BERGISCHE
UNIVERSITÄT
WUPPERTAL

Studies on the optical readout of the ATLAS Insertable b-Layer

From firmware development to commissioning
of the Back-of-Crate card

**Dissertation zur Erlangung
des akademischen Grades**

Dr.-Ing.

vorgelegt von

Marius Wensing

Erstgutachter:	Univ.-Prof. Dr. Ullrich Pfeiffer, Bergische Universität Wuppertal, Fachbereich Elektrotechnik, Informationstechnik, Medientechnik
Zweitgutachter:	Univ.-Prof. Dr. Peter Mättig, Bergische Universität Wuppertal, Fachbereich Mathematik und Naturwissenschaften

Wuppertal, im August 2015

Die Dissertation kann wie folgt zitiert werden:

urn:nbn:de:hbz:468-20160502-095258-2
[http://nbn-resolving.de/urn/resolver.pl?urn=urn%3Anbn%3Ade%3Ahbz3A468-20160502-095258-2]

Abstract

The ATLAS Experiment is one of the world's largest experiments in particle physics and is based at the Large Hadron Collider (LHC) at CERN in Geneva. During the LHC shutdown in 2013-2015 the ATLAS Pixel Detector as the innermost tracking detector has been upgraded by a new fourth layer, the Insertable b-Layer (IBL). The IBL uses new pixel sensors and front-end electronics with higher readout bandwidths to cope with the LHC conditions and due to its position close to the interaction point. Therefore, it requires a totally new designed readout system. The IBL readout system will be based on the readout system of the Pixel Detector and consists mainly of the Readout-Driver (ROD) and the Back-of-Crate (BOC) card.

The BOC card is responsible for sending and receiving the optical data signals from and to the detector and forwarding it to the ROD. This thesis deals with the development of the firmware for the BOC card and its testing during the production and in the full readout system. A main focus will be on the implementation of the fine delay to adjust the detector timing and the comparision of different implementations. A new approach using the partial reconfiguration will be shown as well.

Finally, this thesis will conclude with an outlook on future readout architectures for the upgrade of the ATLAS Inner Detector in 2024. New readout concepts will be shown to tackle the requirements of the future detector.

Contents

1 Introduction **1**

2 Basics of experimental particle physics **3**
2.1 Units 3
2.2 The Standard Model of particle physics 3
2.2.1 Fermions 3
2.2.2 Interactions between elementary particles 5
2.2.3 Higgs boson 8
2.2.4 Beyond the Standard Model 8
2.3 Experimental particle physics 9

3 The Large Hadron Collider and the ATLAS Detector **13**
3.1 The Large Hadron Collider 13
3.2 The ATLAS Detector 15
3.2.1 Overview 15
3.2.2 Subdetectors 16
3.2.3 Coordinate system 17
3.2.4 Trigger and Data Acquisition system 17

4 The ATLAS Pixel Detector and its new fourth layer **19**
4.1 The Pixel Detector during Run-1 19
4.2 The Insertable b-Layer 20
4.3 The IBL Detector Control System 21
4.4 Overview of the IBL readout system 21
4.5 On-detector readout components 22
4.5.1 Pixel sensors and the FE-I4 22
4.5.2 Optoboard 24
4.6 Off-detector readout components 27
4.6.1 Timing-Trigger-Control Interface Module 27
4.6.2 Back-of-Crate card 28
4.6.3 Readout-Driver 28
4.6.4 Readout-Subsystem 30
4.7 Timing of the ATLAS Pixel Detector 30
4.8 Testing and commissioning steps 31
4.9 Outlook on the Layer-1 and Layer-2 readout upgrade 32

5 Requirements for the ATLAS IBL BOC card **35**
5.1 Mechanical requirements 35
5.2 Detector interfaces 35
5.2.1 TTC path to the detector 36
5.2.2 Data path from the detector 37

5.3 ROS interface 38
5.4 Clock distribution 39
5.5 External Interfaces 40
5.5.1 Setup-Bus 40
5.5.2 Ethernet 40
5.5.3 CAN 41
5.6 Summary of the requirements 41

6 Hardware of the ATLAS IBL BOC card 43
6.1 PCB design 43
6.2 FPGAs and direct periphery 43
6.3 Clock circuit 46
6.3.1 First version 46
6.3.2 Production version 47
6.4 Optical plugins 47
6.5 DCS monitoring 49

7 Firmware development for the ATLAS IBL BOC card 51
7.1 Introduction & Overview 51
7.2 Design flows 51
7.2.1 Simulation 51
7.2.2 Synthesis 52
7.3 BCF firmware 52
7.3.1 Global system interconnect and register bank 52
7.3.2 BMF firmware loading 55
7.3.3 Clock monitoring and control 56
7.3.4 Microblaze processor and its software 57
7.4 BMF firmware 58
7.4.1 Clock generation 59
7.4.2 Register bank 60
7.5 BMF transmitter firmware 60
7.5.1 Overview 60
7.5.2 Coarse delay (step I) 60
7.5.3 BPM encoding 61
7.5.4 Coarse delay (step II) 62
7.5.5 Serialiser 62
7.5.6 Fine delay 62
7.5.7 Debugging and monitoring tools 67
7.5.8 Simulation results of the transmitter firmware 68
7.6 BMF receiver firmware 70
7.6.1 Overview 70
7.6.2 Clock and data recovery 70
7.6.3 Channel multiplexer 71
7.6.4 Decoding 71
7.6.5 ROD data bus 72
7.6.6 Data Monitoring 72

7.7 FE-I4 emulator firmware . . . 74
7.8 BMF S-LINK firmware . . . 75
7.9 IBL BOC software library . . . 77

8 Production of the IBL BOC card 79
8.1 IBL BOC production . . . 79
8.2 Overview of the production test . . . 79
8.3 Production test setup . . . 80
8.4 Production test steps and their results . . . 80
8.4.1 Optical & electrical inspection . . . 80
8.4.2 Firmware loading & Network configuration . . . 82
8.4.3 Clock test . . . 83
8.4.4 Loopback tests . . . 83
8.4.5 Optical parameters . . . 85
8.4.6 Backplane & S-LINK . . . 87
8.4.7 Production test database . . . 90
8.5 Failures detected during the production test . . . 90
8.5.1 BCF memory problems . . . 90
8.5.2 Ethernet problems . . . 91
8.6 Summary of the production test . . . 91

9 The IBL BOC card in the system test 93
9.1 Introduction . . . 93
9.2 The Lab infrastructure . . . 93
9.2.1 Overview . . . 93
9.2.2 Front-end chips . . . 93
9.2.3 Optoboard & adapter PCB . . . 95
9.2.4 ROD/BOC readout hardware . . . 96
9.2.5 ROS PCs . . . 96
9.2.6 Software Infrastructure . . . 96
9.2.7 Scanning in the local infrastructure . . . 98
9.3 System test at CERN . . . 100
9.3.1 Overview of the SR1 and Pit setup . . . 100
9.3.2 FE-I4 corruption issue . . . 101
9.3.3 S-LINK issue . . . 104
9.3.4 IBL detector timing . . . 105
9.4 Summary of the system test . . . 110

10 Future readout concepts 113
10.1 LHC and ATLAS upgrade plans . . . 113
10.2 The ATLAS ITk project . . . 113
10.3 Future readout architectures . . . 114
10.4 GBT and Versatile Link . . . 115
10.4.1 Overview . . . 115
10.4.2 GBT studies in Wuppertal . . . 117
10.5 The FELIX readout concept . . . 121
10.5.1 Overview . . . 121

10.5.2 Concepts for ITk readout and calibration 122
10.6 PCIe studies 125

11 Summary and Outlook 129

List of Tables 131

List of Figures 136

List of Abbreviations 137

Bibliography 139

1 Introduction

Experimental particle physics is a field of physics, which is looking into the fundamental building blocks of all the matter we know. These building blocks and their interactions are described by fundamental theories, like the Standard Model (SM) of particle physics. Experimental particle physics is aiming to prove or disprove these theories by testing their predictions against measurements. A basic introduction into experimental particle physics and the SM of particle physics will be given in Chapter 2.

The world's largest experiment in particle physics is the Large Hadron Collider (LHC) at the European Organization for Nuclear Research (CERN). It is located in a 45 to 170 m underground, 27 km long circular tunnel on the border between Switzerland and France near Geneva. In the LHC bunches of protons circulating with almost the speed of light are brought to collision. Particles generated by these collisions can be measured by particle detectors. The results of these measurements can be analysed to verify theories in particle physics. At the LHC four main experiments are operated: ATLAS, CMS, ALICE and LHCb. Chapter 3 will give an overview about the LHC and the ATLAS experiment, in the context of which this thesis was written.

ATLAS is a general-purpose particle detector and one of its subdetectors is the Pixel Detector. The Pixel Detector is a pure silicon detector and it is part of the ATLAS Inner Detector. Its main purpose is the tracking of particle trajectories. The charge which is deposited by a charged particle passing through the pixel cell can be measured to identify the momentum of these particle. The Pixel Detector provides around 80 million readout channels distributed over three layers and six disks.

During a technical shutdown of the LHC in 2013-2015 the Pixel Detector has been upgraded with the Insertable b-Layer (IBL). The IBL has been installed into the Pixel Detector as a new fourth innermost layer. This upgrade adds 12 million pixels to the Pixel Detector and increases its tracking resolution. A more detailed description of the ATLAS Pixel Detector and its IBL upgrade will be presented in Chapter 4.

To readout the data of the additional 12 million channels, the Data Acquisition (DAQ) system has been upgraded as well. IBL is equipped with newly developed front-end electronics, and they require also the off-detector readout system to be adapted. The off-detector readout system consists mainly of a card pair: the Readout-Driver (ROD) and the Back-of-Crate (BOC) card. The ROD is steering the readout process, whereas the BOC card is dealing with the optical interfaces to the detector and the higher-level readout. In the course of the IBL upgrade both cards have been completely redesigned.

This thesis will deal with the development of the BOC card, especially its firmware. Before the firmware development has started, a number of requirements were defined.

The hardware as well as the firmware has to fulfil these requirements to ensure a proper readout of the detector under all circumstances. Some requirements, like interface descriptions, already apply to the hardware or firmware development, and some requirements need to be checked for every single card after production. These requirements will be presented in Chapter 5.

In Chapter 6 will give an overview of the IBL BOC hardware. It was developed by the University of Heidelberg and is equipped with state-of-the-art Field Programmable Gate Arrays (FPGAs) for signal processing and commercial optical plugins to connect to the detector through 80 m of optical fibres.

A central part of this thesis is the development and testing of the FPGA firmware for the BOC card. Chapter 7 will describe the firmware building blocks and their functionality. The focus will be on the signal processing in the data path. To adjust the timing of the detector, delays with a granularity in the range of 100 ps are needed. Their implementation in the firmware using partial reconfiguration will be shown. The data path from the detector will be described focusing on the sampling and decoding of the incoming detector data. Also monitoring tools for the transmission and reception data path will be presented.

Another important aspect is the testing of all cards after their production. In Chapter 8 the test procedure during the production is shown together with results from the testing. Especially, the measurement of the fine delay is an important part of the production test. The production test should ensure that only the best quality is being delivered to CERN.

Similar to the production test, also a system test of the card has been performed. The system test and its results will be depicted in Chapter 9. Parts of these tests were performed in Wuppertal in a local laboratory setup, but the most important tests have been performed at CERN together with the real detector. Results from these tests had also a large impact to the firmware development. Issues of the readout system including the card will be presented together with changes solving these issues.

Finally, the thesis will conclude with an outlook on future readout systems for the next upgrade of the ATLAS Inner Detector. It is foreseen to be installed in 2024. A totally new Inner Tracker will be installed requiring a complete redesign of the readout system. First concepts of these new readout system will be shown as well as evaluation studies performed in Wuppertal to study new readout architectures.

2 Basics of experimental particle physics

2.1 Units

Particle physics is a field of physics dealing with smallest dimensions. Therefore, it has its units adjusted to express physical properties. Most of these properties are derived from energy as a fundamental unit in particle physics.

The energy of particles is given in the unit eV. 1 eV is the energy of an electron accelerated by a voltage of 1 V in an electric field and it corresponds to $1.602 \cdot 10^{-19}$ J.

The total energy of particles can be expressed by $E^2 = m^2c^4 + pc^2$, where m is the rest mass of the particle, p its momentum and c the speed of light in vacuum ($c = 299792458\,\frac{\mathrm{m}}{\mathrm{s}}$ [1]). Therefore, the mass of particles can also be expressed in units of eV$/c^2$ and their momentum in units of eV$/c$. A mass of mass of 1 eV$/c^2$ corresponds to $1.783 \cdot 10^{-36}$ kg.

Also the size of particles is expressed in very small units. A common unit for the size is the femtometer, which is 10^{-15} m.

2.2 The Standard Model of particle physics

The Standard Model (SM) [2, 3] of particle physics is an unified theory explaining what the world as we know is made of, and what the fundamental forces between particles are. Up to the present day the SM describes all known elementary particles and their interactions. Figure 2.1 shows an overview of all particles described by the SM.

The SM distinguishes between two types of particles. The matter particles are the fermions which have a half-integer spin. The interactions between these particles are modeled by the exchange of particles. These particles have a spin=1 and therefore are bosons. Each particle also has its anti-particle with opposite charge.

2.2.1 Fermions

The fermions are the matter particles in the SM. They all are spin= $\frac{1}{2}$ particles and characterised by the Fermi-Dirac statistics. An important principle in the Fermi-Dirac statistic is the Pauli exclusion principle. It states that two fermions cannot have the same quantum state in a system where the wave functions of the two fermions is overlapping. The fermions overall are divided into three generations. Particles of higher generations have nearly the same physical characteristics, but a higher mass and shorter life-time than the corresponding particle of the first generation. Each generation of particles contains two quarks and two leptons.

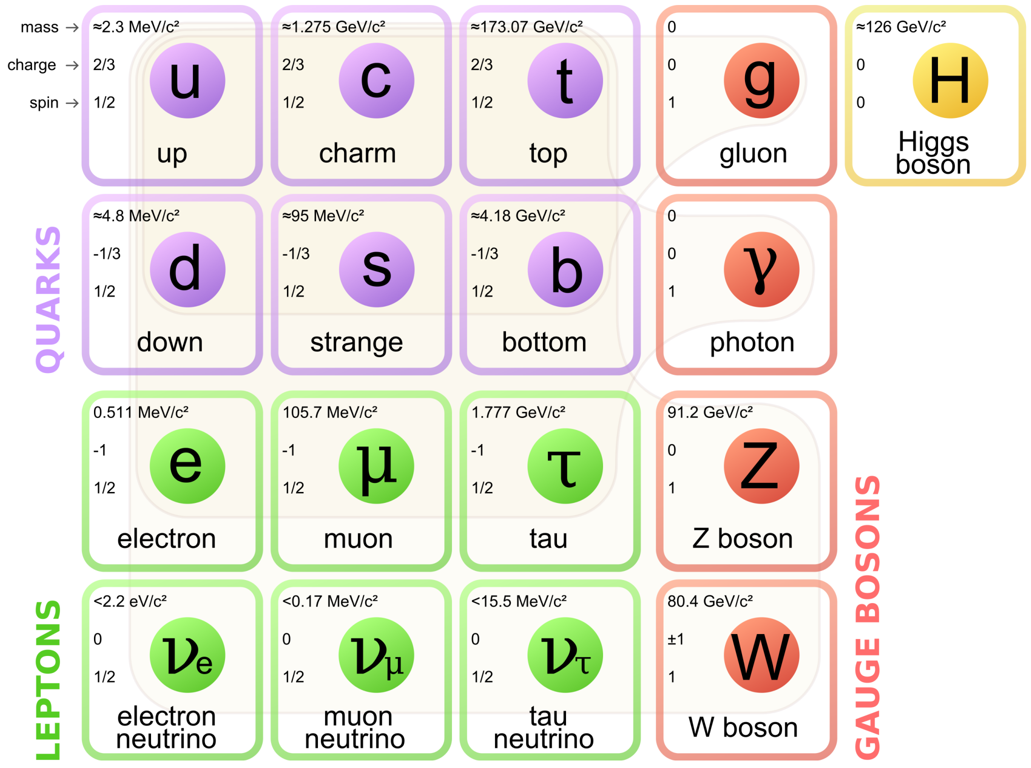


Figure 2.1: Overview of all particles described by the Standard Model of particle physics [4].

The matter we know today is built out of particles of the first generation. The electron, a lepton, was discovered in 1897 by J. J. Thomson. Electrons form the atomic shell and carry an elementary charge of $-e = -1.602 \cdot 10^{-19}$ As [1]. Protons and neutrons form the atomic nucleus, and they consist of a combination of the up- and down-quarks. Two up- and one down-quark form a proton, one up- and two down-quarks form a neutron. Both quarks are charged particles, the up-quark carries a fractional charge of $\frac{2}{3}e$ and the down-quark carries $-\frac{1}{3}e$. Therefore, the proton has a total charge of $+e$, whereas the neutron is uncharged. The last of the four fundamental particles in the first generation was first postulated by W. Pauli in 1930. He tried to explain the energy spectrum of the β^--decay. The energy spectrum of the β^--decay is a wide spectrum that can only be observed if two particles are emitted. As the electron has been the only observed particle, he postulated that another electrically neutral and hard to observe particle is being emitted. He called this particle the “neutron”. Later it was renamed to “neutrino” to avoid misunderstandings with the neutron we know today. In 1953, 23 years after Pauli's postulation, it was discovered by C. L. Cowan and F. Reines [5].

During experiments in the last years, more particles have been found. These particles build the second and third generation. The second generation consists of the charm- and strange quark and the muon with its corresponding neutrino. The third generation is formed by the top- and bottom quark together with the tau and tau-neutrino. The particles of the higher generations are not stable and decay into particles of the first generation. These particles can only be found in particle colliders or cosmic rays, because of their short life-time, and the high energy that is needed to create them.

2.2.2 Interactions between elementary particles

The interactions between elementary particles are described by the SM in the form of exchange particles. These particles are vector bosons. Each type of interaction can have one or more particles associated. The properties of these exchange particles define the strength and the range of the interaction. Figure 2.2 shows all particles of the SM and their interaction. A line between these particles means that they can interact with each other. A closed loop corresponds to a self-interaction.

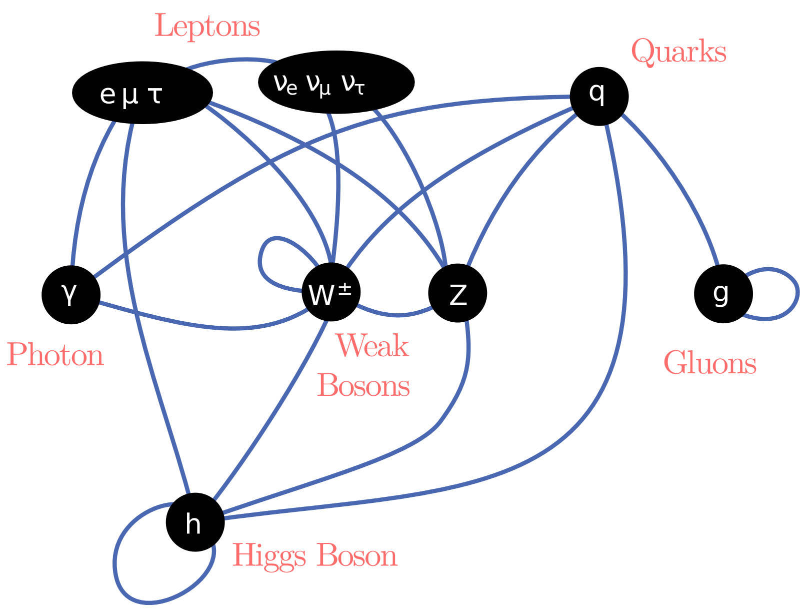


Figure 2.2: Overview of fundamental interactions between particles of the SM. A line between two particles indicates that they interact with each other, a loop corresponds to an interaction between particles of the same type [6].

The SM describes three fundamental forces: the strong force, the weak force and the electromagnetic force. Table 2.1 shows an overview of the three forces and the properties of the corresponding bosons. The gravitation, as the fourth of the fundamental forces in physics can be neglected due to its low relative strength of 10^{-38} compared to the strong force. The exchange particle of the gravitation is the hypothetical Graviton and has not yet been observed.

The electromagnetic force

The electromagnetic force is the most noted force of the fundamental interactions. Its exchange particle is the photon. A large number of physical observations have their origin in the electromagnetic force. For example, the binding of electrons to the atomic nucleus is a consequence of the electromagnetic interaction. It only couples

Table 2.1: Overview of the three fundamental forces in the SM

force	**strong**	**electromagnetic**	**weak**
relative strength	1	10^{-2}	10^{-13}
range [m]	$2.5 \cdot 10^{-15}$	∞	10^{-18}
bosons	8 gluons	photon	W^+, W^-, Z
boson mass [GeV/c^2]	0	0	80, 80, 91

to electrically charged particles. Therefore, all fermions except the neutrinos are influenced by the electromagnetic force.

The weak force

The weak force was first discovered in the β-decay of nuclei. It is called "weak", because its range and strength is lower than the strong force. The exchange particles of the weak force are the $W^{\pm}$ and Z bosons. The limited range is a result of short lifetime of the boson. For example, in the β^--decay a neutron decays into a proton by changing one of its down-quarks into an up-quark. Figure 2.3 shows the Feynman diagram of this decay. In this example the exchange particle is the W^- and it carries the charge difference between the up- and down-quark. Finally, the W^- decays into an electron and an electron anti-neutrino [7].

