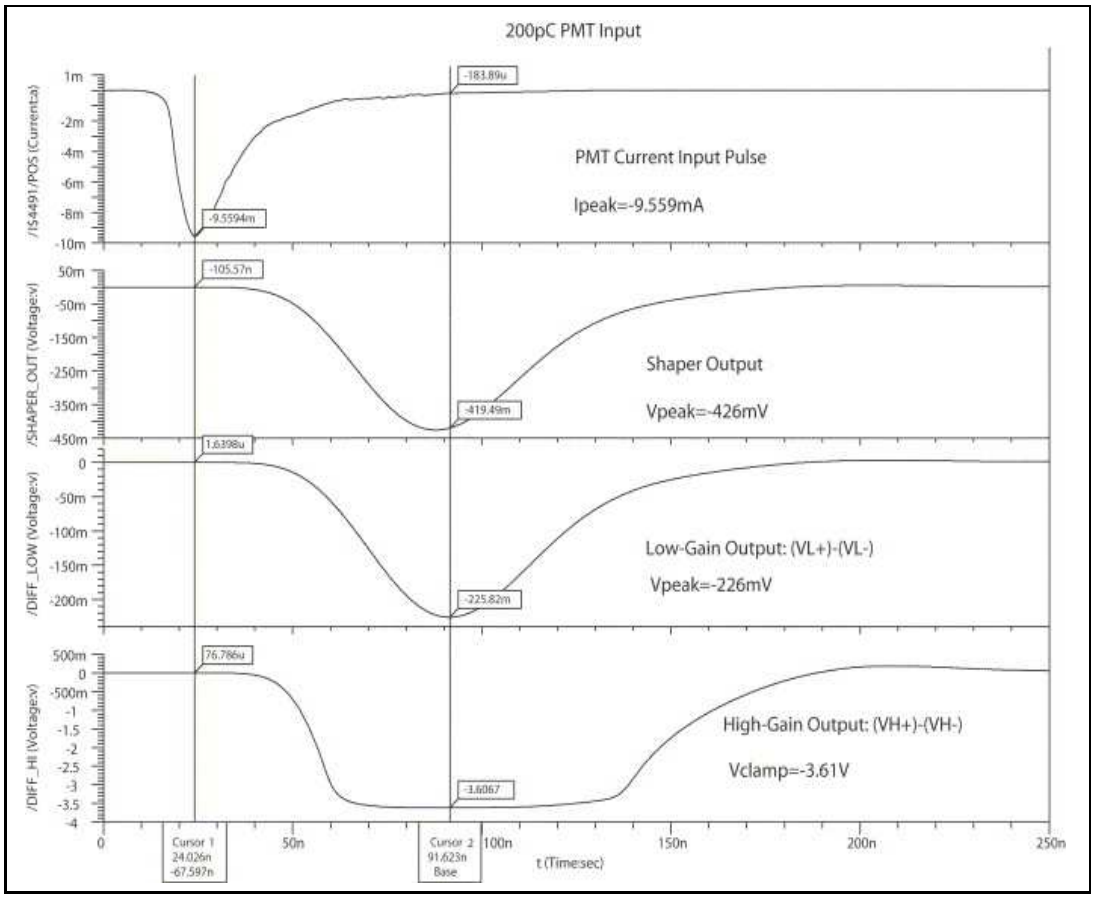

Figure 3.15: CIS input and output of the Shaper, the low-gain amplifier and the high-gain amplifier [23].

Chapter 4

The ATLAS Trigger-DAQ System

4.1 The Trigger Challenge

In order to understand the constraints on the ATLAS-trigger performance it is useful to do some estimations. The cross section for the production of Higgs bosons with a mass of 150 GeV in proton-proton collisions at $\sqrt{s} = 14\,\text{TeV}$ is small: $\sigma_H \approx 10^{-1}\,\text{nb}$. Thus the final LHC luminosity has been designed to be very high: $\mathcal{L} = 10^{34}\,\text{cm}^{-2}\text{s}^{-1}$, so satisfactory statistics for the Higgs discovery can be achieved. With this luminosity the rate for Higgs production becomes: $r_H = \mathcal{L}\,\sigma_H \approx 1\,\text{Hz}$, still assuming a mass m_H of 150 GeV. As one can extract from Fig. 3.4 the total cross section σ_{tot} at 14 TeV is about 70 mb. This gives a rate of $r = 7 \cdot 10^8\,\text{Hz}$, almost 1 GHz for minimum bias events. Thus, the trigger must be very efficient concerning the capture of interesting events.

With one bunch-crossing every 25 ns there are 23 overlapping events per bunch-crossing. As already mentioned in the previous Chapter a high granularity detector is needed in order to separate these events and minimize pile-up effects. High detector granularity, however, causes a considerable event size. The overall ATLAS event size is about 1.5 MB. Clearly this amount of data cannot be recorded at 40 MHz which would result in about 60 TB/s. In fact the affordable rate for mass storage is about 200 Hz or 3 PB/a. This means that the trigger has to reject 99.9995 % of the events, while taking care to keep the desired rare ones.

4.2 Overview of the Trigger-DAQ System

The event data are stored in electronic pipeline memories for 2.5 μs. This is the time available for the trigger to make a decision, whether to read out the detector or not. In order to manage this exercise the trigger is designed in three levels. Fig. 4.1 gives an overview of the ATLAS Trigger-DAQ system, showing the trigger-path on the left and the DAQ-path on the right side.

The Level-1 Trigger reduces the event rate from ~ 1 GHz to 75 kHz within 2.5 μs. In order to minimize the latency it is installed in USA15, a cavern right next to the pit UX15, in a radiation-safe environment. The event data that are accepted by

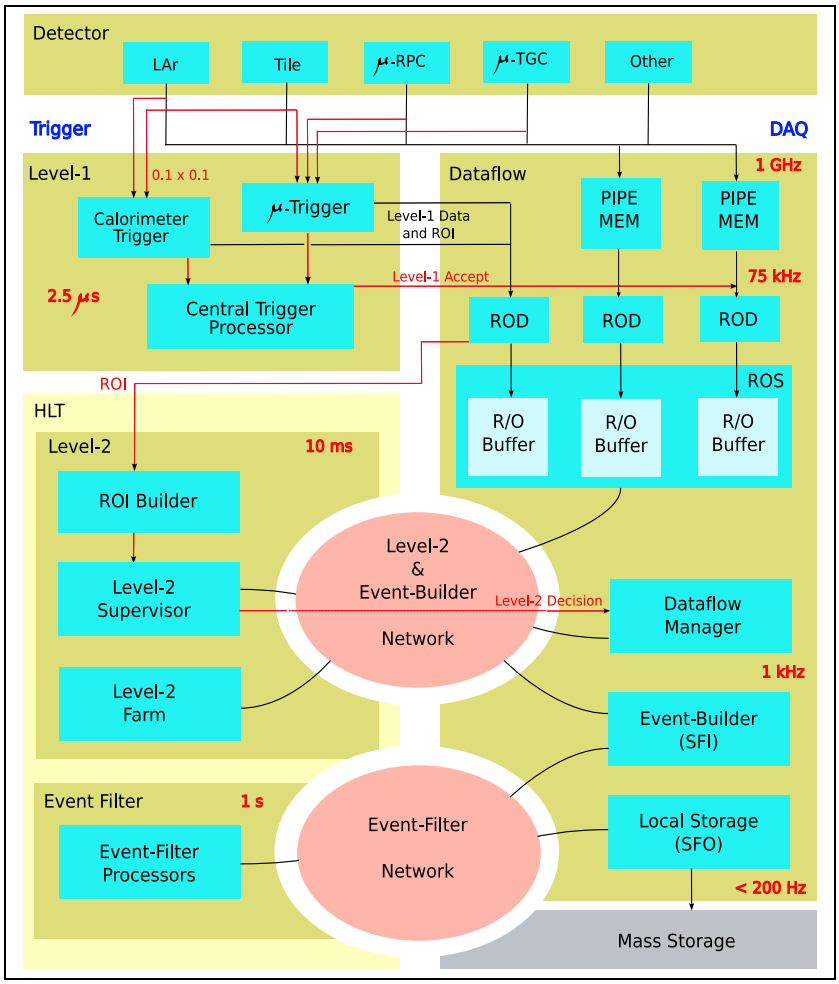

Figure 4.1: Overview of the ATLAS Trigger-DAQ system

Level-1 are buffered in so-called Readout Buffers (ROB) until the Level-2 decision is taken. The Level-2 Trigger reduces the rate from 75 kHz to 1 kHz in 10 ms, which is the time-span, the the data are stored in the ROBs. If the Level-2 decision is positive the full ATLAS event is built, but not yet stored. The reconstructed ATLAS event is analyzed by the Event Filter (EF) which represents the third Trigger-Level. The Event Filter reduces the rate down to the storage rate of 200 Hz in about one second per event. The Level-2 Trigger and the Event Filter are mainly software-based and also referred to as High Level Triggers (HLT). The HLT is running on computer-farms which are installed in SDX1, in a building on the surface of Point 1.

4.3 Implementation

The Level-1 Trigger is a VMEbus multi-crate system using the data from the Muon-Trigger system and a pre-summed version of the calorimeter data. The Level-1 Trigger consists of three blocks, the Calorimeter Trigger, the Muon Trigger, and the Central Trigger Processor. The Muon Trigger does not only analyze data from the Muon System but also gets a copy of the third and final layer of the Tile Calorime-

ter used in coincidence with track-candidates from the barrel muon chambers for background suppression. Both, the Calorimeter and the Muon Trigger have programmable sets of thresholds. In case of the muon system six thresholds are applied to the transverse momentum p_T of the muon. In case of the Calorimeter Trigger the thresholds are applied by algorithms that run on the so-called transverse energy E_T. E_T is defined as the energy E that has been measured in a Trigger Tower weighted with the geometrical factor $\sin\theta$, where θ is the angle associated with the η-coordinate of the TT (Eqn. 3.5). The algorithms are so-called sliding-window algorithms that inspect the transverse energy measured in a connected subset of Trigger Towers (a window) then slide to to the next subset and inspect that one until the η-range under consideration has been covered. The output of the algorithms are candidates for objects like electrons, photons, hadrons, τ-leptons and jets. Multiplicities, i.e. the number of objects passing the thresholds, are computed for all thresholds and sent to the Central Trigger Processor. Additionally the overall scalar transverse energy E_T and missing transverse energy E_T^{MISS} are determined. In case of the overall energy-algorithms the number of thresholds passed is derived. For accepted events, Regions of Interest (ROI) where candidates for particles or jets have been found are sent to the DAQ-path.

The Central Trigger Processor can be programmed with 256 so-called "Trigger items". A Trigger item is a combination of demands on the multiplicities, e.g. two muons with a $p_T > X$ and a missing transverse energy $E_T^{MISS} > Y$. Once a Trigger item is fulfilled a Level-1 Accept (L1A) signal is generated. The L1A is sent to the pipeline memories mainly located in the detector front-end electronics via the Timing, Trigger and Control (TTC) system. The TTC system connects most of the ATLAS electronics, provides a 40 MHz clock synchronized to the LHC bunch-crossing and the possibility to send simple signals from one system to another. After receiving the Level-1 Accept the pipeline memories send the data of accepted events to dedicated modules, the Readout Drivers (RODs). The RODs combine data from several Pipe-Memories, format it in a specified way, and forward it to the Readout Buffers. The Readout Buffers are PCI-boards hosted in a Readout Subsystem (ROS) where the data were kept until the Level-2 decision is taken. Other, than the ROBs and the ROS that are common for the ATLAS experiment, each sub-group is responsible for designing their own RODs using a specified link (S-Link) to the ROBs.

The Level-2 Trigger uses data from all sub-detectors at full granularity, but only in the Regions of Interest determined by Level-1. The ROIs are provided by the ROI Builder. The ROI Builder is a VMEbus system and the only item of the HLT that is implemented in hardware and situated at USA15. It receives information and multiplicities from various parts of the Level-1 Trigger and combines it all into one single record that is sent to the Level-2 Supervisor. The Level-2 Supervisor passes the ROI information to a process running on the Level-2 farm. This process requests the data of interest from the ROS, processes it and sends the decision whether to accept or reject the event back to the Level-2 Supervisor. The Level-2 Supervisor forwards the decision to the Dataflow Manager. Depending on whether an event has been rejected or accepted, the Dataflow Manager instructs the ROS to delete the event-data, or initializes the event building on ~ 100 dedicated dual-CPU nodes of the Event Filter Network, the Sub Farm Inputs (SFI).

Reconstructed events are analyzed by the Event Filter, involving approximately 1600 dual-CPU nodes. Accepted events are then sent to the CERN computer center for mass-storage via the Sub Farm Outputs (SFO) comprising about 30 dual-CPUs.

Chapter 5

The Level-1 Calorimeter-Trigger

5.1 Input Signals

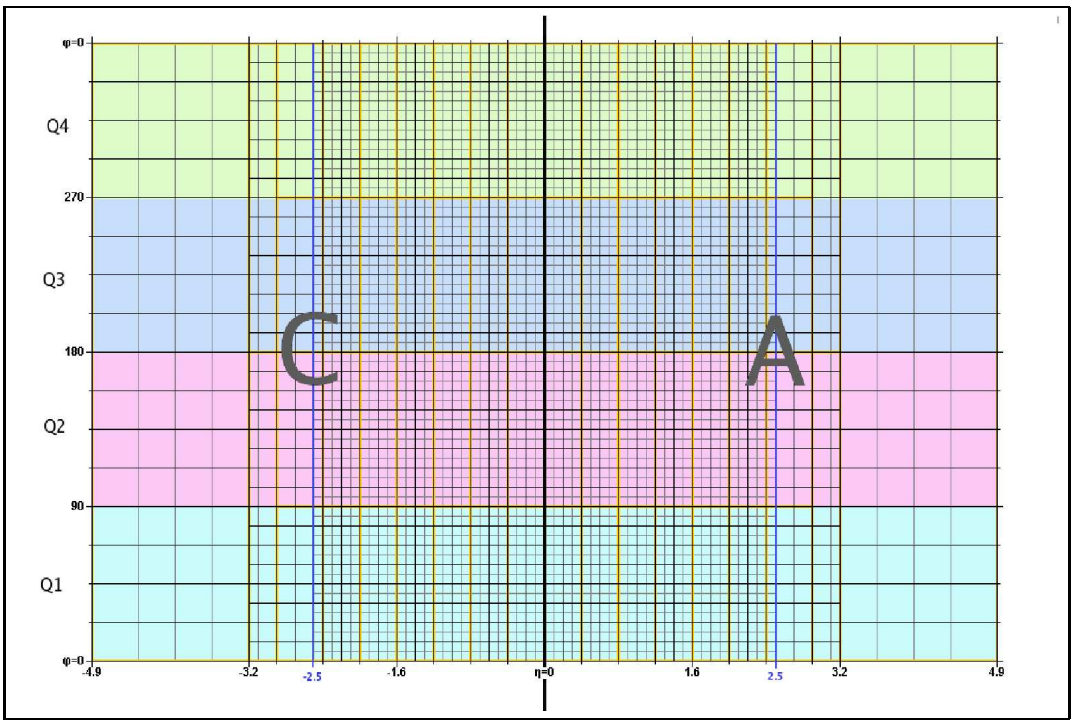

Figure 5.1: Trigger Towers in η-ϕ plane. The orange outlines show which region a PPM is responsible for. The background colors code the four quadrants in ϕ. The same picture holds for the electromagnetic and the hadronic layer.

The Level-1 Calorimeter Trigger (L1Calo) receives $\sim$ 7200 analog signals from the various sub-detectors of the calorimeter. For the purpose of the Level-1 Trigger it is not necessary to process the full calorimeter granularity. Therefore readout cells are summed to so-called Trigger Towers (TT). The summation is generally performed in the calorimeters front end electronics. Only some of the Trigger Towers that span the EM barrel and EM end-cap, are built in USA15. For the low η regime $|\eta| < 2.5$ the Trigger Tower resolution is 0.1×0.1 in $\Delta\eta \times \Delta\phi$. For higher values of η the trigger towers become wider following the decreasing detector granularity. The resolution

is at least 0.2×0.2 until the end of the end-caps ($1.5 \lesssim |\eta| < 3.2$) and 0.4×0.4 for the Forward Calorimeter ($3.2 < |\eta| < 4.9$). With respect to the sign of η, the detector is divided into three parts: A for positive η, B for the plane where $\eta = 0$ and C for negative η.

In the radial direction Trigger Towers are separated in an electromagnetic and a hadronic layer. More precisely this means for a given $\Delta\eta \times \Delta\phi$-region all cells of electromagnetic calorimeters are summed to one Trigger Tower, all cells of the hadronic calorimeters to another one. The ϕ axis in Fig. 5.1 is divided in quadrants as illustrated.

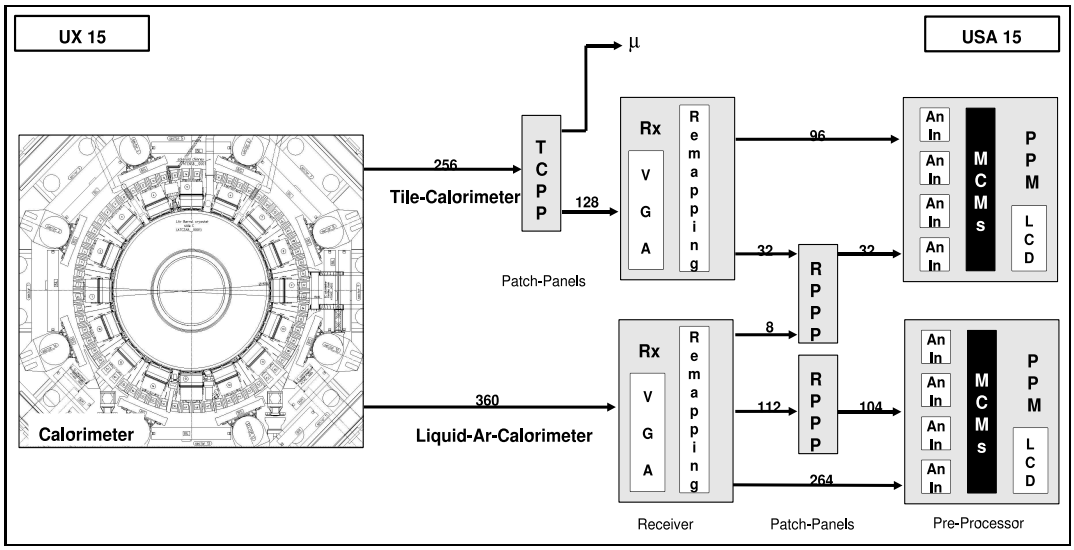

Figure 5.2: Analog cabling from the calorimeters to the Pre-Processor Modules (PPM) of the Level-1 Calorimeter Trigger. The numbers indicate how many cables are needed for the individual tracks.

The differential trigger signals are propagated from the detector in UX15 to the trigger cavern USA15 via $\sim$ 70 m twisted-pair analog cables resulting in a signal attenuation of approximately 40 %. The Tile Calorimeter signals for the Muon Trigger and those for the Calorimeter Trigger are carried over the same cables and separated on dedicated Tile Calorimeter Patch Panels (TCPP). All Calorimeter Trigger channels arrive in so-called Receivers (Rx) which are the last items in the signal chain the detector groups are in charge of. On the Receivers the signals are shaped by analog filters, amplified by Variable Gain Amplifiers (VGA) and channels are remapped using various remapping daughter boards. The Remapping Boards are especially designed for different regions of the detector in order to sort the channels in the way they are expected by the Trigger. On the Calorimeter Trigger side, the signals are received by Pre-Processor Modules (PPM) on Sub D 37 connectors, each providing data from 16 Trigger Towers.

Additional Receiver to Pre-Processor Patch Panels (RPPP) are needed for repacking some of the channels addressing four different tasks:

- The Tile Calorimeter extended barrel covers the range $0.8 \leq |\eta| < 1.7$. In terms of Trigger input, however, all cells for which $|\eta| > 1.4$ are summed into

the $1.4 < |\eta| < 1.5$ Towers. Sorting signals from the Tile Calorimeter, and the hadronic end-cap, in order to bundle the range $1.2 < |\eta| < 1.6$ in one cable is realized using RPPPs.

- The overlap between the EM barrel and the EM end-cap between $1.4 < |\eta| < 1.5$ is summed to one Trigger Tower in the Receiver system. This results in two sets of cables, one with signals from $1.2 < |\eta| < 1.4$ and one with signals from $1.4 < |\eta| < 1.6$. The Pre-Processor (PP) expects input cables, that have signals for the range $1.2 < |\eta| < 1.6$, which are populated by means of RPPPs.
- At the end of the end-caps, $2.4 < |\eta| < 3.2$, the granularity varies from 0.1×0.1 over 0.2×0.2 up to 0.1×0.2 in $\Delta\eta \times \Delta\phi$. To be able to use the same PPMs for the entire η-ranges, cables have to be resorted in that region leaving some of the PPM inputs empty.
- The output of the two hadronic FCALs (FCAL1 and FCAL2) is summed to Trigger Towers in the Receiver system. The result are semi-populated cables that are merged together on RPPPs.

5.2 Overview

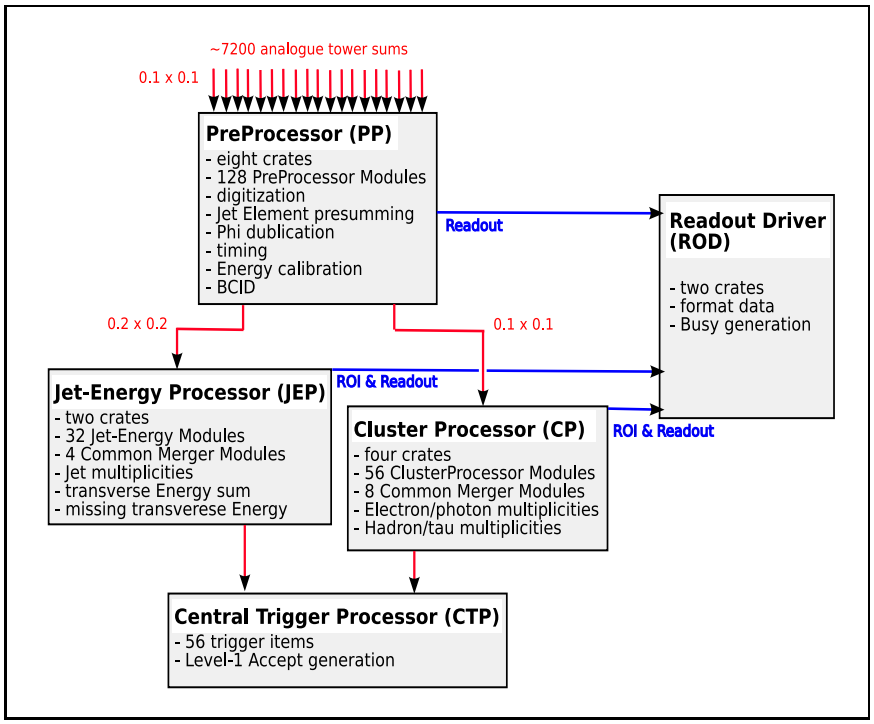

Figure 5.3: Block diagram of the Level-1 Calorimeter Trigger.

The Level-1 Calorimeter Trigger consists of three major processors, the Pre-Processor (PP), the Cluster Processor (CP), the Jet-Energy Processor (JEP) and the Readout Drivers (ROD), Fig 5.3 gives an overview. The whole system is built in hardware, designed as a multi-crate VMEbus system in order to cope with the tight latency constraint of $2.5\,\mu$s. In order to interface the ATLAS TTC system, the

Calorimeter Trigger uses a common Timing Trigger and Control Module (TCM). The TCM is mounted in the rightmost slot (21) of the crate and receives the LHC-clock and broadcast commands via an optical fiber. The optical signal is then decoded and converted into electrical signals on the TCM. Clock and broadcast commands are routed to the other slots in the crate via an auxiliary backplane that is installed on the rear of the VME-backplane on the J0 connectors. On the various L1Calo Modules the electrical TTC signal is received and decoded by common TTC decoder (TTCdec) submodules.

The PP, the largest subsystem of the Calorimeter Trigger, consists of 128 9U Pre-Processor Modules (PPM), in eight crates. Beside 15-16 PPMs, each crate holds a CPU that allows to access the VMEbus and a TCM. The PP has the task to prepare the analog signals for digital processing. Therefore it digitizes the signals to eleven bits and does the timing alignment of all channels with a precision of one nanosecond. Additionally the PP is responsible for Bunch Crossing Identification (BCID), i.e. tagging of the clock-tick that an event is associated with. The transverse energy of each Trigger Tower is estimated by investigating five successive samples and sent to the CP and the JEP via Low Voltage Differential Signal (LVDS) links as an eight bit word. One additional bit is used to encode the BCID-result, and another one for parity protection. The JEP expects a reduced granularity of 0.2×0.2 in $\Delta\eta \times \Delta\phi$. Therefore Trigger Towers are summed in η-ϕ for the JEP-links. The number of CP-links needing the full Trigger Tower granularity is reduced by multiplexing pairs of channels. More details on the PP implementation can be found in the next Chapter.

The two digital processors, CP and JEP, are running sliding-window filter algorithms on the η-ϕ-plane (Fig. 5.1) using the transverse energies estimated by the PP. The results of these algorithms are twofold. On one hand the Level-1 CTP is provided with multiplicities for particle and jet candidates and for the total and missing transverse energy E_T and E_T^{MISS}. On the other hand Regions of Interest (ROI) in which candidates for particles or jets where discovered, are forwarded to Level-2 for further investigation. The CP and the JEP are implemented as a set of custom-made 9U VME-modules, the Cluster Processor Modules (CPM) and the Jet Energy Modules (JEM). For none of the processors the entire η-ϕ-range can be covered by one module only. This means that overlap regions have to be created, so the sliding-window algorithms can be applied. Furthermore data from all modules have to be combined. A common solution for these exercises is implemented in the L1Calo Trigger.

Individual Calorimeter Trigger modules are mapped onto the η-ϕ-plane such that they handle only channels from one ϕ-quadrant and the required overlap region. Fig. 5.1 shows this for PPMs, Fig. 5.4 for CPMs and JEMs. Overlap-regions in ϕ are realized by fanning-out channels at quadrant borders in the Pre-Processor and sending them to the two appropriate modules of the digital processors. The overlap regions in η are realized by fanning signals in and out between the modules in one crate (FIO). The FIO is done using a common custom-built backplane for both, the CP and JEP. All modules mapped onto the same ϕ-quadrant are installed in one crate and communicate via the backplane. The combined information of all CPMs and JEMs is collected and put together by Common Merger Modules (CMM). A

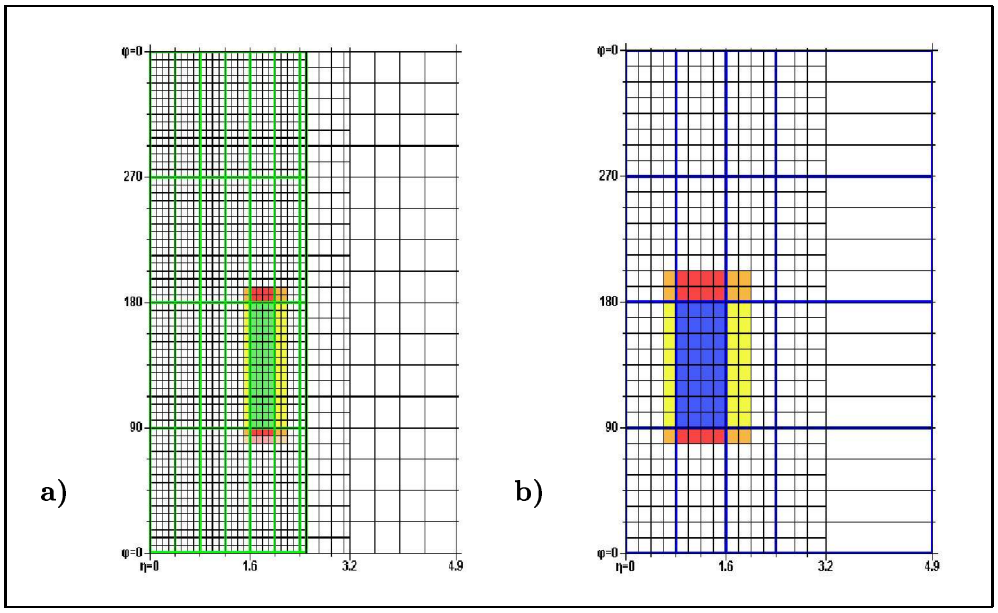

Figure 5.4: Regions addressed by individual a) CPMs and b) JEMs for positive η. Overlap from fanout on PPMs (red) and FIO on the backplane (yellow) are color coded for a sample CPM (green) and a sample JEM (blue).

CMM is the type of module that finally sends the multiplicities to the CTP and the ROI-data to the ROD, from where it is forwarded to the ROI Builder.

Cluster Processor

The Cluster Processor consists of four crates, one for each quadrant in ϕ. Each crate hosts 14 CPMs, 2 CMMs and 1 TCM. The CP is limited to $|\eta| < 2.5$ because it requires a granularity of 0.1×0.1 in $\Delta\eta \times \Delta\phi$. Two algorithms are implemented in the CP, one handling electrons and photons, the other one hadrons and τ-leptons. Fig. 5.5 shows the block diagram of a CPM.

Each CPM receives data from 4×20 Trigger Towers from the electromagnetic and hadronic layer. This corresponds to the CP-output of two PPMs including the overlap in ϕ. Each Trigger Tower delivers a ten-bit word at 40 MHz. In order to reduce the number of tracks and pins on a CPM, the LVDS data is serialized in twenty Serializer FPGAs (Field Programmable Gate Array) by groups of four. More precisely, data from two Trigger Towers are multiplexed in sets of four bits at 160 MBaud. The serialized data are fanned out four times at the Serializer output. One copy goes to neighboring modules and the other copies to one of eight CP-FPGAs where the algorithms run. The CP-FPGAs share tasks and are loaded with different Firmwares. Additionally two dedicated Hit Merger FPGAs calculate multiplicities and transmit them to the CTP via a CMM through the backplane. The input data is pipelined in the Serializer FPGAs, the ROI-data in the CP-FPGAs for readout. The readout is managed by two Readout Controllers (ROC) on receipt of a L1A. The ROCs get the L1A from the TTCdec card and use an optical G-Link to send the data to the ROD. One ROC provides input Trigger Tower data for the DAQ system, the other handles ROI data for the Level-2 ROI Builder.

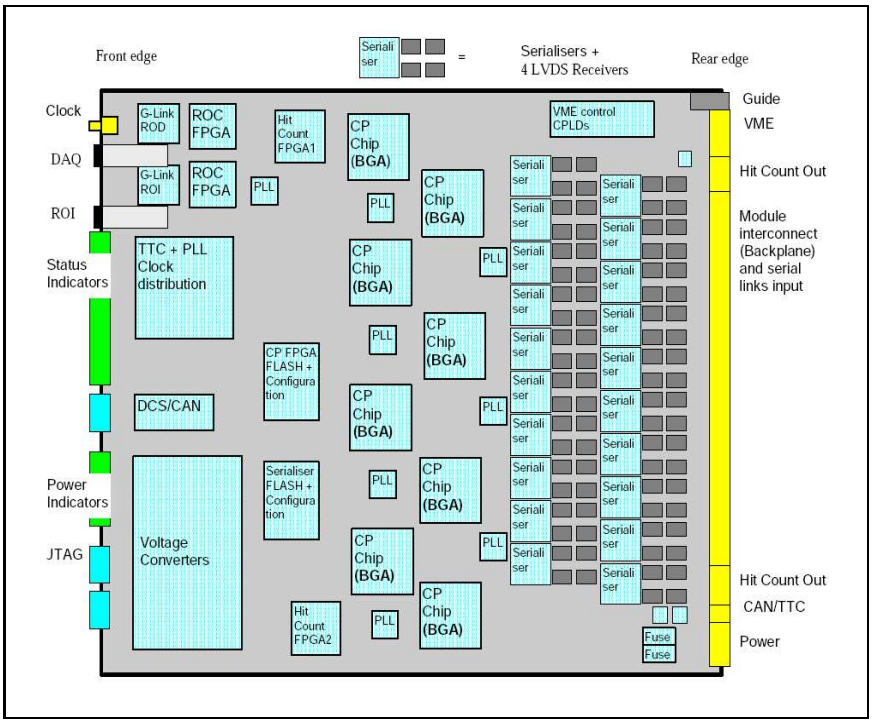

Figure 5.5: Block diagram of a CPM [15].

Jet-Energy Processor

Two algorithms run on the JEP. The Jet Algorithm is searching for jets and determines jet multiplicities. The Energy-Sum Algorithm computes the total and missing transverse energy. Due to the Trigger Tower granularity the Jet Algorithm is limited to the range $|\eta| < 3.2$. The Energy-Sum Algorithm uses the whole range $|\eta| < 4.9$ covered by the calorimeter. Both algorithms process Jet Elements. A Jet Element has a granularity of 0.2×0.2 in $\Delta\eta \times \Delta\phi$ and spans the electromagnetic and the hadronic layer. The JEP is installed in two VME-crates each hosting 16 JEMs, two CMMs and one TCM. On crate handles the ϕ-quadrants one and three, the other one two and four. The JEP crates have the same custom-made backplane used by the CP.

Fig. 5.6 shows a block diagram for a JEM. Each JEM receives 44 electromagnetic and 44 hadronic Trigger Towers from the PP. The channels are received in four input daughter boards, 24 channels per board. Each of the input modules has four 6-channel LVDS deserializers and one Input FPGA. The algorithms are implemented in two FPGAs per JEM, the Jet FPGA and the Sum FPGA, respectively, also referred to as Jet and Sum Processors. The Input FPGAs sum corresponding PP-signals from the electromagnetic and hadronic layer to form a Jet Element and pipeline it. E_y, E_x and E_{sum}, quantities needed by the Sum Algorithm, are also computed on the Input FPGAs. Two data streams are provided by the Input FPGAs, one for each algorithm. The Sum Processor does not require the overlap information and gets data from three of the four Input FPGAs at 40 Mb/s. The data for the Jet Algorithm is fanned out to the Jet FPGA on the board and those on the neighboring boards at 80 Mb/s. The fan-out to the other JEMs is done via the backplane.

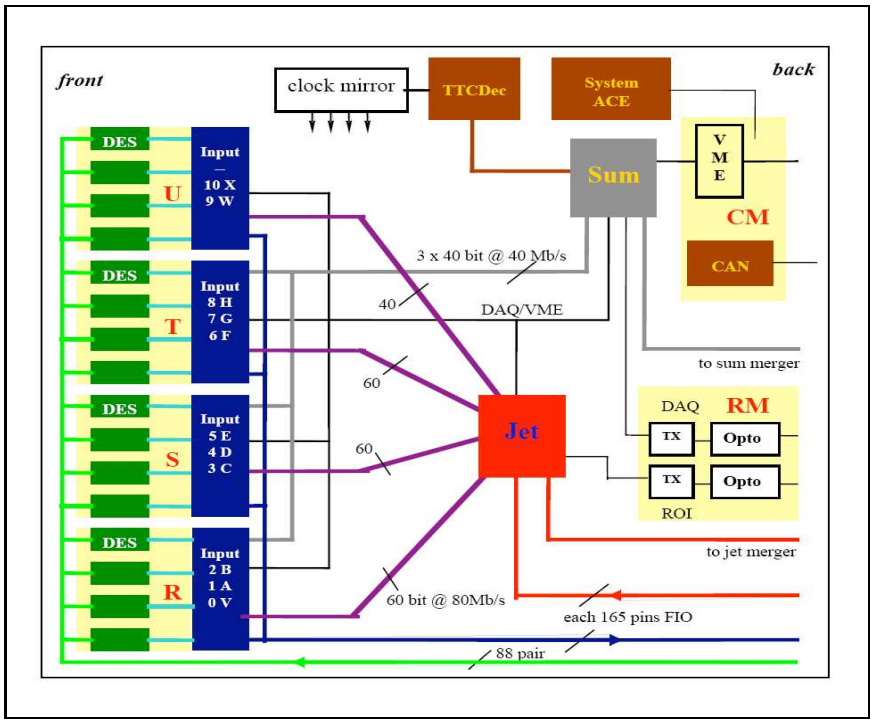

Figure 5.6: Block diagram of a JEM [16].

The multiplicities determined by the Jet and the Sum Processor are transmitted to appropriate CMM via the backplane, too. Once a L1A is received, similar to the CTP, the readout is managed by two ROCs. The ROCs are located on a daughter module, the Readout Module (RM). One ROC handles input data from the Input FPGAs and the Sum Processor for the DAQ, the other handles ROIs from the Jet FPGA for the Level-2 ROI Builder.

5.3 Algorithms

Four sliding-window algorithms run on the transverse energy E_T provided by the Pre-Processor. These are implemented in FPGAs on the modules of the JEP and the CP as described in the previous Chapter. Three algorithms use elements computed from Jet Elements and Trigger Towers in the electromagnetic and hadronic layer. Two of the algorithms evaluate these elements in order to find candidates for electrons, photons, hadrons, τ-leptons and jets according to their signatures in the detector shown in Fig. 5.7. τ-leptons that decay into τ-neutrinos and hadrons have the same signature as hadrons. Once a candidate has been found the corresponding multiplicity is incremented and the ROI is retained. The remaining algorithm derives the total transverse energy E_T and missing transverse energy E_T^{MISS} and delivers the number of given thresholds they surpass.

Electron-Photon/ Hadron-Tau Trigger

The Electron-Photon, Hadron-Tau algorithms are based on the same TT-sum elements that are evaluated in a different manner. The elements are computed by

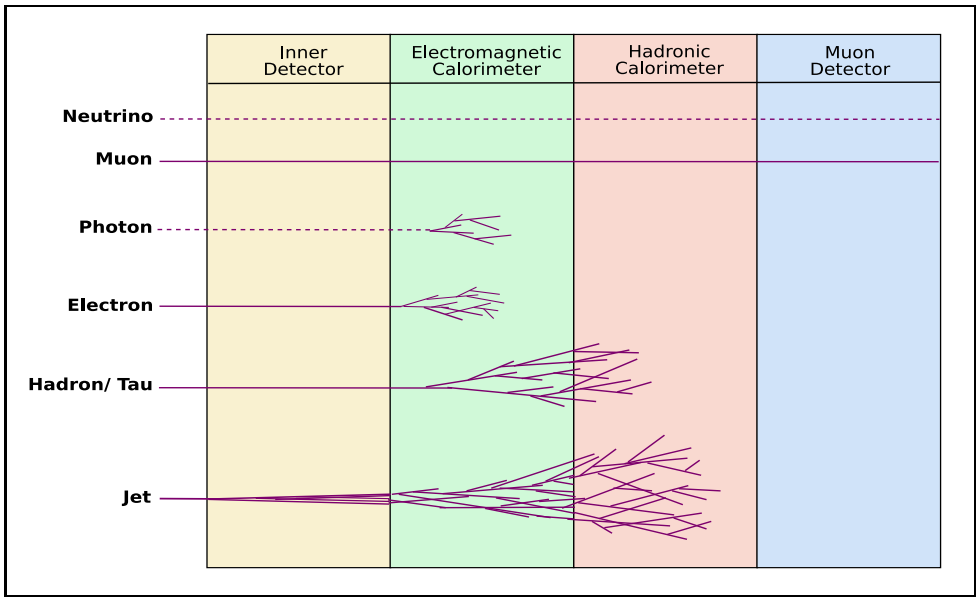

Figure 5.7: Detector penetration for different objects. Particles passing without leaving a trace are indicated by dotted lines.

means of windows with a maximum size of 4×4 Trigger Towers, that are sliding on the electromagnetic and hadronic η-ϕ-layer with a step width of 1 in η and ϕ. The algorithms need a granularity of 1×1 in $\Delta\eta \times \Delta\phi$ and address the range $|\eta| < 2.5$. The elements are illustrated in Fig. 5.8 a), they are:

- **Electromagnetic Clusters:** Four overlapping electromagnetic clusters, each a sum over two electromagnetic Towers.
- **Hadronic Core:** Sum of the four hadronic TTs in the center of the 4×4-window.
- **Hadronic Clusters:** Four hadronic clusters, each the sum of an Electromagnetic Cluster and the Hadronic Core.
- **Electromagnetic Ring:** Sum of the twelve electromagnetic TTs surrounding the center of the 4×4-window.
- **Hadronic Ring:** Sum of the twelve hadronic TTs surrounding the center of the 4×4-window.
- **Cluster ROI:** Sum of the four electromagnetic and hadronic TTs in the center of the 4×4-window.

In order not to double-count ROIs, a ROI is only tagged if the Cluster ROI of a given window position fulfills the conditions illustrated in Fig. 5.8 b). I.e. the Cluster ROI must be more energetic than the ROIs to the right and above and at least as energetic as the ROIs to the left and underneath. Demanding these conditions is referred to as declustering. Additionally the following conditions must be fulfilled for an electron-photon candidate:

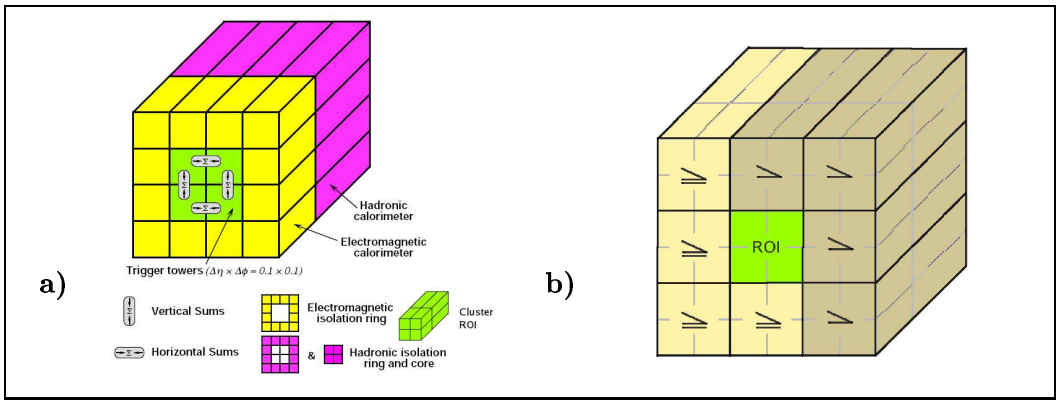

Figure 5.8: Illustration of a) windows and sums used by the Cluster Processor [6] and b) the comparison to neighbor Cluster ROIs for declustering.

- The E_T in the most energetic electromagnetic cluster must be bigger than the electromagnetic cluster threshold under consideration.
- The E_T in the Electromagnetic Ring must be smaller or equal to the electromagnetic isolation ring threshold under consideration.
- The E_T in the Hadronic Ring must be smaller or equal to the hadronic isolation ring threshold under consideration.
- The E_T in the Hadronic Core must be smaller or equal to the hadronic core isolation threshold under consideration.

The additional requirements for a hadron-tau candidate are:

- The E_T in the most energetic Hadronic Cluster must be bigger than the hadronic cluster threshold under consideration.
- The E_T in the Electromagnetic Ring must be smaller or equal to the electromagnetic isolation ring threshold under consideration.
- The E_T in the Hadronic Ring must be smaller or equal to the hadronic isolation-ring threshold under consideration.

Both algorithms, Electron-Photon and Hadron-Tau, have eight programmable sets of thresholds. The threshold-sets are applied to ROIs which contain a candidate for a "physics object" and comprise the same elements used to identify the ROI. If all thresholds in a set are surpassed, the multiplicity for the related object candidate is incremented.

Jet/ Energy-Sum

The basic elements of the Jet Algorithm are Jet Elements. Jet Elements have the granularity 0.2×0.2 in $\Delta\eta \times \Delta\phi$ and span the electromagnetic and hadronic layers. The detector granularity limits the jet algorithm to the range $|\eta| < 3.2$.

Eight sets of window size and threshold must be set in the Jet algorithm. Possible window sizes are illustrated in Fig. 5.9. Jet candidates are tagged using a 2×2

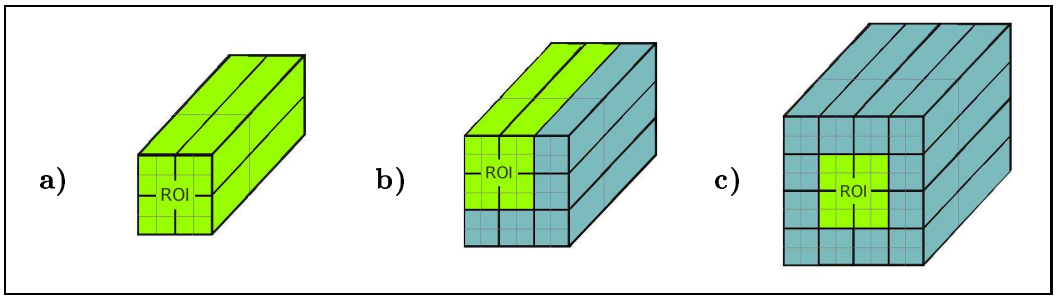

Figure 5.9: Illustration of Jet Clusters for a) 0.4×0.4, b) 0.6×0.6 and c) 0.8×0.8 $\Delta\eta \times \Delta\phi$ window. The windows are built out of Jet Elements, details in the text.

Jet Element Cluster ROI that must fulfill the same decluster conditions used in the Electron-Photon and Hadron-Tau Trigger algorithms, see Fig. 5.8 b) for details. In the case of the 6×6 window the Cluster ROI combines the Jet Elements that have the highest E_T, in case of the other window sizes it is centered. If the decluster conditions are fulfilled and the E_T is bigger than the jet threshold under consideration, the ROI is considered to contain a jet candidate. In that case the eight sets of window size and threshold are applied. Multiplicities are incremented for each set if the total E_T in the window surpasses the associated threshold.

The Energy-Sum algorithm computes the total scalar E_T-sum and missing transverse energy E_T^{MISS} up to the end of the Forward Calorimeter $|\eta| < 4.9$. The E_T-sum result is compared with four, the E_T^{MISS} result with eight thresholds and the information which thresholds have been surpassed is sent to the CTP.

Chapter 6

The Pre-Processor System

6.1 Implementation

The Pre-Processor (PP) is a modular VMEbus system of eight crates. An overview is given in Fig. 6.1. The analog electronics in the Trigger cavern up to the PP is separated into two parts according to the A-side ($\eta > 0$) and C-side ($\eta < 0$) of the detector. This separation is done as it allows to minimize the overall cable length. The signal chain starts at the TCPPs that are located close to the digital processors near the middle of all racks where the detector cables arrive. From the TCPPs they are routed to the Receivers in the outermost racks on each side of the setup. From the Receivers, the signals go their way back to the digital processors in the central racks via the RPPPs and the PPMs. For more details refer to [18] or [19]. The PP crates are ordered in electromagnetic, hadronic, barrel and end-cap crates. I.e. there is one electromagnetic and one hadronic barrel and end-cap crate for each side of the detector. Beside one Timing and Control Module (TCM) to interface the Timing Trigger and Control (TTC) system and a Single Board Computer (SBC) to interface the VMEbus, each PP crate holds 14 or 16 Pre-Processor Modules (PPM). The PPMs form the essential part of the PP. A total 128 PPMs are installed in the complete system.

Pre-Processor Module (PPM)

Each PPM is processing 64 channels in parallel. The major signal-processing is done by custom-built ASICs (PprASIC) for compactness sake. The PprASICs are mounted on Pre-Processor Multichip-Modules (PprMCM) 16 of which are located on one PPM. The PPM is an analog/digital hybrid that receives analog input from the calorimeters, digitizes it and separates it into two data-paths: the realtime-path and the readout-path. The realtime-path provides data to the object finding processors CP and JEP. The readout-path provides the PP-related data for events accepted by the CTP.

Fig. 6.2 shows the block diagram of a PPM. The PPM is a $400 \times 366\ mm^2$, 9U VME board carrying various submodules:

- 4 Analog Input boards (AnIn),
- 16 Multi Chip Modules (MCM),

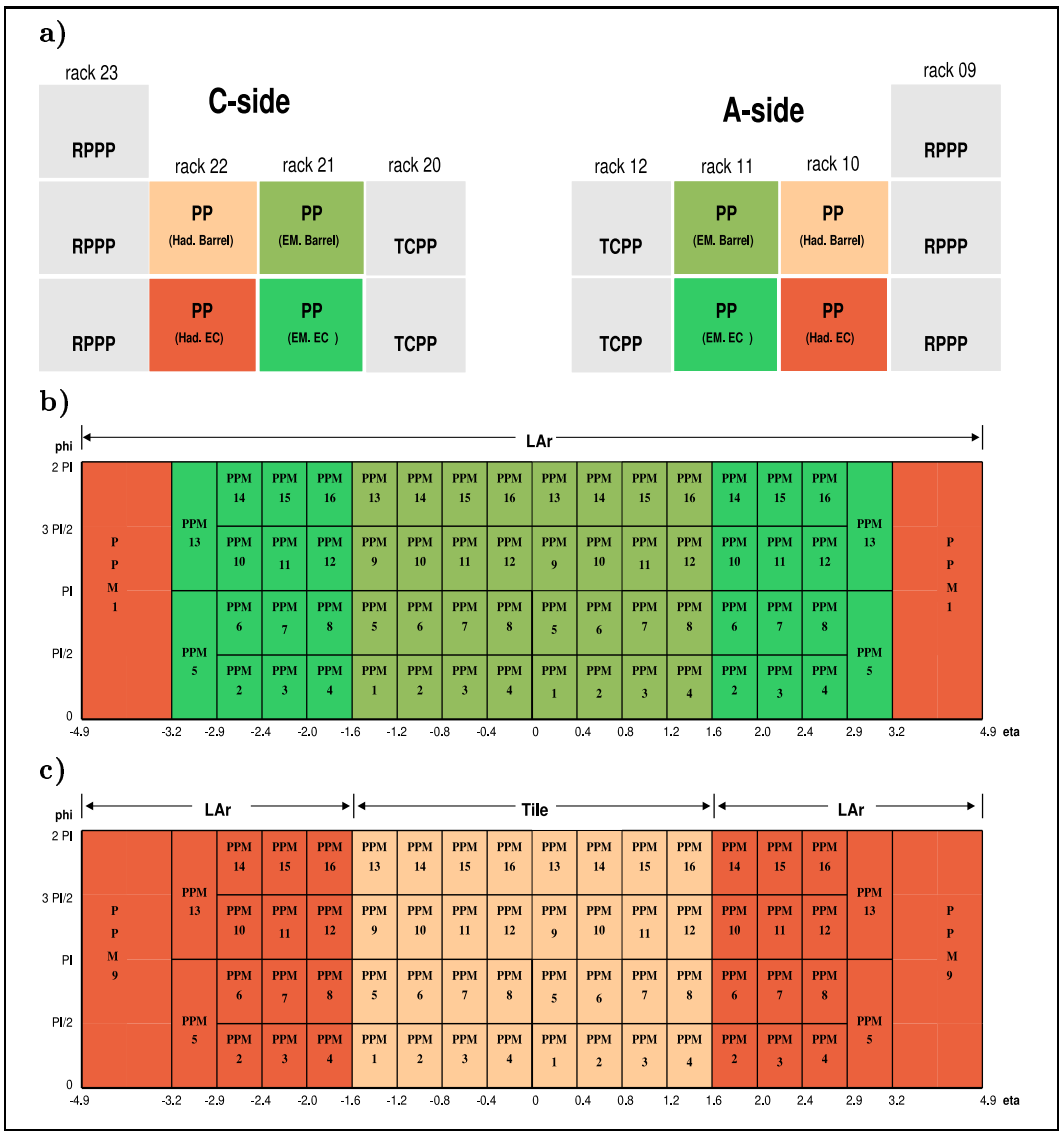

Figure 6.1: Mapping of η-ϕ onto the PP crates. a) Overview of Pre-Processor racks, b) mapping of electromagnetic layer and c) of hadronic layer. The digital Processors and the RODs are located in the central racks 16, 17, 18 and 19, between rack 12 and 20. The Receiver crates are located in the outermost racks: 7, 8, 24 and 25.

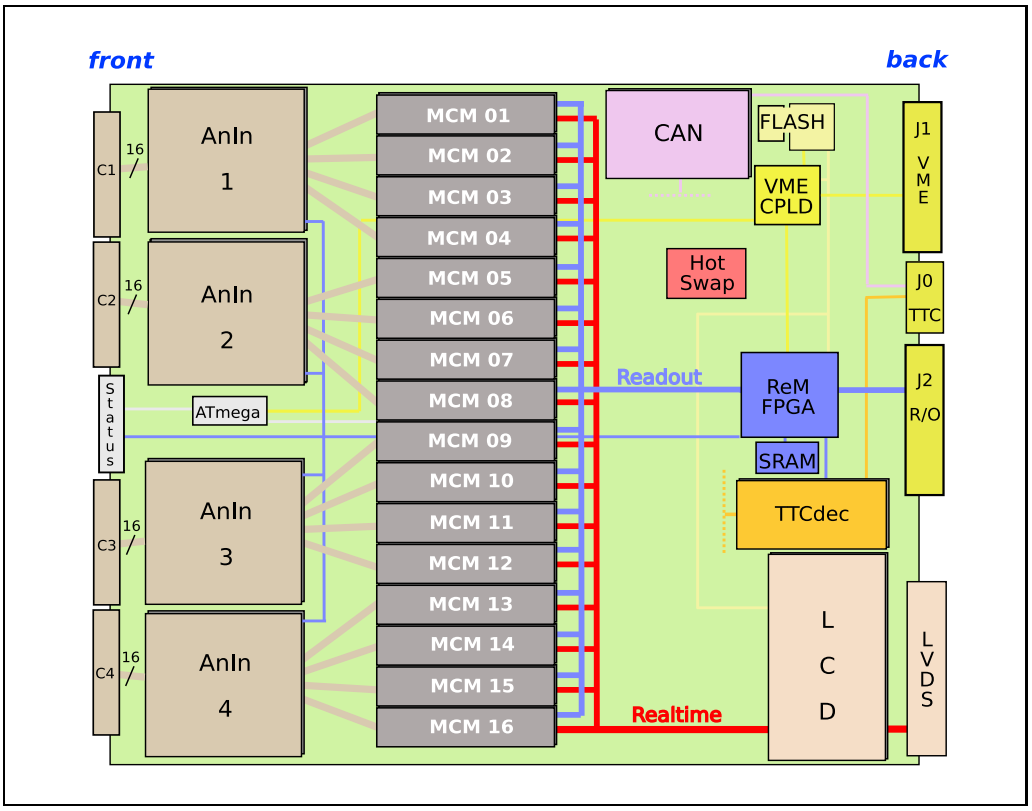

Figure 6.2: Block diagram of a Pre-Processor Module.

- 1 LVDS (Low Voltage Differential Signal) Cable Driver (LCD),
- 1 Timing, Trigger and Control decoder card (TTCdec),
- 1 Control Area Network sub-module (CAN).

Other components are bonded on the PCB:

- 1 Complex Programmable Logic Device (CPLD) handling VME access,
- 1 CPLD with Flash memory handling FPGA flash loading,
- 1 Readout Merger FPGA (RemFPGA), with 4 GB SRAM,
- 1 ATmega Microcontroller,
- 1 Hot-Swap controller,

Analog Input board (AnIn)

Each PPM receives 64 differential, analog signals via four Sub D 37 connectors. The differential signals are routed to four AnIn boards where the differential signal is transformed into a unipolar signal that is later digitized on the MCMs. The baseline of the unipolar signal can be shifted by adding a DC-level, that is programmable via an eight bit Digital to Analog Converter (Baseline DAC). Another eight bit DAC (Threshold-DAC) is used to create a threshold voltage. The threshold voltage

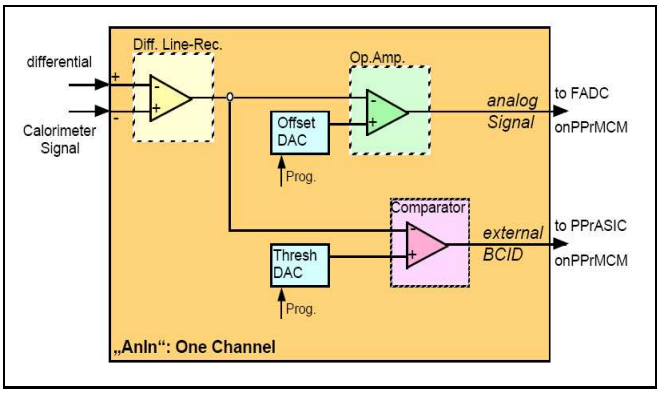

Figure 6.3: One channel on the AnIn board [20].

is fed into a discriminator and compared to a copy of the unipolar signal. The discriminator output is 1 if the signal surpasses the threshold, 0 elsewise. The digital signal is called external BCID. Both DACs are programmed via an Serial Peripheral Interface (SPI) bus that connects the RemFPGA, and all AnIn boards.

Multi Chip Module (MCM)

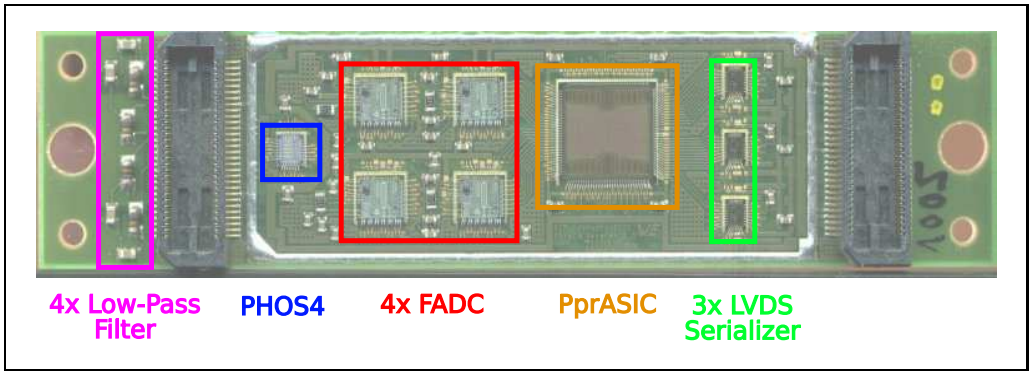

Figure 6.4: Block diagram of a MCM.