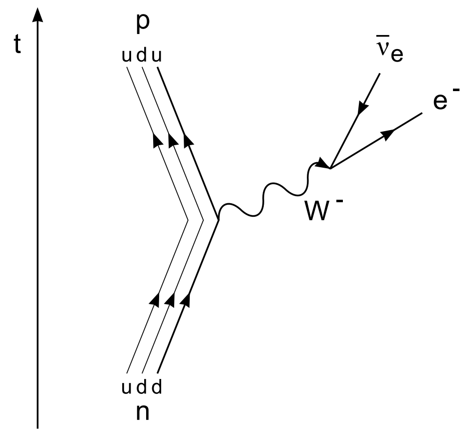


Figure 2.3: Feynman diagram of the β^- decay. One of the down-quarks in the neutron decays into an up-quark. Finally, the resulting W^- boson is decaying into an electron and an electron anti-neutrino [7].

The weak force couples to all fermions and enables them to transform into other fermions. This process is called "flavour change" and converts quarks into other quarks, and leptons into other leptons. These interactions only occur as charged interactions with the $W^{\pm}$ as exchange boson. Flavour changes with the Z boson have not yet been observed.

First experiments in 1973 have shown that the electromagnetic force and the weak force are very similar interactions at high energies. Therefore, both can be combined in the "electroweak" interaction [2]. Glashow, Salam and Weinberg were awarded with the Nobel Prize in Physics 1979 for their prediction of the electroweak interaction [8, 9]. This leads to the theory that all interactions are only different aspects of the same force at low energies and that they unify at high energies. Currently, physicists are searching for the unification of the electroweak and the strong force (see next section) at high energy scales.

The strong force

In the SM, the strong force is exchanged by gluons and it is responsible for the binding of quarks to hadrons. Similar to the charge-coupling of photons, gluons in the SM couple to the colour charge of particles. Only particles carrying colour can interact with the strong force.

The only particles carrying a colour charge are quarks and gluons. There are six different colour charge values: red, green, blue and their anti-colours. A neutral colour "white" can be either achieved by combining red, blue and green or colour and anti-colour. These "white" particles do not interact with the strong force. Also leptons do not carry any colour charge, so they do not interact with the strong force.

The strong force is increasing with the distance. Figure 2.4 shows the potential well of the strong force as a function of the distance. It means that the work which is neccessary to seperate two colour charged particles is increasing with the distance between the two particles. The energy introduced by the work is high enough to create new quark-antiquark pairs. Therefore, only neutral combinations of quarks can be observed. This effect is called "confinement".

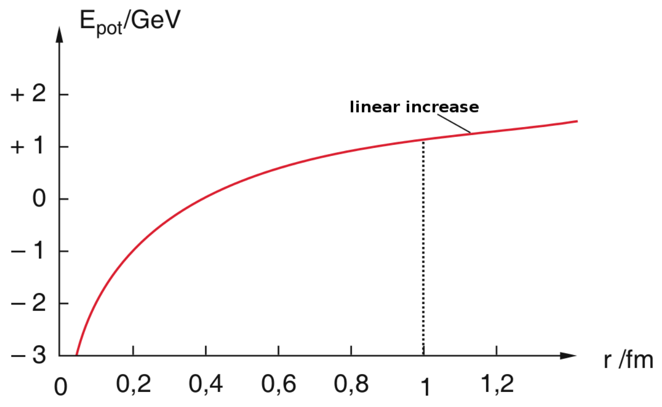


Figure 2.4: Potential energy between two quarks as a function of their distance. For low distances it is comparable to the electrical potential, at high distances it is linear. [10]

Due to the confinement only bound states of quarks can exist. These bound states are called "hadrons" and they split up into different groups. All particles consisting of three quarks are called baryons and particles consisting of a quark-antiquark pair are mesons. The most famous baryons are the proton and the neutron which form the atomic nucleus. Also bound states of more than three quarks exist. The tetraquark as a bound state out of two quarks and two anti-quarks or a meson-meson molecule has been observed by the Belle [11] and LHCb [12] experiments. In July

2015, the LHCb experiment announced the observation of the pentaquark. It is a bound state of four quarks and one antiquark [13]. The process of building bound states of quarks is called “hadronisation”.

If a quark leaves the interaction point the energy of the interaction increases with the distance due to the confinement. At some point this energy is large enough to create a new quark-antiquark pair in the process of “hadronisation”. This process is repeated many times and all created hadrons have the same direction as the first particle. If many of these hadrons appear in a narrow solid angle, this is called a “hadronic jet”. These jets can be measured by particle detectors and allow one to identify the quark which produced the jet.

Interesting effects can be observed, when B-mesons (mesons containing b-quarks) are created by particle interactions in a collider. These B-mesons have a short lifetime in the order of 10^{-12} s, but it is long enough to leave the interaction point by a few millimeters. When a b-quark decays, the emitted particles have the origin of trajectory not in the interaction point, but a few centimeters away. These trajectories are called “secondary vertices” and their identification is the “b-tagging”. To have a good b-tagging efficiency, it is required to identify the origin of a track with a high resolution. This is one reason for having a highly efficient tracking detector near the interaction point.

2.2.3 Higgs boson

In 1964, Peter Higgs [14], François Englert and Robert Brout [15] developed a mechanism to explain the mass of all particles in the SM. This mechanism explains mass as a coupling to a new scalar field, the “Higgs field”. The strength of the coupling is proportional to the mass of the particle for fermions. In the SM theory all interactions between particles are described by the exchange of a boson. For the Higgs mechanism this boson is called the “Higgs boson”.

After its postulation the search for the Higgs boson began. Several large experiments around the world contributed to this. Finally, in 2013, the ATLAS [16] and CMS [17] experiment at the LHC at CERN announced the discovery of the SM Higgs boson. The data leading to the discovery was taken during the LHC runs in 2011 and 2012. Figure 2.5 shows the discovery plot with the combined results of the ATLAS experiment. In the plot it can be seen that a new particle was found at a mass of 126 GeV/c^2, and it is compatible with theoretical predictions for the SM Higgs boson. The probability of statistical fluctuations being misinterpreted as a signal is less than 10^{-9}.

2.2.4 Beyond the Standard Model

Even if the current results of the particle physics experiments conform to what the SM predicts, many theories looking at physics beyond the SM exist. Reasons for these theories are the lack of explaination for dark matter and dark energy or the asymmetry of matter and anti-matter in the SM. Dark matter is matter which does not interact in reply to the electromagnetic force, but does affect other matter by gravitation [2]. Recent measurements predict that around 25 % of the universe

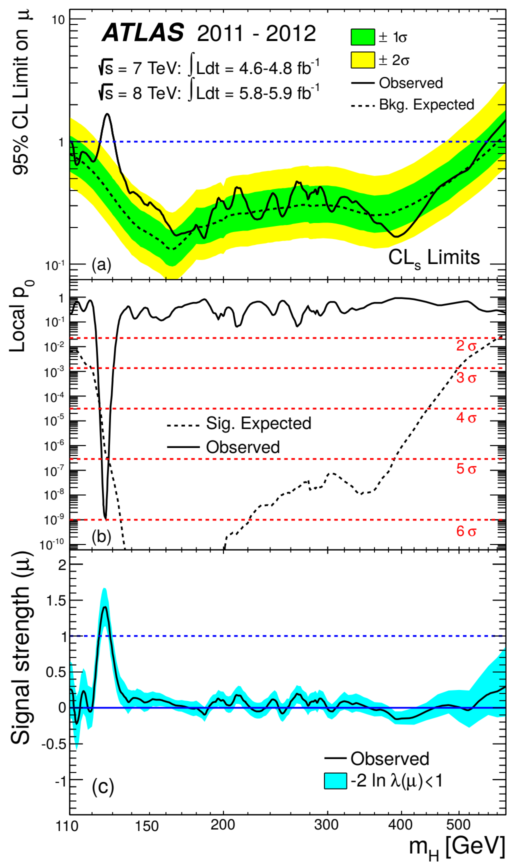


Figure 2.5: Combined results of the Higgs search: (a) exclusion limit on the signal strength, (b) probability of statistical fluctuations being misinterpreted as signal, (c) signal strength of a SM Higgs boson [16]

consist of dark matter [18]. Another reason for theories beyond the SM is the gravitation which cannot be explained by the SM.

One extension to the SM is the supersymmetry (SUSY). In this theory every particle of the SM has its supersymmetric partner (superpartner). The spin of superpartners is differing by a half-integer. Up to the present day, no SUSY particle has been detected in particle physics experiments around the world, and it is not yet clear if this theory is valid.

2.3 Experimental particle physics

The SM itself is a set of concepts, equations, and predictions of how elementary particles behave. To validate the SM, measurements need to be performed. This is the field of experimental particle physics. In the experiments the predictions of the SM are tested against measurements. If the measurements would deviate these predictions, this would be a sign for new physics beyond the SM.

In many particle physics experiments two particles are brought to collision. This collision is known as scattering. One of the first, widely known, scattering experiments has been performed by Rutherford in 1911. He observed the deflection of

α particles (Helium-4 nuclei) passing a thin gold foil. Most of the particles could pass the foil without being scattered, but some were scattered and even totally reflected. This was the first observation of the atomic nucleus and lead to the Rutherford model.

In general, the probability of a specific particle interaction can be expressed by its cross section defined as [19]:

$$\sigma = \frac{\text{number of reactions per unit time}}{\text{beam particles per unit time} \times \text{scattering centres per unit area}} \tag{2.1}$$

The cross section has the unit [m^2], but it is usually expressed in the unit “barn”. 1 barn = 1 b is equal to 100 fm^2. A large cross section corresponds to a large probability for the scattering process and vice-versa.

As a detector can only observe a fraction of all particles, only the differential cross section $\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}$ in a certain solid angle can be measured. To obtain the total cross section σ, it has to be integrated over the observation range:

$$\sigma = \int \frac{\mathrm{d}\sigma}{\mathrm{d}\Omega} \mathrm{d}\Omega \tag{2.2}$$

For example, the differential cross section as function of the scattering angle for Rutherford’s experiment is [7]

$$\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}(\theta) = \left(\frac{Z_1 Z_2 e^2}{16\pi\epsilon_0 E_\alpha}\right)^2 \frac{1}{\sin^4\frac{\theta}{2}} \tag{2.3}$$

Z_1 and Z_2 are the charge numbers of both particles, E_α the energy of the α particle and θ the polar scattering angle.

To calculate the event rate from the cross section, the number of collisions per time and per area is needed. This number is called “luminosity” and can be calculated for a colliding beam experiment by [19]

$$L = \frac{n \cdot N_1 \cdot N_2 \cdot f}{A}, \tag{2.4}$$

where N_1 and N_2 are the number of particles per bunch, n is the number of bunches, f the bunch crossing frequency and A the cross sectional area where the collisions are happening. The unit of the luminosity is m$^{-2}s^{-1}$. The event rate can then be expressed by:

$$\dot{N} = L \cdot \sigma \tag{2.5}$$

To calculate total number of events during an observation period, this equation has to be integrated. As σ is time-invariant, it is constant with respect to the integration:

$$N = \int \dot{N}\,\mathrm{d}t = \int L \cdot \sigma\,\mathrm{d}t = \sigma \cdot \int L\,\mathrm{d}t = \sigma \cdot L_{\text{int}} \tag{2.6}$$

L_{int} is commonly referred as the integrated luminosity. The unit of the integrated luminosity is b^{-1}. It is a characteristic variable for the delivered data of a particle physics experiment.

Figure 2.6 shows a plot of typical cross sections in proton-proton collisions as a function of the collision energy. For most of the processes, like the Higgs generation, the cross section is increasing with the collision energy. This means, the higher the energy and the luminosity in an experiment, the higher is the event rate.

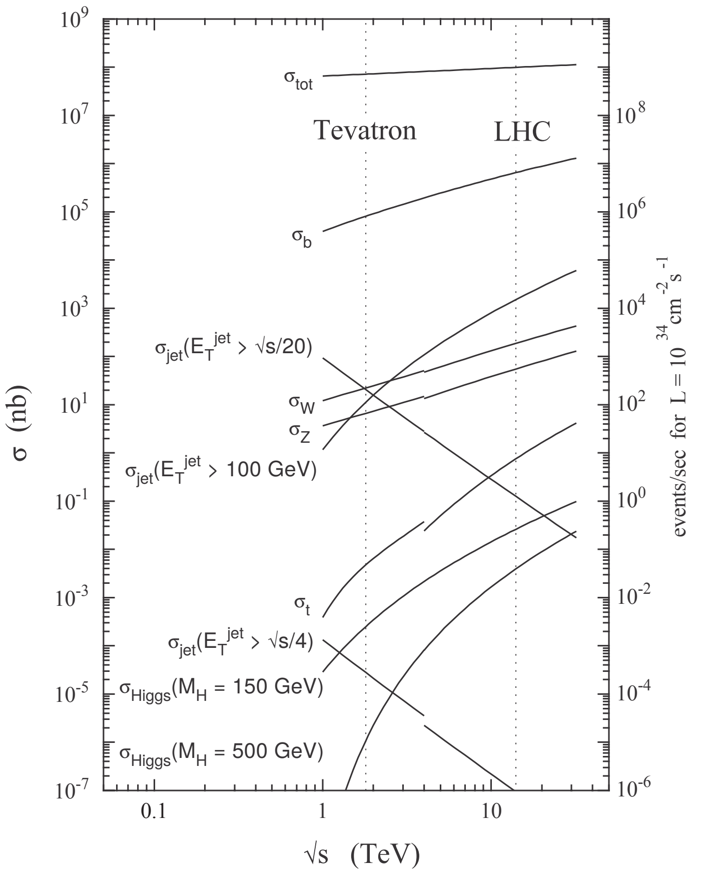


Figure 2.6: Cross sections and event rates in proton-proton collisions as a function of the collision energy. The dotted lines mark the collision energies of the Tevatron and LHC experiments. [20]

3 The Large Hadron Collider and the ATLAS Detector

3.1 The Large Hadron Collider

The Large Hadron Collider (LHC) [21] at the European Organization for Nuclear Research (CERN) near Geneva, Switzerland, is world's largest particle accelerator and collider. It utilizes the former 27 km long tunnel of the Large Electron-Positron Collider (LEP) on the border between Switzerland and France. The tunnel was built end of the 1980's. It is located between 45 and 170 m below the surface with an inclination of 1.4 % towards the Lake Geneva.

The LHC accelerates two opposing particle beams of either protons or heavy ions[1]. To expedite these particles, several stages of accelerators are needed. The particles are first speeded up in a linear accelerator (LINAC). From there they are injected into the Proton Synchrotron (PS). The next stage of acceleration happens in the Super Proton Synchrotron (SPS). Through two lines the beams are finally injected into the LHC. Figure 3.1 shows a sketch of the LHC accelerator complex.

The energy of the protons is designed to be up to 7 TeV, whereas the ions can be accelerated to an energy of 574 TeV. In a collision of the two particles, the energy of both particles is added to a total center-of-mass energy of $\sqrt{s} = 14\,\mathrm{TeV}$ for protons and $\sqrt{s} = 1148\,\mathrm{TeV}$ for heavy ions. The particles are injected into the LHC in bunches of 10^{11} particles with a spacing of 25 ns. The two beams circulate most of the time in independent beam pipes, but at the collision point they are combined into a single beam pipe and both particle beams are deflected by magnetic lenses (see Figure 3.2). In the focus of the two deflected beams the particles interact with each other. During a collision only a very minor fraction of the particles in the beams interacts. The number of interacting particles per bunch crossing is called "pile-up". Another important measure of a collider is the luminosity, which was already introduced in Section 2.3. For the LHC the luminosity was designed to be $10^{34}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$ with 40 MHz bunch crossing frequency. During Run-1 from 2009 until 2013 a beam energy of 4 TeV and a luminosity of $0.77 \cdot 10^{34}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$ at 20 MHz bunch crossing frequency has been reached [23]. Starting mid 2015, Run-2 aims for an initial beam energy of 6.5 TeV at the design luminosity and bunch crossing frequency.

To keep the particle beams on a circular trajectory, strong magnets are needed. The LHC magnets use field strengths of up to 8 T. This can only be realised by using superconducting materials. The magnets for the LHC are composed of titanium and niobium. They need to be cooled down to 1.9 K (-271.25 ° C) by superfluid helium.

[1]fully stripped $^{208}Pb^{82+}$-ions

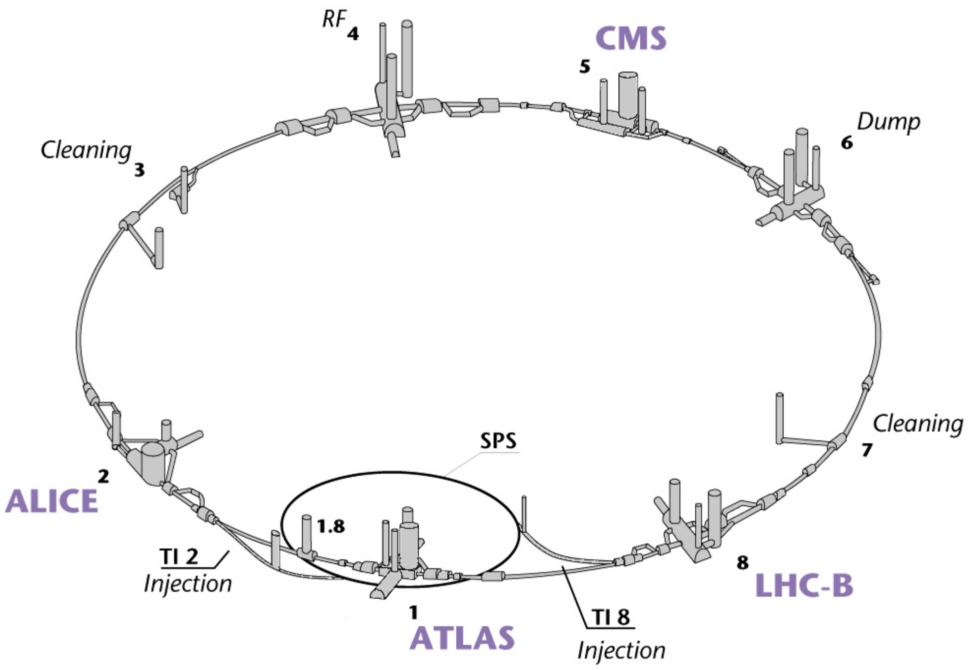


Figure 3.1: Overview of the LHC accelerator complex and the locations where the experiments and LHC services are placed (modified from [22]).

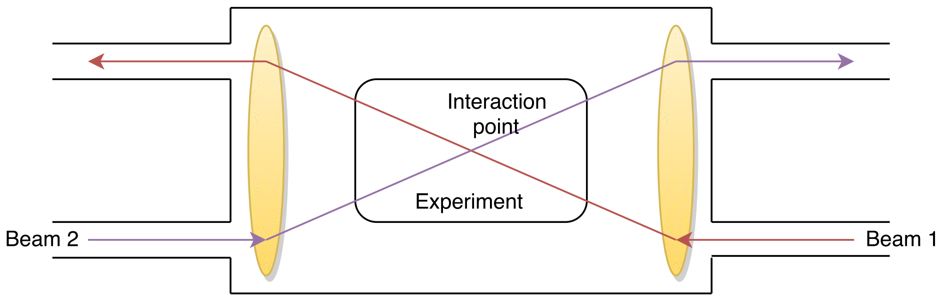


Figure 3.2: Principle of a colliding beam experiment, where two seperate beams are deflected by magnetic lenses (yellow) and colliding in the interaction point.

Distributed along the LHC ring are four points at which the two particle beams can collide. Around the collision points the experiments of the LHC are constructed. The two general-purpose experiments are ATLAS (A Toroidal LHC ApparatuS) and the CMS (Compact Muon Solenoid). The two smaller experiments are LHCb (LHC beauty), which is specialised on research on b-Physics, and ALICE (A Large Ion Collider Experiment), which is exploring quark-gluon plasmas. As this thesis is in the context of the ATLAS experiment, the next section will give an introduction of the experiment.

3.2 The ATLAS Detector

3.2.1 Overview

As stated in the previous section, ATLAS [24] is one of the four main experiments at the LHC. It is a multi-purpose detector meaning it is not specialised to a certain measurement, but is sensitive to a wide range of particle types. ATLAS is located on the Meyrin site (Point 1) at CERN in a cavern 100 m below the surface.