Both, the digital and the analog signal are routed on the shortest way to one of 16 MCMs. On the MCM the analog signals pass a $\sim 20\,\mathrm{MHz}$ R-C low-pass filter for noise suppression. Each MCM handles four channels at the time and carries nine Application Specific Integrated Circuits (ASIC):

- 4 commercial 10 bit Flash Digital to Analog Converters (FADC).
- 1 four channel delay generator (PHOS4), developed at CERN .
- 1 Pre-Processor ASIC (PprASIC), developed at the Kirchhoff Institute of Physics (KIP).
- 3 commercial LVDS Serializers.

The PHOS4 provides four $40\,\mathrm{MHz}$ clocks, whose phases can be programmed via an Inter-Integrated Circuit bus (I^2C) in steps of 1 nanosecond individually. The

PHOS4 output clocks are used by the FADC's defining the latch-phase of the signal. The digital signals and the external BCID are routed to the PprASIC. The PprASIC does Bunch Crossing Identification (BCID) and estimates the transverse energy with an eight bit resolution. Pipeline memories in the PprASIC buffer the FADC data as well as the E_T estimation and the BCID result for readout. The PprASIC can be read out and controlled by means of two Serial Interfaces (SIF) per chip, each responsible for two channels. The Realtime data for the same channel pairs a SIF addresses is multiplexed and sent to two of the three LVDS Serializers, providing the 1×1 η-ϕ granularity information for the CP. A copy of the data from all channels of a MCM is summed to a 2×2 η-ϕ cell and send to the third LVDS Serializer for the JEP.

LVDS Cable Driver (LCD)

The LVDS links provided by the MCMs operate at 480 MBaud and must be driven from the MCMs to the digital processor crates over a distance of ~ 11 m. To do so the signals have to be shaped in order to compensate frequency dependent effects in the LVDS cables. This shaping is commonly called pre-compensation. Furthermore some of the channels have to be fanned out in order to provide ϕ-overlap regions. All $3 \times 16 = 48$ LVDS links are routed to a LVDS Cable Driver (LCD) submodule dedicated to these tasks. The LCD board is located at the bottom back of the module and carries four X2CV_250 FPGAs, two for each digital processor. The output of the LCD board is routed to a connector at the bottom back of the modules. Dedicated pin-through backplanes attached to the rear side of the crate backplane connect these connectors with the cables.

Timing, Trigger and Control decoder (TTCdec)

The PPM is clocked via the TTCdec card or an on-board crystal clock that takes over in case no TTC signal is available. The TTCdec card is provided with input by the TCM. The TTC signal is routed to the PPMs through an auxiliary backplane that is mounted on the J0 connectors at the rear of the crate. The TTCdec card holds a TTC Receiver (TTCrx) chip. The TTCrx is an ASIC developed at CERN that decodes an electric TTC signal. The TTCrx has various Status and Control registers that can be accessed with an I^2C-bus. Examples for Status registers are a 12 bit Bunch counter that is incremented at each clock cycle or an 24 bit Event counter that is incremented on receipt of an L1A. Control registers can e.g. be used to delay TTC signals or steer the behavior of the counters.

CPLD and ATmega

If a PPM is inserted in a crate and switched on, a Hot Swap controller powers the module, and a dedicated VME CPLD allows the communication with the board. The functionality at that state is, however, very limited. Essentially all one can do is to read the board ID that is assigned to the board with a resistor array and communicate with the ATmega Microcontroller. The ATmega is addressed by the VME CPLD, controls a 9×9 LED matrix on the front panel, and measures MCM core voltages and temperatures. In order for the PPM to be fully operational firmware

(F/W) must be loaded in the various FPGAs. The F/W bit-files are either loaded from VME or the Flashloader CPLD via an a eight bit data bus. The Flashloader CPLD has an eight MB Flash Memory attached that can hold 6 different bit-files for the RemFPGA and one bit-file for each of the LCD FPGAs.

Readout Merger FPGA (RemFPGA)

The RemFPGA is the central device of the board, it interfaces:

- the VME CPLD,
- DACs on the AnIn boards via a SPI bus,
- PHOS4 Delays via an I^2C-bus,
- Configuration Registers of the TTCrx chip via another I^2C-bus,
- 16×2 MCM Serial Interfaces,
- a four MB SRAM,
- a Rear G-Link Transition Module (RGTM) mounted on the rear side of the crate at the J2 connector,
- Status and Error LEDs mounted on the front panel.

The task of the Rem FPGA is to allow the communication with the sub-components of the PPM listed above. Beside that it is responsible to read out the MCMs after receipt of an L1A and forward the data to the RGTM where it is serialized and sent to the ROD via an optical G-Link.

Control Area Network (CAN)

Independent of what has been described so far, a Control Area Network (CAN) is realized using one CAN controller per PPM. The CAN controller is monitoring temperatures and voltages at low frequency and can switch a board off in order to protect it if something unexpected happens. The CAN bus is implemented on the same auxiliary backplane that distributes the TTC-signal to the PPMs.

6.2 Processing Chain

In order to match the 1.0 V digitization window of the 10 bit FADC which goes from 1.9 to 2.9 V the AnIn board is designed such that a 3.0 V differential input window is mapped onto a 1.2 V unipolar output window. The FADC covers therefore a 2.5 V window of the differential input signal, with an adjustable offset. The AnIn output baseline is 1.65 V for a zero Baseline DAC setting and 2.25 V for the maximum DAC value of 255. Given the FADC window this corresponds to a digitization window of 0.685 to 3.185 V for a zero DAC value and -0.875 to 1.625 V for the maximum DAC value with respect to the differential AnIn input signal. For input values higher than 3.0 V the AnIn board saturates and behaves non-linearly. For normal operation however, the saturation is not an issue. The FADC window will always

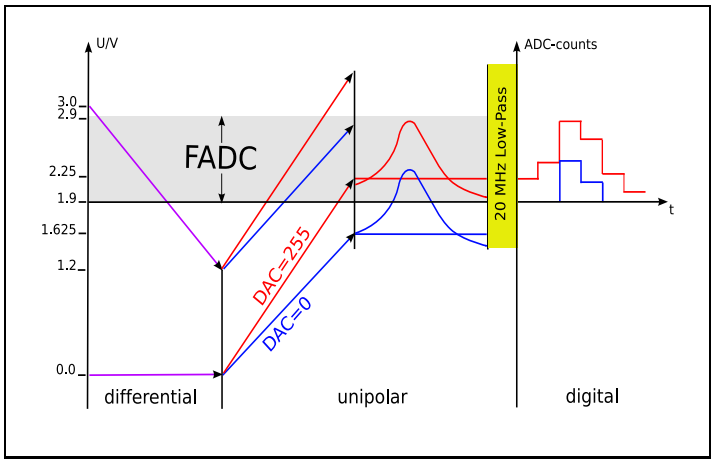

Figure 6.5: Analog Signal chain from the differential input to FADC.

be adjusted such that the input baseline is digitized. This means the AnIn output baseline will always be set to $\geq 1.9\,\mathrm{V}$, and consequently the maximum input values that are digitized will always be $\leq 2.5\,\mathrm{V}$. The differential input is gauged such that a 0.1 V input signal approximately corresponds to a transverse energy of 10 GeV in the Trigger Tower. This means that one FADC-count approximately corresponds to 244 MeV. Therefore the PP input saturates at maximum transverse energies of $E_T \geq 300\,\mathrm{GeV}$ and for transverse energies $E_T \geq 250\,\mathrm{GeV}$ if the baseline is just starting to be digitized. In case of the Tile Calorimeter the E_T-gauging is done using the Receiver VGAs, in case of the LAr Calorimeter it is mainly done in the Tower Builder Boards in the FE crates and the Receiver VGAs are only used for fine-tuning.

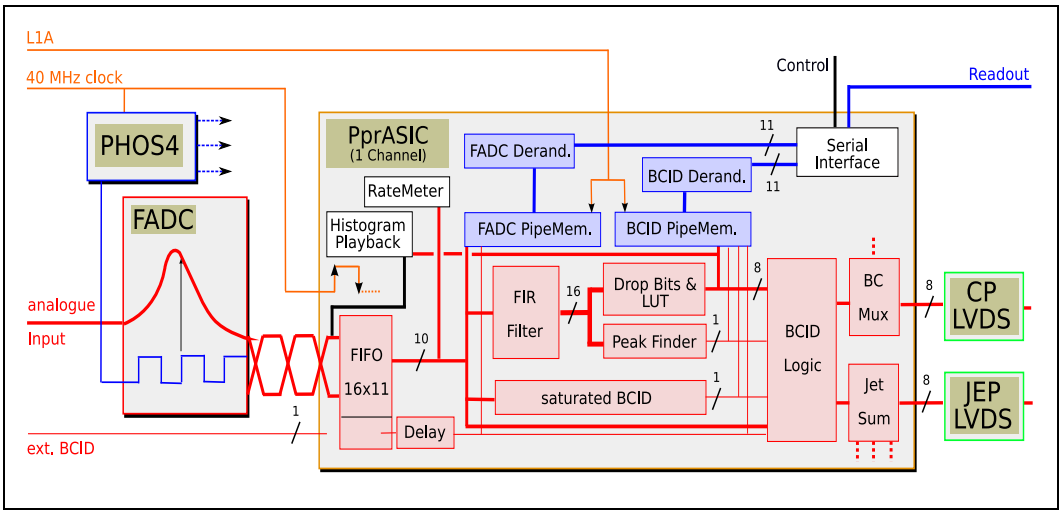

Figure 6.6: Block diagram of the Signal chain for one channel on the MCM; the data stream goes from left to right.

Most of the signal-processing done in the Pre-Processor takes place on the MCMs. Fig. 6.6 shows a block diagram of the data stream on the MCM for one Trigger Tower channel.

A 40 MHz frequency clock which corresponds to a period of 25 ns is derived from the TTC system and distributed to the PHOS4 and the PprASIC. The PHOS4 generates four output clocks which individually can be delayed between 0 and 24 ns with a resolution of 1 ns. The delayed clocks are used by the FADCs' which means that the PHOS4 delays determine the latching time of the analog signal within a 25 ns interval. The FDACs ten bit digital output is routed to the PprASIC. At the time the FADC is just about to latch, the output data become corrupted. The phase of the PprASIC clock and the time the data needs from the FADC to the PprASIC are fixed. As a consequence there will be PHOS4 settings for which the data arriving at the PprASIC are corrupted with respect to the PprASICs clock. This issue is addressed by allowing to select the clock edge (positive $\leftrightarrow$ negative) at which the PprASIC latches input data. For more information refer to Appendix B. The FADC data are also referred to as raw data.

In the PprASIC the FADC data and the external BCID signal from the AnIn board run through a First-In-First-Out (FIFO) buffer with an programmable length from 1 to 16 clock ticks. The FIFO can be used to synchronize channels. After the FIFO the FADC data are processed for transverse energy estimation and BCID, the external BCID goes through an additional delay. That way the FADC processing result and the external BCID arrive synchronously in the BCID Logic. The external BCID is the fall back version of three signals that can be used to select the clock tick at which the event data were registered in a given channel.

After the FIFO, the FADC data is fanned out into six paths. One path goes into a 256×11 b memory that can be used to produce a distribution of the Raw data. The same memory can be loaded with playback data, so-called test-vectors that can be injected into the upcoming processing chain replacing the FADC data. The memory is referred to as Histogramming or Playback Memory, depending on which mode it is operated in. Another path goes into a RateMeter logic that produces rates of raw data words that pass a programmable threshold. The third copy of the FADC data is pipelined into a dedicated FADC PipeMemory and preserved for potential readout, and the fourth one goes to the BCID Logic.

The fifth path of the FADC data is used by the saturated BCID Logic. The saturated BCID Logic generates a one-bit signal similar to the external BCID in order to select event data. The saturated BCID is dedicated for saturated signals. A good explanation of the saturated BCID can be found in [21].

The last FADC data copy is used for estimation of transverse energy E_T and the default BCID. As a first step the FADC data is filtered by means of a Finite Impulse Response (FIR) filter. The FIR filter covers a window of five successive samples and can be programmed with a four bit accuracy. In the first and the last FIR Filter coefficient the Most Significant Bit (MSB) is used as a sign. The maximum FIR Filter output is (3*15+2*7)*1023=60357 (0xEBC5), thus 16 bits deep.

The FIR Filter output is forked into two streams one of which is used for E_T estimation. The final E_T has an accuracy of eight bits and is derived using a ten to eight bit Look Up Table (LUT). A Drop Bits logic transforms the 16 bit FIR output into the 10 bit LUT input. This is done by selecting ten bits from a given start bit. The start bit has to be chosen such that for given FIR coefficients and 0x3ff FADC data the highest non-zero bit of the FIR output is still captured. As dropping

	Resolution		Maximum FIR Coefficient		
D	**Start bit**	**Δa_i**	**a_0** ($\pm$3 b)	**a_1-a_3** (4 b)	**a_4** ($\pm$3 b)
r	0	1	$\pm$7	15	$\pm$7
o	1	0.5	$\pm$3.5	7.5	$\pm$3.5
p	2	0.25	$\pm$1.75	3.75	$\pm$1.75
	3	0.125	$\pm$0.875	1.875	$\pm$0.875
B	4	0.0625	$\pm$0.4375	0.9375	$\pm$0.4375
i	5	0.03125	$\pm$0.21875	0.46875	$\pm$0.21875
t	6	0.015625	$\pm$0.109375	0.234375	$\pm$0.109375

Table 6.1: Effective FIR Filter coefficients a_i for given Start bits in the Drop Bits logic. Column 2 shows the precision Δa_i coefficients can be chosen with, columns 3 to 5 the maximum possible coefficients. It is obvious, that the Start Bit should be set as high as possible for the maximum coefficient required to improve the precision.

a bit corresponds to a division by two, one can think of the FIR Filter/Drop Bits ensemble as of an effective FIR Filter as shown in Table 6.1.

A Peak Finder generates the default BCID signal using the remaining FIR Filter output. For each clock tick, the Peak Finder compares the current data f_0 with the data from the previous f_{-1} and following f_{+1} clock tick. The Peak Finder can be operated in two modes, either setting its output to one if

$$f_{-1} < f_0 \geq f_{+1} \tag{6.1}$$

or

$$f_{-1} < f_0 > f_{+1} \tag{6.2}$$

is fulfilled.

The LUT has to be programmed to derive the final E_T estimation out of the FIR Filter result. The LUT output is commonly called LUT data or BCID result, not to be confused with the 1 bit BCID signals. It is 8 bits deep, and the aim is to get an approximate GeV/count ratio of 1. The LUT data is split into three paths. One path goes to the Histogramming Memory that thereby can be used to create E_T distributions as well as FADC data distributions. The second path goes to a dedicated BCID Pipeline Memory in order to keep it for read-out. The last path goes to the BCID Logic.

The BCID Logic has the task to select LUT data using the various BCID signals. This is done for three energy regions individually. The energy regions can be programmed via two thresholds either using the Raw FADC or the LUT data as a reference. The input signals to the BCID Logic are:

- the Raw FADC data,
- the eight bit E_T estimation from the LUT output,
- the synchronized 1-bit signal from the external BCID ($extID$),
- the 1-bit result from the saturated BCID logic and ($satID$)
- the 1-bit Peak Finder output (PF).

For each of the energy regions one can individually choose which BCID signals should influence the decision and how. Any combination of BCID signals can be connected via logical ANDs[1], ORs[1] and XORs[1] in order to determine whether a given clock tick should be selected or not. Let's assume $\circ$ indicates to ignore the following bit, this means that any of the combinations:

$$\{\circ\}extID\ \{\&,|,\wedge,\circ\}satID\ \{\&,|,\wedge,\circ\}PF \tag{6.3}$$

can be used as decision for each of the energy regions. If the decision for a given clock tick and the appropriate energy range is 1, the BCID Logic outputs the LUT result. If the decision is 0 the BCID Logic output is forced to zero.

The output of the BCID Logic for one channel is duplicated. One copy is multiplexed with the data of one additional channel and sent to one of the two CP Serializers. The other copy is summed with the data of three additional channels, all together covering the η-ϕ range of a Jet Element, and sent to the JEP Serializers. From the Serializers the data leave the MCM and are transmitted to the LCD board and the digital Processors.

[1]Bit operators: AND: &, OR: |, XOR: $\wedge$

Chapter 7

Calorimeter-Trigger Calibration

The PP is the part of the Level-1 Calorimeter Trigger that is responsible to prepare the analog detector signals for digital processing. Therefore the analog signals have to be synchronized, and an E_T estimation has to be done that holds over the whole energy range addressed by the Calorimeter Trigger. In fact some synchronization of LVDS signals can be done at the input of the digital Processors, but essentially calibrating the Calorimeter Trigger means calibrating the PP.

The previous Chapter describes how Trigger Tower signals are used in the PP. All signal processing is entirely implemented in hardware, mostly done in ASICs, especially the PprASIC. This means that it is possible to modify the principal processing chain only by building new hardware, i.e. a huge effort. This does not mean that the PP design is completely rigid, a bulk of parameters allows to steer the behavior of the system in many ways. Practically the parameters are set by writing values into registers of the PPM that are accessible via VME. Typically registers are sub-divided into Bit Fields (BF), each associated with a parameter.

		PprASIC		
AnIn	**PHOS4**	**Channel**	**SIF**	**Global**
2	1	58	6	3

Table 7.1: Number of parameters in the PP.

Table 7.1 gives an overview of the number of parameters that influence the signal chain in the PP. The numbers given for the AnIn board, the PHOS4 and the PprASIC Channel refer to one channel. The LUT and the Playback Memory, that would have 1024 and 256 parameters for their own, have been suppressed in this summary. Not listed are parameters at a PPM level, like e.g. TTCrx settings. The number listed under PprASIC-SIF refers to behavior relevant for pairs of channels assigned to a SIF, the number listed under and PprASIC-Global to the behavior relevant for all channels on a PprASIC. The range of values the parameters can have varies from 2^1 to 2^{11}, and inter-dependences exist between them. Therefore the given numbers do not reflect the diversity of the PP configuration directly, however, they do give a hint.

Not all of the PP parameters are related to calibration, in fact they can rather be divided into three sets. The first set includes those settings, that are aligned to the

timing of the signals. It comprises all the settings meant to assure that data related to one event are digitized properly, synchronously leaves the PP in the realtime path and is correctly selected in the L1A triggered readout path. The second set of parameters comprises those parameters that influence the result achieved for the transverse energy. The third set of parameters contains all the Bit Fields that do not belong to any of the other sets. The third set makes up the main part of the parameters and contains parameters that allow to run the system for debugging or tests. It also includes parameters not interfering with the data taking and e.g. control the behavior of the Rate Meter Logic.

Roughly speaking calibration in the context of this theses means finding values for the first two sets of parameters. The settings in the third set are either nailed down to certain values by the fact the system is used for data taking, or they must have no effect the realtime or readout data.

Because of the manner the Bit Fields are implemented, the borders between the three parameter-sets discussed above are not absolutely exact. Some of the parameters that naturally belong into the configuration area for example do effect the timing in an indirect way. In order to get some desired results, parameters have to be seen in combination. In order to ease the life of the user, an abstract layer of configuration parameters is introduced and a mechanism has been elaborated that takes care of register inter-dependences when changing the configuration. More on this later, the next two sections introduce the most important Bit Fields for calibrating the PP.

7.1 Timing Parameters

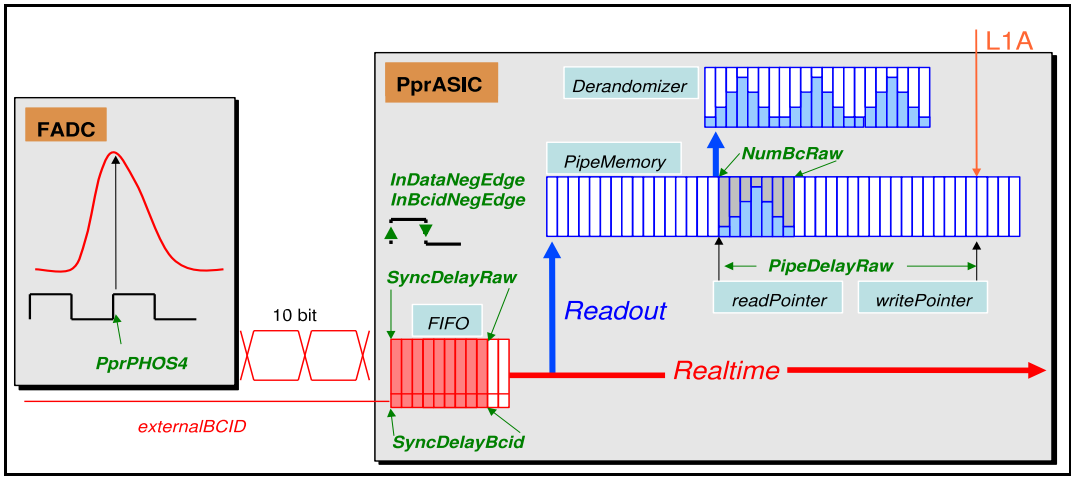

Figure 7.1: Overview of the timing aligned parameters of the PP. The readout path is illustrated for the FADC data only as for the BCID data it looks similar.

Fig. 7.1 gives an overview of the Bit Fields involved in the timing-calibration. The sketch shows only the FADC Pipeline Memory, in fact there are two, one for FADC and one for BCID data. The PprPHOS4 Bit Field controls the PHOS4 delay, and therewith the latching time of the FADC. The PprPHOS4 should be adjusted

to latch the calorimeter signal at its maximum value.

The Bit Fields controlling the depth of the FIFO at the PprASIC input are called SyncDelayData and SyncDelayBcid, those controlling the latch-edge InDataNegEdge and InBcidNegEdge. The FIFO depth is clearly a typical timing related parameter. The task of the FIFO is to synchronize the realtime data path across the PP system. With the play of 15 clock ticks this is clearly feasible; 15 clock ticks correspond to 75 m of cable[1] which is about the length of the cables from the detector to the trigger cavern.

Note that there are inter-dependences between the Bit Fields synchronizing the realtime path. The PprPHOS4 value is determined by the latency between the detector and the PP. Once the PprPHOS4 value is given the latch edge is fixed, see Appendix B for details. Choosing the latch edge, however, effects the clock cycle, data are captured in the PprASIC. PHOS4 delays between 0 ns and 12 ns, that are latched at the negative edge, and the successive PHOS4 delays 13 ns to 24 ns with the positive latch edge, one clock cycle later. This means, that for PHOS4 delays between 13 ns to 24 ns the FIFO depth has to be decremented by one in order to have the same effect as it has for PHOS4 delays between 0 ns and 12 ns.

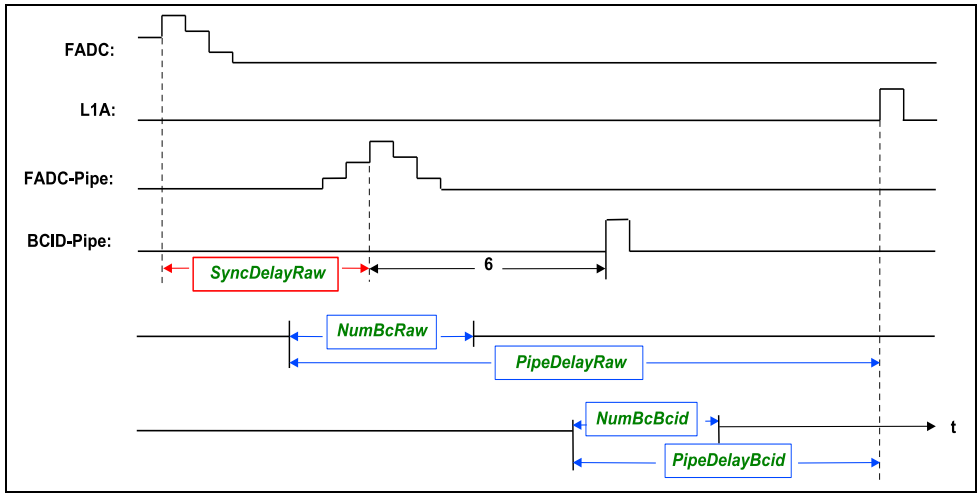

Figure 7.2: Signals and parameters involved in the readout path on a common timeline.

Once correct settings for the Bit Fields introduced so far are found, the realtime path is synchronized. The remaining parameters address the timing of the readout path. The pipeline memories are 128 cells deep and used to bridge the time the calorimeter needs to perform an L1A decision and to compensate differences in the propagation time of the L1A from the CP to the various PP channels. Both the FADC and the BCID Pipeline Memories have a read and a write pointer. The write pointer is free running and continuously copying data from the realtime path into a circular queue. The read pointer follows the write pointer with an offset that can be programmed via the PipeDelayRaw, PipeDelayBcid Bit Field, respectively. If an L1A arrives at the PprASCIC a given number of cells is copied from the pipeline

[1] assuming a signal speed of $\frac{1}{5}\,\frac{\mathrm{m}}{\mathrm{ns}}$ in the cable

memory to the Derandomizer where it waits to be read out. The number of cells that is read out is determined with the NumBcRaw, NumBcBcid Bit Field, respectively.

Fig. 7.2 gives an overview of the signals and parameters involved in the timing of the readout path. The parameters should be adjusted such that the maximum in the Raw Pipeline Memory falls into the center of the raw window and the BCID result in the BCID Pipeline Memory is in the BCID window. Given the default Peak Finder is used in the BCID Logic, there is a fixed offset of 8 clock ticks between the maximum of the raw data in the Raw Pipeline and the LUT data in the BCID Pipeline. Once the parameters for the FADC readout are known, the parameters for the BCID readout can be computed. The default parameters for NumBcRaw and NumBcBcid are 5 and 1.

7.2 Energy Parameters

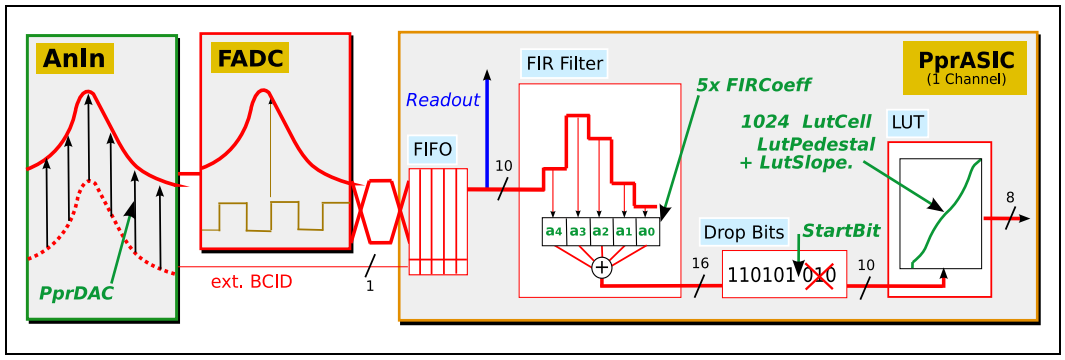

Figure 7.3: Overview of the energy-calibration aligned parameters of the PP.

The energy-calibration related parameters are illustrated in Fig. 7.3. The very first parameter is the DAC offset: PprDAC. The next parameters that influence the energy-calibration are the FIR Filter coefficients and the Drop Bits Logic, that can be seen in combination as an effective FIR Filter (Table 6.1). The FIR Filter intended to improve the Signal to Noise Ratio (SNR). Last not least, there is the 10 to 8 bit Look Up Table. The LUT can be loaded with a straight line, defining an y-axis intercept, LutPedestal, and a slope LutSlope. In order to have full flexibility it is also possible to set each LUT cell individually.

In order to understand how to set best the FIR coefficients let us consider an estimation. Assume $\{\mathbf{x}_i\}$ is a set of N 5-dimensional vectors each containing a noisy signal, i.e. five consecutive ADC samples of a signal pulse. Moreover let $\mathbf{a}$ be the vector containing the five FIR coefficients.

As the size of a non-saturated pulse is proportional to the transverse energy E_T deposited in a Trigger Tower, it can be computed as:

$$E_T(\mathbf{x}_i) = k\ \mathbf{a}\mathbf{x}_i, \tag{7.1}$$

where $\mathbf{x}_i$ is a sample vector and k is a factor. The mean transverse energy $\langle E_T \rangle$ is:

$$\begin{aligned} \langle E_T \rangle &= \frac{k}{N} \sum_i \mathbf{a}\mathbf{x}_i \\ &= k\ \mathbf{a}\ \frac{1}{N} \sum_i \mathbf{x}_i \\ &= k\ \mathbf{a}\boldsymbol{\mu} \end{aligned} \tag{7.2}$$

where $\boldsymbol{\mu}$ is the expectation value of $\{\mathbf{x}_i\}$. The variance of the transverse energy is:

$$\begin{aligned} \sigma_E^2 &= \frac{1}{N-1} \sum_i \big(E_T(\mathbf{x}_i) - \langle E_T \rangle\big)^2 \\ &= \frac{k^2}{N-1} \sum_i \big(\mathbf{a}\mathbf{x}_i - \mathbf{a}\boldsymbol{\mu}\big)^2 \end{aligned} \tag{7.3}$$

which is equivalent [1] to:

$$\begin{aligned} \sigma_E^2 &= \frac{k^2}{N-1} \sum_i \mathbf{a}\big(\mathbf{x}_i - \boldsymbol{\mu}\big) \otimes \big(\mathbf{x}_i - \boldsymbol{\mu}\big)^T \mathbf{a}^T \\ &= k^2\ \mathbf{a} \left[\frac{1}{N-1} \sum_i \big(\mathbf{x}_i - \boldsymbol{\mu}\big) \otimes \big(\mathbf{x}_i - \boldsymbol{\mu}\big)^T \right] \mathbf{a}^T \\ &= k^2\ \mathbf{a}\ C\ \mathbf{a}^T \end{aligned} \tag{7.4}$$

where C is the noise covariance matrix. The signal to noise ratio:

$$SNR \equiv \frac{\langle E_T \rangle^2}{\sigma_E^2} = \frac{(\mathbf{a}\boldsymbol{\mu})^2}{\mathbf{a}\ C\ \mathbf{a}^T} \tag{7.5}$$

has a maximum for:

$$\boxed{\mathbf{a} = C^{-1}\boldsymbol{\mu}} \tag{7.6}$$

The FIR Filter should be derived according to 7.6; such a Filter is called a Matched Filter. Choosing a Matched Filter is also beneficial for the Peak Finder BCID. The Matched Filter output gives a good measure for the probability that the signal contains a pulse according to $\boldsymbol{\mu}$. Note further, that assuming white noise makes C the unity matrix. For more details refer to [22].

The LUT is the last item in the processing chain. Therefore it must assign the FIR Filter output to the final transverse energy estimation.

In more detail that means:

- Compensate nonlinearities in the system.
- Subtract the pedestal that the FADC data are sampled with in order to measure the baseline.
- Inhibit output for values smaller than a given E_{thresh} to suppress noise.

Fig. 7.4 shows how the content of a LUT in typically looks like. Pedestal subtraction and noise suppression is done by zeroing the LUT content.

[1] using Einstein notation: $(a_k b_k - a_k \mu_k)(a_l b_l - a_l \mu_l) = a_k (b_k - \mu_k)(b_l - \mu_l) a_l = a_k [(b-\mu)_k (b-\mu)_l] a_l$

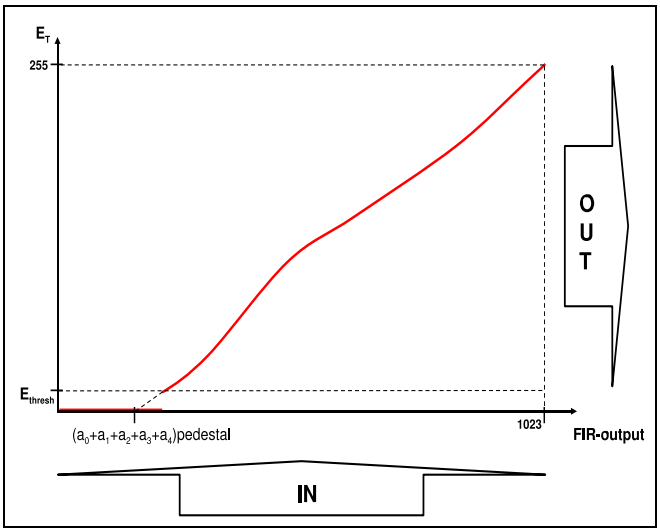

Figure 7.4: LUT input/output diagram.

7.3 Setup(s)

This thesis was written in the ATLAS commissioning phase. During such a phase hardware is permanently installed, tested, repaired, reinstalled, retested, exchanged etc. The number of possible activities is often already restricted by what is available in the system at a given time. Actually, a significant amount of the tests done addresses the system commissioning, installation and integration and is presented in the next Chapter. Under conditions as described, it is naturally difficult to talk about one System Setup.

On the other hand the principal possibilities of what can be done/tested w.r.t. the calibration, are effected by a subset of the overall changes only. With respect to the calibration there are some typical setups that have been used, and are going to be used. All setups in this Chapter are typical in this sense and presented to give an idea of the principle possibilities the system provides. However, it is possible to mix certain aspects of the setups shown here, and this has been done. Minor variations from what is presented here and what was actually used for particular tests are possible.

Heidelberg Setup

A test-system that has been installed in the ATLAS laboratory of the Kirchhoff-Institut für Physik in Heidelberg. It is illustrated in Fig. 7.5 and consists of five major items:

- A Timing, Trigger and Control (TTC) crate.
- A Nuclear Instrumentation Module (NIM) crate.
- A Pre-Processor (PP) crate.
- An Arbitrary Function Generator (AFG).

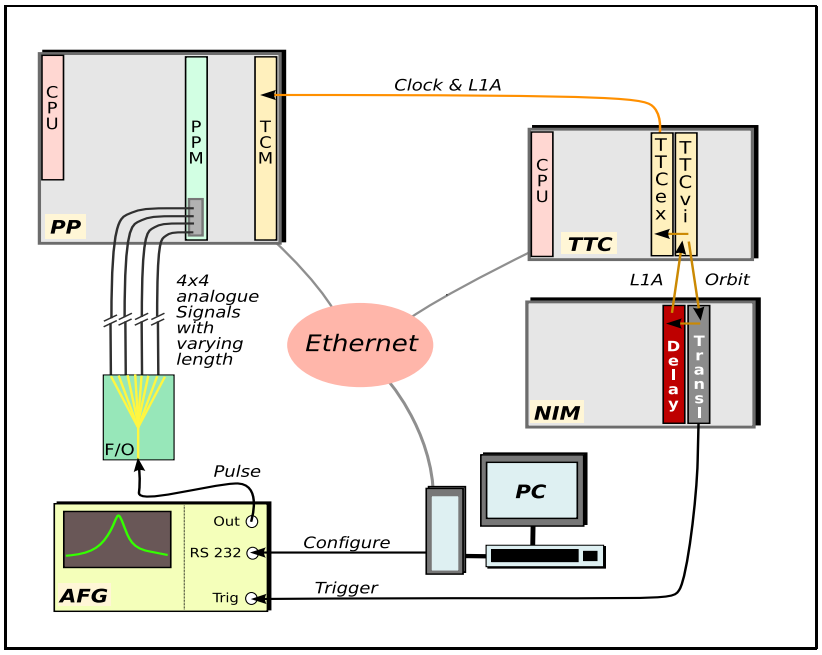

Figure 7.5: Overview of the Heidelberg Setup.

- A standard Personal Computer (PC).

The TTC crate contains two modules that are produced and provided by CERN and needed for the TTC infrastructure: the TTCvi and the TTCex. The TTCvi and the TTCex allow to generate TTC signals as they will be used in the ATLAS experiment, i.e. to encode a 40 MHz clock and broadcast commands via an optical link. TTCvi stands for TTC VME Interface and provides an interface between the TTC system and the user. The TTCvi provides a clock and mechanisms to generate all kind of signals as they are used in the final ATLAS system. The generation of a L1As for example can be triggered by a VME command, free running counters or an external trigger. TTCex stands for TTC Emitter/Receiver. The TTCex gets electrical signals that are generated by the TTCvi and encodes them into 10 optical links that can be used for test purposes.

In the Heidelberg Setup the TTC system is used in combination with a NIM crate in order to generate the L1As and trigger an AFG to generate an analog pulse. The TTCvi provides an orbit counter output that generates a signal at the same frequency at which a proton bunch surrounds the LHC orbit: 11223 Hz $\approx$ 10 kHz. The orbit frequency corresponds to a period of 3564 (0xdec) LHC clock-ticks. Providing a clock-synchronous signal the orbit counter output is duplicated in the NIM crate. One copy is used to trigger the AFG, the other copy is delayed and fed back into the external L1A input of the TTCvi, triggering the L1A generation. The electrical signal generated by the TTCvi is passed to the TTCex and from there to the Timing and Control Module (TCM) of the PP crate.

The PP crate receives 16 analog input signals from a custom built 1 to 16 channel Fan-Out (F/O) board that is fed by the AFG. Only a quarter of a PPM is covered by the F/O board, but this is enough for test purposes. In order to generate a timing skewness between the channels, the signals are routed on four cables with different

length from the F/O board to the PP crate. The readout of the PPM is done via VME due to the lack of a ROD and a ROS.

A standard PC that is connected with the VME crate-controllers in the TTC and the PP crate via ethernet provides access to the modules. The PCs' RS-232 interface is used to program the AFG.

It should be mentioned at this point that this is not the only test-setup in Heidelberg. There are other ones mainly used for MCM and PPM production testing. In the framework of the PPM production tests, F/O boards are used in a cascade allowing to feed analog input to a whole PPM. Moreover modules have been developed that can check the PP readout and realtime output. For the calibration test setup, however, these modules are not very helpful. Firstly they would bring only little benefit w.r.t. what can be tested, and secondly they are not ATLAS standard. In fact it is rather foreseen to extend the test bench with JEMs, CPMs, RODs and a ROS once the production status of these items and the ones needed to use them allows so.

L1Calo at CERN

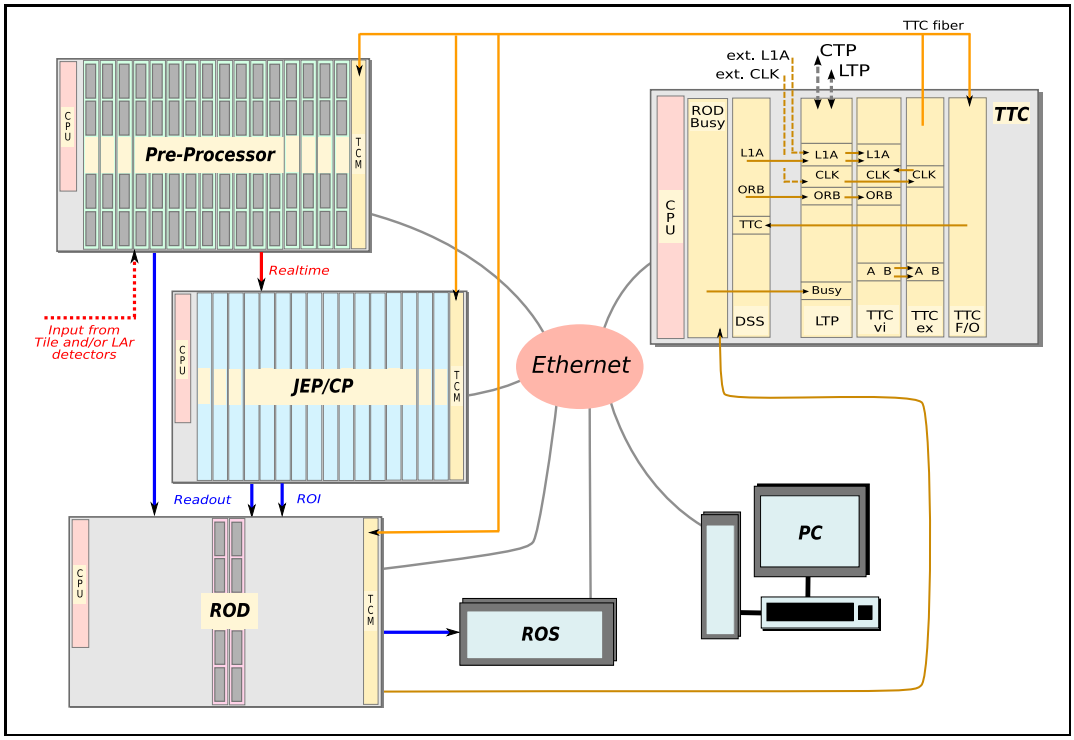

Figure 7.6: Overview of the Calorimeter Trigger Setup at CERN. Dashed lines indicate optional, connections.

Fig. 7.6 shows the setup of the system installed in the trigger cavern at CERN. The individual blocks are to be understood as logical units, the number of the individual items and the cabling between them may vary. The setup corresponds to the design of the Calorimeter Trigger as it is described in Chapter 5.

The focus of this section is on the TTC system whose setup is very much linked to how the system is operated. Concerning the TTC system, the ATLAS experiment is divided into about 40 so-called partitions one of which is the Calorimeter Trigger. Within the Calorimeter Trigger, the TTC signals are coming from a dedicated 6U VME crate and distributed via optical fibers.

The TTC crate holds

- a Single Board Computer to interface VME (CPU),
- a ROD Busy fan-in module,
- a Data Source Sink (DSS),
- a Local Trigger Processor (LTP),
- a TTCvi,
- a TTCex
- and an optical-to-electrical TTC fan-out board.

The TTCvi and TTCex are explained in the Heidelberg Setup, however not very much detailed. It should be added here, that in fact there are two channels going from the TTCvi to the TTCex, not only one. The channels are referred to as A and B channel. The A channel is used to transmit the clock signal and the L1As, the B channel is used to transmit commands and data. Beside that, four "inhibit" outputs can be programmed to send clock-synchronous signals a programmable time relative to the orbit counter. Another feature of the TTCvi is the so-called "B-Go" mechanism. Broadcast commands can be programmed into TTCvi registers and are encoded into the B channel on receipt of an external B-Go input signal.

The LTP can be seen as the root of the partition. Each ATLAS partition contains at least one LTP that provides a bi-directional connection to the CTP. This communication is realized with dedicated LTP-cables. Each LTP has two LTP-cable connectors which allows to operate them in a Daisy Chain. Moreover the LTPs have an internal memory that allows to generate TTC signals as well as NIM connectors that allow to provide them locally. To summarize, the LTP can be considered as a switch that allows to either run the system with the CTP, in a Daisy Chain, or locally using the LTPs memory or local signals. Additionally each LTP has a busy-input that suppresses trigger output if active. The busy signals are generally generated by the RODs. If the LTP is operated as slave in a Daisy Chain mode or connected to the CTP, the busy signal is forwarded and the L1As are suppressed by the Master LTP or the CTP, respectively. In the L1Calo Setup busy signals from all RODs are combined on a single link to the LTP on a busy signal fan-in module.

The LTP provides the largest flexibility to program how the TTC-system is used, consequently all inputs to the TTCvi are directly or indirectly fed from the LTP. The TTCex gets a clock input from the LTP and forwards it to the TTCvi. Furthermore it receives the A and B channel signals from the TTCvi and creates an optical link that is fanned out in a optical fan-out unit.

The DSS is developed at the Rutherford Appleton Laboratory (RAL) for and by the L1Calo collaboration. It is designed for test purposes and allows to send and

receive data. The DSS holds ten 32 kB Memories that are interfaced by FPGAs. Two of the FPGAs are directly connected to S-Link connectors on the motherboard, the rest to connectors that can hold daughter boards. There are three types of daughter boards: LVDS, G-Link and General purpose In Out (GIO). The DSS in the TTC crate holds a GIO daughter module and is used to generate local input to the LTP. Therefore one of the DSS memories is loaded with a sequence of L1As, B-Gos and Orbit-Counter Resets for the time span of eight orbit cycles. This pattern is output with 40 MHz up to a programmable memory address. After that during a number of N orbit-cycles no output is produced, then the procedure is restarted. The DSS uses a TTC signal from the optical to electrical TTC fan-out board to generate its clock.

Tile Charge Injection System (CIS)

As shown in Fig. 7.6 the CERN L1Calo system can receive external input signals at the analog PPM input and the TTC crate. One system used to generate these inputs is the Charge Injection System of the Tile Calorimeter. The CIS implementation on the detector is explained in section 3.4, here the control and timing setup is focused on.

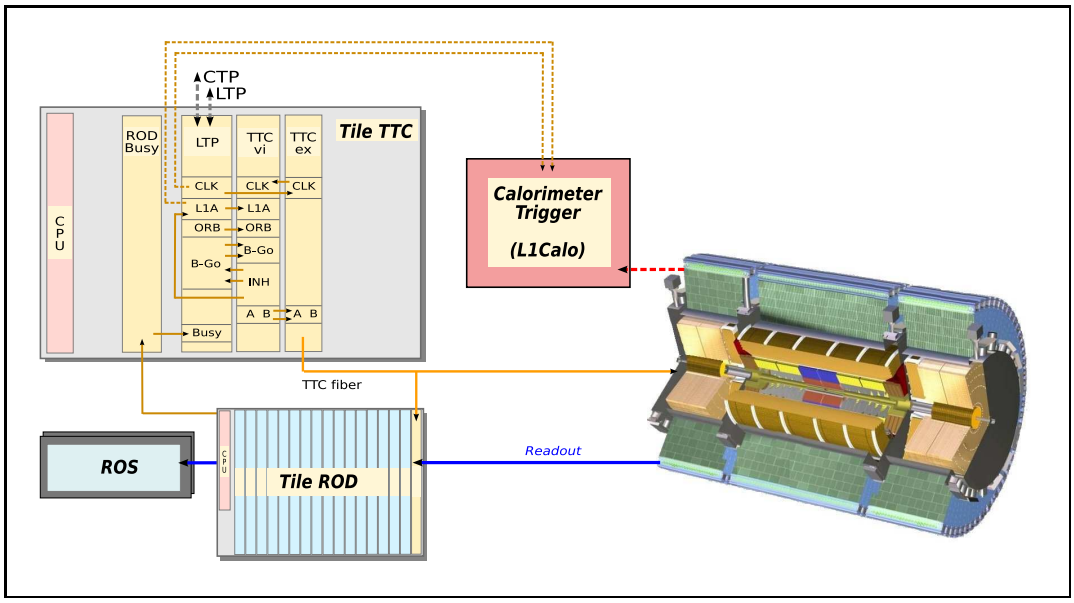

Figure 7.7: Overview of the CERN Setup when using Tile calorimeter CIS. Calorimeter picture: [13].

The Tile Calorimeter system is divided into four partitions, one for each extended barrel and two for the central barrel. The division into partitions corresponds to the alignment of super-drawers along the beam-axis. Each partition is controlled with its own TTC crate. Fig. 7.7 shows how a Tile Calorimeter TTC crate is used to run the Charge Injection System.

The clock distribution, the TTCex and the busy-handling are set up exactly as in the L1Calo Setup. The rest of the setup addresses the CIS configuration that is

based on the orbit counter. The orbit signal is passed from the LTP to the TTCvi. The TTCvi has three of the inhibit signals active, the Inhibit<i> registers are used to tune the timing. One signal (INHIBIT_0) is used to trigger the charging of the capacitors, another one (INHIBIT_1) for the charge injection into the signal chain, and the last signal (INHIBIT_2) triggers the L1A generation. The first two inhibit signals go to the external B-go inputs of the LTP and from the B-go outputs of the LTP to the B-go inputs of the TTCvi. The B-go<i> registers of the TTCvi are programmed according to the actions required: charge capacitors, inject charge. The INHIBIT_2 signal goes from the TTCvi to the local input of the LTP and from the L1A output of the LTP to the external L1A input of the TTCvi. On the TTCvi all signals are encoded in the A and B channel and passed to the TTCex. Fig. 7.8 gives an overview of the CIS timing.

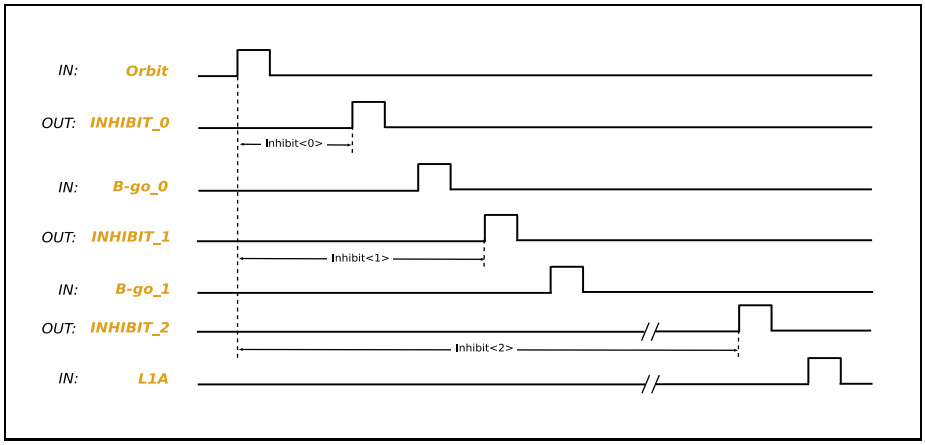

Figure 7.8: Sketch on the CIS timing at the TTCvi in and outputs.

The setup of the CIS system has historical reasons. It stems from a time where no LTPs existed. Based on the orbit counter, the CIS could perfectly live without an LTP by simply passing the inhibit outputs directly back to the B-go inputs of the TTCvi. Going the way through the LTP, however, allows to configure how to run the system by configuring the LTP, i.e. by software. Note that the CIS is controlled by non ATLAS Trigger DAQ (TDAQ) based software. However, support for the TDAQ-software is planned. The design of the Tile CIS is very flexible w.r.t configuring individual channels: All calorimeter cells can be individually configured to fire with different charges.

If more than one Tile Calorimeter partition is used, the LTPs of the TTC crates are Daisy chained and have a common orbit and clock, the charge injection is still done in the individual partitions.

When the Calorimeter Trigger is running with the CIS, it uses the clock and the L1As generated by the Tile TTC system. In principle this could happen by using a Tile Calorimeter fiber as input for the optical fan-out unit of the L1Calo Setup. However, in practice this is not possible, as L1Calo cannot handle readout with the orbit frequency without the busy-mechanism. There is no LTP cable connecting the two systems that would allow to pass the busy signal from the Calorimeter Trigger to the Tile Calorimeter TTC crate. Therefore a NIM copy of the Tile Calorimeter

clock and L1A is routed to the Calorimeter Trigger and used as "Local" input of the Calorimeter Trigger LTP. This way L1Calo can generate its own busy signal without interfering with the Tile Calorimeter. Another advantage this setup has is, that L1Calo can profit from runs, which the Tile Calorimeters do to test their system, without disturbing them.

Cosmic Runs

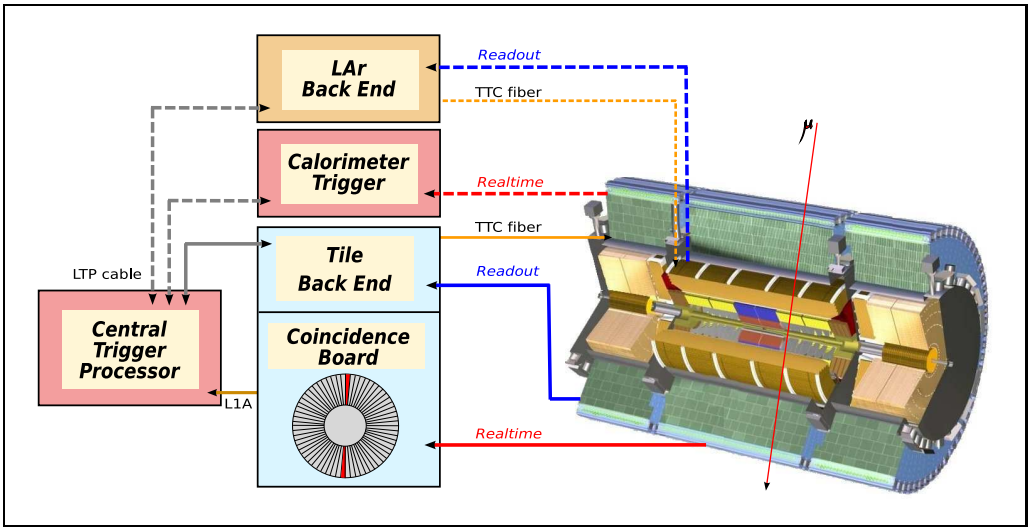

Figure 7.9: Setup for cosmic runs with integrated Central Trigger Processor. Dashed lines indicate optional connections. Calorimeter picture: [13].

The Tile Calorimeter Collaboration has a Coincidence Board that can handle 96 Trigger Tower channels that are processed in a two-dimensional coincidence matrix of rank 48. This logic allows to trigger on coincident signals in 48 top Trigger Towers and 48 bottom Trigger Towers and is used to detect cosmic muons. Unfortunately the Coincidence Board is using the signals that are normally used by the Calorimeter Trigger. This means that the Calorimeter Trigger is restricted to channels not used by the Coincidence trigger, i.e. channels that have only a little chance to see cosmic data. Therefore the Calorimeter Trigger is normally not participating in cosmic runs but it is foreseen to fed the Coincidence Board with the Level-1 Muon-Trigger input (Fig. 5.2) in order to have the possibility to see cosmic data in the Calorimeter Trigger. However, this requires additional hardware that was not available at the time this thesis was written. The setup shown in Fig. 7.9 has so far only been used in the Milestone runs (M runs) that are done to integrate the various ATLAS sub-systems.

The picture in Fig. 7.9 shows an integrated CTP that is receiving the L1A signals generated by the Coincidence Board. The CTP is then distributing the L1As to the sub-systems via LTP cables. Without CTP, one could use L1As from the Coincidence Board as "Local" input for a Tile Calorimeters' LTP. The Calorimeter Trigger could then be supplied with a NIM copy of the clock and L1A signals from the Tile Calorimeter. The rate of cosmic events is less than a Hertz. At this rate

the Calorimeter Trigger can safely operate without the busy-mechanism and using a Tile Calorimeter fiber in the Calorimeter Trigger would also be possible.

Laser System

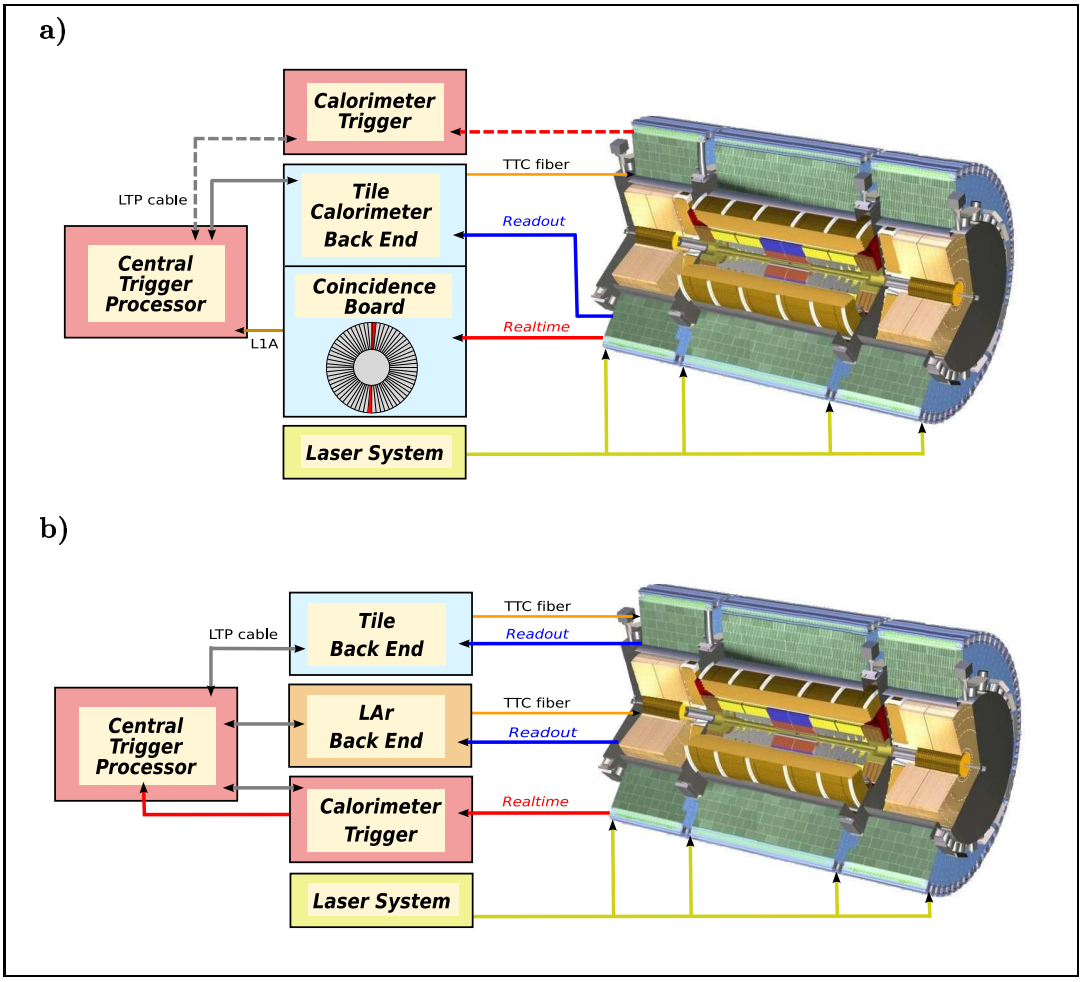

Figure 7.10: Setup using the Laser system with integrated Central Trigger Processor. Dashed lines indicate optional connections. Calorimeter picture: [13].

The Laser System (LS) is described in section 3.4. The fibers are designed such that pulses are injected into the system almost synchronously. Within one super-drawer, fiber lengths differ maximally by 5 m. The fiber length to the barrel and extended barrel is adjusted in order to simulate the timing typical for a particle that comes from the interaction point. All in all the maximal time difference for the pulse injection at different channels can be assumed to be about two clock ticks. This makes the LS very useful for the timing-calibration and allows to operate it in almost the same setup that is used for Cosmic runs. In fact the LS allows to adjust the timing for Cosmic runs providing signals for all channels at once and with a higher rate. The LS is, however, not synchronous with the LHC clock phase, but one can certainly assume that the same holds for cosmic muons.

Fig. 7.10 shows two setups using the laser system. As it supplies only the Tile Calorimeter, it is not really useful to integrate the LAr Calorimeter in an LS run. The accurate timing of the Laser System makes it possible to use it for a coarse timing-calibration. Therefore triggers could be generated as in the final system, i.e. in the CTP. Fig. 7.10 b) shows such a setup.