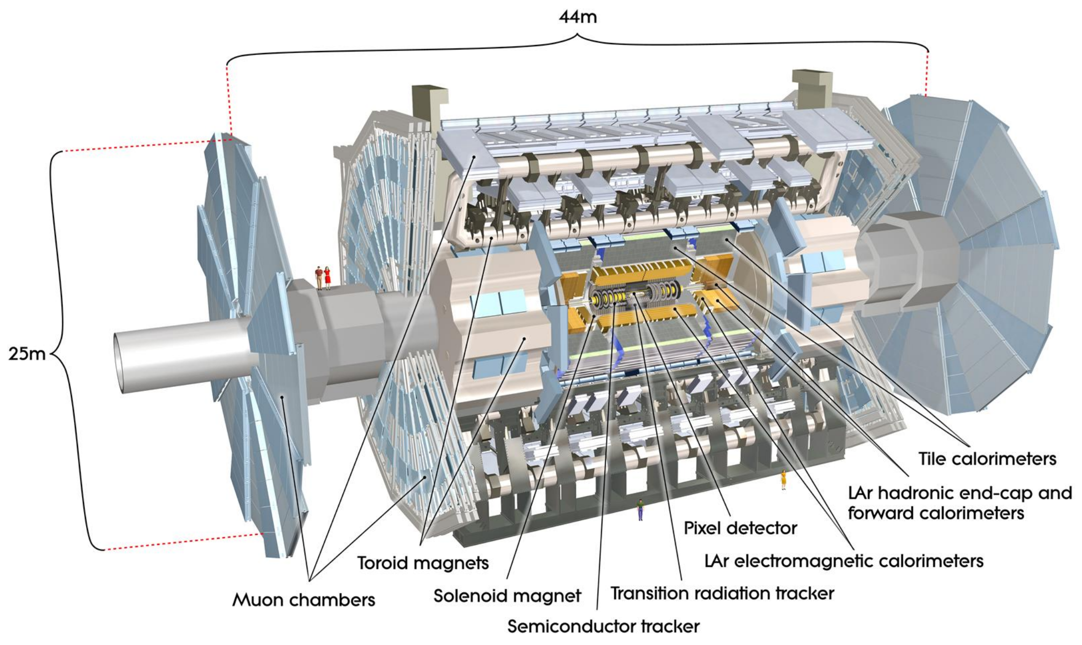

44m
25m
Tile calorimeters
LAr hadronic end-cap and forward calorimeters
Pixel detector
LAr electromagnetic calorimeters
Toroid magnets
Solenoid magnet
Transition radiation tracker
Muon chambers
Semiconductor tracker

Figure 3.3: Computer-rendered image of the ATLAS Detector [25].

The structure of the detector is comparable to an onion. Several layers of different detectors are built around the interaction point. This is typical for a colliding beam experiment and allows one to detect particles flying in all directions around the interaction point. Figure 3.3 shows a computer-rendered image of the ATLAS Detector. It has a length of 44 m, a diameter of 25 m, and a weight of 7000 t.

3.2.2 Subdetectors

The ATLAS Detector is built from a number of subdetectors. The placement of the subdetectors follows general design strategies of high energy physics colliding beam experiments. The general design of ATLAS is shown in Figure 3.4.

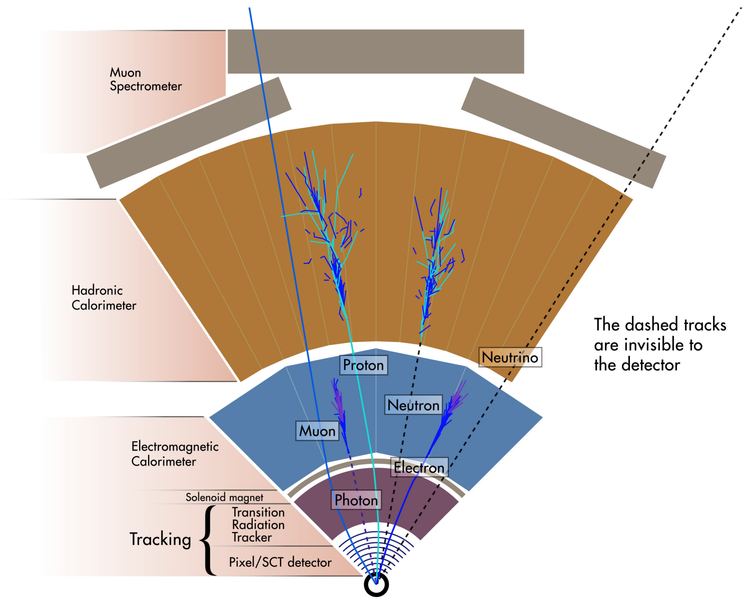


Figure 3.4: Conceptual design of the ATLAS Detector as a general-purpose particle detector in colliding beam experiments. The innermost part consists of a tracking detector. It is followed by the calorimeters and the muon subsystem [26].

The innermost part of ATLAS, close to the interaction point, is a tracking system, the "Inner Detector". It allows the measurement of the trajectory of charged particles. A strong solenoidal magnetic field with a field strength of 2 T forces charged particles on a circular trajectory allowing to measure the particle's momentum. The Inner Detector is built out of three subdetectors: the Pixel Detector, the Semi-Conductor-Tracker (SCT) and the Transistion-Radiation-Tracker (TRT). The Pixel Detector and SCT are pure silicon detectors, whereas the TRT is constructed out of drift tubes. The Inner Detector has tight constraints regarding the material inside the detector volume. It has to be ensured that the material inside the Inner Detector is kept at minimum to avoid that particles lose their energy.

The next stage of detectors are the calorimeters. The task of the calorimeters is the measurement of the energy a particle is depositing in the detector material. The energy of photons, electrons and positrons is measured in the first calorimeter stage: the electromagnetic calorimeter. These particles interact with the detector material by producing particle showers. Most of the hadrons can pass the electromagnetic calorimeter, but are stopped by the next stage: the hadronic calorimeter. They create hadronic showers by interacting with the detector material. Parts of the calorimeter system are used as a trigger source for the ATLAS trigger system.

The last detector stage is the muon system. Together with the Inner Detector it is used to precisely measure the tracks of muons and do a fast identification of muons. The full muon system is placed inside a toroidal magnetic field allowing the measurement of the muon momentum. Like the calorimeters the muon system also provides input to the ATLAS trigger system.

3.2.3 Coordinate system

To express coordinates in the ATLAS Detector Cartesian coordinates are used. Figure 3.5 shows a picture of the ATLAS coordinate system. The origin of the coordinates is the interaction point in the center of ATLAS. The x-axis is pointing towards the center of LHC ring, the y-axis is pointing upwards and the z-axis is pointing in the same direction as beam 2. The positive z-direction is referred as the A-side, the negative z-direction is called the C-side.

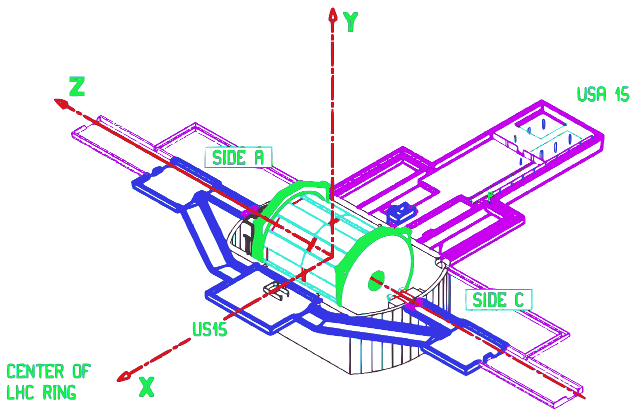


Figure 3.5: ATLAS coordinate system and the placement of the counting rooms [27].

To better cope with the physics point of view and the symmetry of ATLAS around the interaction point, radial coordinates can be used. The z-axis remains the same, but x- and y-axis are expressed by the azimuthal angle Φ and the polar angle θ. Usually θ is not directly used, but it is expressed by the pseudorapidity η. It is defined as [28]:

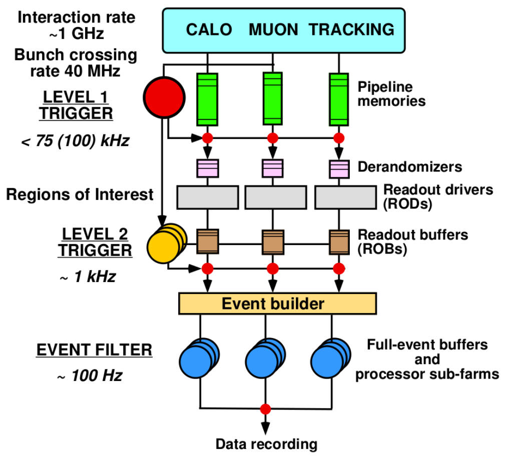
$$\eta = -\ln \tan \frac{\theta}{2} \tag{3.1}$$

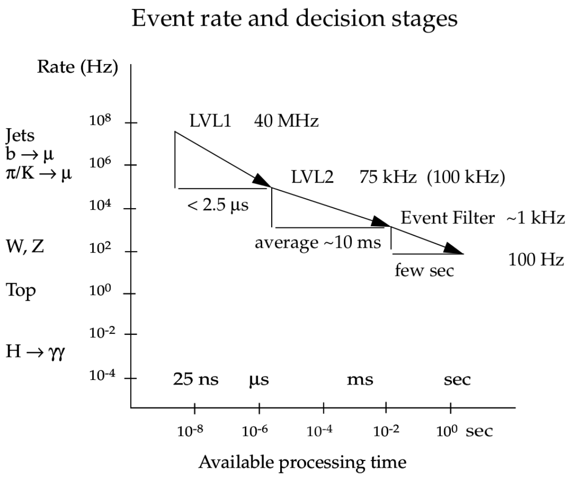
With this definition the plane orthogonal to the beampipe at $z = 0$ can be expressed as $\eta = 0$, whereas the forward and reverse direction are expressed by $|\eta| = \infty$. The maximal observation range of ATLAS covers a region up to $|\eta| \approx 5$.

3.2.4 Trigger and Data Acquisition system

The Trigger and Data Acquisition (TDAQ) system is a multi-stage global readout system for ATLAS. Its main purpose is to readout the detector and reduce the number of events to be written to the ATLAS storage. An overview of the TDAQ system is shown in Figure 3.6a. With the designed bunch crossing frequency of

40 MHz and a pile-up of 25 ATLAS generates an event rate of about 1 GHz. By using a three-stage trigger system interesting events are selected and the event rate written to disk is reduced to 100 Hz in the end.





(a) Overview of the Trigger and Data Acquisition **(b)** Event rates and decision times for the trigger system

Figure 3.6: Trigger and Data Acquisition system [24].

Level-1 trigger (LVL1)

The first trigger stage (LVL1) [29] is a pure hardware trigger with a maximum trigger frequency of 75-100 kHz. The decision has to be taken after a latency of 2.5 μs. The trigger decision is derived from special regions in the calorimeters and muon systems. The trigger decision is distributed from the Central Trigger Processor to the detector subsystems via the Timing-Trigger-Control (TTC) system. The LVL1 trigger system can trigger on "electrons/photons, hadronically decaying tau leptons, muons, jets and (missing transverse energy) E_T^{miss}" [29]. Until the trigger decision has been made, the front-end electronics need to store the hits in their internal buffers. The hits for a triggered bunch crossing are sent out to the Readout-Drivers (RODs), where they are collected and combined into event fragments. These event fragments are transferred to the Readout-Subsystem (ROS). It stores them until the next stage trigger decision has been made.

High-Level-Trigger (HLT)

The High-Level-Trigger (HLT) [20] consists of the Level-2 trigger system (LVL2) and the Event Filter (EF). It is a pure software trigger system. The LVL2 trigger uses regions of interest (RoI), which have been identified by the LVL1 trigger system. The LVL2 trigger system reduces the event rate to around 1 kHz. The latency of the LVL2 decision has to be less than 10 ms. The last stage of filtering is performed in the Event Filter. It reduces the event rate to around 100 Hz. In total the incoming data rate of all detectors has been decreased to a few 100 MB/s. During a year of running ATLAS is generating about 3200 TB of raw data [30].

4 The ATLAS Pixel Detector and its new fourth layer

4.1 The Pixel Detector during Run-1

The ATLAS Pixel Detector [31] is the innermost subdetector of ATLAS. It is a pure-silicon detector for tracking purposes. Figure 4.1 shows a computer-rendered image of the Pixel Detector. Its package is about 1.4 m long and has a diameter of 0.4 m which is about 60 times smaller in diameter than the whole ATLAS Detector.

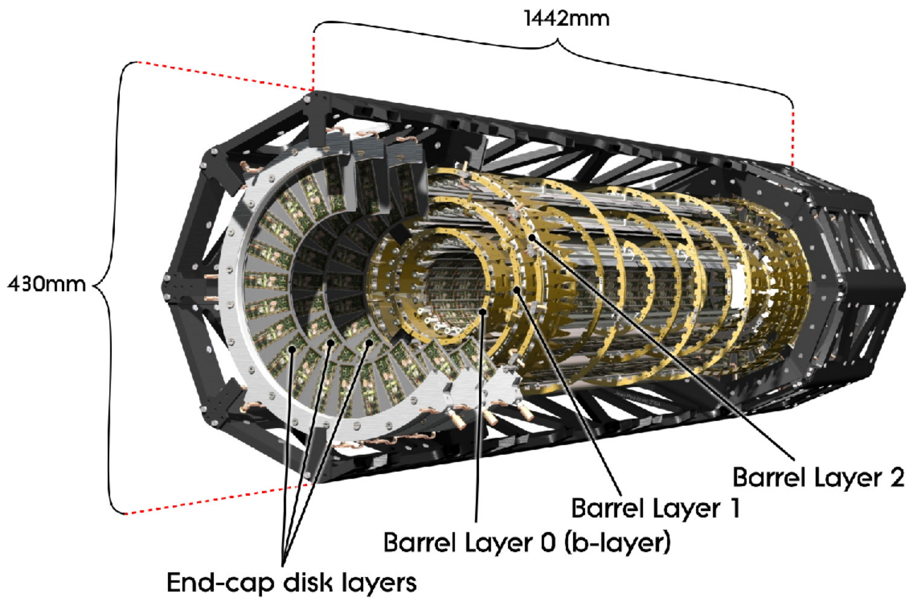


Figure 4.1: Computer-rendered image of the ATLAS Pixel Detector [32].

The Pixel Detector can be divided into two main sections. One section is the barrel section, which is placed around the beam pipe. The other section is the disk section in the forward region of the detector. The full Pixel Detector consists of three barrel layers and three disks per side. The barrel layers are called b-Layer, Layer-1 and Layer-2. Each layer is constructed out of 800 mm long mechanical support structures, the "staves". They carry the pixel sensors and also play an important part in the cooling of the detector as they conduct the heat away from the electronics to the cooling pipe. The Pixel Detector was designed to cover all tracks up to a pseudorapidity of $|\eta| = 2.5$ with three hits. This allows precise tracking and the reconstruction of primary and secondary vertices.

Overall, a total number of 80 million readout channels is provided by the Pixel Detector. This is almost 90 % of all available readout channels in the ATLAS Detector. Table 4.1 shows the placement and the parameters of the different pixel layers and disks.

Table 4.1: Placement and parameters of the different pixel layers and disks [33]

Item	**Radius** [mm]	**z-Position** [mm]	**Staves / Sectors**	**Modules**	**Pixels** ($\times 10^6$)
b-Layer	50.5		22	286	13.2
Layer-1	88.5		38	494	22.8
Layer-2	122.5		52	676	31.2
Disk 1		± 495	2×16	2×48	4.4
Disk 2		± 580	2×16	2×48	4.4
Disk 3		± 650	2×16	2×48	4.4

4.2 The Insertable b-Layer

During the LHC shutdown 2013-2015 the ATLAS Pixel Detector has been upgraded by a new innermost layer, the Insertable b-Layer (IBL) [33, 34]. Figure 4.2 shows the constructed IBL during its insertion into the ATLAS Pixel Detector in May 2014.

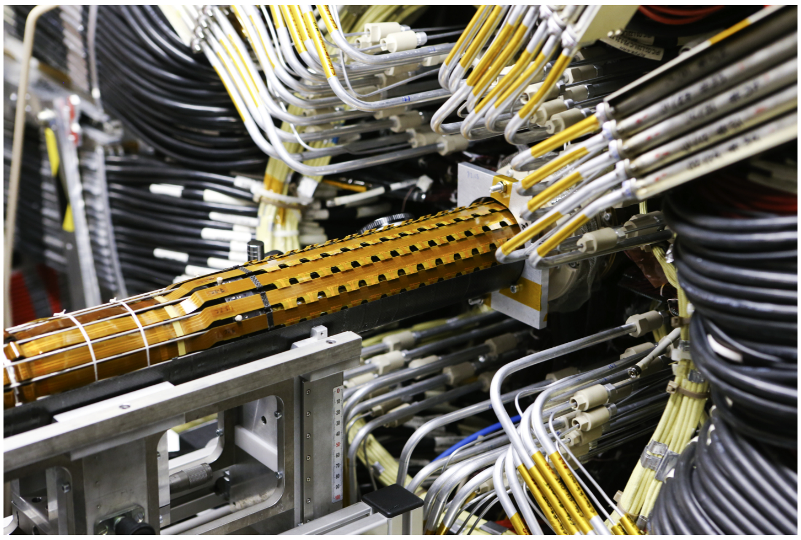


Figure 4.2: IBL during its insertion into the ATLAS Pixel Detector in May 2014 [35]

The planning for the IBL upgrade started in 2008. The first technical design report (TDR) [33] was published in 2010. It presents several motivations for the upgrade of the Pixel Detector. The main aspect is the improvement of the tracking robustness and precision. During its lifetime the Pixel Detector will suffer more and more from radiation damage. This causes the loss of detector modules and will decrease the tracking precision. With the IBL upgrade this loss in precision can be compensated and the precision can be even increased due to the proximity to the interaction point.

Another reason for the IBL upgrade are luminosity effects. The Pixel Detector is designed to handle the LHC luminosity of $10^{34}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$, but it is expected that the LHC will exceed this value in the next years of operation. With the higher than expected luminosity also the required readout bandwidth increases resulting in inefficiencies. These inefficiencies are also compensated by the IBL.

As LHC allows the installation of a new beampipe IBL will be installed between the new smaller beampipe and the b-Layer at a radius of 33 mm. With the IBL upgrade 12 million readout channels are added to the Pixel Detector. They are distributed over 14 staves, each carrying 16 readout modules. A more detailed description of the IBL module and the readout system of the IBL will be presented in the following sections.

4.3 The IBL Detector Control System

The IBL Detector Control System (DCS) [36] is a crucial part for the safety of the detector and people working on the detector. The DCS is responsible that the detector is always in good operating conditions to allow proper data taking.

Figure 4.3 shows an overview image of the IBL DCS. It can be split up into different parts. In blue the cooling of the detector is shown. To reduce the noise and radiation damage in the sensors, the detector has to be cooled down to a temperature of around $-25\,^{\circ}\mathrm{C}$. IBL uses an evaporative CO_2 cooling which can remove up to 1.4 kW of heat from the detector. The powering of the detector is shown in dark turquoise. It is divided into the high-voltage (HV), low-voltage (LV), and powering of the optical link (SC-OL). Several monitoring paths allow the supervision of temperatures, humidity, voltages all over the detector. They are also input signals to a hardwired interlock system, shown in red. It is the last line of defense, and can switch off parts of the detector to prevent damage or injury of people.

The DCS control software is programmed in WinCC (formerly known as PVSS) developed by Siemens. It is divided into a front-end panel and a back-end server system connecting the sensors and actuators. The detector is controlled by a Finite-State-Machine (FSM). It handles all the shifter commands, like turning on the detector. Also it supervises the detector with a software interlock system.

4.4 Overview of the IBL readout system

The IBL readout system is based on the Run-1 Pixel readout system, which is described in detail in [37]. The readout of the detector is split into two parts. The readout electronics located inside the detector volume belong to the “on-detector” readout system. The readout components in the counting rooms are part of the “off-detector” readout system. The off-detector readout system is based on the Versa Module Eurocard (VME) interface. Both systems are connected to each other with about 80 m of optical fibres.

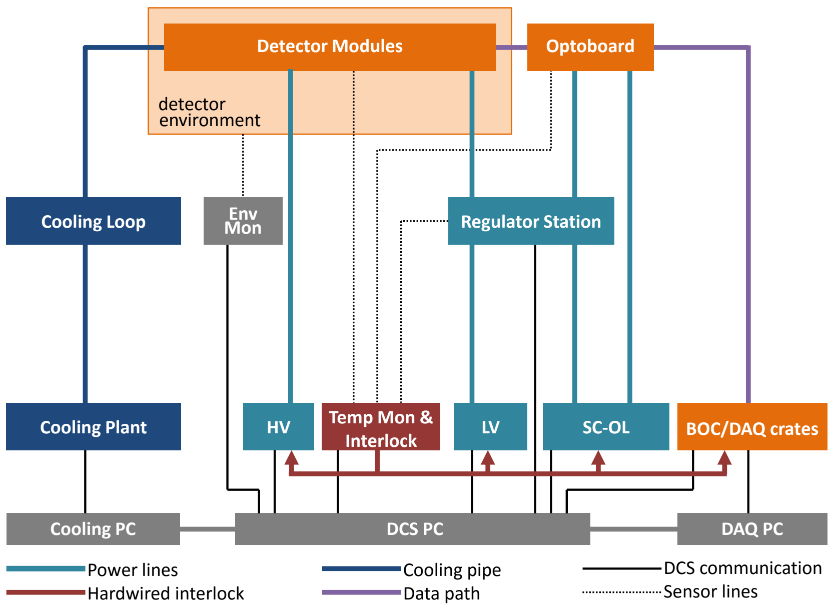


Figure 4.3: Overview of the ATLAS IBL Detector Control System. It is divided into four major parts: cooling, powering, monitoring and the interlock system [36].

Figure 4.4 shows an overview of the full IBL readout system. Central part are the 14 IBL staves with the corresponding VME readout cards. On the right side the software infrastructure for the detector readout and calibration is presented. The left side of the overview the interface to the global ATLAS TDAQ system is depicted. In the following two sections all important components of the readout system will be described shortly.