LAr pulser system

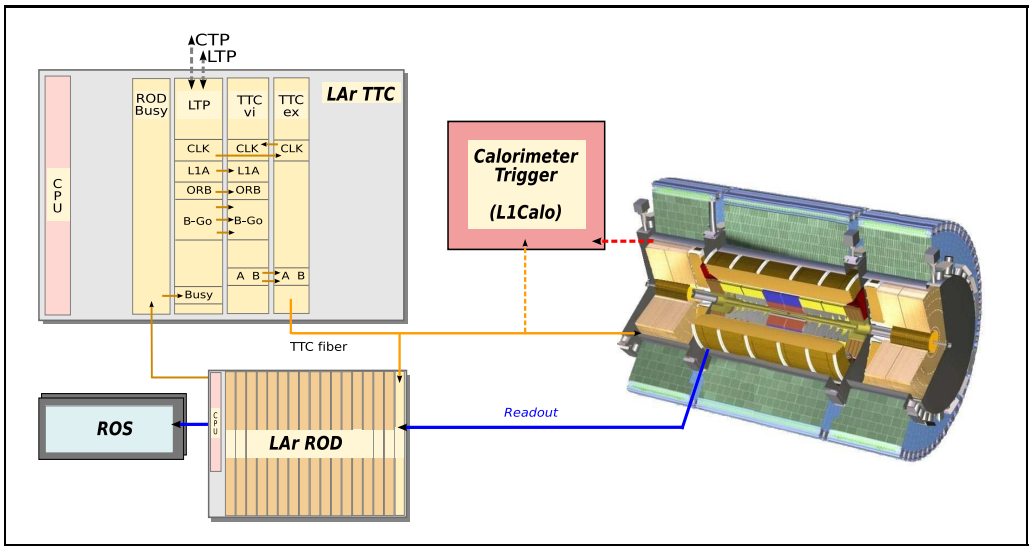

Figure 7.11: Setup for runs using the LAr pulser system. Calorimeter picture: [13].

Fig. 7.11 shows a setup for the LAr pulser system. The configuration of the Front End is done using the B-go mechanism. B-go signals and L1As are generated using the LTPs internal pattern generator. The LTP pattern generator is implemented using four 1 M deep and 4 b wide SRAMS, and can generate frequencies ≥ 38 Hz. B-gos and L1As are encoded on the TTC signal and broadcast to all Front End crates via an optical fiber. As the L1A frequency can be adjusted to a value that is uncritical for the Calorimeter Trigger readout, the busy signal is not mandatory to operate the Calorimeter Trigger. Therefore it can be provided with L1As and clock using a LAr TTC fiber.

The LAr system is composed of six partitions, four for the A and B side of the electromagnetic barrel and the electromagnetic end-cap, one for the hadronic end-caps and one for the the FCALs. If the pulser system is used in more than one partition the LTPs of the according TTC crates are operated in a Daisy Chain.

Beam Runs

Finally the system is designed to be operated with LHC beams. The setup that is used for beam runs is shown in Fig. 7.12. Clock and L1As are provided by the CTP, see Chapter 5 for details. The Calorimeter Trigger, calorimeter partitions and the rest of the ATLAS Experiment is connected to the CTP via LTP cables. The LTP

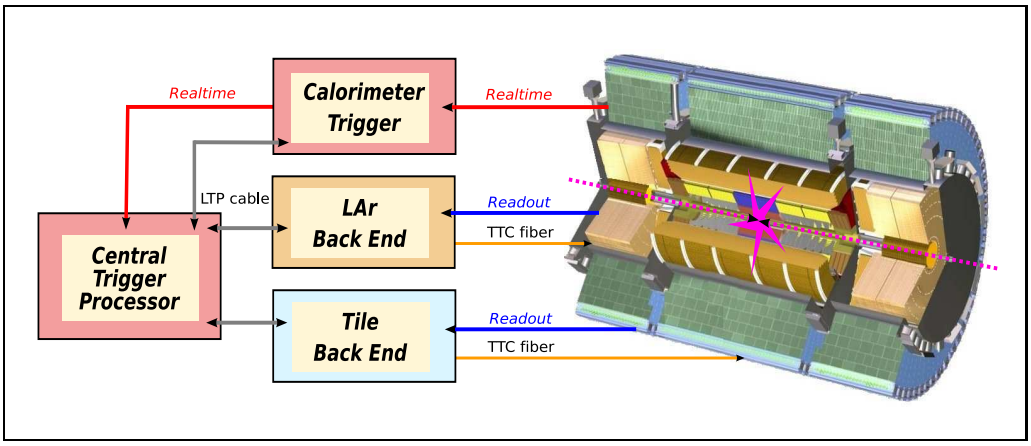

Figure 7.12: Setup for beam runs. Calorimeter picture: [13].

cables provide a clock signal and L1As in one direction and transmit busy signals in the other.

7.4 Software

The calibration software of the Calorimeter Trigger is based on the ATLAS Trigger-DAQ software (TDAQ S/W), i.e. the software that is used to configure and monitor the ATLAS data acquisition during a run. This does not comprise the software for offline data analysis and Monte Carlo studies. The TDAQ software has to control all items that are explained in Chapter 4 in order to have a consistently running system. Of course this task is very comprehensive and so is the TDAQ software. Here the principle design and a selection of the software used for the Calorimeter Trigger calibration is introduced.

Apart from diverse external packages in ATLAS there are three major software areas: the LHC Computing Grid (LCG) software, the TDAQ S/W and the software provided by the various ATLAS detector and trigger groups like e.g. L1Calo. Fig. 7.13 gives a coarse and incomplete overview of the packages of the three areas. Note that some of the packages shown are really implemented as a set of packages. As these sets belong together logically, they are referred to as one package in the context of this thesis for simplicity's sake.

The LCG provides packages used by all LHC experiments. The SEAL package contains all kind of libraries, tools and frameworks, covering a broad range of unrelated functionalities. SEAL addresses all kind of problems that commonly arise when writing software and do not have standard solutions.

The COmmon Relational Abstraction Layer (CORAL) provides a common Application Programming Interface (API) to different database technologies: Oracle, MySQL and SQLite. This allows to run the same code independent of which kind of database is available at different sites. POOL stands for Pool Of persistent Objects for LHC. POOL is designed to store the data produced by the LHC experiments in a distributed manner. In some sense POOL is the interface to the offline analysis world. POOL mainly addresses event and detector data but also physics and detec-

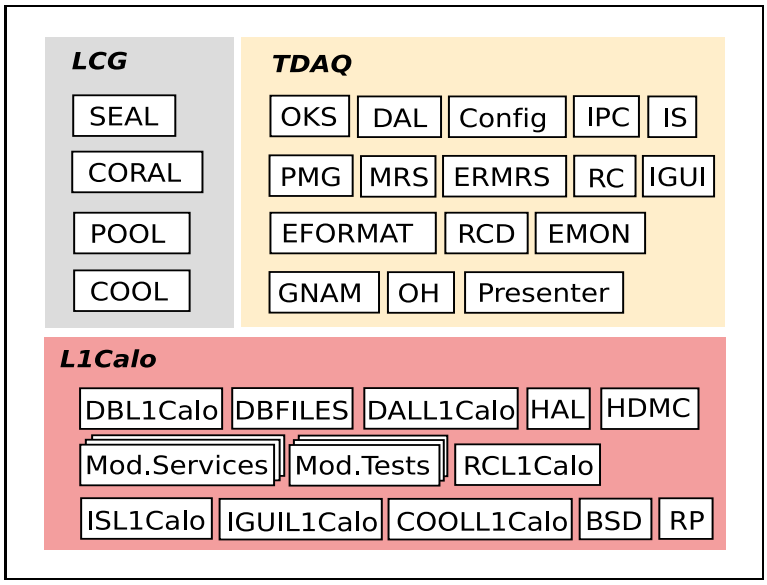

Figure 7.13: Coarse and incomplete overview of the LCG, ATLAS-TDAQ and L1Calo software packages. The TDAQ depends on the LCG software and the L1Calo on both the LCG and the TDAQ software.

tor simulation and bookkeeping data. COOL (COnditions Objects for LHC) uses CORAL and provides an API to store and retrieve all kind of "conditions data". Where event data comprise the measurements done with the detector, the conditions data describe the condition of the detector and the trigger itself. This includes e.g. calibration constants.

The Object Kernel Support (OKS) is a library that manages in-memory objects and can be used as a kind of fast database with restricted database functionalities. OKS has a C++ API and follows an Object Oriented (OO) approach. Class definitions and their instances are stored in eXtendensible Markup Language (XML) files. The XML files are divided into two sets. One set, the schema files, contains class definitions. The other set, the data files, contains the description of class instances including their member data values. The Data Access Layer (DAL) package provides a code generator that can read the schema files and generate Java and C++ code for the defined classes. A Configuration class provided by the Config package parses the XML files using an OKSKernel object. The Configuration object creates instances of DAL generated classes, according to how they are described in the data files. The Configuration object can then be used to query the OKS data content from within an user application.

In ATLAS, OKS is used to define a setup configuration, i.e. the configuration of all involved TTC partitions. The configuration of a partitions comprises the configuration of the hardware and the software setup, where the software setup describes all the processes that have to run on the various machines.

The Inter Process Communication (IPC) package is one of the most vital ones in the TDAQ S/W. IPC is a Common Object Request Broker Architecture (CORBA) implementation. It uses a public domain package called Inter Language Unification (ILU) and allows processes written in different programming languages and running

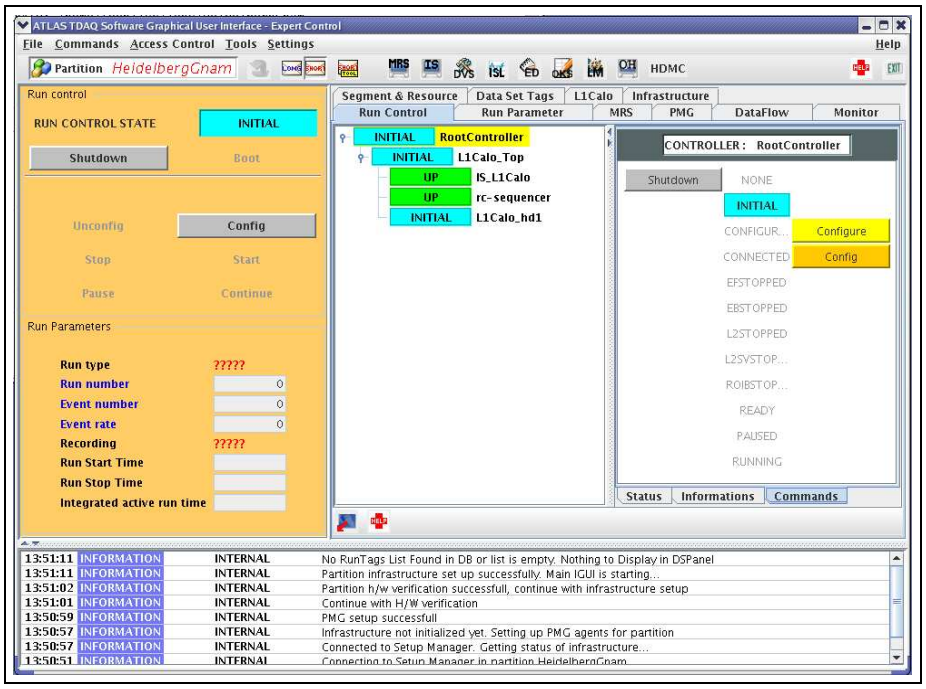

Figure 7.14: Screenshot of the Integrated Graphical User Interface (IGUI).

on different computers to work together.

Like many TDAQ packages the Information Service (IS) package uses the IPC package. IS provides services used to share information between processes. The processes can provide (insert, update, remove) and search information in form of so-called IS variables in information-repositories. Beside that, processes can subscribe for notification if an IS variable is changed.

The Process Manager (PMG) package allows to manage distributed processes on a network. Once the PMG infrastructure is set up all kind of processes can be started and monitored on all machines in a network.

The Message Reporting System (MRS) and Error Message Reporting System (ERMRS) provide the infrastructure for a common information-, debug- and error-message handling in the TDAQ framework.

The Run Control (RC) is used to control the data taking activities of the TDAQ components. Each TDAQ group is responsible for implementing run-controllers for their system using the API and base classes defined in the RC package. The run-controllers have the task to bring the TDAQ system into a state that is appropriate for data taking. Of course, bringing up a system like the TDAQ is not trivial. in order to do so in an organized and synchronous way the run-controllers are designed as state machines with several specified states. To allow the transition of all run-controllers from a single point of control, the run-controllers are organized in a hierarchical tree. run-controllers that are higher in hierarchy take care of the run-controllers they are responsible for. State transitions are evoked by the user, i.e. the shift crew, from the Integrated Graphical User Interface (IGUI).

The IGUI provides the interface to the user, a screenshot is shown in Fig. 7.14.

It is implemented in JAVA and consists of three blocks. On the left side it allows to toggle between the Run Control states. At the bottom MRS and ERMRS messages are displayed. On the right there is the IGUI tab pane that is dedicated to the TDAQ subsystems. The tab pane is configurable, i.e. tabs that have been implemented in a specified way can be added to the tab pane. This allows the various sub-groups to have their own area in the IGUI. The communication between the IGUI and the back-end software generally happens via IS.

The EFORMAT package implements the ATLAS data event format. The data is in a bytestream that is produced in the setup shown on the DAQ side of Fig. 4.1. The bytestream contains a hierarchical structure of nested fragments that are identified by header words in the stream. The hierarchy of the fragments reflect the setup the bytestream is produced in. On the highest level it is initiated by an Event Header that indicates that the following data belong to one ATLAS event. The Event header contains fragments for the various ATLAS sub-detectors. With respect to the integration in TDAQ the trigger groups are generally regarded as sub-detectors. The sub-detector fragments are split into fragments that contain data from one ROS. The ROS fragments contain ROB fragments, and the ROB fragment contain ROD fragments. The data in a ROD fragment are supplied by the RODs of the individual sub-detectors and may vary in length. The end of a ROD fragment is marked by ROD trailer. The bytestream format up to the ROD level is common for all of ATLAS and handled by the EFORMAT package. The code required to handle the data in a ROD Fragment must be provided by the sub-detector collaborations.

The EMON, RCD, GNAM, OH and Presenter packages of the TDAQ S/W handle the system monitoring. Therefore the data stream of the DAQ path is wiretapped for monitoring at various points in the TDAQ system. These points are the ROD, ROS and the SFI. The difference in the data from the tree sources lies in the trigger level they have passed. ROD data is Level-1 filtered only, ROS data have passed the Level-2 stage and SFI data the Event filter. As the RODs are hardware-wise located in crates, an infrastructure is required to read out ROD crates via VME and encode the data in ATLAS EFORMAT. This infrastructure is provided by the ROD-Crate-DAQ (RCD) package.

The EMON package provides Event Monitoring Services (EMS) the monitoring raw data is published to. The data in the EMS can be retrieved by processes called Monitoring Processes (MP). One kind of MPs are GNAM processes. GNAM stands for (GNAM is Not AtlMon), the GNAM processes follow a workflow that is illustrated in Fig. 7.15.

GNAM is implemented as a plugin application. It consists of a GNAM core and the plugin libraries that have to be provided by the sub-detectors. The GNAM core retrieves the event formatted data from the Event Monitoring Service and decodes it up to the ROD level. The data format on the ROD level is sub-detector specific and decoded by a libDecode plugin that has to be provided by a sub-detector collaboration. The libDecode library returns the decoded library in form of so-called Raw Data Objects (RDO). The RDOs are then passed to one or more histogramming plugins (libHisto) by the GNAM core. The libHistos are also provided by the sub-detector groups and produce histograms out of the RDOs. The histograms are

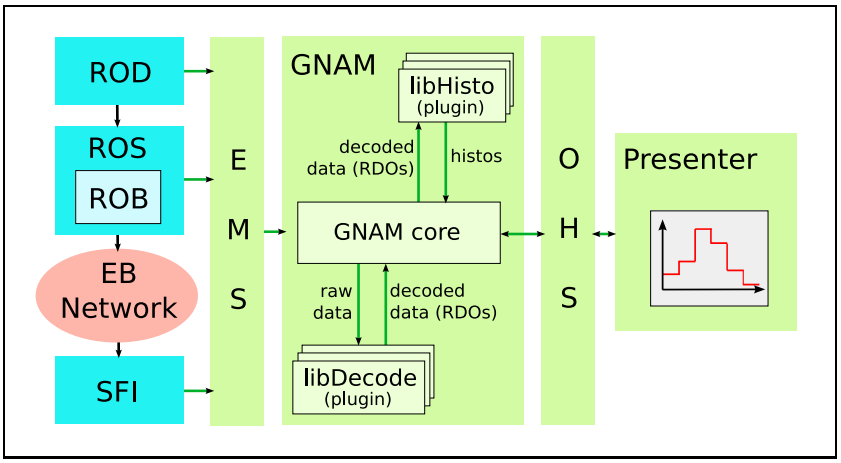

Figure 7.15: Overview of the GNAM workflow.

then published on the Online Histogramming Service (OHS) from the OH package. Several "presenter" programs exist that retrieve the histograms from the OHS and display them in a GUI. Commands like reset, rebin or update can be entered in the presenter and are passed to the GNAM Process via the OHS. In the GNAM process they are forwarded to the histogramming library in charge.

The TDAQ software is started by a script called setup_daq. The OKS database is read from the script that then launches the basic infrastructure, i.e. IPC and PMG processes and several services. Then the script starts the IGUI and the root (top level) run-controller. The root run-controller starts the rest of the processes as defined in the OKS description.

The L1Calo package that reads the OKS configuration is the DBL1Calo package. DBFILES contains L1Calo OKS schema and data files. The code generated from the schema files is in the DALL1Calo package. The Hardware Access Layer (HAL) and the Hardware Description Monitoring and Control (HDMC) package contain code for basic hardware access.

HDMC is a C++ based project developed for L1Calo but designed to be used as a general tool. In HDMC, hardware as for example (VME) buses, FPGAs, memories or registers is defined as HDMC parts. This means that hardware is described by means of classes that inherit the HDMC part class. HDMC parts can be linked together in order to describe a hardware setup. In the pure HDMC framework an ASCII file (Parts-file) is used to define how the parts are linked together. In L1Calo a mixture of ASCII files the OKS is used.

The hardware access to all L1Calo modules is implemented by means of HDMC parts in the various ModuleServices packages. The ModuleTest packages contain all kind of test applications. The RCL1Calo package contains the L1Calo specific run-controllers. These use the Module Services packages in order to perform the transition actions with the hardware. Therefore the ModuleServices classes do not only implement HDMC parts, but also DaqInterfaces that declare the transition actions.

ISL1Calo contains the definitions of the IS Variables needed by the L1Calo soft-

ware to communicate via the Information Services. The code in ISL1Calo is automatically generated using OKS schema files. IGUIL1Calo contains the JAVA classes for the L1Calo IGUI panel. The interface between L1Calo and the COOL database is managed by the COOL1Calo package. The Byte Stream Decoder (BSD) package provides a library to decode the L1Calo ROD specific bytestream. The Runplan (RP) package allows to perform automatically a series of runs as for example required for a calibration procedure.

7.5 Concepts and Procedures

The basic idea of the calibration is to take data while scanning a range of calibration parameters and analyze the result in order to derive calibration settings. To allow the scanning of the parameters to happen system-wide, it is useful to use the Run Control (RC) state machine. L1Calo defines several Runtypes, that can be chosen in the L1Calo tab of the IGUI. The Runtype is required to identify the information how to do a run from OKS and COOL.

Multistep runs

One type of runs used for parameter-scanning are the so-called Multistep-runs (MS-runs). The idea of MS-runs is to scan a parameter range in several steps that are separated by a set of RC state transitions. MS-Runtypes define which parameters to scan. Generally only one parameter is scanned at a time, and the sequence can be defined by setting a Start Value, a Scan-Parameter Increment, and the Number of Sequence Steps. To have the flexibility to analyze the data from a MS-run in different manners, an Analysistype is defined separately from the Runtype. The Sequence Definition, the Analysistype and other variables needed to actually do MS-runs are defined in COOL Folders, i.e. conditions-database tables. These folders are referred to as (Run-)Configuration Folders. There is one Configuration Folder per Runtype. The rows of the Configuration Folders are linked to the parameters the Runtype has, the columns contain different sets of settings for these parameters and are addressed via a Runtype version. Both, the Runtype and the Runtype version can be selected in the L1Calo-tab of the IGUI and are published in IS. Non-standard so-called User Broadcast state transitions are used to toggle from one set of parameter values to another. The transition states involved when stepping are:

1. stopTrigger: ROD goes busy ⇒ suppress L1As.
2. endStep: End of the previous step ⇒ store/analyze data.
3. beginStep0: Empty buffers where necessary and update register values in database objects.
4. beginStep1: Write new settings to hardware.
5. startTrigger: ROD busy goes down ⇒ L1As.

Once a user is starting a Multistep run, a program called sequencer is taking over the stepping. The sequencer knows the run definition from COOL and changes

the IS-variables that are triggering the state transitions accordingly. When the IS-parameters change the L1Calo top-level run-controller is informed and starts the transitions on all run-controllers it is responsible for. The run-controllers know about the definition of the Multistep run from COOL and perform the actions required for stepping, counting the steps themselves.

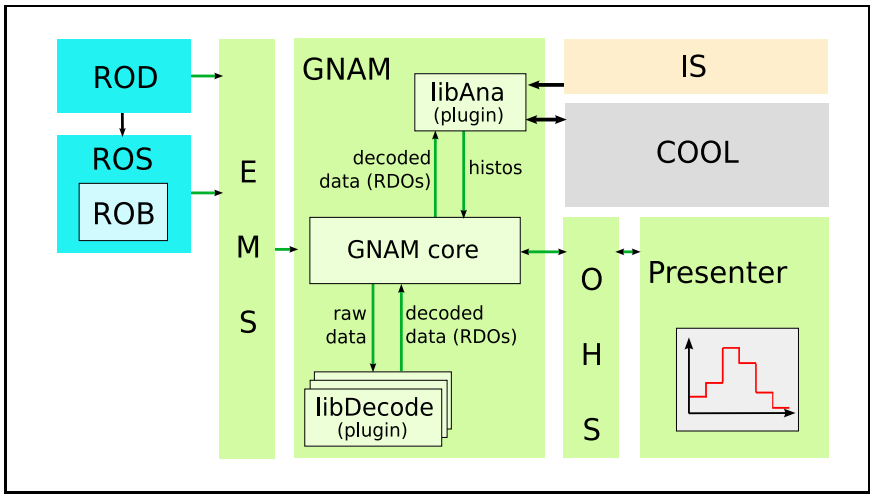

Figure 7.16: Overview of the use of GNAM in a calibration run. The step-number is encoded in the bytestream.

Fig. 7.16 shows how GNAM is used in a Multistep run. As readout data from the entire L1Calo system is published to the Event Monitoring Service (EMS) via the ROD and the subsequent ROS, this is a good place to pick it up in order to analyze it in the context of a calibration. A tool that can be used to do so exists already in the Monitoring framework: GNAM. GNAM does not know about Multistep-runs, but the step-number is encoded into the bytestream by the ROD. Remember, that the data-format within a ROD-fragment is defined by the sub-detectors. Encoding the step-number in the bytestream assures that it is persistified together with the data. The ROD knows about the step-number from the ROD-crate run-controller that is updating a register at each step.

The GNAM core uses a decoder library in order to decode the ROD fragments. The decoder library uses the L1Calo bytestream-decoder and produces RDOs that contain the step-number. This way a histogramming plugin gets information about the step-number. The idea of using GNAM for calibration purposes is to write a histogramming plugin, use it for data analysis and call it analysis plugin. Of course such a plugin needs information of the actual run and therefore reads the Runtype and Runtype version directly from IS. The analysis plugin creates a COOL access, not only to read run-parameters like the Runtype, Runtype-version the run-con etc... but also to store analysis results to COOL. The PPM analysis plugin is designed such that the analysis in each step is kept minimal and a final analysis of all steps is done at the end of the run. Histograms are created using the analysis results that are published to the OHS and can be browsed by means of a Presenter.

Roughly spoken a MS-run involving a GNAM process with analysis plugin can be summarized like this:

1. User: set Runtype, Runtype version and start MS-run.
2. Run-controller: configure H/W.
3. GNAM: take data ⇒ minor analysis.
4. Sequencer: toggle step or go to 7.
5. GNAM: take data ⇒ minor analysis.
6. Go to 4.
7. GNAM: analyze data from all steps,
   store result to COOL and publish Histograms to OHS.
8. Sequencer: Stop MS-run

Calibration Runtypes

A summary of the calibration aligned parameters is given in Section 7.1 and Section 7.2. Not all of these parameters must/can be derived via scans by means of MS-runs. For some parameters scans are not required as there are fixed relations between parameters, i.e. having a good setting for one determines a good setting for another one. For parameters in the digital part of the system scans are not necessary as the influence of parameter changes for given input data can be computed.

Nevertheless all the information required in order to find suitable settings for all calibration-parameters must be measured in the Calibration runs. Storing all data recorded in a calibration run, however, would be be too much for a database technology like COOL. On the other hand the information stored from the calibration runs must be sufficient to derive register settings for all kind of configurations, considering the parameter inter-dependences. Practice has shown that the best solution is to stay close to the measured information without trying to translate it into the PPM register model. The result of a calibration run should contain as much as possible of the pure calibration-run result with a moderate number of variables to store.

All of the Pre-Processor (PP) related MS-runs give a feedback whether the run was successful or not on a channel by channel basis. If serious problems occur the run fails. Serious problems may come from inconsistencies between the data, the COOL database and the OKS Configuration or the data quality. If the data do not allow to derive Calibration Results, a run will fail. This can either be the case because the data are corrupted, the number of steps is insufficient, or no data have been taken at all. A run that has not failed is classified as successful. If a Calibration Run is successful, the derived Calibration Results can still be critical. An acceptance range for all Calibration Results is defined in the Configuration Folders. If Calibration Results are not in the given range, they are classified as critical. Another way a Calibration-Result may become critical is when the RMS of a data-sample that has been taken with the same configuration, e.g. within one step, exceeds a programmable threshold. If a Calibration-Result becomes critical it is up to the user to make the decision whether to use it or not.

Several Runtypes most of which are MS-runs are used to scan the parameter range of the PPM that is required to derive register settings for all calibration parameters.

DAC Scan

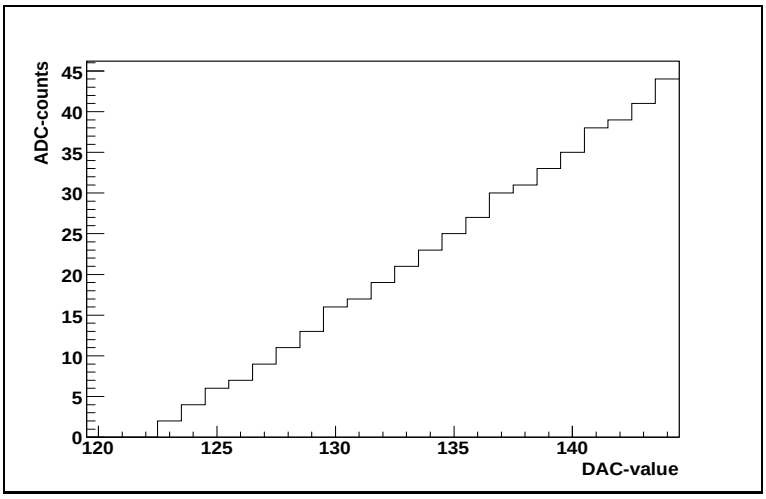

Figure 7.17: Result of a DAC Scan using Tile Calorimeter input. Note that there are no error bars due to limitations in the GNAM framework that still have to be sorted out.

The DAC Scan addresses the only energy-calibration related parameter in the analog part of the Pre-Processor, the Baseline DAC or simply DAC setting on the Analog Input board. The influence of this parameter on the Processing chain is explained in section 6.2.

To be able to measure baseline drifts, the complete noise distribution, and signal undershoots as shown in Fig.3.11, the offset of the digitization window of the FADC has to be adjusted. The baseline has to correspond to some positive number of FADC-counts. This value is called pedestal and a typical pedestal value is 40 FADC-counts. With a pedestal value of 40 FADC-counts the Calorimeter Trigger input spans an energy-range from -10 to 240 GeV.

The purpose of the DAC Scan is to determine a DAC-value FADC-count relation, that allows to decide which DAC value is needed for a desired pedestal. To do so, the DAC Scan has to be performed measuring noise with as many of the electronics in front of the PP switched on as possible. No signal (pulse) must be digitized during a DAC Scan which can be achieved by either having all possible signal sources switched off or making sure that the timing settings of the PP avoid that any signal is digitized.

During a DAC Scan only the PprDAC Bitfield is incremented, Fig. 7.17 shows a measured DAC-value pedestal relation or DAC Ramp. At the end of a DAC Scan a Linear Regression is applied to the measured data, and only the result of that regression, i.e. y-axis (ADC-counts) intercept, the slope and the cross-correlation coefficient are stored to COOL.

Pedestal Run

The Pedestal Run is not a MS run, thus no parameter is scanned. In a Pedestal Run the DAC value is adjusted according to a given pedestal value and data is taken. The Pedestal Run allows to check whether the derivation of the DAC setting from the DAC Scan actually works.