4.5 On-detector readout components

4.5.1 Pixel sensors and the FE-I4

The most important part of the on-detector readout system are the front-end electronics together with the pixel sensors. IBL uses two different types of silicon sensors: planar in the central region and 3D sensors in the forward regions at the end of the staves. Figure 4.5 shows a sectional drawing of both sensor types. The planar sensors have their electrodes implanted on the top and bottom side, whereas the 3D sensors have electrodes implanted into the silicon. This reduces the collection distance and decouples it from the active thickness of the sensor. As a result, 3D sensors are more radiation hard than planar sensors [38]. Both sensors have a pixel size of $250\,\mu\text{m} \times 50\,\mu\text{m}$. The planar sensor has a total number of 53760 pixels, the 3D sensor has only 26880 pixels. Both sensors are bump-bonded to the front-end electronics.

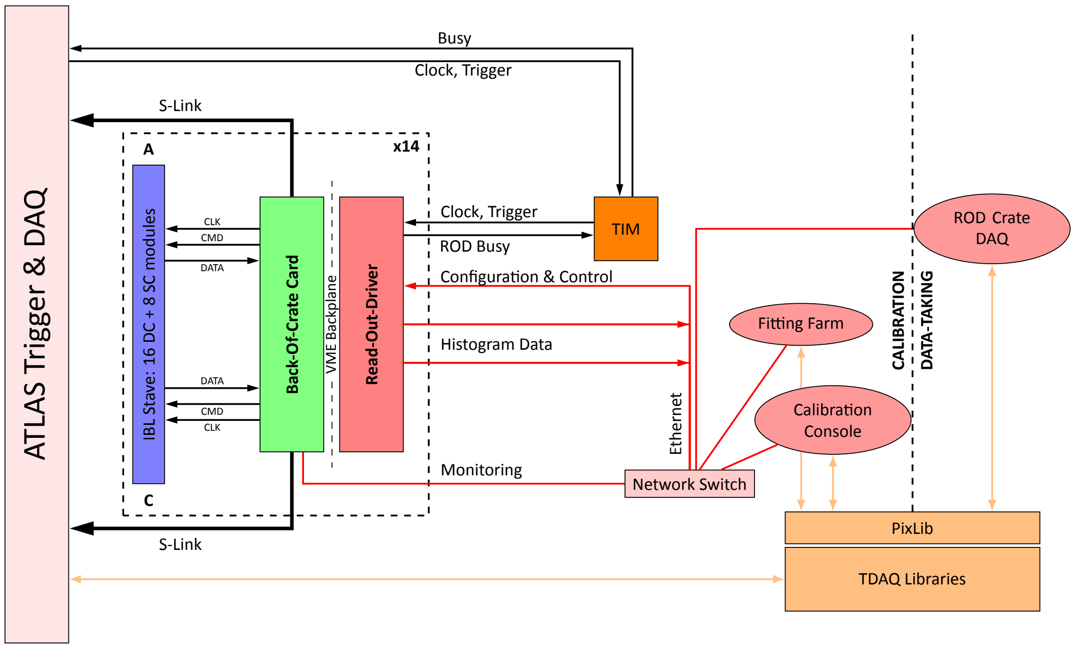


Figure 4.4: Overview of the IBL readout system. Central part (dotted box) are the 14 staves with the corresponding ROD/BOC readout cards. Also the connections to TDAQ and the high-level calibration system are shown. Drawing by Karolos Potamianos.

The front-end chip is called “FE-I4” [39] and provides the readout for 26880 pixels. A planar sensor together with two front-end chips forms a double-chip module, whereas a 3D sensor with one front-end chip is a single-chip module. Figure 4.6 shows the distribution of the modules along the stave. From the DAQ point of view two single-chip modules are consolidated, so a DAQ module always consists of two front-end chips. These two front-end chips share a single serial line for TTC information. To distinguish between the two chips, addressed commands are used.

The purpose of the front-end chip is the measurement of the charge which is injected into the pixel if an ionising particle passes through the sensor. In Figure 4.7 the analog part of the front-end electronics for each pixel is shown. The charge is converted into a voltage by a two-stage feedback current amplifier. This voltage is then compared to a per-pixel threshold. Thereby, the duration where the voltage is above the threshold can be measured. This time is the time-over-threshold (ToT) and is an indicator for the amount of injected charge.

Figure 4.8 shows an overview block schematic of the FE-I4. In the upper area the pixel matrix of 40 double columns and 336 rows, giving 26880 pixels, is placed. Four pixels share a common digital logic block which collects and buffers the hits until a LVL1 trigger decision is taken. A LVL1 trigger decision results in a trigger command being sent to the front-end chip via its command inputs. This command starts the collection of hit information (row, column and ToT) for all the hits belonging to the triggered bunch crossing. The hit information is aggregated in the end-of-column logic and then sent to the end-of-chip logic. There it is serialised and transferred it

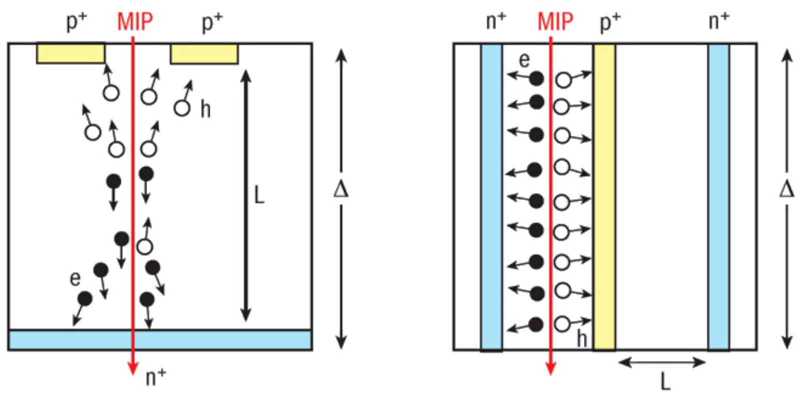


Figure 4.5: Sectional drawing of the two pixel sensor types (planar: left, 3D: right) used in the IBL detector. A minimum ionising particle (MIP) creates electron/hole pairs. The collection distance L is decoupled from the active thickness Δ for the 3D sensor making it more radiation hard [38].

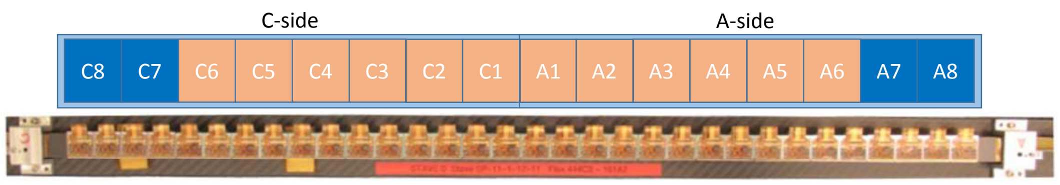


Figure 4.6: Picture of the first prototype IBL stave with placement of the sensor types (blue: 3D, orange: planar) and DAQ module names. The 3D sensors are the outermost two modules of the stave. Stave photography taken from [40].

to the off-detector side. The FE-I4 sens the data at a nominal rate of 160 Mbit/s using the 8b10b encoding, which will be described in Section 5.2.2.

The communication protocol for commands to the front-end chip is shown in Table 4.2. It defines a number of serial commands which are split into several fields. In general, the front-end chip distinguishes two types of commands. Fast commands, like triggers, can only be issued in data taking mode. In contrast slow commands, like register reads or writes, can only be executed in configuration mode. To address multiple chips sharing the same serial command line, a four bit ChipId is added to all slow commands. If the most significant bit in the ChipId is set (ChipId ≥ 8), it will be treated as a broadcast command to all front-end chips on the serial command line.

4.5.2 Optoboard

The optoboard [41, 42] is the on-detector interface between the front-end chips and the optical fibres to/from the off-detector area. It is a bi-directional opto-electrical converter. Figure 4.9 shows an image of the optoboard for the IBL detector. It has 8 input channels and 16 output channels. The optical fibres are directly connected to the BOC card.

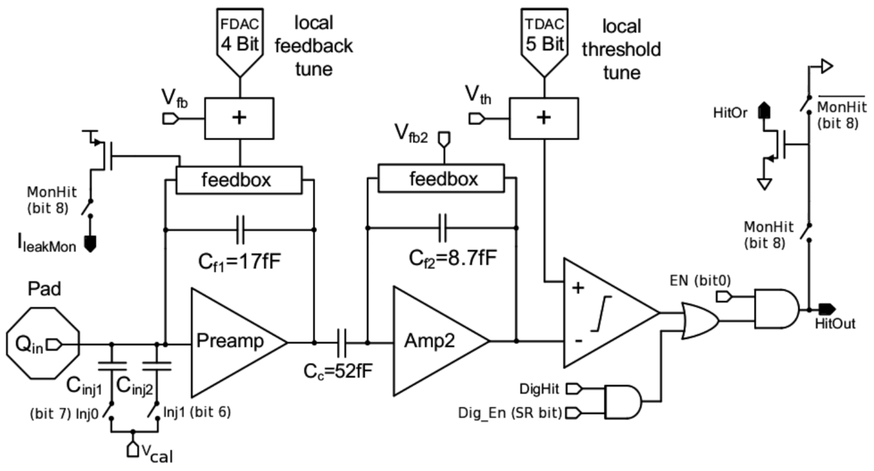


Figure 4.7: Overview of the analog part of a pixel cell in the FE-I4 front-end electronics. The injected charge is amplified in a two-stage feedback current amplifier and then compared to a local threshold [39].

Table 4.2: List of all FE-I4 serial commands

Name	**Type**	**Field 1**	**Field 2**	**Field 3**	**Field 4**	**Field 5**	**Field 6**
size (bits):		5	4	4	4	6	
LVL1	fast	11101	-	-	-	-	-
BCR	fast	10110	0001	-	-	-	-
ECR	fast	10110	0010	-	-	-	-
CAL	fast	10110	0100	-	-	-	-
RdRegister	slow	10110	1000	0001	ChipId	Address	-
WrRegister	slow	10110	1000	0010	ChipId	Address	Data
WrFrontEnd	slow	10110	1000	0100	ChipId	xxxxxx	Data
GlobalReset	slow	10110	1000	1000	ChipId	-	-
GlobalPulse	slow	10110	1000	1001	ChipId	Width	-
RunMode	slow	10110	1000	1010	ChipId	sssccc	-

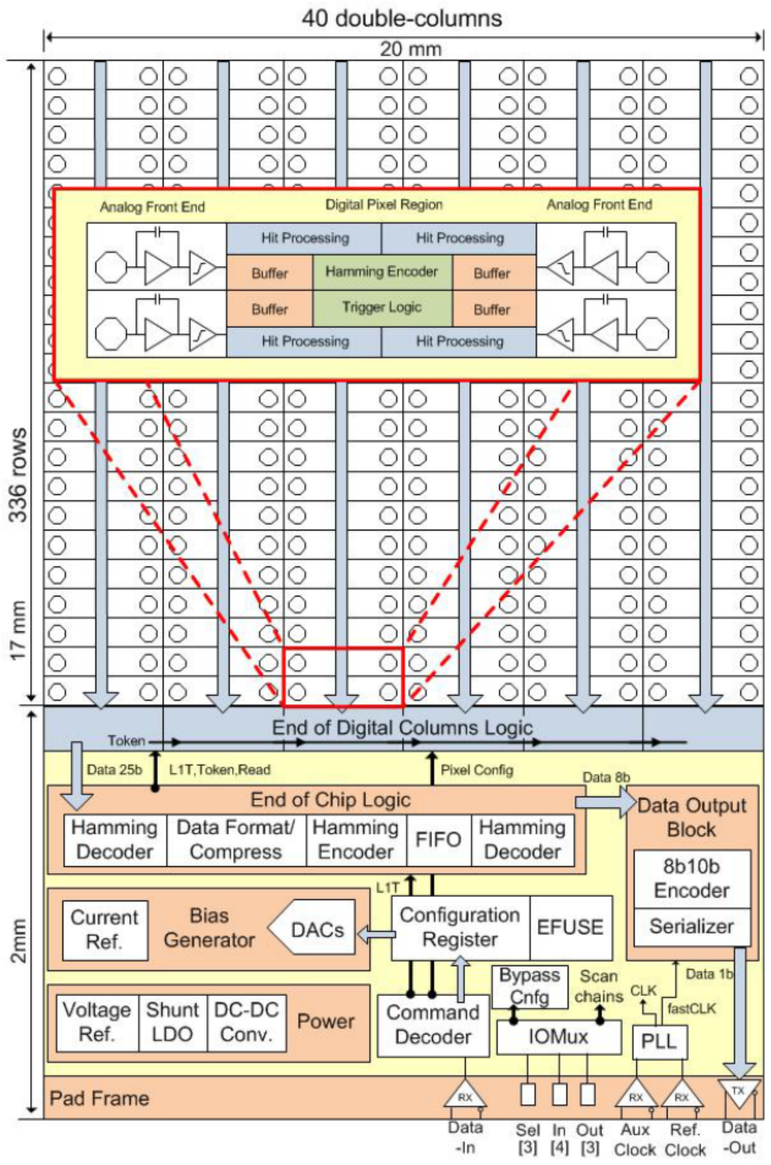
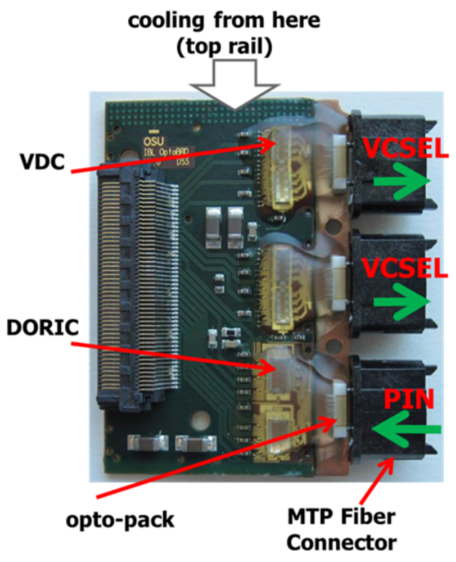
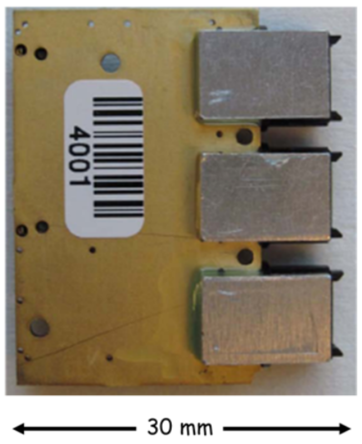


Figure 4.8: Block schematic of the FE-I4 front-end chip. The upper part is the pixel matrix with 26880 pixels in 40 double-columns and 336 rows. The enlarged part shows a four pixel digital region. In the lower part the powering and the command and data processing is shown [39].



(a) top view



(b) bottom view

Figure 4.9: Picture of the IBL optoboard. It is equipped with two 8-channel VCSEL arrays and an 8-channel PiN array [42].

In the downlink direction to the detector, the optoboard carries an array of 8 PiN diodes for receiving the optical signals. These signals are TTC signals and they transmit a clock signal and a serial data stream in a biphase-mark (BPM) encoded signal. The BPM encoding will be described later in Chapter 5. The PiN diodes convert the incoming light into a current. This current is measured by the Digital Optical Receiver IC (DORIC). It amplifies the signal and decodes the BPM signals into clock and data signals.

The uplink direction from the detector to off-detector electronics is using a vertical-cavity surface-emitting laser (VCSEL) array which is driven by the VCSEL driver chip (VDC). The serial signal is generated directly in the front-end chips. The VDC allows one to control the laser current on a per-chip basis using an external analog voltage input (VIset). This voltage can be controlled through the DCS.

4.6 Off-detector readout components

4.6.1 Timing-Trigger-Control Interface Module

The Timing-Trigger-Control Interface Module (TIM) is the central timing and trigger stage in the off-detector readout system. The TIM is a VME card and receives the LHC bunch crossing clock and the trigger signals from the ATLAS central trigger processors. Afterwards, it distributes the clock and trigger signals to all off-detector cards in the VME crate. Also it is responsible for the propagation of the busy signals from the RODs to the central trigger processors. If the buffers in one of the RODs get full, it asserts the busy signal which stops the trigger generation.

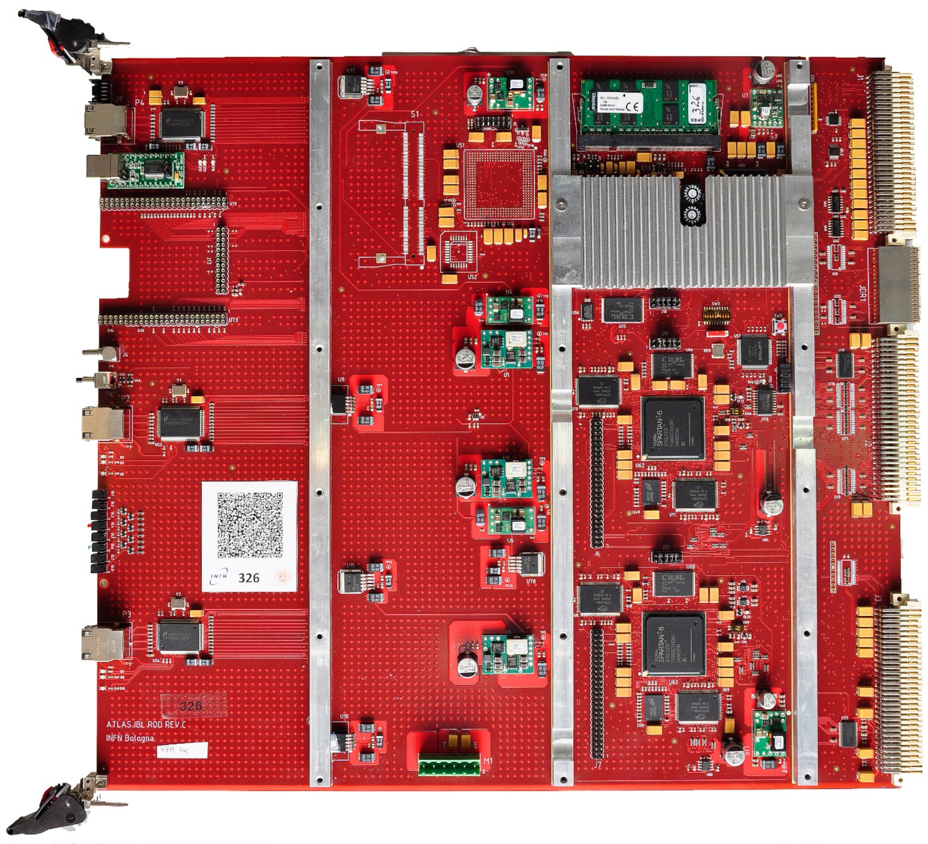


Figure 4.10: Picture of the ATLAS IBL ROD.

4.6.2 Back-of-Crate card

The Back-of-Crate (BOC) card is the optical interface card for the off-detector side. It is directly connected to the detector. The main task of the BOC card is the signal encoding and decoding for the detector interfaces. As it is the central part of this thesis it will be described in the following chapters.

4.6.3 Readout-Driver

The Readout-Driver (ROD) is a major element in the Pixel Detector readout and the ATLAS TDAQ system, which has been presented in Section 3.2.4. A photo of the ROD is shown in Figure 4.10. It has two main purposes. During data taking the ROD receives the trigger information from the TIM and distributes it to the detector modules. After sending the trigger command it waits for the hit information to arrive and combines it into an event fragment. The fragment is sent out via the BOC card to the higher-level readout systems.

Another important use case is the calibration of the detector. In the front-end electronics several parameters can be tuned, like amplification values, thresholds etc. During calibration, a procedure called "scan" is executed. A scan tries to find the best values for these parameters for each pixel of the detector. As shown in Figure 4.11, a scan procedure is defined by a number of nested loops. During the scan

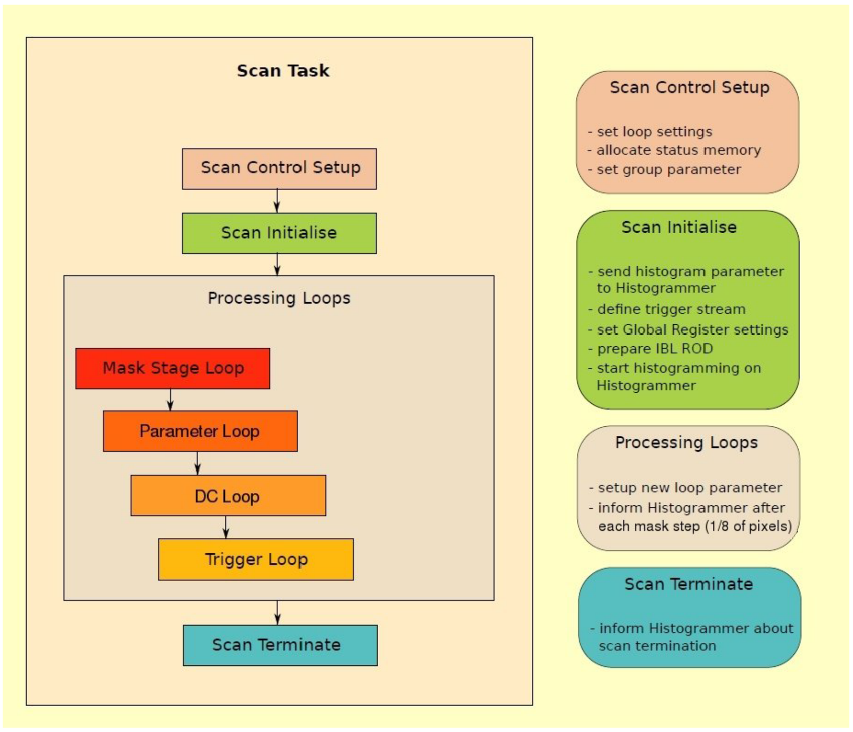


Figure 4.11: Overview of the IBL scan procedure running on the ROD. Modified from [43]. The scan is executed as a series of nested loops over all pixels.

loops, the hit occupancy as well as the ToT values for each pixel are histogrammed. These histograms are shipped out of the ROD via Ethernet to server PCs, which do further histogram processing. Processing results are fed back into the calibration process forming the tuning loop.