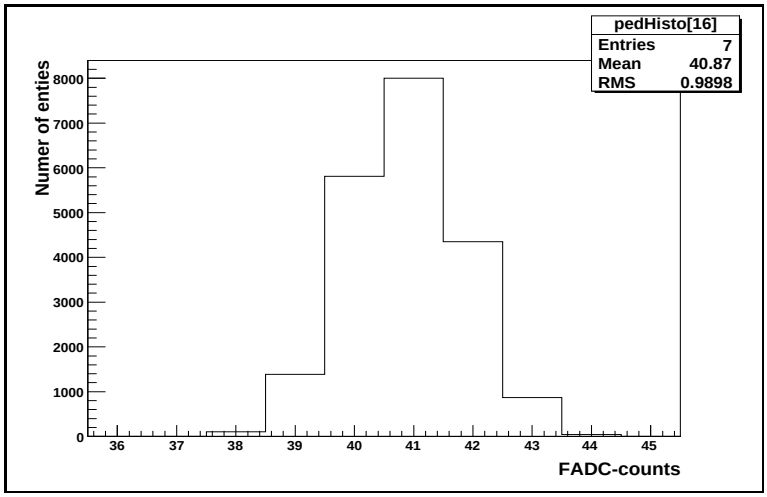

Figure 7.18: Result of a Pedestal Run using Tile Calorimeter input, with all of the FE electronics switched on. The DAC has been adjusted to get a pedestal of 40 FADC-counts.

Fig. 7.18 shows the pedestal distribution measured in a Pedestal Run. Due to the slope of the DAC Ramp, which is approximately 2 FADC-counts/DAC-increment, the pedestal can only be adjusted with a precision of ± 2 FADC-counts. This precision is acceptable as the precision of the final energy estimation is four times smaller than the precision of the FADC. The FADC result has 10, the BCID result eight bits. The Results of the Pedestal Run that are sored to COOL are the mean and RMS of the pedestal distribution.

Readout-Pointer Scan

Section 7.1 gives an overview of the timing related parameters. Concerning the readout path, the SyncDelayRaw and the PipeDelayRaw setting have the same effect. However, only the SyncDelayRaw setting has an effect on the realtime path. Let's assume the propagation time of an L1A from where it is generated to all PP channels is the same or at least known. Under this assumption settings for the SyncDelayRaw parameter can be determined from the PipeDelayRaw settings, given the correct values for latter are known. Moreover correct settings for PipeDelayBCID parameters can be derived from the PipeDelayRaw values as there is a fixed offset of eight clock ticks between the PipeMemory for the raw and the BCID data.

When running with a pulser system one can assure that there is a signal in all channels for every L1A. In that case, a simple approach to find a good setting for the PipeDelayRaw parameter would be to adjust the PipeDelayRaw and NumBCRaw parameter in order to readout a whole PipeMemory. Then one could search for the

maximum value and store the cell position it was found in. Unfortunately this is not possible due to technical reasons. The number of slices that can be read out via the ROD is limited to a programmable number between one and 5, the NumBcRaw must be between one and 5, the default value is five.

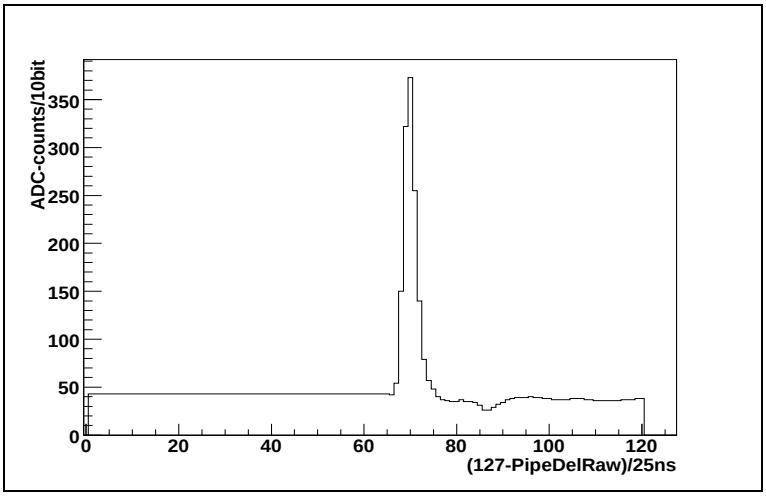

Figure 7.19: Result of a Readout-Pointer Scan using Tile Calorimeter input and the Charge Injection System. The content of the Raw PipeMemory has been pieced together reading out sets of five consecutive memory cells.

The purpose of the Readout-Pointer Scan is to reconstruct the whole of the RAW PipeMemory buffer, by reading it out in consecutive pieces of one to 5 cells in the MS-run framework. Therefore the PipeDelayRaw parameter is incremented in steps of the NumBcRaw value and the PipeMemory content is pieced together. Fig. 7.19 shows the result of an Readout-Pointer Scan. The Readout-Pointer Scan allows to adjust the timing up to the precision of one clock tick. The only finding of the Readout-Pointer Scan that is stored to COOL is the position of the maximum value.

The number of events that is captured in a step, e.g. for one section of the PipeMemory, depends on the MS setup and the trigger rate. Usually more than one event is captured in a step. The default analysis of the Readout-Pointer Scan averages over all of these events. However, this makes only sense if one expects the same data for all events, i.e. if a pulser system is used. Obviously the situation is different when using a data source, like cosmic muons or beam that cannot be controlled in the same way. In these cases an alternative Analysistype has to be used that takes the maximum value over all events for a given memory entry. This way the reconstructed PipeMemory content has a maximum at the correct position, given there has been a signal in one of the events captured when looking at the correct section.

PHOS4 Scan

Using a Readout-Pointer Scan, the timing of the Pre-Processor can be determined to a precision of one clock tick, which corresponds to 25 ns. The PHOS4 chip on the

MCM, however, allows to adjust the latching time of the FADC up to a precision of one nanosecond. It has been shown that it is possible to determine the maximum position of a signal with 1 ns precision by fitting the 25 ns precision data [24]. This is the way the fine-timing of the PP could be derived when the signals are non-repetitive, i.e. come from cosmic muons or the beam.

Given there are repetitive input signals, i.e. a pulser system is used, the position of the maximum value can directly be measured by stepping the PHOS4 value from 0 to 24. The PHOS4 Scan is the MS-Runtype, that addresses the stepping of the PHOS4 value. In section 7.1 we have seen that there are inter-dependences between the PHOS4 value, the ASIC input-data latch edge (InDataNegEdge) and the Delay of the Raw data have to be considered when changing the PHOS4 value (Appendix B). Therefore not only the PHOS4, but also the InDataNegEdge and the PipeDelayRaw setting are changed in a PHOS4 Scan.

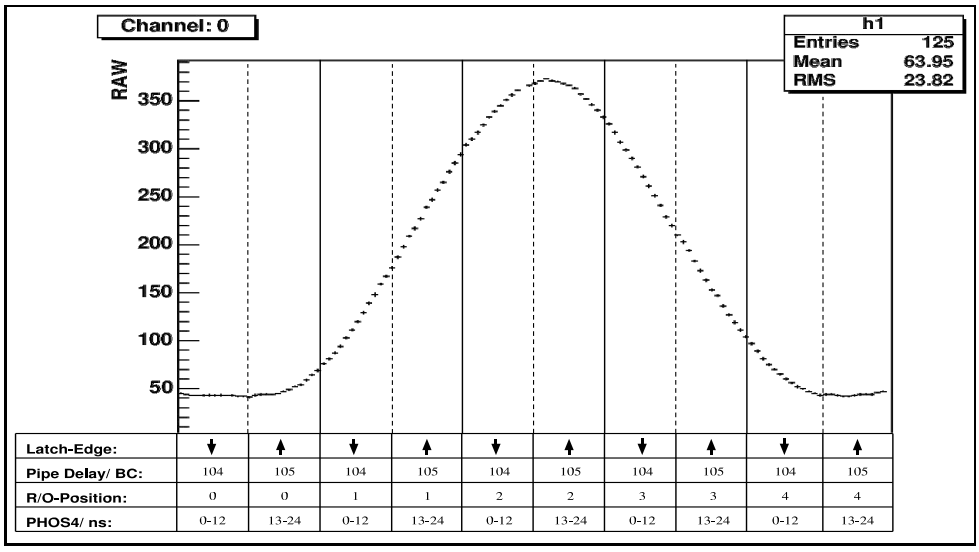

Figure 7.20: Result of a PHOS4 Scan in the Heidelberg Setup. with the reconstructed pulse shape and the aligned parameter settings.

Fig. 7.20 shows the result of a PHOS4 Scan in the Heidelberg Setup. The PHOS4 Scan can not only be used to find the peak position of the signal, but also to measure the shape of a repetitive signal with a one nanosecond resolution. In fact the PP can be used as a 7200 channel, 1 GHz sampling oscilloscope.

By default the results of a PHOS4 Scan that are stored to COOL are the PHOS4-delay for the maximum value, the InDataNegEdge and the PipeDelayRaw setting that takes the correct PHOS4 setting into account. Beside that, the analysis of the PHOS4 Scan identifies the channels which have an input signal by checking whether a programmable number of consecutive slices is above a programmable threshold. Channels that are identified to have an input signal are labeled.

Energy Scan

All of the Energy related parameters, except the DAC are in the digital part of the signal chain. As Fig. 6.6 shows the readout of the FADC PipeMemory delivers

exactly the data that is input for the part of the signal chain, dedicated to energy-calibration. If one measures the input to that chain for known energies it is possible to find calibration settings for the FIR Filter, the Drop Bits logic and the LUT. Please refer to Fig. 7.3 for details.

Both, the Tile Calorimeter and the LAr Calorimeter have pulser systems that allow them to induce pulses with given nominal energies into the signal chain. The idea of the Energy Scan is to use these Pulser systems, scan over a given energy range and gather the information required to do an energy-calibration.

Obviously an Energy Scan needs to combine the calorimeters and the Calorimeter Trigger in a combined run. At the time this thesis is written combined runs including the Calorimeter Trigger are just being started and still on a basic level. For practical reasons performing an Energy Scan additionally implies that the pulser systems can be controlled via the standard ATLAS TDAQ software. This is true in case of the LAr and aimed at, but not yet possible, in case of the Tile Calorimeter.

Another problem is that it is not yet decided how Energy Scans will be implemented. One possibility would be to use the Multistep framework. However, MS runs so far are only adopted by the Calorimeter Trigger, and it is not clear that the Calorimeters will adopt this framework. Without MS-runs it would be possible to send pulses with different energies in an agreed pattern related to the event number, i.e. the L1A: send first a 100 L1As with pulses of one energy, then another 100 with another energy etc. This scheme may, however, become problematic when the L1As are not generated by the pulser system or RODs go busy. The easiest, but also most time consuming, option would be to dedicate entire runs to a certain energy.

COOL Folders and Run Organization

So far we have seen two examples for the usage of COOL Folders, one of which are the Run-Configuration Folders. The other Folders, although they have not been explicitly mentioned, are the folders used to store the Calibration Results: the Calibration-Result Folders. An overview of all Pre-Processor aligned COOL Folders is given in Appendix C.

In order to minimize the amount of redundant data in COOL, there is one folder for the results of every Calibration Run. Beside that there is a set of folders that contain default register values for all registers of the Pre-Processor and one folder that contains calibration results for all of the Calibration Runs, the Validated Calibration Results Folder. In order to reveal trends in the system it is foreseen to do Calibration Runs a few times per month. In the time after a beam dump and before injection of a new one, this can be done without interfering with the data acquisition. It is, however, not advisable to change the calibration settings each time they have changed only marginally. Therefore it is foreseen to start runs using the settings in the Validated Calibration Results Folder and only update these ones if parameters have significantly changed.

Fig. 7.21 illustrates the workflow of a Calibration run. The diagram is simplified for comprehensibility's sake. It should not be considered as a strict UML diagram but give an idea, how a Calibration Run in principle works.

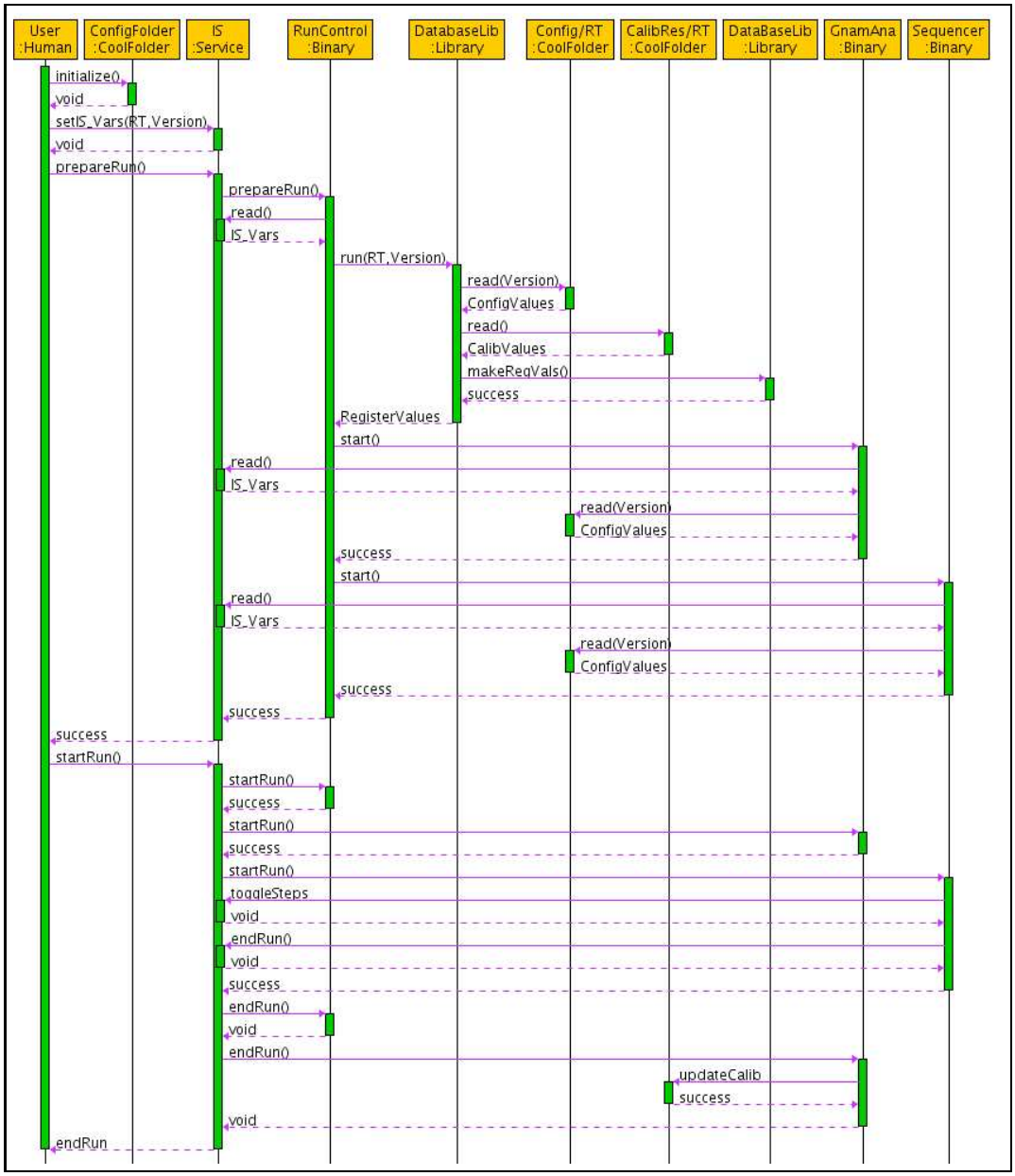

Figure 7.21: Sequence diagram for the workflow of a Calibration Run. Details in the text.

The first thing a user has to do before starting a run is to initialize the COOL Folders which can be done using dedicated programs. Once the folders are initialized and the run infrastructure is set up the user has to select a Runtype and Runtype version via the IGUI. Once this is done one has to go through all the transitions of the RC state machine that prepare the system for data taking. In the diagram the relevant issues of this phase are summarized in the prepareRun() method. Each time a transition is done the run-controllers are informed. The first thing the run-controllers do is to determine the Runtype and the Runtype version from IS. Once they have this information, they read from COOL the Run-Configuration Folder, that contains the information how to setup the system.

It has been mentioned above, that the data stored in the Calibration-Result Folders is close to the information that has been measured in the Calibration Runs. The Run-Configuration Folder contains the information which Calibration Results to use, i.e. Calibration Results from which Calibration Run to use. Additionally it has the information required to determine register values for the Pre-Processor Modules using the Calibration Results. The assignment of register values is done in the Database Library, indicated by the makeRegVals() method in the diagram. An example: The DAC Scan result are the slope and the y-axis intercept of the DAC Ramp. The Configuration Folder of a run contains the desired pedestal value, and the Database Library derives an adequate DAC value, using the relation $DAC = (pedestal - intercept)/slope$. As mentioned above, it is not yet decided how the Energy Scan is implemented. However, the results of an Energy Scan are already known and the part determining calibration settings from Energy Scan results is implemented.

The class in the Database Library that is computing the register values is called PprRegValMaker. The findings of the PprRegvalMaker are assigned to data members of a class called PpmCal. It is PpmCal objects that are used by the run-controllers to configure the PPMs. Before any of the data members in a PpmCal object gets updated they are initialized with values stored in the Default register values Folders, refer to Appendix C for details. This part is not shown in the diagram.

If Calibration Results from more than one Calibration Run are used, the procedure described above is repeated, each time updating values of the PpmCal objects. The order in which the various Calibration Results are used to update the PpmCal objects is fixed. It is the same order in which the Calibration Runs are done, to generate a Pre-Processor calibration from scratch when using a pulser system: first the DAC Scan, then the Pedestal Run followed by the Readout-Pointer and the PHOS4 Scan. Last not least the Energy Scan. As the order of Calibration Runs is fixed, they are limited in the Calibration Results they can use. The DAC Scan e.g. is alway done with Default register values, the Pedestal Run can use DAC Scan results etc... .

One task of the run-controllers is to start all kind of processes, as they are defined in the OKS configuration. In case of a Multistep Calibration Run this must include a GNAM process with decoder and analysis plugin, "GnamAna" in the diagram, and the Sequencer program. Both the GnamAna and the Sequencer program first determine Runtype and Runtype version from IS and then read the

Run-Configuration Folder to figure out what they have to do.

Once the user has prepared the run, (s)he can start it by moving to the associated RC state. This is the point where the sequencer takes over, toggles through the steps and finally ends the run. At the end of the run the GnamAna process updates the Calibration Result Folder with the new results.

Note, that the Calibration Run is only a special case of a normal run. The diagram shown in Fig. 7.21 holds for any non-calibration run, except that there are no sequencer and no GnamAna process. In case of a non-calibration run it is the user, who ends the run, not the sequencer.

Runplans

The calibration framework as presented so far allows to derive calibration parameters by dedicated Calibration Runs. The major disadvantage of the applied scheme w.r.t. one where the whole required parameter range is scanned at a time, is that a calibration takes longer and requires several runs to work together consistently. Setting this up is certainly not trivial for a non-expert.

On the other hand, parts of the system sometimes need to be calibrated before one can address other ones. The applied scheme allows to check the part of the calibration done before using it for the one still to be done. Moreover the configuration of the hardware during the individual steps of a MS-run can easily be reconstructed knowing the content of the Run-Configuration Folder and the Calibration Folders that have been used. Beside that, one can redo/check individual parts of the calibration, without being forced to do the whole lot. Organizing the calibration in parts that logically belong together also allows to play with the individual Runtype and Analysistype. This can be used in order to find a way to do an initial Calorimeter Trigger calibration or address other tasks.

One of the disadvantages of the current scheme is addressed by the Runplan (RP) software package [25]. The Runplan package has been developed as the need grew to automate the calibration procedure involving several runs. It addresses the organization of a set of consecutive (Calibration-)Runs in a user-friendly way.

When first using MS runs the workflow was very different from how it has been presented here. The calibration data for the PPM were stored in XML-Files on a hardware associated way. The PpmCal objects where initialized from these files. As the ROD readout was not available, the data were read out via VME and stored to files. An analysis of these files was launched by hand and output the XML-Files that were used for the next run. This scheme had lots of restrictions. The configuration of the PP was very limited, i.e. tight to the register model without any abstraction layer. But also the configuration of the whole procedure was hard. Some parameters had to be adjusted in OKS, some in resource file, others via IS or in the command line, that was passed to the analysis programs. Moreover the user had to take care of many things in order to do successfully a Calibration Run. A first attempt to automate the procedure was done using shell scripts. The underlying procedure did not change, but using scripts made the operation much more user-friendly and less error prone. There were several scripts that were dedicated to individual tasks. However, in in order to do something that was not addressed by a script one had to

use the system as before.

Bringing more flexibility into the system, while keeping it user-friendly and concentrating all configuration parameters at one place is addressed by the Runplan package. The idea of Runplans is to implement all kinds of activities that have to be done during a run, e.g. set IS Variables, do a RC state transition, copy XML files around, do a MS-run, analyze a MS-result etc. in so-called RunPlanActions (RPA). RPAs can be configured via parameters and are pieced together into a Runplan. A Runplan can be set up writing an ASCII file in a dedicated format. The arrangement of RPAs in the Runplan and their parameter settings allow to automate and configure any procedure that consists of individual runs.

The Runplan file is parsed by a RunPlanHandler that controls its execution. There are two RunPlanHandlers in the L1Calo software repository. One is command-line based and rigidly executes a Runplan, the other one has a GUI and allows to split the Runplan into sections (RunPlanSections). RunPlanSections can then individually be executed. At the time Runplans were introduced, ROD readout became available. The RPAs that are dedicated to data analysis can read both data files gathered from VME or via the ROD-ROS stream. Furthermore they can store their results to both, XML-files and COOL.

In order to use the Event Monitoring Service as data-source, the RPAs responsible for the data analysis were extended to handle RDOs as input, too. A GNAM analysis plugin has been written that uses these RAPs which results in a design as previously described (Fig. 7.21).

7.6 Initial Calibration

In section 7.3 setups using various signal sources were discussed. Section 7.5 gives an overview of the Runtypes, i.e. parameter scans that are implemented. Several scenarios are possible using the available Runtypes and given setups in order to find settings for an initial calibration of the ATLAS Level-1 Calorimeter Trigger.

An initial energy-calibration can be done by using the LAr pulser system and the Tile Calorimeter CIS. Tests done at CERN show that adequate timing parameters for the pulser signals can determined using a Readout-Pointer and a PHOS4 Scan. Using these timing settings an Energy Scan can be realized. From the result of the Energy Scan settings for the energy-calibration related parameters can be derived based on the nominal energy values adjusted in the pulser systems that are provided by the calorimeter groups. The energy-calibration related settings derived with the pulser systems will only be a first step for the calibration of the ATLAS Calorimeter Trigger. This step is, however, mandatory to start the data acquisition. Once data taking has started, it is foreseen to correct these setting, using the findings of offline analysis that are based on physics channels like e.g. Z^0 decays.

When there is no beam the derivation timing-related parameters is not as straight forward as the derivation of initial settings for parameters associated with the energy-calibration. The pulser systems can only be used to measure timing relations between subsets of the calorimeter channels. The Laser System is not clock-synchronous but it can fire all Tile Calorimeter channels with a precision in time

that allows to use the signals for coarse timing measurements. The length of TTC fibers to the LAr Front End Crates varies. Therefore an overall timing of the LAr Calorimeter cannot be derived using the LAr pulser system. It is, however, possible to determine the timing relation between subsets of the LAr Calorimeter that are linked to one FE crate. The timing of Trigger Towers that are linked to one FE crate is precise to one nanosecond.

$$D = \begin{pmatrix} 0 & d_{12} & X & d_{14} & d_{15} & X & \dots & d_{1n} \\ d_{21} & 0 & d_{23} & X & d_{25} & X & \dots & d_{2n} \\ X & d_{32} & 0 & X & d_{35} & X & \dots & X \\ d_{41} & X & X & 0 & d_{45} & X & \dots & d_{4n} \\ d_{51} & d_{52} & d_{53} & d_{54} & 0 & X & \dots & X \\ X & X & X & X & X & 0 & \dots & d_{6n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & X & d_{n4} & X & d_{n6} & \dots & 0 \end{pmatrix}$$

Figure 7.22: Matrix that contains the delays between individual channels. Unknown entries are marked with an X.

A practical way to store the delays is to write them into a delay matrix of the dimension $n \times n$, where n is the overall number of channels. Such a matrix obviously is anti-symmetric, a sample is shown in Fig. 7.22. Note that a delay matrix does not need to be completely filled in order to contain all the required information. In fact it would be enough if one row of the matrix is filled. It is also possible that none the rows are completely filled but the matrix contains all of the required information anyway. In the example shown in Fig.7.22, e.g. the d_{13} entry can be filled using that $d_{13} = d_{15} + d_{53}$. A simple algorithm can be used to fill all entries a matrix contains information for. The approach to find the timing relation between all channels would be to fill as many of the matrix entries as can be measured and then run the algorithm to fill all possible entries.

Once the relative timing is found for subsets of channels the deviations between the subsets have to be determined in order to get an overall timing alignment. Therefore it is enough to find delays between sample channels of the different subsets. When there is no beam a measurements providing this information could be done using signals from cosmic muons. Cosmic muon signals, however, have not yet been measured in the Calorimeter Trigger, as the Calorimeter Trigger input cables are used for the Coincidence Board in order to generate L1As in cosmic muon runs. In order to use cosmic muons for a timing alignment in the Calorimeter Trigger, L1As for cosmic muon runs have to be generated different from how they are generated now. This could be done using the calorimeter signals for the Level-1 Muon Trigger as input for the Coincidence Board. As only up to 5 cells can be read out in a PPM using the ROD-ROS stream it will, however, be painful to find the muon signals in the PipeMemory. This especially is true under the aspect of the low rates of usable cosmic muons in the ATLAS pit, which is smaller than one Hz. Note that at this rate it would be feasible to overcome the ROD-ROS readout limitations by using VME readout. However, it is doubtful that cosmic muons will be used for timing

measurements.

Another way to determine the timing between all channels or calorimeter subcomponents is to use beam signals. By forcing the PP output of all channels but one to zero one can trigger on the signature one gets if this single channel has a signal, e.g. one "electron", using the CTP. Generating triggers this way assures, that all signals in the PP are timed w.r.t. this one enabled channel and therefore can be compared. Latter would not be he case if more than single channel were switched on without having them aligned beforehand. Due to the readout limitations the signal measurement would have to be performed by means of a Readout-Pointer scan or using VME readout. The Readout-Pointer Scan would have to be long enough that for each step enough events are captured so that one provides a signal in the channels that delay information is needed for. Admittedly the event-rate for such a measurement cannot be expected to be very high. However, once the delays between some channels are known and it is possible to adjust their timing, one can use them all to generate triggers. This way the event-rate can be increased. Note that the time between bunch-collisions must be large compared to the skew between the individual channels in order for this method to work.

Once the coarse timing is found, the fine timing using the PHOS4 delay has to be done. As beam signals are not repetitive the PHOS4 scan is not feasible for this purpose. In fact the fine timing of beam signals must be done by fitting the signals measured in 25 ns resolution with an adequate function [24].

Chapter 8

Commissioning and Integration

The current ATLAS schedule envisages the commissioning of all ATLAS components to be finished by April 2008, the commissioning of the Pre-Processor is finished in Autumn 2007. Parallel to the ongoing hardware installation, tests are being done in order to uncover problems in the system. Beside that the integration and combined operation of sub-systems is addressed in so-called Milestone Runs.

The commissioning tests done with the Pre-Processor are twofold. There are tests checking whether the cabling connectivity between the detector and the Calorimeter Trigger is correct and tests that address the signal quality.

8.1 Connectivity Tests

The cabling from the calorimeter to the Calorimeter Trigger is described in 5.1. Wrong connections can occur because cables were wrongly connected at the TCPPs, the Receivers or the RPPPs. Furthermore the connectors are soldered onto the cables in place which makes it possible that individual channels in one cable are wrongly connected. It is also possible, that individual wires from the differential pairs of one channel have been interchanged.

The connectivity from the calorimeters to the Pre-Processor is tested making use of the usual pulser systems, i.e. the LAr pulser system and the CIS. However, neither the control of the pulser systems via the TDAQ software, nor the integration of the calorimeters and the Calorimeter Trigger in combined runs is at a state that would allow to automate a connectivity test in a combined (MS-)run. As the connectivity has basically only to be checked once at the start of the experiment, implementing a dedicated Runtype for connectivity tests would be a little bit excessive anyway.

The way connectivity tests are done is that for each checking step of a test, the pulser systems are configured individually and the signals are measured on the Pre-Processor side using a program called ppmwatch. Ppmwatch is running on the crate controllers and allows to configure and readout PPMs via VME and can therefore handle the part of the PP that is associated to one PP crate. Ppmwatch has been designed for low level hardware testing and debugging, it can be used to do all possible kinds of register manipulations and to readout individual PPMs or the whole crate. Two-dimensional overview plots of the η-ϕ range covered by the crate under consideration can be created in ppmwatch. These plots can either show

the BCID result or the maximum value of the raw data that has been read out for each channel. In the context of the connectivity tests the overview plots that show the maximum value of the raw data are used. A detailed description on ppmwatch can be found in Appendix D.