Each ROD is equipped with four FPGAs. One FPGA controls the VME interface and is responsible for programming and reset control of the whole card. It is called the Program-Reset-Manager. The main control of the card is provided by the Master FPGA. It includes a PowerPC microprocessor core running at 400 MHz which executes parts of the low-level scan and configuration engine. The Master FPGA is also responsible for receiving the trigger information from the TIM and sending out trigger commands to the front-ends. The main data processing is performed by the two Slave FPGAs. They receive the hit information from the BOC card and build the event fragments to be sent out to the higher-level readout. During calibration they form histograms of the data. Therefore, they are equipped with 4 MB of low-latency ZBT-SRAM[1] for fast read-modify-write operations. The Master and both Slave FPGAs have a 1 Gbit/s Ethernet connection to the local network for histogram transfer to the server and card control.

Every ROD is paired with a BOC card for interfacing the optical connections. Each card pair is connected to two half-staves, the A- and the C-side of one stave. In total 14 cards for the readout of the IBL are needed. One additional card pair

[1]Zero-Bus-Turnaround Static Random-Access-Memory

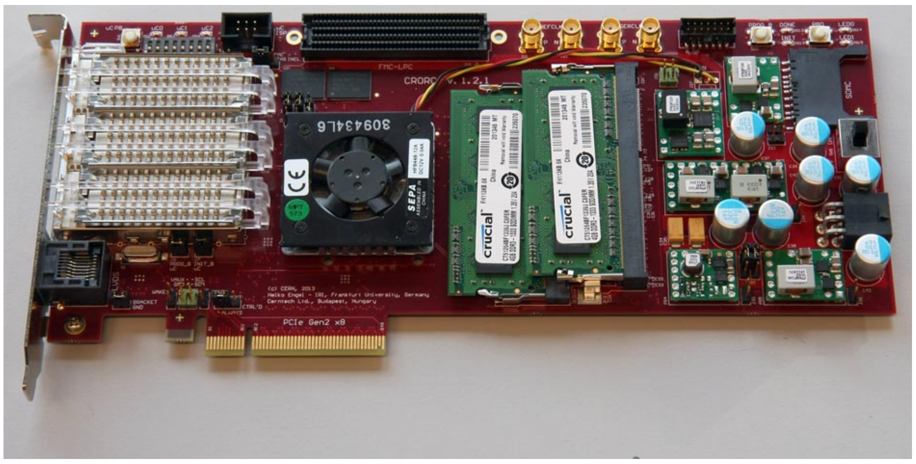


Figure 4.12: Picture of the latest ROBIN card receiving the detector data in the ROS-PCs

is connected to the ATLAS Diamond Beam Monitor (DBM) [44], which has been installed together with the IBL and uses the same type of front-end electronics.

4.6.4 Readout-Subsystem

The Readout-Subsystem (ROS) is a buffering stage in the TDAQ readout scheme. It stores the incoming events from the ROD until an LVL2 trigger decision has been made. The ROS consists of a number of server PCs equipped with the Readout-Buffer-Input (ROBIN) card. Figure 4.12 shows a picture of latest version of the ROBIN card. It is a Peripheral Component Interconnect Express (PCIe) FPGA card with 12 optical input channels. Four input channels are needed for the readout of an IBL stave, so in total 56 input links are required. These links use CERN's S-LINK protocol [45] at a data rate of 2 Gbit/s.

When an LVL2 trigger decision has been made, the ROS-PC gets the corresponding event number of the fragment and then picks the right event data from its internal memory. Afterwards, the event is shipped out via two 10 Gbit/s Ethernet links to the processing farms.

4.7 Timing of the ATLAS Pixel Detector

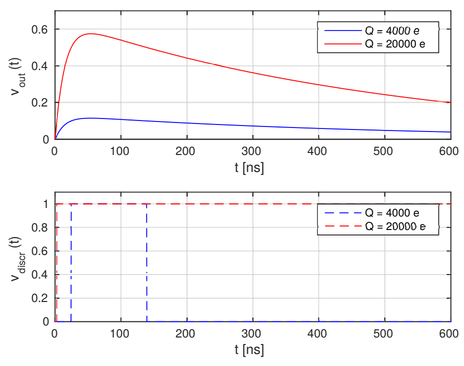
The high bunch crossing rate of the LHC results in harsh timing constraints on the readout of the detector. Having collisions every 25 ns requires also the readout to have a granularity of 25 ns. Therefore, in normal operation only hits inside a 25 ns window are read out. However, in the readout of silicon pixel detectors there is an important effect influencing the readout efficiency. This effect is called "time walk" and to understand it, the pixel sensor and the front-end electronics have to be considered in more detail.

As described in Section 4.5.1, the front-end chip converts the amount of injected charge in the pixel sensor into a ToT value. The response of the charge amplification in the front-end electronics can be approximated by [46]

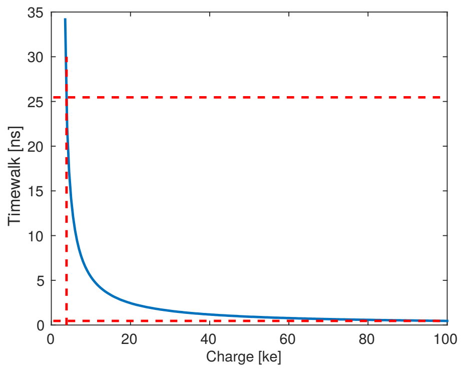
$$v_{out}(t) \approx \frac{Q}{C_f} \cdot \frac{\tau_f}{\tau_f - \tau_r} \left(e^{-\frac{t}{\tau_f}} - e^{-\frac{t}{\tau_r}} \right) \tag{4.1}$$

where Q is the injected charge, C_f is the feedback capacitance, τ_r is the rise time and τ_f the feedback time constant. Figure 4.13a shows the response to different amounts of injected charge. The position of the maximum of both pulses is at the same time, but at different voltages [46]. That leads to different points in time for the threshold crossing. Pulses generated by only a small amount of charge cross the threshold later than large-charge pulses. This effect is called “time walk”. If the readout window of the front-end chip is not in the right position, it can happen that small-charge pulses cross the threshold too late. Then the hit will not be registered in the readout window where it originally belongs to, but in the next readout window. During operation the front-end chip is configured only to send out the data of one readout window, so the small-charge hit is lost.

Figure 4.13b shows the time walk as a function of injected charge. The red-dotted horizontal line is the optimal readout window. It catches all the high-charge hits just at the beginning, and also collects low-charge hits at the end of the window. For the front-end chip that means their clock has to be well-aligned with respect to the bunch crossing, to maximize the efficiency of the readout system. The alignment of the timing is also part of the requirements for the BOC card, which will be shown in the next chapter.



(a) Effect of different charge values on the pulse detection



(b) Time walk as function of charge and position of the optimal readout window

Figure 4.13: Visualisation of the time walk effect

4.8 Testing and commissioning steps

The testing of all components is divided into several stages. First, all components are tested during their development and production. These tests are performed by the responsible institutes and are independent from each other for the most part.

After the production of the cards and their individual testing, they are installed together with the detector in the setup at CERN.

When the detector is installed and all services as well as the DCS is ready, the system test can start. Basic communication tests will then be performed. During these tests the communication between all components in the readout system is tested. An important part there is the communication between the detector and the BOC card off-detector, as this test builds the basis for all other tests. These tests run in parallel with the LHC upgrade and commissioning, so no beam in the machine is present at that stage. During this stage the calibration of the detector with the readout system is heavily tested.

After carfully checking the standalone operation of the full IBL readout chain, it is integrated into the global ATLAS readout system. During a number of milestone runs the data taking with the complete ATLAS detector including IBL is tested to a large extent. The first milestone runs are scheduled to a time, when there is no beam in the machine. In this time, the data taking is limited to cosmic data taking. That mean, events from cosmic interactions passing through the ATLAS detector are being recorded. with finishing the LHC commissioning and first stable beams in the machine, the data taking with collission data can start.

4.9 Outlook on the Layer-1 and Layer-2 readout upgrade

As the luminosity of the LHC machine is developing better than expected, also the link occupancy of the readout links for the Pixel Detector increased. Table 4.3 shows the estimated occupancy at a trigger rate of 100 kHz. As the link occupancy is getting close to 100 % or even exceeds it, it was decided to upgrade parts of the Pixel Detector readout electronics for Layer-1 and Layer-2. During the nSQP[2] upgrade [47] new optoboards have been installed together with new fibres to the off-detector side. This allows doubling the data rate from 40 to 80 Mbit/s for Layer-2, and from 80 to 160 Mbit/s for Layer-1. The b-Layer already runs at the maximum module speed, so it cannot be upgraded.

Table 4.3: Estimated link occupancy of the Pixel Detector readout links for a trigger rate of 100 kHz [48].

Bunch spacing	**Pile up**	**Link occupancy [%]**			
		b-Layer	Layer-1	Layer-2	Disks
50 ns	37	51	45	69	40
25 ns	25	47	42	65	37
	51	71	67	88	52
	76	95	97	148	75

The Pixel off-detector electronics can handle a total input data rate of 2.5 Gbit/s and 32 input channels. This data rate can be split up into 32 channels running at

[2]New service quarter panel

40 Mbit/s or 80 Mbit/s or 16 channels at 160 Mbit/s. So for Layer-1 twice the cards as currently installed would be needed. This amount of card pairs is not available and a new production cannot be started due to obsolete parts. So it was decided to exchange the old hardware with the new cards that have been developed for the IBL upgrade. They can handle the incoming data rates easily. Currently, the adaption of the firmware for the Layer-1 and Layer-2 cards is being prepared [49].

5 Requirements for the ATLAS IBL BOC card

The Back-of-Crate (BOC) card is an important part in the optical data transmission path between the detector and the counting-rooms. It serves as the optical interface card for the ROD in the off-detector readout system.

The name “Back-of-Crate card” has been chosen, because it is located in the rear side of the VME readout crate. The main reasons for that are the optical interfaces. They are driven by lasers and the rear side of the crate can be covered with a door to ensure laser safety. The door is connected to the laser interlock system which is switching off the lasers if the door is opened.

This chapter will depict the requirements for the BOC card. Some of these requirements are motivated by card interfaces, others have their origin in the efficiency of physics data taking.

5.1 Mechanical requirements

Due to the readout being a VME-based readout system the BOC card has to fulfil several mechanical constraints. These constraints are described in the VMEbus Specification Manual [50].

The Pixel Detector uses a 9U-VME crate. It extends the height of a standard double height crate by an additional VME connector at the bottom of the crate. This VME connector is placed on a custom backplane and carries detector specific signals. The design of the custom backplane has been adopted from the Pixel Detector VME crates. So the width of the PCB has to be 368 mm. The depth of the VME crate is 400 mm for the front side and 220 mm for the back side. This means the depth of the PCB must be 220 mm as well.

By specification, the thickness of the PCB has to be 1.6 mm. For the BOC card it will be increased to around 2.2 mm to provide more mechanical stiffness. This will prevent the PCB from bending during removal or insertion of a card. To fit the guide rails a rabbet has to be milled at the left and right edge of the board. The total height of the components mounted on the top side of the card has to be less than 13.71 mm to not collide with a card in the neighbouring slot.

5.2 Detector interfaces

The requirements for the detector interfaces are mainly driven by the on-detector electronics. First, the TTC link to the detector will be described. Afterwards, the link from the detector will be depicted.

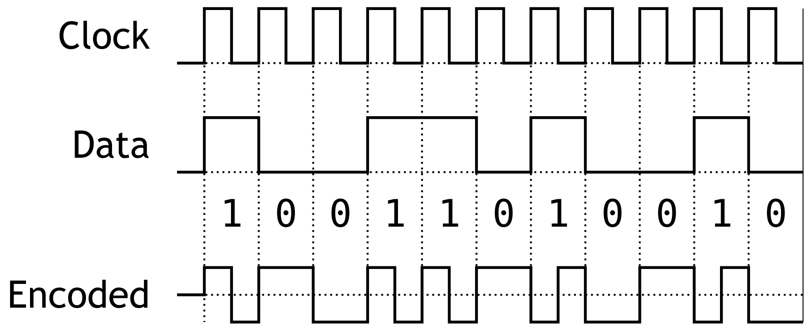


Figure 5.1: Example waveform of the BPM encoding

5.2.1 TTC path to the detector

The TTC path to the detector has the task to distribute the LHC clock signal together with a serial trigger and configuration signal from the ROD to the detector modules. IBL is using the same TTC distribution as the Pixel Detector. So all concepts of the Pixel Detector were adopted by IBL.

The TTC information is received by the ROD and forwarded to the BOC card using separate serial data lines on the VME backplane. Each detector module has its own TTC serial line. The serial lines run a data rate of 40 Mbit/s synchronous to the LHC bunch crossing clock.

To send the serial TTC information together with the LHC clock signal on a single transmission line, the biphase-mark (BPM) encoding is used. Figure 5.1 shows the BPM encoding principle of a serial data stream. The BPM encoding is comparable to the Manchester encoding [51]. The output signal is alternating at the beginning of each bit on a rising edge of the clock signal. It also has a transition in the middle of the bit at the falling edge of the clock signal if the transmitted data bit is '1'. This results in a 20 MHz pulse to be generated, when a '0' is transmitted, and a 40 MHz pulse otherwise. The clock and data recovery of this signal is performed in the DORIC on the optoboard (see 4.5.2). The duty-cycle required by the optoboard should always be in the operating range of the optoboard between 46 and 54 % [41].

The timing of the TTC data path is an essential parameter of the BOC card, as it affects the readout efficiency. Due to the time walk effect, low-energy hits can migrate into the following bunch crossing. This effect was already described in Section 4.7. To maximize the readout efficiency, the readout window has to be moved to the very beginning of the high-energy hits. The size of the readout window for IBL is 25 ns. On the Pixel BOC card, the window could be moved in steps of 280 ps [37]. The lower-limit for this adjustment is the jitter of the LHC bunch crossing clock which is in the range of some ten picoseconds [52]. With new technologies it is planned to decrease the step size of the readout window adjustment by nearly a factor of 3. Therefore, the IBL BOC card has to provide a fine delay with a step size of 100 ps or less. The total delay to cover is in the range of a few bunch crossing. Also it needs to be ensured that the timing is stable over a long period of data taking. The total latency introduced by the BOC card and the BPM encoding of the serial data should not exceed 150 ns.

The optical interface is serviced by commercial optical SNAP12 [53] plugins. Each card can be equipped with four of these plugins giving 48 output channels in total. Only the inner 8 optical fibres are used, so a total of 32 optical channels is available. 16 channels remain as spares. For IBL only two plugins serving 16 channels need to be mounted. For the Layer-2 readout all four plugins will be used.

The requirements on the optical output power for these plugins are tightly coupled to the optoboard. It has certain requirements on the optical output power of the SNAP12 plugins, which were analysed in detail in [54]. The DORIC chip, which is receiving the TTC signals, requires at least $100\,\mu$A input current coming from the PiN diode. With a conversion factor between the light power and output current of the PiN diode after irradiation of 0.475 A/W this gives a minimal optical input power at the optoboard level of $210\,\mu\text{W} = -6.77\,\text{dBm}$. Together with the attenuation of fibres and connectors of about 1.34 dB, and a safety margin of 3 dB this gives a minimum output power of -2.43 dBm or $570\,\mu$W. It needs to be ensured by the optical plugin that its output power does not fall below this value even after a long time of operation.

5.2.2 Data path from the detector

The data path from the detector carries all the hit information from the detector front-end electronics to the off-detector hardware. Compared to the readout of the Pixel Detector, IBL uses a totally redesigned data path.

As well as in the TTC path new commercial optical plugins are used. Four SNAP12 Receiver plugins will be mounted serving 48 input channels. 32 of these channels are used for the readout of the IBL and 16 remain as spares.

The major change was the use of the 8b10b encoding, which was developed by Widmer and Franaszek [55]. It maps incoming 8-bit wide symbols into 10-bit output symbols. Compared to the standard NRZ[1] signals used in the readout of the Pixel Detector, this has many advantages. First, the 8b10b encoded signal is totally DC-balanced and has a bounded disparity. The disparity is the difference between the number of 1's and the number of 0's in a data word. So over a long observation time, the count of 1's and 0's in the signal is equal. This allows the usage of AC-coupled transmission on the data lines and also enables the receiver to recover the sending clock from the data stream. To achieve the DC-balance, the encoded data words are wisely chosen to have either a balanced disparity or a disparity of ± 2. The encoder keeps track of the running disparity, which is always ± 1. Only data words with a disparity of ± 2 can affect the disparity of the total data stream. If the current disparity is -1, the decoder chooses a word with $+2$ as disparity and vice-versa. Using this algorithm the running disparity fluctuates always around 0. Also the number of consecutive 1's or 0's is limited to 5, to allow the clock and data recovery from the signal. The 8b10b encoding is widely used where high-speed serial data transmission is taking place: PCIe, SerialATA, USB 3.0 etc.

Additional to 256 data symbols, the 8b10b encoding allows transfer of 12 special data words, also known as "k-words". Table 5.1 shows an overview of the k-words in the 8b10b encoding. All of them have a Hamming distance of 2 to all other data

[1]Non Return To Zero

Table 5.1: List of all possible k-words in the 8b10b encoding

Input		**Encoded bit sequence**	
	Hex	**disparity = +2**	**disparity = -2**
K.28.0	0x1C	001111 0100	110000 1011
K.28.1	0x3C	001111 1001	110000 0110
K.28.2	0x5C	001111 0101	110000 1010
K.28.3	0x7C	001111 0011	110000 1100
K.28.4	0x9C	001111 0010	110000 1101
K.28.5	0xBC	001111 1010	110000 0101
K.28.6	0xDC	001111 0110	110000 1001
K.28.7	0xFC	001111 1000	110000 0111
K.23.7	0xF7	111010 1000	000101 0111
K.27.7	0xFB	110110 1000	001001 0111
K.29.7	0xFD	101110 1000	010001 0111
K.30.7	0xFB	011110 1000	100001 0111

words. So a bitflip in a data-word cannot result in a k-word. Also these k-words can be used to signal special low-level functions, such as Start-Of-Frame (SOF, K.28.7), End-Of-Frame (EOF, K.28.5), or Idle (K.28.1). These three k-words have a special property that allows one to reconstruct word boundaries inside the serial data-stream and therefore they are calld "comma-words". Their bit sequence cannot occur across word boundaries. Using the comma-words allows reconstruction of the word boundaries.

The data rate which has to be handled for each serial line is 160 Mbit/s. So in total a data rate of $32 \times 160\,\mathrm{Mbit/s} = 5.12\,\mathrm{Gbit/s}$ has to be handled by the BOC card. After the 8b10b decoding the net data rate is 4.096 Gbit/s.

The BOC card is responsible for handling the incoming 8b10b data and checking its integrity. If errors are detected, they need to be reported to the ROD. After decoding the data, the BOC card hands over the decoded data to the ROD. Between the two cards 96 data lines synchronous to an 80 MHz board clock are used. These 96 lines are divided into 8 seperated bus systems, each with 12-bit width. Each bus transfers data of 4 front-end chips. 8 lines of each bus are assigned to the decoded 8b10b data word, 1 line indicates if the data is a k-word or a normal data word, 2 lines address the front-end chip the data belongs to and 1 line indicates the bus state is valid and data is being sent. All together the data bus allows a transfer rate of 5.12 Gbit/s, which is larger than the net data rate of the incoming data, so no back pressure functionality is needed. The bus occupancy will be 80 % at maximum.

5.3 ROS interface

The interface to the higher-level ROS system is using the High-speed Optical Link for ATLAS (HOLA) S-LINK transmission protocol [45]. Every BOC card has to provide four independent S-LINK interfaces to the ROD. Each S-LINK interface consists of a dual-HOLA block shown in Figure 5.2. Therefore, the data will be

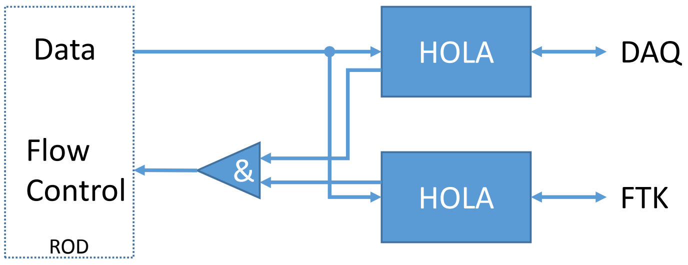


Figure 5.2: S-LINK interface in the dual-HOLA implementation providing access to the IBL DAQ and the ATLAS FTK project

Table 5.2: Signal description of the S-LINK signals between ROD and BOC card

Signal name	Width	Description
UDATA	16	Double-Data-Rate User Data Signals
UWE_N	1	Low active write enable
UCTRL_N	1	Low active control word enable
UTEST_N	1	Low active test mode enable
URESET_N	1	Low active reset
UCLK	1	Interface clock for source-synchronous transmission
LDOWN_N	1	Low active link down indicator
LFF_N	1	Low active flow control indicator

duplicated in the BOC card. One path serves the usual DAQ data path to the ROS, the other path is a copy of the DAQ data and will be provided to the ATLAS Fast TracKer (FTK) project [56].

The S-LINK interface between BOC card and ROD is defined by a number of signals between both cards. Table 5.2 lists all the required interfaces. The interface is running at a clock frequency of 40 MHz thus allowing a total transfer rate of 1.28 Gbit/s using double-data-rate signaling.

The connection to the ROS and FTK readout will be made by using commercial optical Quad Small Form-factor Pluggable (QSFP) transceivers. Each plugin consists of four transmitters and four receivers, so two QSFP plugins per BOC card are needed to serve all outgoing S-LINK connections.

5.4 Clock distribution

Another task of the BOC card is the clock distribution and management for a ROD/BOC card pair. The BOC card receives a 40 MHz clock from the TIM which is synchronous to the LHC bunch crossing clock. This clock is distributed to the front-end electronics as well as to the ROD, so everything in the system is synchronous to this clock. The BOC card is also responsible for adjusting the clock phase of the incoming TIM clock. This first stage delay just needs to shift the clock by one clock cycle (25 ns), to adjust the sampling point of signals coming from the TIM

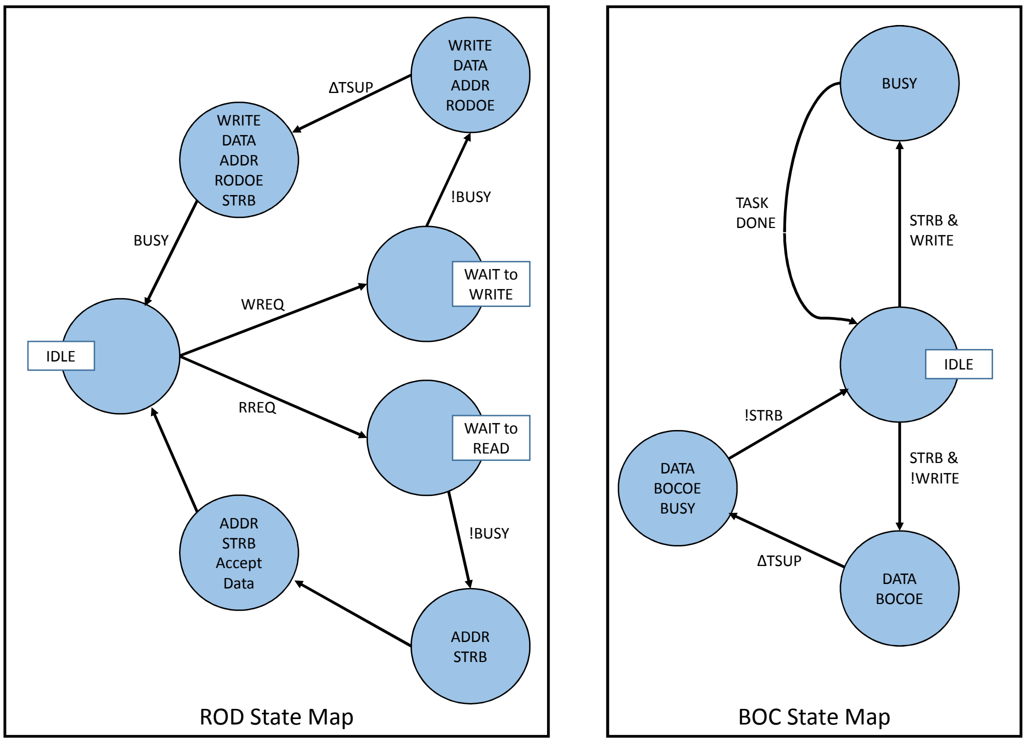


Figure 5.3: Setup-Bus communication state machine [57]

properly. Besides delaying the clock signal, jitter cleaning needs to be performed using a Phase-Locked-Loop (PLL).

5.5 External Interfaces

In addition to the important signals for data taking, the BOC card serves some external control and monitoring interfaces. These interfaces will be described in this section.

5.5.1 Setup-Bus

The Setup-Bus is the main control bus between the BOC card and the ROD. It is an asynchronous 8-bit wide bus providing an address space of 16-bit. So in total 64 kB of memory can be addressed. The three control signals "strobe", "write_enable" and "busy" are used to indicate the bus state. Figure 5.3 shows a state machine with the bus states.

5.5.2 Ethernet

The BOC card needs also to provide Ethernet connectivity to allow control from the internal network. The interface will be copper based standard IEEE 802.3 Gigabit Ethernet.

5.5.3 CAN

The CAN interface is providing information on the card operation to the DCS. Experiences with the old Pixel Detector readout showed that lasers on the Pixel BOC card were dying because of too much humidity and temperature [58]. So it was requested to have a full monitoring of humidity, temperatures, and voltages available to the DCS.

5.6 Summary of the requirements

To summarize the most important requirements the BOC card and its firmware has to meet:

- PCB has to be electrically and mechanically compatible to the VME 9U crate standard.
- TTC signals need to be BPM encoded and delayed before transmitting them to the detector.
- The TTC delay needs to be adjustable up to several bunch crossings with a stepsize of about 100 ps per step.
- Latency introduced in the TTC path should not exceed 150 ns.
- Decoding and forwarding of the 8b10b data received from the front-end electronics.
- Error monitoring and reporting of the data from the detector.
- Forwarding of the decoded data to the ROD at full rate.
- Distribution of the LHC clock signal to the ROD and the front-end electronics.
- Full S-LINK dual-HOLA implementation with QSFP plugins for IBL DAQ and FTK.
- Laser interlock mechanism has to be provided to ensure laser safety.
- Monitoring of voltages, temperatures and humidity should be available to the DCS via CAN.
- Setup-Bus connection to the ROD and Gigabit-Ethernet for card control.

6 Hardware of the ATLAS IBL BOC card

The design of the BOC card for the IBL detector is a result of the requirements that were put on the card. This chapter will describe the final hardware version of the card. It was designed by the Andreas Kugel and Nicolai Schroer [59] from the Institute of Computer Engineering of the University of Heidelberg (ZITI). Figure 6.1 shows a picture of the produced hardware.

6.1 PCB design

The PCB was designed as a 9U-VME card. The PCB consists out of 12 layers where 6 layers are dedicated for signal routing and the other 6 layers for powering. In total the PCB is 2.4 mm thick. On both sides the PCB has been milled down to 1.6 mm to fit the guide rails in the VME crate.

6.2 FPGAs and direct periphery

To offer good flexibility and easy reprogrammability, FPGAs were chosen to handle all the highly parallel input data streams. A processor system would not be able to process this amount of data. Especially, a lot of custom interfaces are not provided by commercial products. As there was already a lot of experience using Xilinx FPGAs, the decision was to use them on the BOC card as well. At the time of design, the Spartan-6 [60] was the state-of-the-art low-cost FPGA from Xilinx.

The largest device of the Spartan-6 family gives a total amount of 184,304 flip-flops and 4.8 MBit of internal block RAM. The normal I/Os of the device can handle data rates of up to 1 Gbit/s, but it also provides Multi-Gigabit-Transceivers (MGTs) with a data rate of up to 3.125 Gbit/s.

On the BOC card, three Spartan-6 FPGAs are mounted. One FPGA is responsible for the controlling of the card and handling the external interfaces (Ethernet, Setup-Bus etc.). This FPGA is referred as the BOC Control FPGA (BCF). The two other FPGAs are siblings and are called BOC Main FPGAs (BMFs). The BMFs control the data paths to the detector and to the higher-level readout. Each FPGA is responsible for the readout of 8 DAQ modules, which is equivalent to 16 front-end chips.

The BCF is a Spartan-6 LX75T FPGA. Being responsible for the control of the card, it is connected to all main control interfaces, such as the Setup-Bus and Ethernet. The two BMFs are Spartan-6 LX150T FPGAs which is the largest device of the family. The two BMFs are referred as BMF north and BMF south, depending

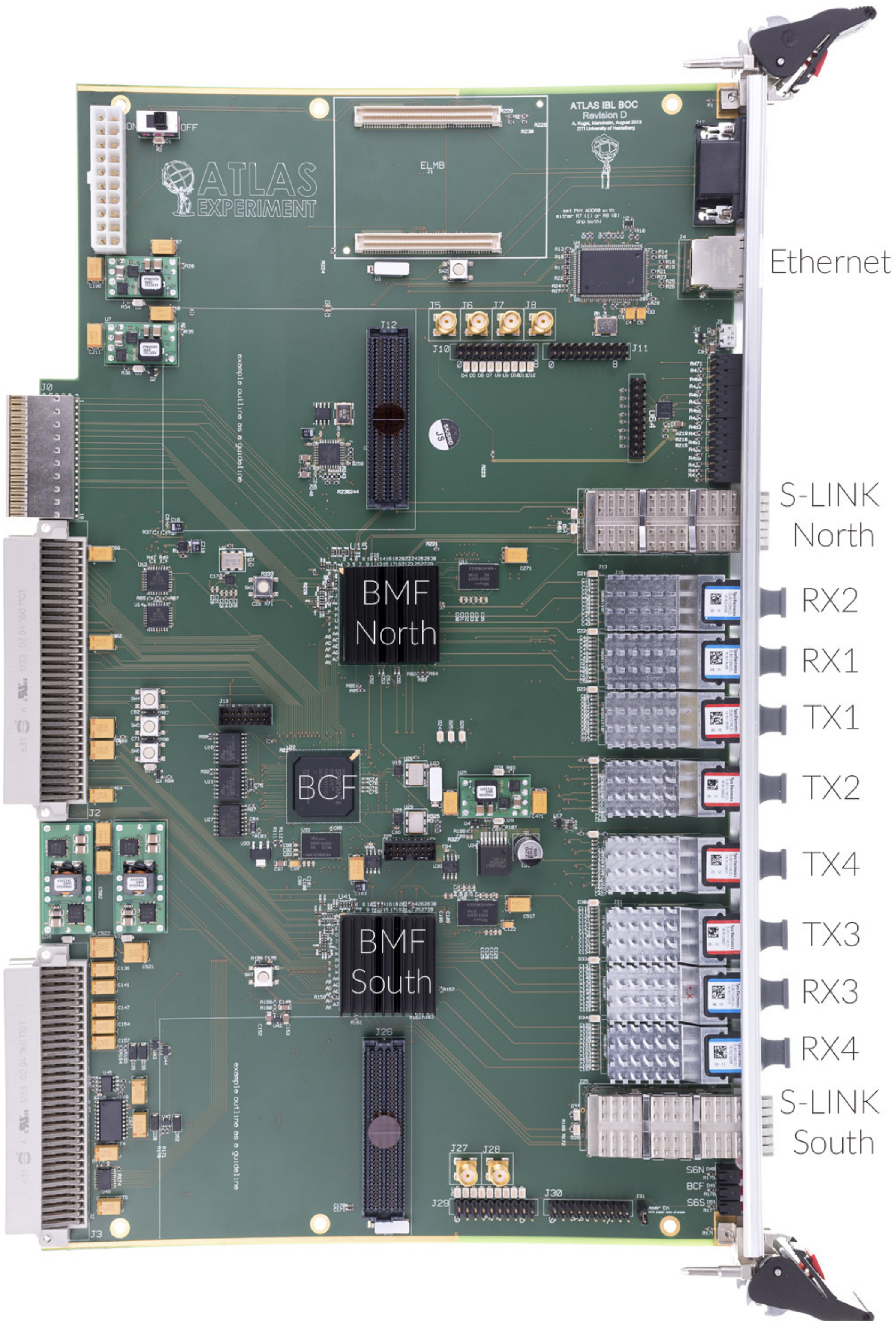


Figure 6.1: Photography of the produced IBL BOC card (revision D). On the right side, all optical interfaces are placed and on the left side the VME backplane is shown. Between both, the FPGAs are mounted. Also an Gigabit-Ethernet connection is provided on the front-panel. [49]

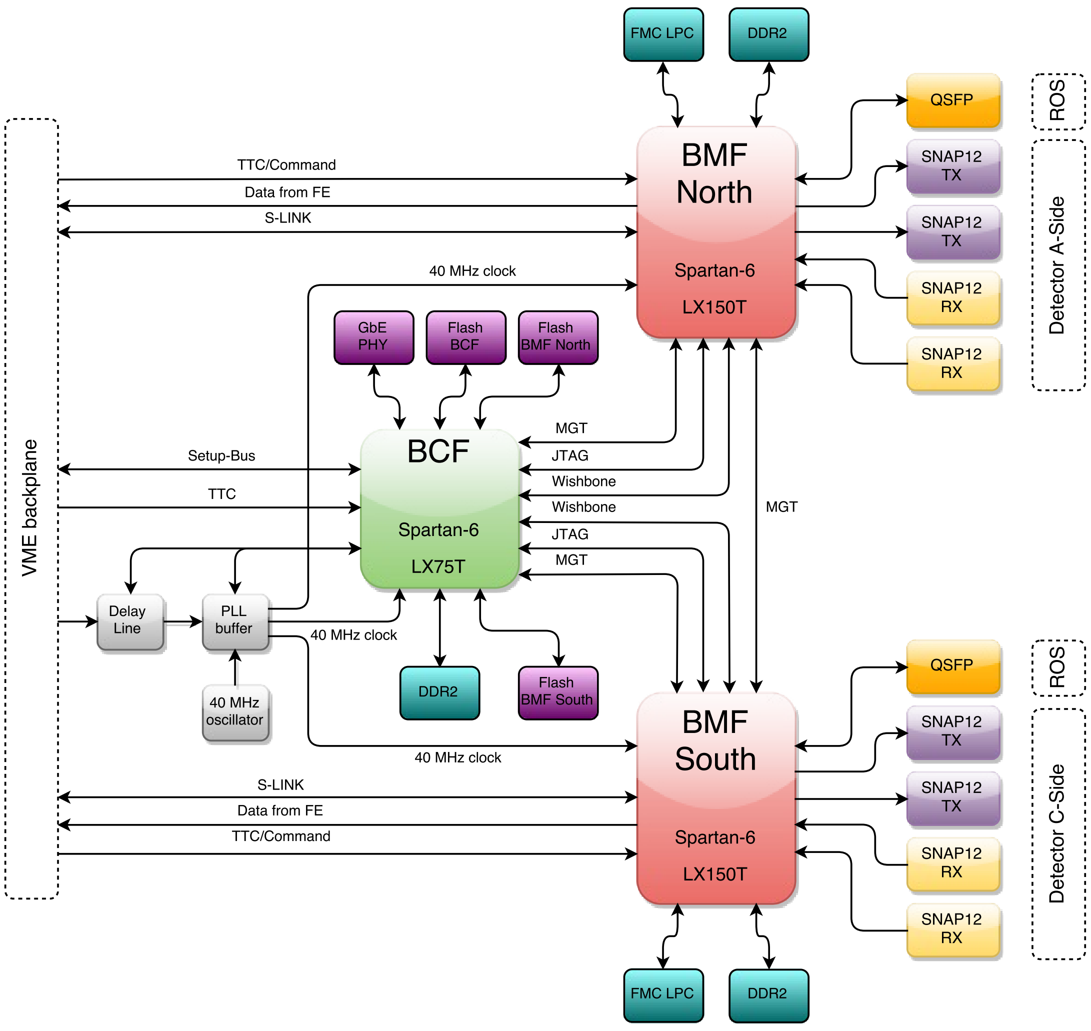


Figure 6.2: Overview of the FPGA connectivity on the IBL BOC card.

on their position of the card. Each BMF has a number of external connections, like the 48 differential pairs between the FPGA and the optical plugins.

Figure 6.2 shows the connectivity of all FPGAs and their peripherals. The two BMFs are symmetric in their connections. Each BMF is connected to the detector via 48 differential signal pairs. They are directly wired to the optical plugins. The QSFP plugins to the ROS is connected to four of the MGT lanes. The signals to the backplane are 8 TTC command signals and 48 data signals carrying the hit information. Optionally each BMF has an expansion port using the FPGA Mezzanine Card (FMC) standard [61]. Both BMFs are connected to the BCF with a 24 line wide bidirectional bus and one MGT lane. Also the Joint Test Action Group (JTAG) interfaces[1] of the BMFs are connected to the BCF. This allows the BCF to program the BMFs. Between both BMFs also an MGT lane for direct data exchange is wired.

[1]de-facto standard for programming and debugging interfaces

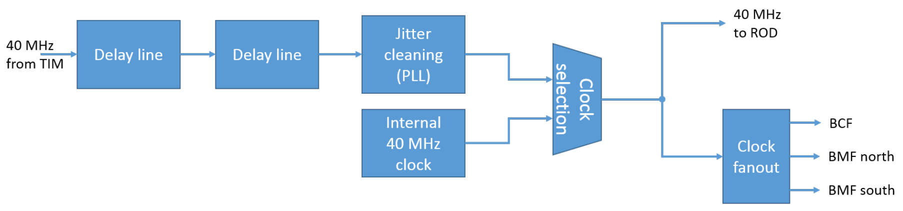


Figure 6.3: Overview of the clock circuit on the IBL BOC card.

All FPGAs have 64 MB of DDR2 SDRAM memory attached, which allows to store data that does not fit into the internal FPGA memory. It uses a 16-bit data bus between memory and FPGA.

6.3 Clock circuit

The clock circuit on the card has an important role. It was designed according to the requirements on the clock distribution. The goal is to distribute the LHC clock to the FPGAs on the card as well as to the ROD.

Figure 6.3 shows an overview of the clock circuit on the card. On the left side the incoming LHC clock from the TIM is shown. It can be shifted by about 20 ns using two cascaded programmable delay lines, which can be configured by the BCF. This allows one to adjust the LHC clock for each card individually. After delaying the clock it will be jitter cleaned and distributed to all the FPGAs and the ROD. If no LHC clock is received from the TIM, for example in standalone operation, an internal 40 MHz oscillator is used. During hardware testing it was found that the first version of the clock circuit used a PLL as jitter cleaner which did not have a fixed deterministic phase between input and output clock. So the clocking circuit needed to be redesigned. Both clock circuits will be described in the next sections.

Additionally to the LHC clocking path, multiple auxiliary clocks are provided. An 100 MHz clock from an on-board oscillator is distributed to all three FPGAs. It can be used for applications which do not need to be synchronous to the LHC clock signal. Additional four dual-frequency oscillators provide the reference clocks for the MGTs which can be either 100 MHz for S-LINK operation or 156.25 MHz for other applications.

6.3.1 First version

The first version of the clock circuit on the BOC card is shown in Figure 6.4. It used a Texas Instrument PLL chip, the CDCE62002 [62], which is a PLL for clock generation and jitter cleaning with two outputs. Beside the reference input clock an auxiliary clock input can be used. One of its output is routed to the backplane and supplies the ROD with the LHC clock, the other output is fed into a 1-to-4 clock fanout supplying the on-board FPGAs.

The PLL is freely configurable from the BCF using a simple three-wire Serial Peripheral Interface (SPI). It allows changing the multiplier and divider values for

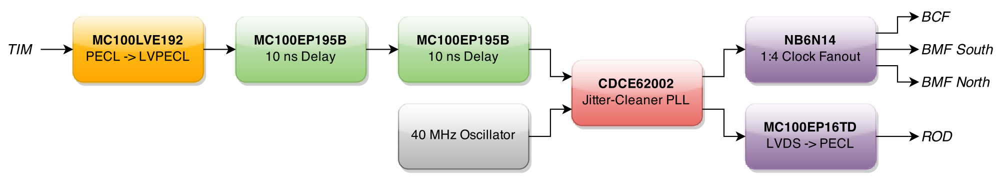


Figure 6.4: First version of the clock circuit as it was used in revisions A, B and C with the CDCE62002 as jitter-cleaner

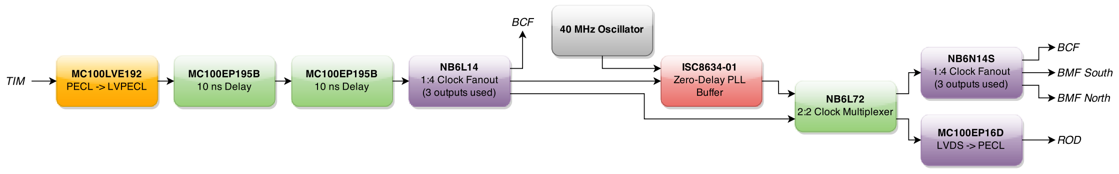


Figure 6.5: Production version of the clock circuit with the ISC8634-01 as zero-delay PLL buffer for jitter cleaning

the PLL and the clock outputs. An initial configuration can be stored in the internal configuration flash memory, which will be automatically loaded during power-up.

During testing of the cards, serious issues with this clock circuit appeared. A major requirement on the clock circuit for the distribution of the LHC clock is that the phase alignment of that clock is predictable. It turned out that the CDCE62002 device does not fulfil this requirement. The phase alignment of the clocks changes randomly after every power-cycle of the board which is unacceptable and resulted in a redesign of the clock circuit.

6.3.2 Production version

In the production version of the card the CDCE62002 has been replaced by a ICS8634-01 which is a zero-delay PLL-buffer. It provides a feedback input, so one of the outputs can be fed back into the PLL and so the delay between input and output is compensated. Additionally, the PLL-buffer can be bypassed to allow direct access to the LHC clock. To allow the measurement of the VME clock, it is also fed directly into the BCF. The new clock circuit is shown in Figure 6.5.

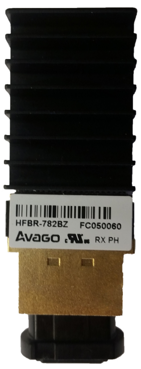
With the new clocking circuit the time difference between input and output clock could be fixed to around 20.5 ns, when using the PLL-buffer, and 19.4 ns, when bypassing it. This behaviour is predictable and the new clock circuit therefore usable. More measurements will be presented in Chapter 8.