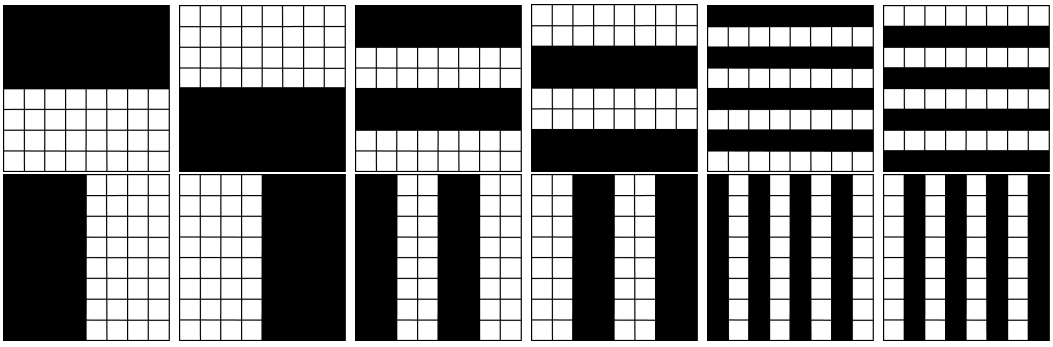

Figure 8.1: Set of 12 patterns that can be used to check connectivity of 64 channels.

A connectivity test for a system of parallel channels must assure that the channels on the sending side are connected to the channels of the receiving side according to the design of the system. In particular it must be assured, that no channel on the receiving side is connected to more than one channel on the sending side and no channel on the sending side is connected to more than one channel on the receiving side.

As mentioned in section 5.1 the Calorimeter Trigger receives ~ 7200 analog signals from the various sub-detectors of the calorimeter. We assume that one can only reliably differentiate channels that are entirely switched on from channels that are switched off and that it is not safe to use different signal levels on the individual channels in order to identify them. Under this assumption the straight-forward solution for a connectivity test would be to test each channel individually. With a non-automated procedure as it is used for the connectivity tests, it would, however, be painful to check each channel individually. A method has therefore been worked out that allows to check the connectivity of a large number of channels while minimizing the number of checking steps [26].

The basic idea of the method used for connectivity tests is that, assuming it is only reliable to differentiate channels that are switched on from channels that are switched off, it is most efficient to check the connectivity for half of the channels at each checking step. Varying the patterns of channels that are switched on allows to make statements on the connectivity of subsets of the channels. If e.g. half of the channels is switched on and one sees the expected pattern on the receiving side, one can almost state that a misalignment of channels is only possible within the half of the channels that has been switched on. However, it is still possible that one of the channels that has been switched off is connected to one of those that has been switched on. A way to rule this possibility out is to check the inverse pattern, i.e. switch of all channels that had been switched on and vice-versa. If the test of the inverse pattern shows the expected result one can treat the two halves independently. Therefore the same idea can be applied to the two halves of channels in parallel. This allows to state that channels can only be misaligned within four

quarters of the channels. Now the four quarters can be checked in parallel and so on, until one arrives at a channel level. Fig. 8.1 shows an example of patters that can be used to check the connectivity of 64 channels displayed in a 8×8 matrix.

In case one would have a system with two channels, two patterns would be required in order to assure that the two channels are connected correctly. Using the scheme described above means that each duplication of channels leads to two additional checking steps. Therefore $2n$ patterns are required in order to check the connectivity of 2^n channels. In other words, the number of steps needed to check a system with N channels is $2\log_2 N$. Since the next higher 2^n number of 7200 is $8192 = 2^{13}$, the minimum number of steps needed check all calorimeter to Pre-Processor connections is 26.

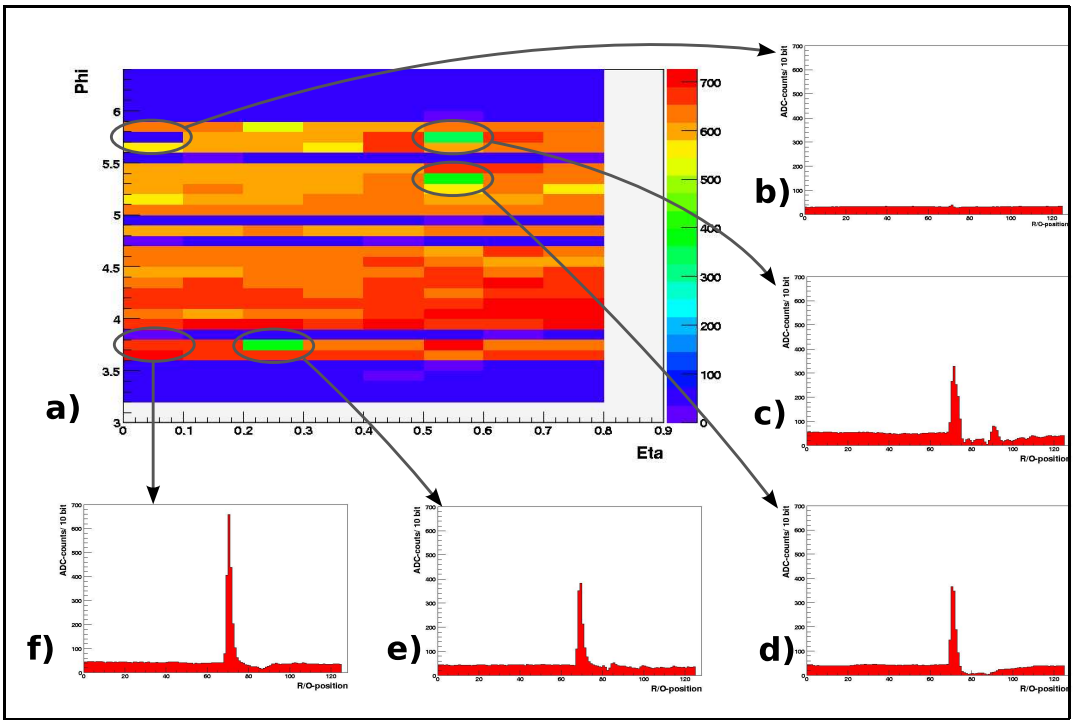

Figure 8.2: a) Overview-plot showing the color-coded maximum of the raw readout for an η-ϕ-region in the hadronic layer that was addressed by a connectivity test. The blue background corresponds to Tile Calorimeter modules, that did not yet have power supplies. b)-f) The content of the raw readout data used in the overview-plot for some selected channels.

The connectivity tests are a still ongoing issue. If the schedule allows, they are done in parallel to the Commissioning of both Detector and Calorimeter Trigger items. An updated status can be found in [27]. Fig. 8.2 shows a some example plots from a test-session with the Tile Calorimeter CIS. The correct connectivity could be affirmed for all channels addressed in this test except the one shown in Fig. 8.2 b) which did not show any signal. As the pedestal noise for this channel is digitized, the reason why it does not show a signal must be in the CIS, a loose connection or in the Receiver system.

The plots shown were measured with a charge of 150 pC injected into one cell

of the Trigger Towers. 150 pC approximately correspond to an energy of 150 GeV. No pedestal or timing adjustment has been done for this test. Therefore one has to expect that the data are not always latched at the maximum value of the peak. In combination with the missing pedestal adjustment this explains the variety of signal levels in Fig. 8.2 a).

Some few channels, however, show values that cannot be explained with the missing timing and pedestal adjustment. The readout for these channels that need further investigation by the Tile Calorimeter group, is shown in Fig. 8.2 c)-e). In fact one benefit the calorimeters have from tests with the Pre-Processor is that they provide a fast way to identify problematic channels. The pulse shown in Fig. 8.2 f) has a symmetric shape which is an indication that it has been latched close to the maximum value. After subtracting the pedestal the height of this pulse corresponds to $\sim$ 620 ADC-counts. With an FADC-count energy relation of 244 MeV/ADC-count this corresponds to 151.28 GeV which is about the energy that has been injected.

8.2 Signal-Quality Tests

In the previous section we have discussed the procedure used for connectivity tests that are done in parallel with the installation of the Pre-Processor and the calorimeters. Some problematic channels have been found during the connectivity tests and investigated by the calorimeter groups. However, the electronics currently installed is not yet the final one. The calorimeter groups have not yet fished their repair cycles and done all the tests they plan to do with their readout system. Therefore one cannot expect all channels to show nice signal pulses. Note, that this is not an issue for the connectivity tests. These are intended to check the cabling between the detector Front End Electronics an the Calorimeter Trigger which is already final.

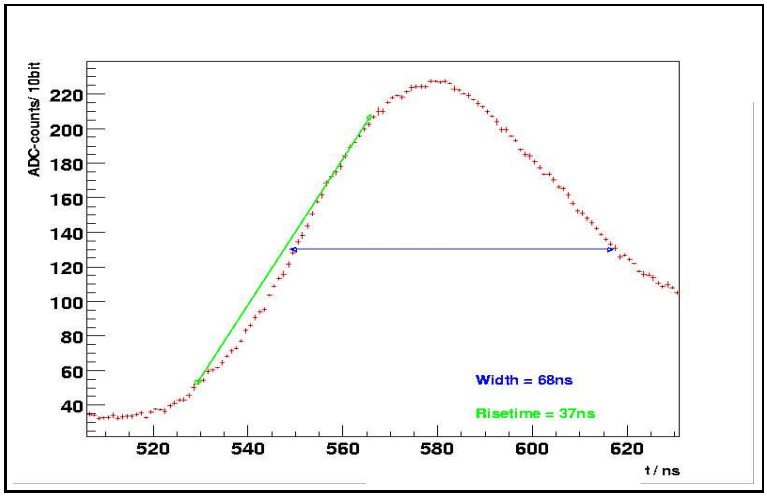

Figure 8.3: Result of a PHOS4 Scan done with the CIS of the Tile Calorimeter. The 10 % – 90 % risetime (green) and the FWHM (blue) have been determined with a program analyzing the pulse shape.

Once the installation is finished, the calorimeter groups have done their tests and the connectivity of all Calorimeter Trigger input channels is approved, it is planned

to to dedicated signal-quality tests. The signal-quality tests will be done on the basis of the Calibration Runtypes introduced in section 7.5. In order to measure the pulse shape with the precision of one nanosecond a PHOS4 scan is performed. An analysis of the signal shapes can then determine properties, like the 10 %-90 % risetime and the Full Width Half Maximum (FWHM) that are the used to identify problematic channels. Fig. 8.3 shows results for a test that has been done to prove the method.

8.3 Milestone Integration Runs

Rather technical aspects of the integration are addressed by so-called Milestone Runs (M Runs). The duty of Milestone Runs is to integrate the ATLAS sub-detectors in a combined run. A combined run in the latter sense is a run with a common TTC distribution, based on an overall OKS configuration and controlled from one single point. Many technical problems are solved during Milestone Runs, and the compatibility of the overall system is checked. The aim of the Milestone Runs is to progress more and more towards a final ATLAS-like running, each time integrating more and more parts of the experiment.

The Level-1 Calorimeter Trigger first participated at the M3 Run that took place from June 4-18 2007. For L1Calo the challenge of the M3 Run was simply to participate and have a stably running system. M3 mainly was dedicated to cosmic muons, and the L1As were generated with the Coincidence Board. As the Coincidence Board requires the Calorimeter Trigger input cables this meant that only noise could be measured in the Calorimeter Trigger. The timing setup was done using the Laser system.

The Calorimeter Trigger channels that where not used for the Coincidence Board have been calibrated as described in section 7.5. Calibration settings for these channels have successfully been derived using a DAC, a Pedestal and a Readout-Pointer Scan together with GNAM. A PHOS4 Scan would have been useless, as the Laser system is not firing synchronously to the TTC clock. Therefore the clock phase cannot be aligned to the signal pulses. All calibration results were stored to the COOL database and used to determine register values according to the procedure illustrated in Fig. 7.21.

Chapter 9

Summary and Outlook

This thesis addresses the work required to successfully operate the Pre-Processor of the Calorimeter Trigger in the context of an ATLAS run. Software has been developed that allows to load the programmable devices and grant access to the configuration registers of the Pre-Processor Modules. This software was developed in the framework of the common ATLAS TDAQ-software. It is based on the hardware description of the experimental setup, linked to the ATLAS conditions database and implements states of the run-controlling state machine. This makes it possible to use the Pre-Processor system in an ATLAS run. The configuration of the Pre-Processor system can be set up in the ATLAS conditions database. An abstraction layer of the hardware related configuration parameters has been developed, that allows to set up the configuration of the Pre-Processor system in an intuitive and result-oriented way. These parameters are stored in the conditions database and used to configure the system. Different versions of these settings can be stored in parallel, and the user makes the selection which version to use at the beginning of a run. A procedure has been implemented that uses this configuration and combines it with calibration measurements in order to derive the required hardware settings.

Calibration procedures have been developed for the Pre-Processor that are based on dedicated Calibration Runs. The Calibration Runs can be used to perform the calibration measurements that are required to configure the Pre-Processor system successfully. During these runs a certain range of the parameter space of the Pre-Processor is scanned, and the measured data are analyzed. The output of the analysis are the calibration results required for the procedure described above. These results are directly stored to the conditions database. Additionally the user gets a feedback whether the calibration results are suspicious or a Calibration Run completely failed. In order to do the parameter scans the concept of Multistep Runs has been introduced. Multistep Runs use the run-controlling state machine in order to change the parameter values from one value to another synchronously in the entire system. In order to gather all the calibration results needed for the Pre-Processor, about four different Calibration Runs are required. A dedicated software package, the Runplan package, has been developed in order to set up and automate an entire calibration procedure consisting of these runs.

A setup has been installed in Heidelberg that can be used to run the software together with hardware and test new implementations. Other setups comprising the calorimeters have been used at CERN in order to test and use the Pre-Processor

calibration with the detector. Due to lack of beam these tests have exclusively been performed with the pulser systems which the calorimeter sub-detectors provide. During the M3 Integration Run the procedure developed in this work has been used in a combined run the first time. Possibilities how to use the result of this work in order to derive calibration settings for the Calorimeter Trigger once there is beam are discussed.

Finally some tests are presented that are done with the calorimeters and the Pre-Processor in the context of the system integration. This addresses mainly connectivity tests which are performed to ensure that the cabling between the detector and the Calorimeter Trigger is correct. Tests are planned that also address the signal-quality. Future Milestone Runs will help to make ATLAS a more and more operational system until it is ready for data taking and a new era begins.

Appendix A

Particle production in parton-parton interactions

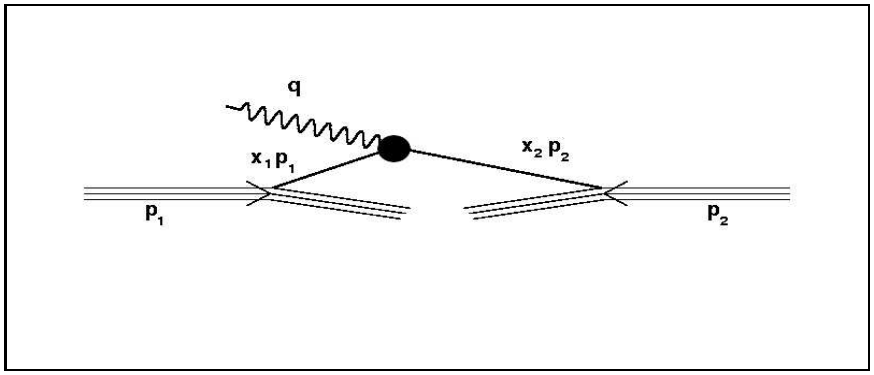

Figure A.1: Parton-parton fusion in p-p cms system.

Two colliding partons can create a massive particle like Z^0, $W^{\pm}$ or the Higgs boson with the invariant mass $M^2 = q^2$ by parton-parton fusion as shown in Fig. A.1. For highly relativistic protons, $E^2 \approx p^2$, the four-momenta of the protons and the created particle in the center-of-mass system of the protons are:

$$\mathbf{p_1} = \left(\frac{\sqrt{s}}{2}, 0, 0, \frac{\sqrt{s}}{2}\right) \tag{A.1}$$

$$\mathbf{p_2} = \left(\frac{\sqrt{s}}{2}, 0, 0, -\frac{\sqrt{s}}{2}\right) \tag{A.2}$$

$$\mathbf{q} = \left((x_1 + x_2)\frac{\sqrt{s}}{2}, 0, 0, (x_1 - x_2)\frac{\sqrt{s}}{2}\right) \tag{A.3}$$

where s is the squared cms energy: $s = (\mathbf{p_1} + \mathbf{p_2})^2 = (14\,\mathrm{TeV})^2$ and x_1, x_2 are the momentum fractions of the involved partons. For the invariant mass of the created particle q^2 one finds:

$$\begin{aligned} & M^2 = q^2 = x_1 x_2 s \\ \Rightarrow \quad & M = \sqrt{x_1 x_2 s}. \end{aligned} \tag{A.4}$$

On the other hand one can define the rapidity y such that $\mathbf{q}$ can be written:

$$\mathbf{q} = M\Big(\cosh y, 0, 0, \sinh y\Big) = \sqrt{x_1 x_2 s}\Big(\cosh y, 0, 0, \sinh y\Big). \tag{A.5}$$

Comparing the zero component q_0 from (A.3) and (A.5) delivers:

$$\sqrt{x_1 x_2 s}\cosh y = (x_1 + x_2)\frac{\sqrt{s}}{2} \tag{A.6}$$

which is equivalent to:

$$x_{1,2} = \frac{M}{\sqrt{s}} e^{(\pm y)}. \tag{A.7}$$

Appendix B

Latching the PprASIC input data

The PprASIC on the Pre-Processor MCMs receives ten-bit digitized data from four FADCs per MCM. The clock phase of the PprASIC and the propagation time of the FADC output to the PprASIC are fixed and determined by the routing on the MCM. The clock-phase of the FADC, however, is programmable in the interval 0 to 25 ns with a 1 ns resolution by setting a delay in the PHOS4 chip.

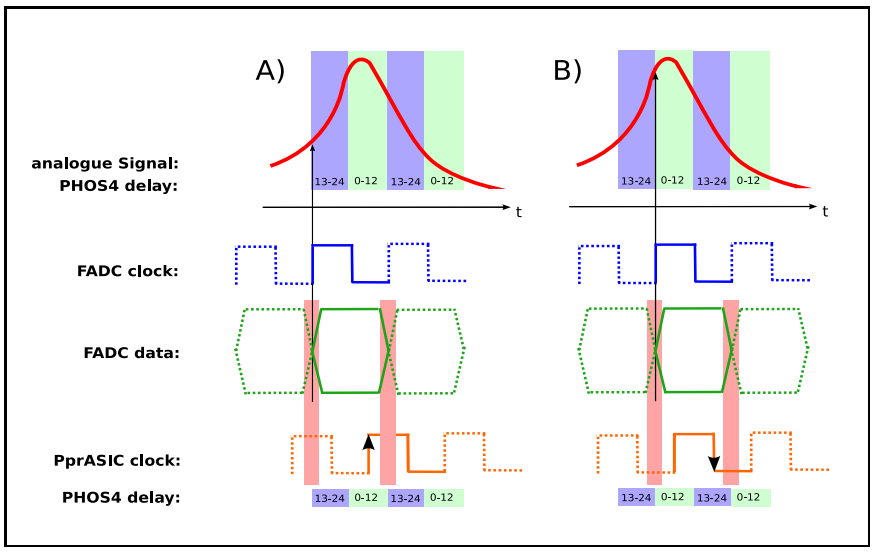

Figure B.1: Timing overview of the signals involved to provide digital Trigger Tower input to the PprASIC. When increasing the PHOS4 delay, the FADC clock and FADC data are moving and the analogue signal is latched at a different position. The red area indicates corrupted FADC data and is moving, too. The situation is shown for A) a PHOS4 delay of 13 ns and B) a PHOS4 delay of 0 ns.

At the time the FADC is latching data its output becomes corrupted. Hence corrupted data arrives at the PprASIC for some delay values in the PHOS4 w.r.t. the PprASIC clock. Due to the dimensions of the MCM the propagation time of the clock signal and the data is short compared to 1 ns. Therefore the data from all FADCs become corrupted at the same PHOS4 setting. This issue is addressed

by allowing to choose the clock-edge at which the PprASIC latches it input data. Selecting the input data latch-edge has been implemented for all input channels of a PprASIC individually, in order to be able to choose arbitrary PHOS4 delays for different channels.

Fig. B.1 gives an overview of the timing situation. Skews between the signals are neglected for comprehensibility's sake. Experimentally one finds that the data are corrupted for a delay of 4 ns when latching with positive latch-edge (↑) and a delay of 16 ns when latching with negative latch-edge (↓). Therefore the PprASICs are operated with negative latch-edge for PHOS4 delays between 0 and 12 ns and positive latch-edge for PHOS4 delays between 13 and 24 ns. Within one clock cycle of the PprASIC clock, in this scheme, the data latched with positive edge belongs to an earlier time w.r.t. the analog signal than the data latched with negative edge. In other words, in order to capture the data for PHOS4 delays 13 to 24 following the ones in the interval 0 to 12 in the PprASIC, one has to go to the next PprASIC clock cycle.

Appendix C

COOL Folders

COOL Folders are database tables as they are implemented in the COOL conditions database. The COOL database is implemented on top of the CORAL package that provides an interface to the most common database technologies: MySQL, SQLite and Oracle. COOL provides a C++ API.

The data stored in COOL are organized in COOL Folders. A COOL Folder can be considered as a table where variables of different type, e.g. integers, boolean or strings are be stored in columns. A variable is always aligned to a column, the data is stored in rows. The rows are queried by the Channel ID. The Channel ID is implemented as a column and used as key identifier in all COOL Folders. Beside the Channel ID every COOL Folder has a column that encodes an Interval Of Validity (IOV) and determines a period the data is valid for. If data are written to COOL an IOV always has to be supplied. Generally one uses the IOV, that goes from the moment the data are written until infinity. Data from all IOVs are retained and can be queried.

Several COOL Folders are used to store Conditions Data for the Pre-Processor. Beside the Run-Configuration Folders and the Calibration-Result Folders per Calibration Run, there are Folders that contain default register values. One Folder that combines Calibration Results from all Calibration Runs is the Validated Calibration Results Folder. Here an overview of all PP associated COOL Folders is given.

Default Register Values

The COOL Folders containing the default register values for the Pre-Processor are used to fill a class from the DBL1Calo package called PpmCal that is used by the PpmServices package to write Registers to the Modules. The PpmServices package implements HDMC parts, wherefore it is organized very close to the Pre-Processor hardware, and so is the PpmCal class. The data in PpmCal is separated into Registers that are on the PPM level, the MCM level and the ASIC-channel level. Beside that a separation between registers is done with respect to their nature, i.e. whether they are aligned to the timing-calibration, the energy-calibration or the configuration of the PP. As there are no energy-calibration aligned registers at a PPM level, one ends up with $(3 \times 3 - 1)$ sets of registers: timing-calibration and configuration aligned registers on a PPM level and timing-calibration, energy-calibration and

configuration-aligned registers on a PPM, MCM and ASIC-channel level.

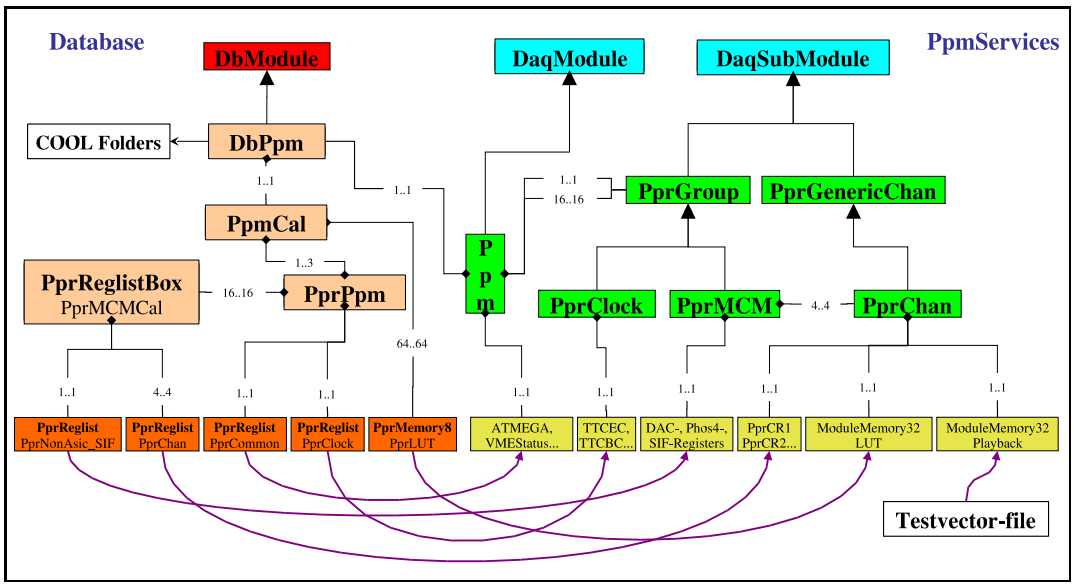

Figure C.1: UML class-diagram on the Database and PpmServices classes. PpmCal contains three PprPpm objects for energy-calibration, timing-calibration and configuration-aligned registers. All of the PprPpm classes reflect the Ppm hardware structure, containing two PprReglists (data-containers for a list of registers) for the Ppm, and 16 PprReglistBoxes (containers for PprReglistBoxes). Each of the 16 PprReglistBoxes is associated with a MCM and has five PprReglists, one for Registers on the MCM level and four for the four channels on a MCM. The PpmCal Structure is filled when parsing the database via the DbPpm class. DbPpm is also the class that is passed to PpmServices so it has database access.

The same division that exists in PpmCal exists for the COOL folders. There are $(3 \times 3 - 1)$ COOL Folders containing default values for the Pre-Processor Registers, that can be used to fill the database classes shown in Fig. C.1

Run-Configuration Folders

Run Configuration-Folders contain the configuration to do a run. Several configurations can be started at a time and queried via the Channel ID. Some of the columns in the Run Configuration-Folders are specific to the Runtype, others exist in all Pre-Processor Calibration-Run Folders:

- **AnalysisType** (string): Defines how to analyze a Calibration Run.
- **ResultFolderType** (string): Folder Type to store the Calibration result in.
- **Data Source** (string): Determines where to get the event data from.
- **MaxMeanDeviation** (unsigned): Defines the range within which data that should by measurement be the same are considered as consistent. If this range is exceeded a Calibration Result is tagged as critical.

Other columns are common to all Multistep-runs:

- **ScanParameter** (string): Defines which parameter to scan.
- **ScanStartValue**(int): Defines the value of the ScanParameter to start with.
- **ScanIncrement** (int): Defines the ScanParameter increment per step.
- **NumSequenceSteps** (int): Defines the number of steps.
- **SequenceMode** (string): Intended to be used to choose the mode the sequencer-program uses to do the stepping, but not yet used.
- **MillisecondsPerStep** (int): Defines how many μs to wait per step.
- **TriggersPerStep** (int): Intended to be used to define the number of L1As allowed per step, but not yet used.

DAC Scan Configuration Folder

Beside the columns that are common to all Pre-Processor Calibration Runs and all Multistep Runs, the DAC-Scan Folder contains the following variables:

- **PipeDelayRaw** (unsigned): Defines where in the RAW PipeMemory to start to read out.
- **NumBcRaw** (unsigned): Defines how many cells from the RAW PipeMemory to read out.
- **MinDacSlope** (double): Defines a lower limit for the slope of the DAC ramp.
- **MaxDacSlope** (double): Defines an upper limit for the slope of the DAC ramp.
- **MinDacOffset** (int): Defines a lower limit for the y-axis intercept.
- **MaxDacOffset** (int): Defines an upper limit for the y-axis intercept.

Pedestal Run Configuration Folder

Beside the columns that are common to all Pre-Processor Calibration Runs, the Pedestal-Run Folder contains the variables that allow to use the result of the DAC Scan:

- **UsePprDacScanResults** (bool): Indicates whether or not to use the DAC-Scan Result.
- **ExpectedPedestal** (unsigned): Pedestal to adjust.

Furthermore it contains variables dedicated to the Pedestal Run itself

- **PedestalMargin** (unsigned): Defines an acceptance range for the noise distribution.
- **DstPedestalType** (string): Defines where to store the result of the Pedestal Run.

Readout-Pointer Scan Configuration Folder

Beside the columns that are common to all Pre-Processor Calibration Runs and all Multistep Runs, the Readout-Pointer Scan Folder contains the variables that allow to use the results of the DAC Scan and the Pedestal Run:

- **UsePprDacScanResults** (bool): Indicates whether or not to use the DAC-Scan Result.
- **ExpectedPedestal** (unsigned): Pedestal to adjust.
- **UsePprPedestalRunResults** (bool): Indicates whether or not to use the Pedestal-Run Result.

PHOS4 Scan Configuration Folder

Beside the columns that are common to all Pre-Processor Calibration Runs and all Multistep Runs, the PHOS4 Scan Folder contains variables that allow to use the results of the DAC Scan and the Pedestal Run and the Readout-Pointer Scan:

- **UsePprDacScanResults** (bool): Indicates whether or not to use the DAC-Scan Result.
- **ExpectedPedestal** (unsigned): Pedestal to adjust.
- **UsePprPedestalRunResults** (bool): Indicates whether or not to use the Pedestal-Run Result.
- **UsePprReadoutScanResults** (bool): Indicates whether or not to use the Readout-Pointer Scan Result.
- **DstMaxPos** (bool): Indicates at which position of the window that is read out from the PipeMemory to put the maximum of the signal.
- **DstNumRaw** (unsigned): Number of cells to read out.