6.4 Optical plugins

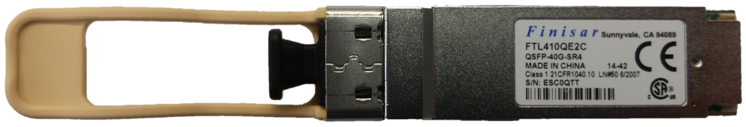
During the production of the PCBs also the optical plugins were evaluated. The SNAP12 plugins are mainly produced by three companies: Avago Technologies, Reflex Photonics Inc. and Tyco Electronics. Especially the output power of the SNAP12 TX plugins is important, as it has to fit to the optoboard requirements and should provide over its full lifetime an output power of at least -2.43 dBm or 570 μW. The vendor which was selected to be used is Avago Technologies and for these plugins the output power has been measured during qualification of the plugins at the University of Bern. The mean light power for all tested plugins is $(846 \pm 92)\,\mu$W [63]. This is above the required 570 μW. Figures 6.6a and 6.6b show a picture of the chosen SNAP12 TX and RX plugins for IBL.

The QSFP plugins, which are used on the BOC card are standard QSFP plugins produced by Finisar. These plugins are widely used in commercial optical Ethernet cabling. Figure 6.6c shows one of the QSFPs, which were used on the IBL BOC card.

(a) SNAP12 TX



(b) SNAP12 RX



(c) QSFP

Figure 6.6: Optical plugins used on the IBL BOC card

Another plugin related circuit on the BOC card is the interlock circuit. As already described in the previous chapter, the BOC card has to disable the lasers during an intervention to achieve complete laser safety. Beside the two door switch signals coming from the DCS system through the VME backplane, the BOC card can generate a third internal interlock signal. The lasers are only activated, when all three interlock signals permit the enabling. If the external interlock signals from DCS are not available (e.g. in a laboratory environment), they can be overwritten with a jumper on the card. The full interlock circuit is shown in Figure 6.7.

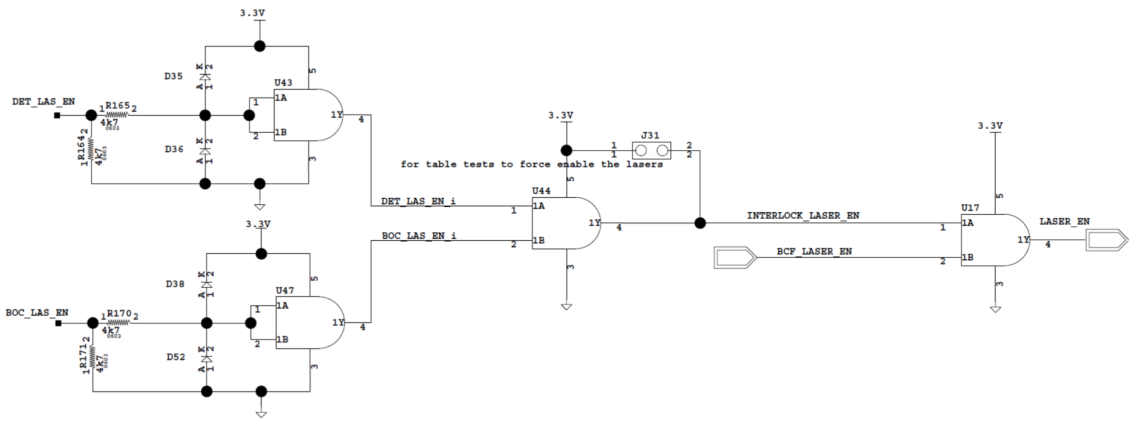


Figure 6.7: Schematics of the interlock circuit on the BOC card. The two backplane signals "DET_LAS_EN" and "BOC_LAS_EN" are ANDed with the internal interlock signal "BCF_LASER_EN". A jumper allows to overwrite the external interlock signals. All signals are active high meaning a high-level is enabling the lasers.

6.5 DCS monitoring

The DCS monitoring of voltages, temperatures and humidity is provided by an ELMB[2] [64], which is widely used in ATLAS DCS. It provides 64 analog inputs and 18 general purpose digital I/O lines. Figure 6.8 shows an image of the ELMB. On the BOC card only the analog inputs are used. The mapping of the analogue channels is shown in Table 6.1.

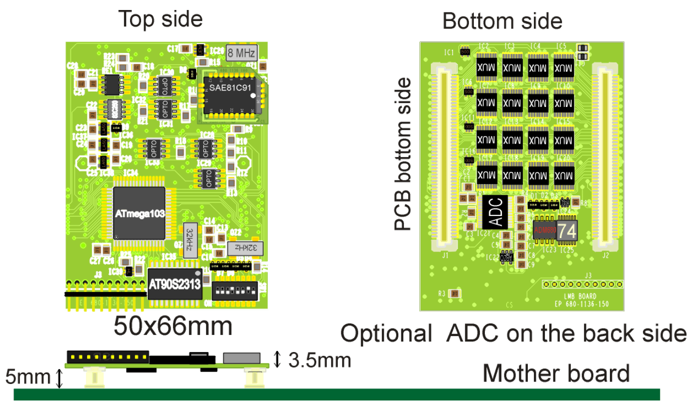


Figure 6.8: Picture of the Embedded Local Monitoring Board for all DCS-relevant measurements [65].

[2]Embedded Local Monitoring Board

Table 6.1: Mapping of the ELMB channels on the IBL BOC card

Channel	Type	Location
0	NTC 10k	SNAP12 RX1 North
1	NTC 10k	SNAP12 RX2 North
2	NTC 10k	SNAP12 RX1 South
3	NTC 10k	SNAP12 RX2 South
4	NTC 10k	SNAP12 TX1 North
5	NTC 10k	SNAP12 TX2 North
6	NTC 10k	top edge
7	NTC 10k	SNAP12 TX1 South
8	NTC 10k	SNAP12 TX2 South
9	NTC 10k	below ELMB
10	NTC 10k	BMF North
11	NTC 10k	BCF
12	voltage	V_{core} Spartan-6 (1.2 V)
13	voltage	V_{MGT} (1.2 V)
14	voltage	VCC Ethernet chip & FMC (2.5 V)
15	voltage	VCC DDR2 (1.8 V)
16	NTC 10k	Ethernet chip
17	NTC 10k	QSFP North
18	NTC 10k	Ethernet chip
19	NTC 10k	QSFP South
20	NTC 10k	BMF South
21	voltage	Ethernet chip (1.8 V)
22	NTC 10k	Voltage regulator
23	NTC 10k	Voltage regulator
24	NTC 10k	Voltage regulator
25	NTC 10k	Voltage regulator
26	NTC 10k	Voltage regulator
27	NTC 10k	Voltage regulator
28	voltage	VME 5 V (divided by 2)
29	voltage	VME 3.3 V (divided by 2)
30	Humidity	middle
31	Humidity	top
32	Humidity	bottom

7 Firmware development for the ATLAS IBL BOC card

7.1 Introduction & Overview

In the previous chapter the hardware of the IBL BOC card was introduced. This chapter, as a central part of this thesis, will focus on the firmware development for the FPGAs. The firmware is split into two main parts. Most of the control logic is handled in the BCF firmware, whereas the signal processing is mainly done in the BMF firmware. The two BMFs on the card share the same firmware files, except they have a different pin assignment.

Most of the firmware is written in VHDL[1]. Some firmware modules in the S-LINK block are written in Verilog, because they were taken from other developments. For some FPGA-specific modules, like First-In First-Out (FIFO) buffers, embedded processors, or the MGTs, Xilinx IP[2]-Cores have been used.

7.2 Design flows

7.2.1 Simulation

During the firmware development, the functionality for most of the firmware blocks has been verified with simulations. All simulations are based on VHDL test benches, whereas some generate waveforms to be reviewed manually, others having an automatic testing against test cases implemented. These automatic tests can run after any modification of the firmware to ensure that all the firmware blocks still work properly.

The simulation environment is based on two HDL[3] simulators. Simple designs without Xilinx IP-Cores were simulated using GHDL [66]. GHDL is a pure VHDL simulator, which is based on the GCC[4] compiler. Waveforms can be written to disk and viewed with a waveform viewer. For designs including Xilinx IP-Cores, Mentor Graphics Questasim has been used for simulation and as waveform viewer.

[1]Very High-Speed Integrated Circuit Hardware Description Language
[2]Intellectual Property
[3]Hardware Description Language
[4]GNU compiler collection

7.2.2 Synthesis

The synthesis design flow follows the standard FPGA design flow from Xilinx' ISE[5] tools. For the synthesis Xilinx ISE in version 14.7 has been used. In general, the flow can be split into these steps:

1. Generation of Xilinx IP-Core netlists
2. Synthesis of the VHDL and Verilog code to netlist
3. Assembly of all netlists into a global netlist
4. Annotation of the pin and timing constraints
5. Mapping of the netlist to FPGA design elements
6. Placement of the design elements and routing
7. Generation of the FPGA bit file

As a result of the synthesis flow, a bit file is generated. This bit file contains the configuration of all the hardware elements inside the FPGA. For the BCF it has typically a size of about 2.5 MB and for the BMF it has a size of about 4.2 MB.

During the firmware development, a command line build flow based on the GNU Make utility has been created. This has several advantages over running the synthesis from ISE's graphical user interface, as it allows the project management in pure text files. This is is important when using version control systems. Also it allows the usage of automatic build systems, like a nightly build script, which synthesizes new firmware versions during the night.

7.3 BCF firmware

As stated in the overview, the BCF firmware is mainly responsible for controlling the card operation, i.e. managing all the external control interfaces. An overview of the firmware is shown in Figure 7.1. All important firmware blocks will be described in the following sections.

7.3.1 Global system interconnect and register bank

A large number of configuration and status registers are distributed in the FPGAs. They need to be accessed from either the ROD or from internal logic. For this task a register interconnect and a register bank is needed, which will be described in this section.

Because it is widely used at CERN in the Open Hardware Repository (OHWR) [67], and because of its simplicity, the Wishbone interconnect [68] bus was chosen. To be compatible with the Setup-Bus interface between the BOC card and the ROD, the Wishbone interconnect uses 16-bit address width and 8-bit data width, allowing a total of 64 kB to be addressed. Additional to the data and address signals

[5]Integrated Synthesis Environment

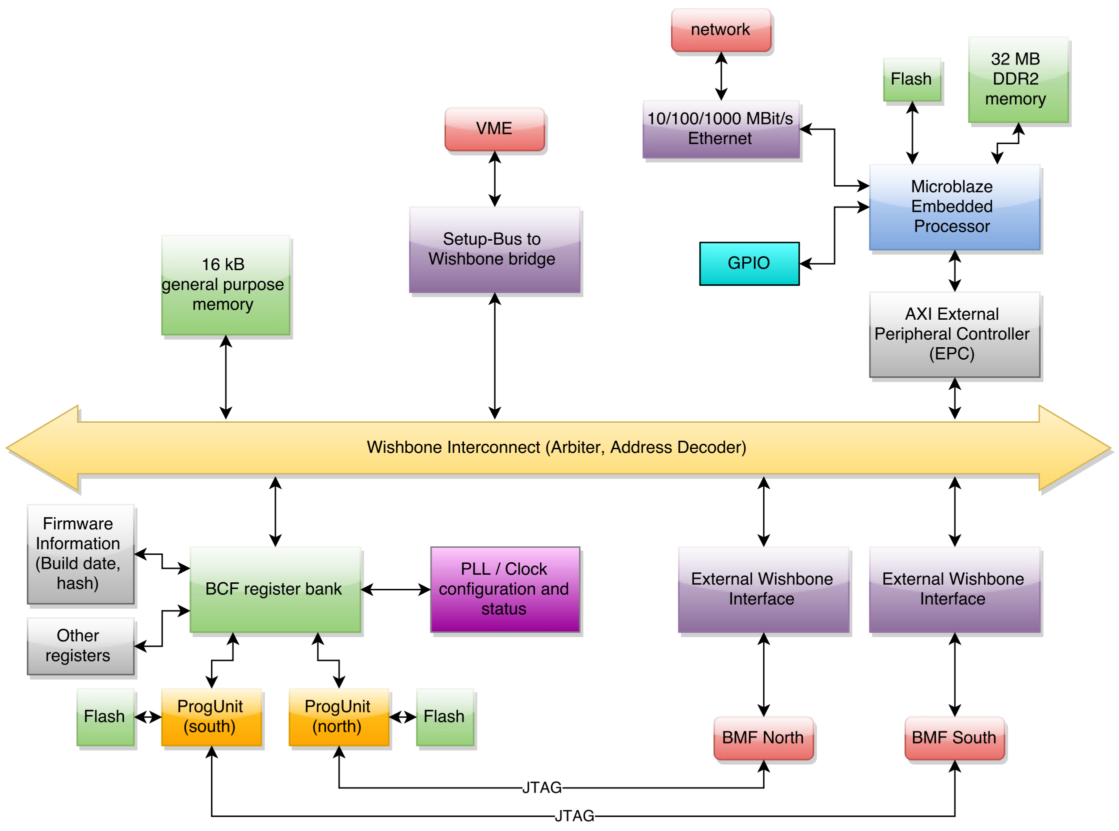

network
10/100/1000 MBit/s Ethernet
Flash
32 MB DDR2 memory
VME
Microblaze Embedded Processor
Setup-Bus to Wishbone bridge
16 kB general purpose memory
GPIO
AXI External Peripheral Controller (EPC)
Wishbone Interconnect (Arbiter, Address Decoder)
Firmware Information (Build date, hash)
BCF register bank
PLL / Clock configuration and status
External Wishbone Interface
External Wishbone Interface
Other registers
Flash
ProgUnit (south)
ProgUnit (north)
Flash
BMF North
BMF South
JTAG
JTAG

Figure 7.1: Overview block schematic of the BCF firmware. Central part is the Wishbone interconnect to which several master and slave modules are connected.

the Wishbone interface uses four signals: “cycle”, “strobe”, “write enable” and “acknowledge”. A master starts a bus cycle by pulling the “cycle” and “strobe” signal high. The value of the “write enable” signal determines if the bus cycle is a read or a write cycle. The slave finishes the bus cycle by acknowledging the transfer. When all control signals are back in their idle state the bus cycle is finished.

Usually, the Wishbone bus is used only internally in the FPGAs, but for the BOC card is was also used for connecting registers of the BMFs. Due to limited lines between both FPGAs only 12 of the 16 address bits are forwarded to the BMFs. This allows each BMF to have 4 kB of memory to be addressed. For asynchronous operation the original Wishbone bus had to be modified and does not allow burst or pipelined transfers. Only single read or write operations are possible.

To distinguish bus accesses between multiple masters and slaves, a simple arbiter and address decoding module has been designed. The arbiter has to ensure that concurrent accesses from multiple masters are handled correctly. Bus interactions must not interfere with each other. Therefore, a state machine handling the access to the bus has been created. Figure 7.2 shows arbiter finite state machine. If a master starts a bus cycle by pulling its cycle signal high, the arbiter detects this and grants access to the master until it has released the bus. After the release a clean up state was inserted to ensures that the slaves have finished the bus cycle as well. If multiple masters request the bus at the same time, the arbiter prioritizes the master with the lower interface number. One of the Wishbone master implements the Setup-Bus accesses from the ROD and maps them to the Wishbone system. This allows the ROD to directly access BOC registers. Another Wishbone master is an external peripheral controller from the BOC internal processor. This allows the processor to act in the same way as the ROD can act on the register, which makes standalone operation of the card possible. Accesses from the ROD always have priority over accesses from the internal processor.

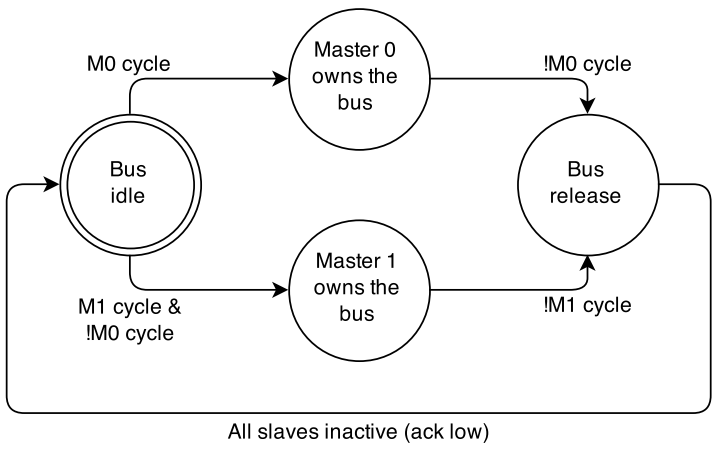


Figure 7.2: Finite state machine of the Wishbone arbiter logic for two masters

The address decoding logic in the BOC card provides four bus slaves to be addressed. The upper two bits of the address decide which slave will be requested. Table 7.1 shows the address map of the BOC card. Each FPGA register bank has its own Wishbone slave connection. Additional to the register banks 16 kB of general purpose memory are available.

Table 7.1: Mapping of the Wishbone slaves in the address decoding logic

Base address	Length	Description
0x0000	0x4000	BCF register bank
0x4000	0x1000	BMF south register bank
0x8000	0x1000	BMF north register bank
0xC000	0x4000	BCF general purpose memory

All register banks are implemented as Wishbone slaves and allow the addressing of a configurable number of registers. Each register holds 8-bit of data. The register bank has a decoding block, mapping the Wishbone signals into four signals per register: “data input”, “data output”, “write enable” and “read enable”. The two “enable” signals are always valid for one clock cycle, independent how long the external Wishbone strobe and cycle signals are. This allows the Wishbone bus to run in asynchronous operation, but having the internal register bank synchronous to the internal clock.

Simulation and results

To ensure the functionality of the Wishbone interconnect and the register bank, several test benches were created, which automatically test for errors in the behaviour of the firmware blocks.

The arbiter and address decoding logic are tested using random bus accesses from two concurrent masters. Each master starts a bus access and waits for its response. The response of a read transaction is then compared to the expected value. In case the read value does not match the expected value, an error is be thrown from the simulation environment. To also test the address decoding logic, four slaves mirroring requests were added. Each slave answers the request in a specific way allowing to distinguish if a request is routed to the wrong slave. During a simulation of one second about 1.7 million bus cycles for each master finished error-free. Figure 7.3a shows the waveform for a concurrent bus cycle by both masters coming from this simulation. Both masters try to access the bus at the same time and the arbiter will prioritize the first master and the second master has to wait. Additional tests have also been performed in the hardware. An integrated Chipscope logic analyser has been used to monitor parts of the Wishbone interconnect. Figure 7.3b shows a concurrent bus access and again the first master is winning in the arbitration process.

7.3.2 BMF firmware loading

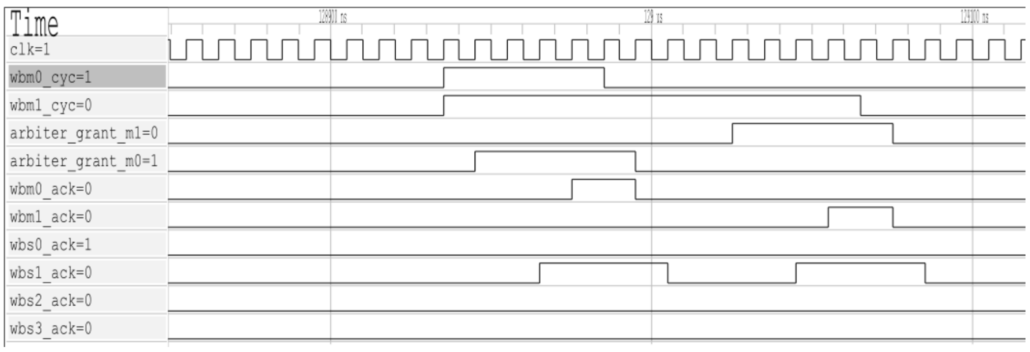
One important part of the BCF firmware is the loading of the BMF firmware. The two BMFs do not have a permanent storage connected, so after power-up they are unconfigured. The configuration needs to be performed by the BCF using the JTAG connections between the FPGAs. The corresponding firmware module is called “ProgUnit” and it is instantiated twice, so each BMF is managed by its own programming module.

The main purpose of the programming module is to load a firmware file from an flash memory connected through the SPI and start the JTAG programming. For the JTAG programming the ACE Player [69] was used. It is provided by Xilinx and will convert the content of the firmware file in the flash memory into JTAG operations.

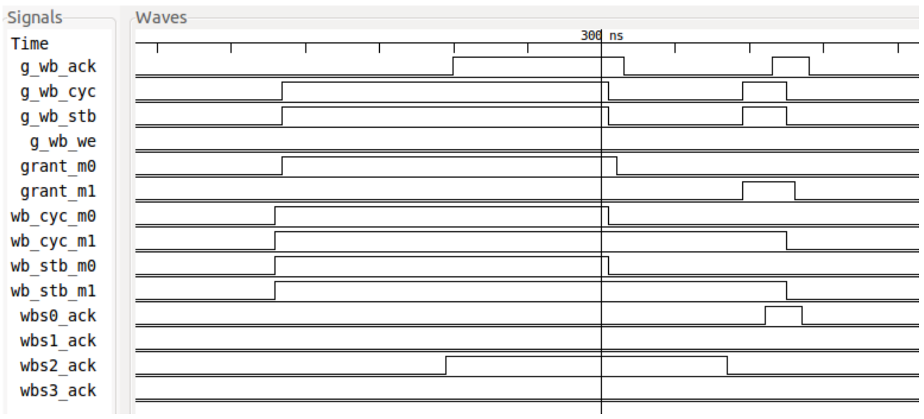
The ProgUnit allows to reprogram the firmware file in system. Flash operations, like erasing and programming can be started by the user. The data to be programmed into the flash memory will be stored in a FIFO buffer. It can hold one page[6] of 512 bytes. To verify the data written to the flash a CRC32-checksum is used.