Furthermore it contains variables dedicated to the PHOS4 Scan itself

- **PedestalSource** (string): Indicates where to get the Pedestal value (that is required by the analysis) from.
- **MinWidth** (unsigned): Gives a lower limit for the width of the signal pulse.
- **MaxWidth** (unsigned): Gives an upper limit for the width of the signal pulse..
- **MinRisetime** (unsigned): Gives a lower limit for the 10% to 90% risetime of the signal pulse.
- **MaxRisetime** (unsigned): Gives an upper limit for the accepted 10% to 90% risetime of the signal pulse.
- **SignalSignificance** (unsigned): Indicates the threshold that has to be surpassed in order to identify a channel to carry a signal .

- **NrSignificanceSamples** (unsigned): Indicates how many samples around the maximum value have to surpass the SignalSignificance-level in order to identify a channel to carry a signal.
- **CreateWRtPlots** (bool): Indicates whether to create plots with the width and the 10% to 90% risetime of the signal pulse.
- **UseFifo** (bool): Indicates whether to use the findings in order to synchronize the realtime path on a PPM level.

General Run Configuration Folder

This Folder is dedicated to a general run, i.e. no calibration run. It contains variables that allow to use the results of the DAC Scan, the Pedestal Run, the Readout-Pointer Scan and the PHOS4 Scan:

- **UsePprDacScanResults** (bool): Indicates whether or not to use the DAC-Scan Result.
- **ExpectedPedestal** (unsigned): Pedestal to adjust.
- **UsePprPedestalRunResults** (bool): Indicates whether or not to use the Pedestal-Run Result.
- **UsePprReadoutScanResults** (bool): Indicates whether or not to use the Readout-Pointer Scan Result.
- **DstMaxPos** (bool): Indicates at which position of the window that is read out from the Raw PipeMemory to put the maximum of the signal.
- **DstNumRaw** (unsigned): Number of cells to read out.
- **UsePprPhos4ScanResults** (bool): Indicates whether or not to use the PHOS4-Scan Result.
- **MaxPosition** (unsigned): Contains the Information how the Maximum of the signal was adjusted while doing the PHOS4 scan.

Calibration-Result Folders

The COOLL1Calo package provides a scheme to derive a Channel ID that encodes the module type (e.g. PPM, CPM, JEM...) the crate the module is installed in and the channel on that module. As Calibration results are aligned to individual channels, this scheme is used in all Calibration Result Folders.

The columns that are common to all Pre-Processor Calibration Result Folders are:

- **Success** (bool): Indicates whether the Calibration Run was successful.
- **Critical** (bool): Indicates whether the Calibration Run was critical .
- **ModuleId** (unsigned): Module ID PPM the Calibration was done for.
- **Histograms** (string): Location where histograms that where generated during the calibration are stored.

DAC Scan Result Folder

The columns specific for a DAC Scan Result Folder are:

- **Offset** (double): y-axis intercept from the linear regression of the DAC Ramp.
- **Slope** (double): Slope from the linear regression of the DAC Ramp.
- **CrossCor** (double): Cross correlation coefficient from the linear regression of the DAC Ramp.

Pedestal Run Result Folder

The columns specific for a Pedestal Run Result Folder are:

- **PedMean** (double): Mean value of the noise distribution.
- **PedSigma** (double): RMS of the noise distribution.

Readout-Pointer Scan Result Folder

The columns specific for a Readout-Pointer Scan Result Folder are:

- **MaxDaqPtrDelayRaw** (unsigned): Position of the maximum of the signal-pulse in the PipeMemory.

PHOS4 Scan Result Folder

The columns specific for a PHOS4 Scan Result Folder are:

- **InDataNegedge** (unsigned): Latch-edge for the ASIC input data..
- **Phos4Delay** (unsigned): PHOS4 delay for the maximum of the signal-pulse.
- **MaxP4DelayRaw** (unsigned): Correction for the PipeDelayRaw value.
- **SignalTag** (bool): Indicates whether the channel was identified to have a signal or not.

Energy Scan Result Folder

This Folder contains the results of an Energy Scan. Even though the Energy Scan itself is not yet implemented the Folder to store the results expected from a Energy Scan has been prepared. Most of the Energy Scan Results are now encoded in strings, even though they contain numerical values. This will be changed as soon as L1Calo has moved to a version of COOL that allows to store Binary Large OBjects (BLOBs).

- **Energies** (string): Contains all the energies that have been scanned.
- **MeanRaw_0** (string): Mean values of the first readout sample at all energies.
- **MeanRaw_1** (string): Mean values of the second readout sample at all energies.

- **MeanRaw_2** (string): Mean values of the third readout sample at all energies.
- **MeanRaw_3** (string): Mean values of the fourth readout sample at all energies.
- **MeanRaw_4** (string): Mean values of the fifth readout sample at all energies.
- **SigmaRaw_0** (string): RMS of the first readout sample at all energies.
- **SigmaRaw_1** (string): RMS of the second readout sample at all energies.
- **SigmaRaw_2** (string): RMS of the third readout sample at all energies.
- **SigmaRaw_3** (string): RMS of the fourth readout sample at all energies.
- **SigmaRaw_4** (string): RMS of the fifth readout sample at all energies.
- **MeanBcid** (string): Mean values of the BCID result at all energies.
- **SigmaBcid** (string): RMS of the BCID result at all energies.
- **MeanPedestal** (double): The mean value of the Pedestal the Energy Scan was done with.
- **SigmaPedestal** (double):RMS of the Pedestal.
- **LutOffset** (double): Estimation of the LUT y-axis intercept.
- **LutSlope** (double): Estimation of the LUT slope.

Validated Results Folder

This Validated Results Folder contains contains all columns from all Result Folders that are listed above.

Appendix D

The Ppmwatch program

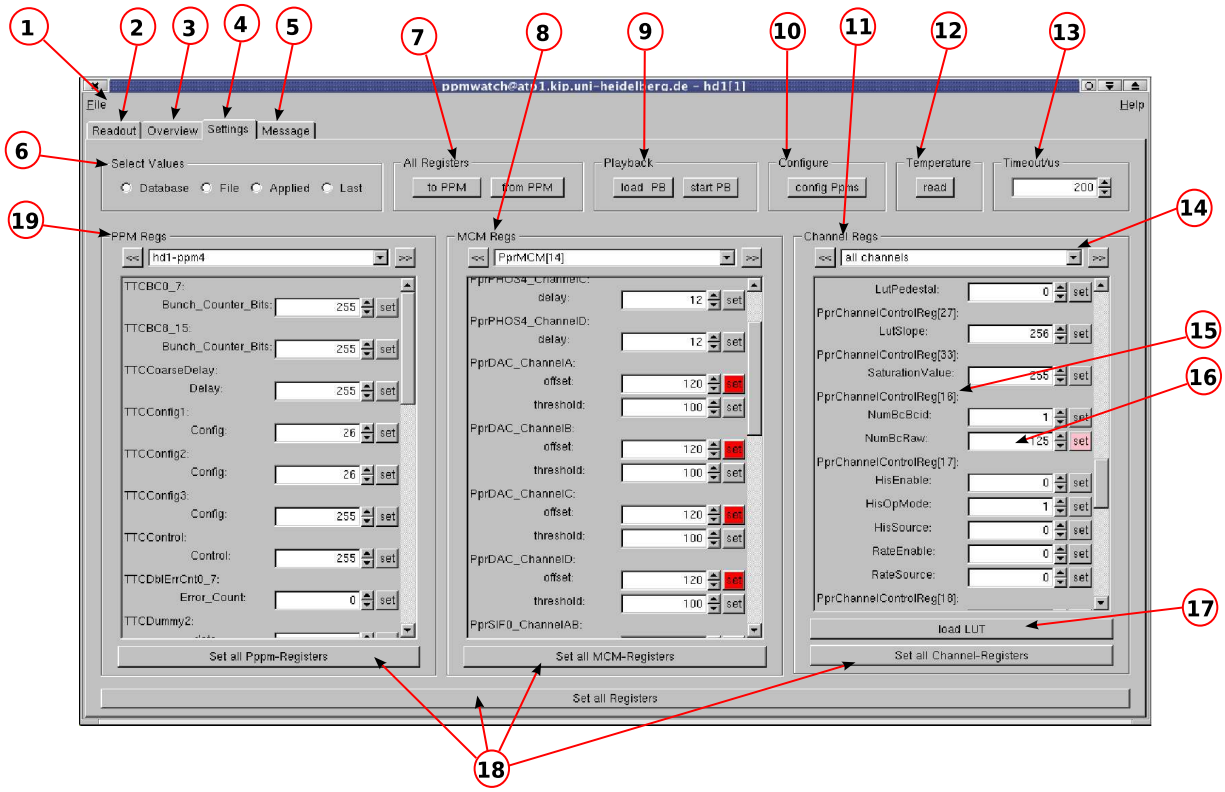

Figure D.1: Screenshot of ppmwatch with the Settings Tab selected, details in the text.

The ppmwatch program is a program that can be used for low-level hardware tests and debugging. Ppmwatch is implemented in the PPMTest package of the L1Calo software repository and makes use of the PPMServices package for hardware access. The program is purely dedicated to PPMs that are installed in one Pre-Processor crate and has to be started on the crate-controller CPU. Once ppmwatch is started it reads the hardware and register configuration of the crate associated with the crate-controller from the OKS configuration and COOL and shows up with a Graphical User Interface (GUI).

The GUI is shown in Fig.D.1, Fig.D.2 and Fig.D.2. Labels have been added to some of the widgets that have been kept unique throughout all three Figures. Therefore in the text the widgets are simply identified with the label number in brackets.

The Ppmwatch GUI essentially comprises a File Menu (1), and four Tabs (2)-(5). The File Menu allows to exit the program and to load and save register settings for individual PPMs from and to XML-files, respectively. The first Tab, the "Readout Tab" is dedicated to the Raw and BCID readout on a channel by channel basis. The second Tab, the "Overview Tab" allows to create overview plots of the η-ϕ-range addressed by the crate under consideration. The third Tab is the "Settings Tab" and can be used to change the setup of the PPMs and the fourth Tab. The "Message Tab" can be used to display a message on the 9×9 LED-matrices on the front-panels of the PPMs. Admittedly writing messages on the front-panels of the PPMs is a gimmick, though an inevitable one.

The Settings Tab

The screenshot of ppmwatch in Fig. D.1 shows the Settings Tab. Ppmwatch keeps four copies for all settings register Settings of the PPMs. The registers are sorted in registers that are on a PPM level, on an MCM level and on an ASIC channel level. In the GUI they are shown in lists under the Combo-boxes (14), that allow to select the PPM (19), MCM (8) or ASIC channel (11). It is possible to select an individual PPM, MCM, and ASIC channel, but also to select all PPMs, all MCMs and all ASIC channels. In case all PPMs, MCMs or ASIC channels are selected register manipulations will be done to all registers that have been selected.

The register lists have a label for each register (15) showing its name. The Bit Fields of the are listed under the register label. Each Bit Field has a Number Entry and a "Set" button (16). The Number Entry shows the register value that has been selected, the "Set" button can be used to write a Bit Field to the hardware. The color of the "Set" button indicates whether the value shown in the Number Entry differs from the one that is applied in the PPMs. If the button is grey the value corresponds to the one in the PPM, if it is red it does not. If the button is rose, one of the selections in (19), (8) or (11) indicates to write to more than one register, at least one of which currently has a value different from the one displayed in the Number Entry. In order to set the values for more than one Bit Field at a time according to the selection in (19), (8) and (11), the "Set all XXX" buttons (18) can be used.

The copy of register values to show in the GUI can be selected in (6). One can either select the values that are defined in the (OKS and COOL) database, the values as they have been applied to the hardware or the values that have lastly been changed in the GUI. Additionally it is possible to load register values from a file and view those. In order to write all values of the selected copy to the hardware or update the copy shown if "Applied" is selected in (6) by reading them from the PPMs, (7) can be used. The playback memory can be loaded and started with (9) and the selected PPMs can be reconfigured as defined in the database by pressing (12). (13) is not yet used and intended to define the timeout for a handshake between

Software and the Firmware in the RemFPGA.

The Readout Tab

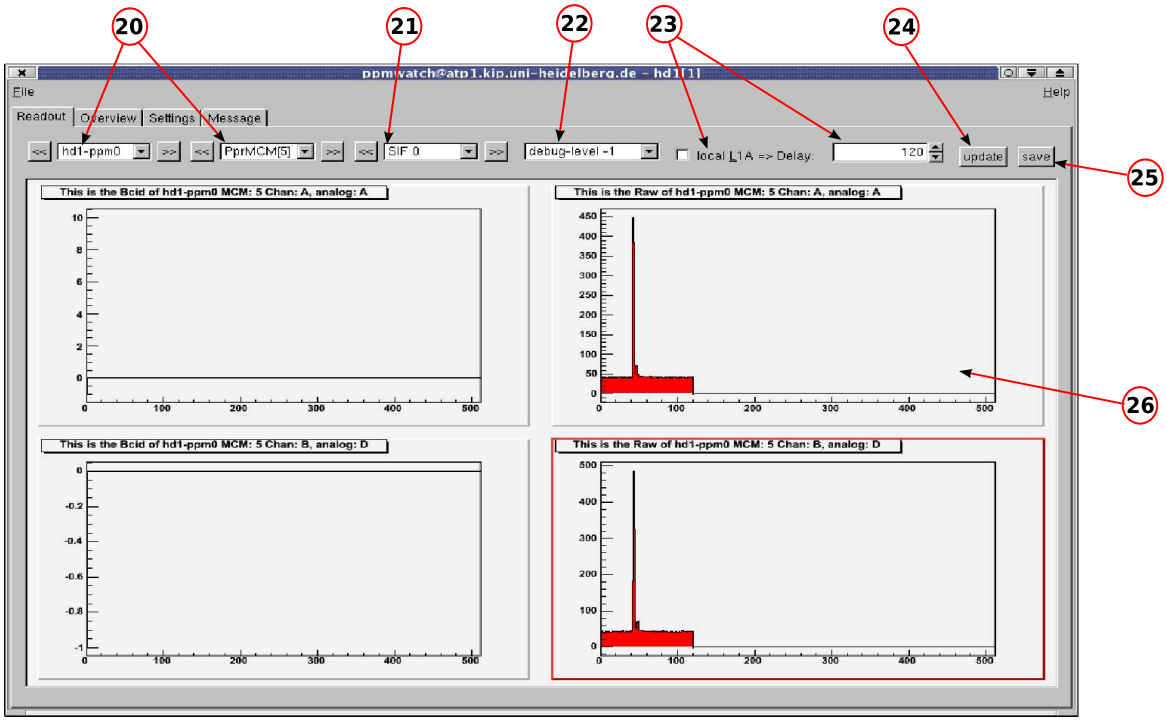

Figure D.2: Screenshot of ppmwatch with the Readout Tab selected, details in the text.

The Readout Tab, Fig. D.2, is dominated by a canvas (28) that shows the BCID and Raw readout for one Serial Interface (two channels) of an MCM in four histograms. The histograms are always shown for the Serial Interface (SIF) selected in (20) and (21). When (24) is pressed the PPMs are read out and the histograms are updated. In order to debug the readout path a debug-level can be selected in (22). The debug level steers the amount of information that is dumped to the standard output when pressing (24). In case there are no other triggers available L1As can be generated in the RemFPGA using (23). All histograms shown in ppmwatch can be saved to a file with (25).

The Overview Tab

Overview maps of the η-ϕ range addressed by the crate under consideration can be generated in the Overview Tab. The map is shown in the canvas (33). The screen shot shown in Fig. D.3 has been taken at Heidelberg in a setup where only one

PPM was installed and input was generated for one Analog Input connector of that PPM.

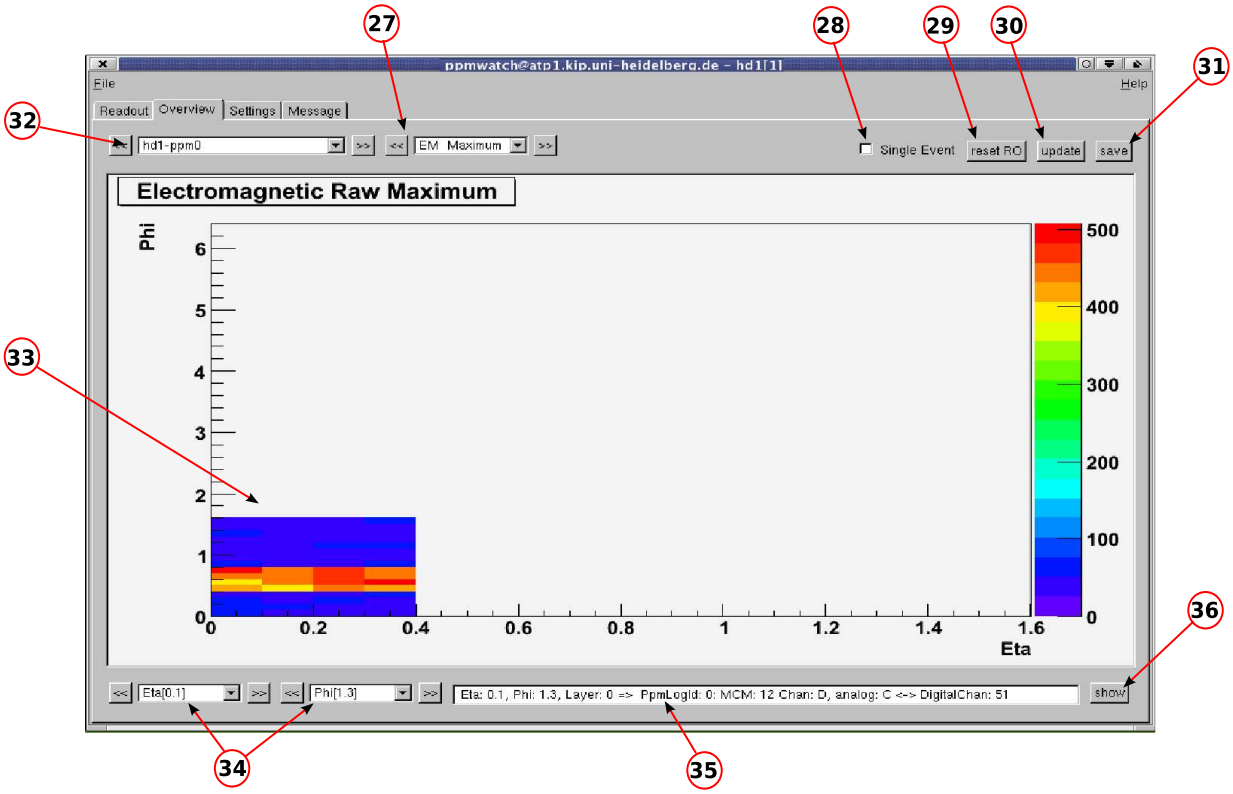

Figure D.3: Screenshot of ppmwatch with the Overview Tab selected, details in the text.

The readout of the PPMs that are selected in (32) can directly be done from the Overview Tab (30). As the VME readout sometimes has problems with high rates of L1As the PPMs can be configured to only accept one L1A at a time (28). Additionally the readout path can be reset for the selected PPMs (29). The overview plot to display can be selected in (27).

Ppmwatch allows to identify the hardware channel that corresponds to a given η-ϕ region. In order to select the η-ϕ region of interest, one can either use (34) or directly click into the canvas. The hardware channel of the selected η-ϕ region is then displayed in (33). When clicking (34) ppmwatch changes to the Readout Tab and displays the readout for selected channel.

Glossary

ADC Analog to Digital Converter

AFG Arbitrary Function Generator

ALICE A Large Ion Collider Experiment

AnIn Analog Input (board)

API Application Programming Interface

ASIC Application Specific Integrated Circuit

ATLAS A Toroidal LHC Apparatus

B Byte

b Bit

BCID Bunch Crossing Identification

BF Bit Field

BLOB Binary Large OBjects

BSD Byte Stream Decoder

CAN Control Area Network

CB Calibration Board

CERN European Center for Nuclear Research

CIS Charge Injection System

CMM Common Merger Module

CMS Compact Muon Solenoid

cms center of mass

COOL COnditions Objects for LHC

CORAL COmmon Relational Abstraction Layer

CORBA Common Object Request Broker Architecture

CP Cluster Processor

CPLD Complex Programmable Logic Device

CPM Cluster Processor Module

CSC Cathode Strip Chambers

DAC Digital-to-Analog Converter

DAL Data Access Layer

DAQ Data AcQuisition

DSS Data Source Sink

EF Event Filter

EM ElectroMagnetic

EMS Event Monitoring Service

ERMRS Error Message Reporting System

F/O Fan-Out

F/W Firmware

FADC Flash Analog-to-Digital Converter

FE Front End Electronics

FEB Front End Board

FIFO First In First Out

FIO Fan In-Out

FIR Finite Impulse Response

FPGA Field Programmable Gate Array

FWHM Full Width Half Maximum

GIO General purpose In Out

GNAM GNAM is Not AtlMon

GUI Graphical User Interface

GUT Great Unifying Theory

HAL Hardware Access Layer

HDMC Hardware Description Monitoring and Control

HLT High Level Triggers

HV High Voltage

I^2C Inter-Integrated Circuit

IGUI Integrated Graphical User Interface

ILU Inter-Language Unification

IOV Interval Of Validity

IPC Inter-Process Communication

IS Information Service

JEM Jet Energy Module

JEP Jet Energy Processor

KIP Kirchhoff Institute of Physics

L1A Level-1 Accept

L1Calo Level-1 Calorimeter Trigger

LCD LVDS Cable Driver

LCG LHC Computing Grid

LEP Large Electron Positron Collider

LHC Large Hadron Collider

LHCb Large Hadron Collider beauty

LS Laser System

LTP Local Trigger Processor

LUT Look-Up Table

LVDS Low Voltage Differential Signal

MRun Milestone Run

MCM Multi Chip Module

MDT Monitored Drift Tube

MP Monitoring Processes

MRS Message Reporting System

MS MultiStep

MSB Most Significant Bit

MSSM Minimal Supersymmetric Standard Model

NIM Nuclear Instrumentation Module

OHS Online Histogramming Service

OKS Object Kernel Support

OO Object Oriented

PC Personal Computer

PEEK Polyetheretherketones

*PHOS*4 Four channel delay generator

PMG Process Manager

PMT PhotoMultiplier Tube

PP Pre-Processor

PPM Pre-Processor Module

PprASIC Pre-Processor ASIC

PprMCM Pre-Processor MCM

RAL Rutherford Appleton Laboratory

RC Run Control

RCD Rod Crate DAQ

RDO Raw Data Object

RemFPGA Readout Merger FPGA

RGTM Rear G-Link Transition Module

RM Readout Module

ROB Readout Buffers

ROC Readout Controller

ROD Readout Drivers

ROI Regions of Interest

ROS Readout Subsystem

RP Runplan

RPA RunPlanAction

RPC Resistive Plate Chamber

RPPP Receiver to Pre-Processor Patch Panel

Rx Receiver

S/W Software

SBC Single Board Computer

SFI Sub-Farm Input

SFO Sub-Farm Output

SIF Serial Interface

SM Standard Model

SNR Signal-to-Noise Ratio

SPI Serial Peripheral Interface

SQL Structured Query Language

TBB Tower Builder Board

TCM Timing and Control Module

$TCPP$ Tile Calorimeter Patch Panel

$TDAQ$ Trigger Data AQuisition

TGC Thin Gap Chamber

TR Transition Radiation

TT Trigger Tower

TTC Timing Trigger and Control

$TTCdec$ TTC decoder

$TTCrx$ TTC Receiver

UML Unified Modeling Language

VGA Variable Gain Amplifier

XML eXtendensible Markup Language

LAr Liquid Argon

Bibliography

[1] P.W. Higgs:
Broken Symmetries, Massless Particles and Gauge Fields
Phys. Rev. Lett. 1964, 12, 132

Spontaneous Symmetry Breakdown without Massless Bosons
Phys. Rev. 1966, 145, 1156

[2] ILHC collaboration:
http://project-i-lhc.web.cern.ch/project-i-lhc/Overview.htm

[3] S. Weinberg:
A Model of Leptons
Phys. Rev. Lett. 1967, 19, 1264

A. Salam:
Elementary Particle Theory
1968

[4] CERN PhotoLab:
Overall view of the LHC experiments (© copyright: CERN)
Cern Document Server (http://cdsweb.cern.ch)
Document: CERN-AC-9906026

[5] ATLAS Collaboration:
ATLAS Detector and Physics Performance TDR
CERN/LHCC/99-15

[6] ATLAS Collaboration:
ATLAS First Level Trigger TDR
CERN/LHCC/2003-022

[7] ATLAS Collaboration:
ATLAS High-Level Trigger Data Acquisition and Controls TDR
CERN/LHCC/98-14

[8] ATLAS Collaboration:
Liquid Argon Calorimeter TDR
CERN/LHCC/96-41

[9] A.D. Martin et al.:
Parton distribution and the LHC: W and Z production
Eur.Phys. J.C 14, 133-145 (2000)

[10] ATLAS Collaboration:
ATLAS Inner Detector TDR
CERN/LHCC/97-16

[11] ATLAS Collaboration:
Calorimeter Performance TDR
CERN/LHCC/96-40

[12] JimPilcher et al:
TileCal Electronics
http://hep.uchicago.edu/atlas/electr/electronics.html

[13] ATLAS Twiki:
AtlasTechnicalPaper
https://twiki.cern.ch/twiki/bin/view/Atlas/AtlasTechnicalPaper

[14] H.L. Lai et al:
Phys Rev. D55/1997 (1280)

[15] Steven Hillier et al.:
ATLAS Level-1 Calorimeter Trigger Cluster Processor Module Project Specification (Post PRR)
http://hepwww.rl.ac.uk/atlas-l1/Modules/CPM/CPM_Specification_2_03.pdf

[16] Uli Schäfer et al.:
ATLAS Level-1 Calorimeter Trigger Jet/Energy Processor Module Project Specification (Post PRR)
http://hepwww.rl.ac.uk/atlas-l1/Modules/JEM/JEMspec12.pdf

[17] Eric Eisenhandler:
ATLAS Level-1 Calorimeter Trigger Algorithm
CERN EDMS: ATL-DA-ES-0050

[18] Stephen Hillier for the L1Calo Group:
Level-1 Calorimeter Trigger: Cable Mappings and crate Layouts from Analog Inputs to Processors
CERN EDMS: ATL-DA-ES-0036

[19] Victor Andrei:
PPM Channel Mapping
http://indico.cern.ch/materialDisplay.py?contribId=s0t20&sessionId=s0&materialId=slides&confId=a062538

[20] Paul Hanke:
The Pre-Processor Module for the Level-1 Calorimeter Trigger
DRAFT, not yet published

[21] Ulrich Pfeiffer:
A Compact Pre-Processor System for the ATLAS Level-1 Calorimeter Trigger
Universität Heidelberg, HD-IHEP 99-11, HD-ASIC 50-0899

[22] J Garvay et al:
Bunch Crossing Identification for the ATLAS Level-1 Calorimeter Trigger
ATLAS Internal Note: DAQ-NO-051

[23] K. Anderson et al.:
Design of the front-end analog electronics for the ATLAS Tile Calorimeter
Nucl.Instrum.Meth.A551:469-476, 2005

[24] B. Gosdzik.:
Offline Comparison of TileCal Signals and Trigger Signals
Universität Heidelberg, HD-KIP 07-03

[25] Florian Föhlisch:
L1Calo RunPlan-Handler core package Documentation
http://atlas-l1calo.web.cern.ch/atlas-l1calo/dox/rpL1Calo/html/

[26] Florian Föhlisch, Rainer Stamen:
Efficient connectivity tests for a large number of parallel channels
http://www.kip.uni-heidelberg.de/atlas/db/ConnectivityTests.pdf

[27] *Level-1 Calorimeter Trigger Cabling Tests Twiki page*
https://twiki.cern.ch/twiki/bin/view/Atlas/LevelOneCaloCablingTest

Acknowledgments

In the four years I was working on this thesis I got to work with many people. First there are my colleagues from Heidelberg, some of which worked on the Pre-Processor, some of which worked on other projects. Then there are all the people in the Calorimeter Trigger Collaboration, with some of which I spent a lot of time in RAL, Mainz, CERN and Heidelberg. After we started commissioning hardware to CERN I also worked together people from other parts of ATLAS like the Calorimeters and the CTP.

I want to thank everyone who helped me with my work, and apologize that I cannot mention them all here. My gratefulness goes to:

- Prof. Karlheinz Meier for giving me the opportunity to work on this interesting project, his explanations and for his support;
- Prof. Ulrich Uwer for kindly having agreed to be the co-referee of this work;
- Paul Hanke for many useful and practical answers to my questions and for doing much of the work in the background;
- Klaus Schmitt and Peter Stock for teaching me all I wanted (or needed) to know in a very enjoyable working climate while giving me the impression that I found all these things out myself;
- Prof. Hans-Christian Schultz-Coulon for all he did for our group and his valuable advices how to present my results;
- Rainer Stamen, for the good teamwork and for reading parts of this thesis;
- Victor Lendermann, Pavel Weber and all the other colleagues from Heidelberg for many discussions;
- Jürgen Stiewe for reading this thesis and answering all my naive questions mainly at the end of the thesis;
- Murrough Landon from whom I learned so many things for patiently sharing all his knowledge;
- Norman Gee and Bruce Barnett for many inspiring debates;
- Monica Dunford and Chaouki Boulahouache who made the tests with the calorimeters possible;

My special thanks go to my family and my friends for their support and amity.