To check if the ProgUnit works correctly a test bench was used. In the test bench a flash model supplied by the flash chip vendor has been used to model the behaviour of the flash memory correctly. The time the flash model needs to erase the full chip was slightly modified. It was decreased to about 100 ms to reduce the simulation

[6]smallest possible amount of data to be written



(a) Simulated waveform



(b) Chipscope logic analyser waveform

Figure 7.3: Simulated and real waveform of a bus of a concurrent bus access by two masters. Both masters start the bus cycle at the same time by asserting their “cycle” signal. Master 0 is prioritised (“grant_m0” high) and Master 1 has to wait for the bus cycle to finish.

run time, as the chip erase consumes most of the simulation time. The test bench verifies all operations that will be performed during operation of the card. It starts with erasing the flash memory. Then it will program a test firmware image and compare the checksum from the flash memory. If everything is correct, it will start the JTAG programming. The status bits of the ACE Player are finally compared if the end-of-file mark has been reached without any error during programming.

7.3.3 Clock monitoring and control

During card operation the external clock from the TIM is monitored and its frequency is measured. Normally, the LHC clock has a frequency of 40.079 MHz [52], which can vary by a few 100 Hz. The BCF will issue a clock good condition if the frequency is between 39.5 and 40.5 MHz. The measurement is performed using a window size of one second. If the frequency of the VME clock leaves the allowed range, the BCF will automatically switch back to an internal fall-back clock and stays there until a manual intervention takes place.

A clock failure will also be reported to the ROD using a dedicated signaling line on the VME backplane, called “BOC_OK”. It will be set to ’1’ only if the card is clocked by the VME clock and its frequency is good. Otherwise the signal will be reset to '0'. The ROD will report this error to the higher-level readout systems as a BOC clocking error via the S-LINK.

7.3.4 Microblaze processor and its software

The Microblaze processor [70] is mainly responsible for the Ethernet communication. Especially TCP/IP communication is too complex to handle it fully in FPGA logic, so a processor subsystem was added to the BCF firmware. Internally the Microblaze subsystem uses an Advanced eXtensible Interface (AXI) interconnect [71] which is the standard Microblaze interconnect bus for the Spartan-6 FPGA family and it is highly supported by Xilinx. The processor has access to 64 MB of DDR2 memory and 32 kB of internal FPGA memory. Another 8 MB of flash memory are connected through the SPI configuration interface to the processor. It also provides a simple serial port interface for debugging.

To connect to the BOC register bank this Microblaze processor uses an AXI External Peripheral Controller (EPC) as a bridge to the Wishbone interface. This EPC translates the AXI signals into the Wishbone signals. This EPC serves the purpose of a Wishbone master allowing it to access the BOC register bank.

The software part of the Microblaze processor consists of a bootloader and the main software. After powering up the card, the Microblaze processor will boot from its FPGA internal memory, which contains the bootloader software. This bootloader software first performs a memory check and then copies a software image stored in the SPI flash memory into the external memory. Afterwards, it proceeds with the software execution from the external DDR2 memory. This process is needed, because the main software does not fit into the limited internal FPGA memory.

The main software is responsible for the Ethernet communication and therefore it is running a Xilkernel [72] real-time operating system and the lwIP[7] stack [73]. Both are fully supported by the Xilinx development environment. The real-time operating system allows it to run multiple concurrent tasks, where the assignment of computation time is managed by an scheduler in the operating system.

To allow reading and writing registers from the Ethernet connection a small server task is embedded in the main software. It is based on the IPbus [74] protocol developed for the CMS Level 1 Trigger system, but it was ported from UDP to TCP. The reason for using TCP is the reliability of the communication as TCP ensures not to loose any data. Originally, the first prototype version of the firmware had a hardware module for the IPbus communication integrated. To be compatible with the developed tools, this protocol was adopted to the Microblaze implementation. The Microblaze converts the IPbus transactions into Wishbone bus cycles through the AXI EPC.

Another important task provides a virtual JTAG cable that can be used to connect from the Ethernet to the JTAG ports of the BMFs. This can be used to run all debugging tools from Xilinx, which are normally available through the JTAG port per remote. A dedicated JTAG programmer is not needed anymore.

[7]light-weight IP

7.4 BMF firmware

The BMF firmware is mainly responsible for the signal processing of the data between ROD, detector and the higher-level readout systems. Figure 7.4 shows an overview of the global firmware structure in the BMF. In general, the firmware is split into five main parts, which will be described in the following sections:

1. Global BMF logic for clocking and control
2. Transmitter data path to the detector
3. Receiver data path from the detector
4. Front-End emulation
5. S-LINK data path to the ROS

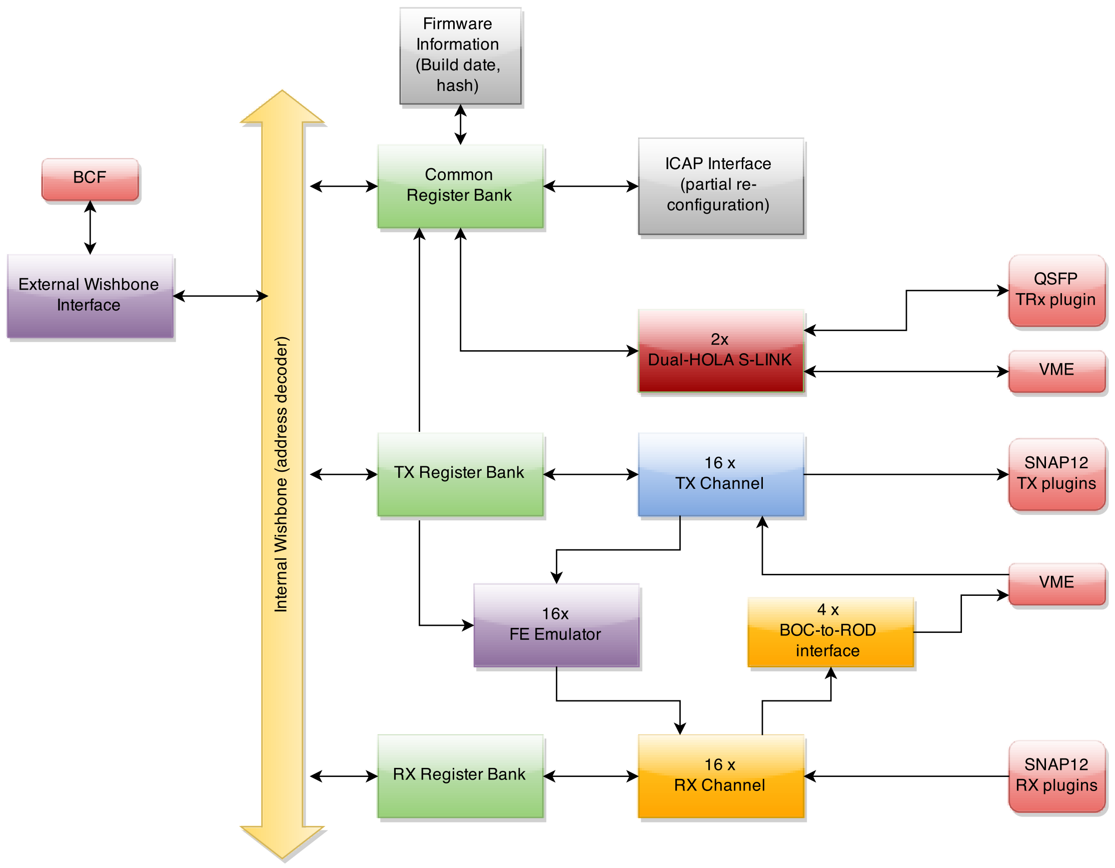


Figure 7.4: Overview of the BMF firmware blocks. The register bank splits up into three segments for the RX, TX and common firmware parts.

7.4.1 Clock generation

The clocking is an essential part in the BMF firmware as it has to provide different clocks to different firmware modules in the BMF. Two of these clocks are dedicated high-speed clocks, which are directly routed to the I/O clock regions. All clocks are derived from the 40 MHz LHC clock available on the backplane.

The input clock is fed into a dedicated PLL. The PLL in the Spartan-6 FPGA family supports an oscillator frequency in the range of 400 to 1080 MHz. The BOC firmware uses an oscillator frequency of 640 MHz which is the input clock frequency multiplied by 16. From this oscillator frequency a number of output frequencies of the PLL are generated. Table 7.2 shows the different output divider settings, the resulting frequency, and the usage of the different clock domains in the BOC firmware.

Table 7.2: PLL clock output configuration and usage in the BMF firmware

Output Channel	**Divider**	**Frequency [MHz]**	**Usage**
1	1	640	RX oversampling
2	2	320	TX BPM encoding & coarse delay
3	4	160	RX serial data
4	8	80	RX backplane
5	16	40	Register bank, TX serial data, RX data processing
6	64	10	Partial reconfiguration

For the two high-speed clocks there is also an additional strobe signal generated. This “SERDESSTROBE” signal makes it possible to align the high-speed clocks to the lower-speed clocks. Figure 7.5 shows the alignment of the SERDESSTROBE signal with respect to its low-speed clock for an oversampling factor of 8. The SERDESSTROBE will be generated on a falling edge of the low-speed signal and is valid for one clock cycle of the high-speed clock. The SERDESSTROBE can be used as an alignment pulse, to be always aligned on the falling edge of the slow clock signal.

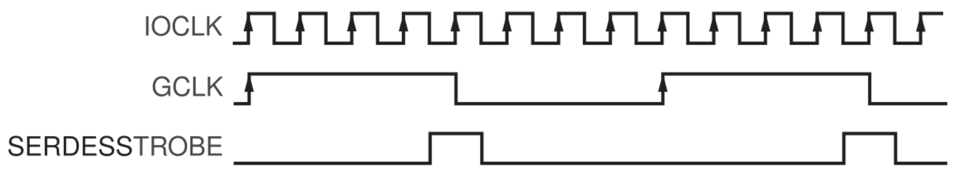


Figure 7.5: SERDESSTROBE generation for clock alignment with an oversampling factor of 8. The strobe signal is generated on the falling edge of the low-speed clock (GCLK) and it is valid for one cycle of the high-speed clock (IOCLK) [75]

7.4.2 Register bank

The register bank used in the BMF is the same as in the BCF. The available Wishbone memory space of 4 kB was divided into three sections. Table 7.3 shows a mapping of these three sections.

Table 7.3: Addressing scheme for the register banks in the BMF firmware. r: register number; c: channel number

Bit number												**Description**
11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	r	r	r	r	r	r	r	BMF global registers
1	0	c	c	c	c	c	r	r	r	r	r	BMF RX registers
1	1	c	c	c	c	c	r	r	r	r	r	BMF TX registers

7.5 BMF transmitter firmware

7.5.1 Overview

The transmitter path in the BOC card has to handle the incoming serial data from the ROD and pass it to the front end chips. Before sending the data to the front-end chip, it has to be encoded with the LHC clock and the signals have to be delayed to adjust the timing of all detector modules with respect to the LHC bunch crossing. Figure 7.6 shows an overview block diagram of the transmitter data path in the BMF. On the left side the serial data arrives to the BMF, whereas on the right side the data leaves the BMF towards the detector.

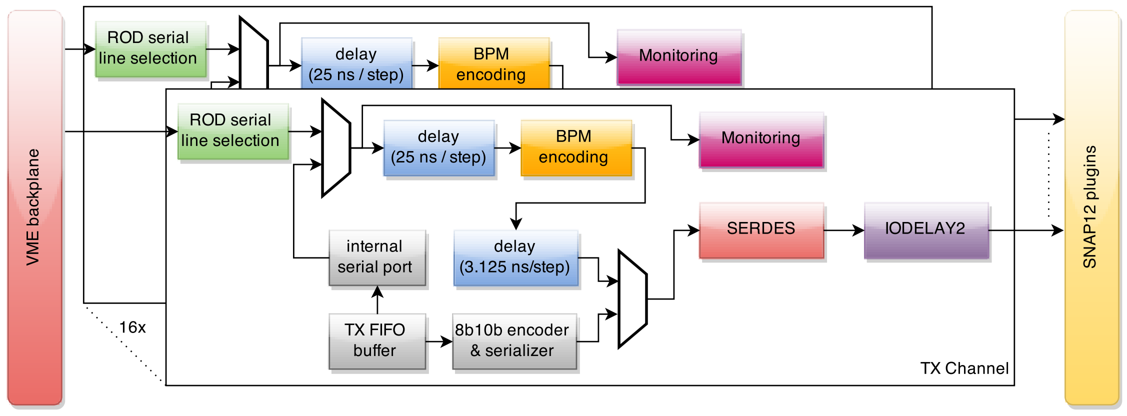


Figure 7.6: Block schematic of the BMF transmitter firmware

7.5.2 Coarse delay (step I)

To insert a delay to the signals, several delay lines have been implemented in the BMF transmitter path. The first stage works in the 40 MHz clock domain and can provide a delay of up to 775 ns with a step size of 25 ns. It is implemented as a variable length shift register. If the delay is set to zero, the input is directly fed to the output, whereas a variable number of flip flop stages are inserted to provide the required delay.

7.5.3 BPM encoding

The BPM encoding has the function to transfer the LHC clock signal together with a serial data stream to the front-end chip. As described in Section 5.2.1, the BPM encoding inserts a transition in the data signal at the beginning of the bit, and another transition in the middle when the transmitted data signal is ’1’. This results in a 20 MHz pulse for a transmitted ’0’ and a 40 MHz clock pulse for a transmitted ’1’.

To generate these signals at least a sampling frequency of $2 \cdot f_{\max} = 2 \cdot 40\,\text{MHz} = 80\,\text{MHz}$ is needed. In the BMF firmware a sampling frequency of 320 MHz has been chosen, to also cope with requirements from the coarse and fine delay, which will be described in the next sections. The processing of the signals is done in the 40 MHz clock domain, meaning 8 sampling points of the 320 MHz clock domain are processed in each 40 MHz clock cycle. The transition into the 320 MHz clock domain happens later in the serialiser block.

The BPM encoding has to provide these 8 sampling points, which are generated from the incoming serial data stream. The algorithm for generating these sampling points is simple. The 8 sampling points are numbered from 0 to 7, where point 7 is the transmitted first. To ensure a transition at the beginning of each bit, the points 4 to 7 are always an inverted copy of the points 0 to 3 of the previous cycle. If a ’0’ is transmitted, the points 0 to 3 get the same value as point 4 to 7 which means there is not transition in the middle of the bit. Otherwise, they keep their value. Figure 7.7 depicts an example waveform. Also shown is the latency of this solution, which is 65 ns from encoder input to data output.

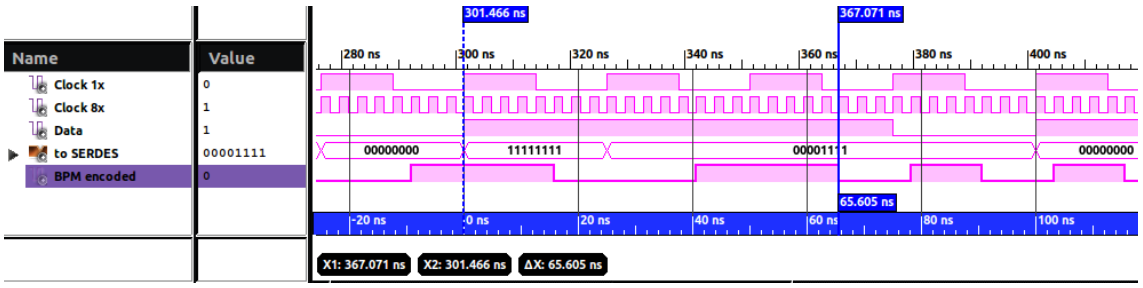


Figure 7.7: Simulation of the BPM encoding example. “Data” is the input signal in the slow clock domain, “to SERDES” are the samples to the serialiser block and finally the upsampled and BPM-encoded output data is shown. The latency between input and output is 65 ns.

7.5.4 Coarse delay (step II)

After the upsampling of the data during the BPM encoding, the second stage of the coarse delay can be applied to the signal. It provides a step size of 3.125 ns. This is needed, because the fine delay cannot cover a range of 25 ns. The implementation of this coarse delay module is again a shift register, but now shifting in 8 bits per cycle into the shift register and at the same time 8 bits will be shifted out. The number of flip-flop stages the signal has to pass can be selected with a barrel shifter. As the first stage of the coarse delay provides already a step size of 25 ns, this stage serves a total delay of 21.875 ns or seven 3.125 ns steps.

7.5.5 Serialiser

The serialiser in the BMF uses the Serialiser/Deserialiser (SERDES) block in the FPGA, which is described in detail in [76]. Each SERDES primitive can serialize 4 bits of data and two primitives can be cascaded to get serialisation factors of up to 8 bits. Figure 7.8 is showing the connectivity for a cascaded SERDES block with an serialisation factor of 8, which is used in the BMF firmware.

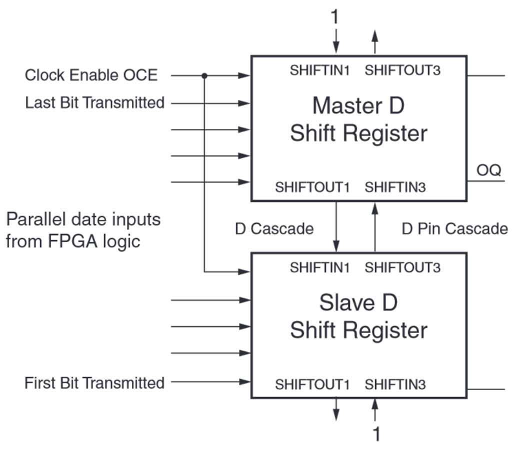


Figure 7.8: Connection of two cascaded SERDES primitives for a serialisation factor of 8 [76].

7.5.6 Fine delay

The last and most complex step of signal processing in the transmitter data path is the fine delay. As stated in the requirement chapter the fine delay has to provide a step size of $t_{\text{step}} = 100\,\text{ps}$ per step, without distorting the duty-cycle of the signal to a value outside the optoboard operating range. This requirement cannot be achieved in normal FPGA fabric anymore, as it would need a clocking of $f = t_{\text{step}}^{-1} = 10\,\text{GHz}$.

During the design phase of the firmware several different implementations of the fine delay have been analysed to find the optimal solution. In general three possible solutions were identified:

1. External delay lines

2. Internal propagation delay in the FPGA fabric

3. IODELAY2 primitive in the FPGA I/O blocks

As a requirement on the fine delay is that everything should be handled inside the FPGA, the external delay line solution was not analysed, but still it was kept as a possible solution if the FPGA internal solutions are not sufficient. Also having to delay 32 channels per card would need a lot of external chips on the card.

Internal propagation delay

The first version of the fine delay implemented into the BOC firmware was using the propagation times inside the FPGA as a reference for the delay [77]. It is called "MSR delay", because it was also used to tune the mark-space ratio of the signal. This method is based on the delay implementation used in the old Pixel Detector readout hardware [78].

The implementation of the MSR delay is based on three different delay stages, as shown in Figure 7.9. The first stage is responsible for shortening incoming data pulses, which means the duty-cycle of the signal is reduced. In the second stage (pulse broadening), the duty-cycle can be increased. Finally, the delay is applied to the signal. Central part of all three stages is the delay module. This module is built out of FPGA primitives and the resulting delay is created by the propagation time of signals through these FPGA primitives. By selecting the length of the primitive chain with a multiplexer, the delay can be changed.

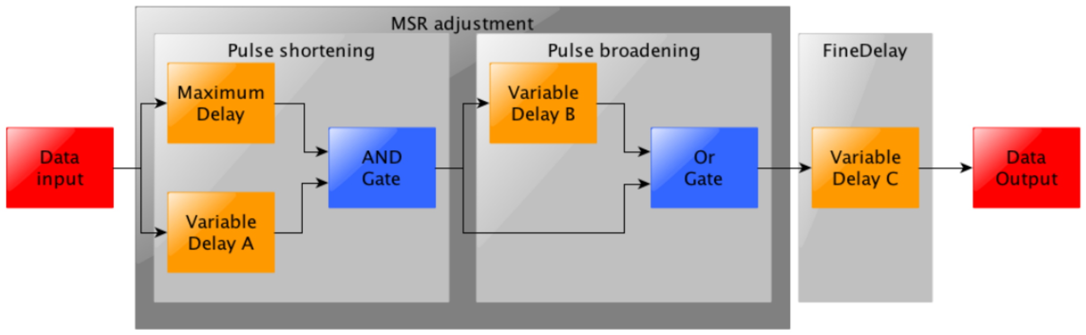


Figure 7.9: Block schematic of the MSR delay using internal propagation delays to adjust delay and mark-space ratio (MSR) [77]

This type of implementation has several disadvantages. The first disadvantage is that propagation times inside the FPGA rely heavily on external influences, like temperature or supply voltage, each device itself has its own propagation time, and also the routing has a large impact on the delay. Some of these factors can be compensated, but not all of them. To not rely on the synthesis toolchain, the user has to do a manual routing process. Another big disadvantage is that all tunable parameters (i.e. the three delay settings) have an impact on the duty-cycle and the total delay of the MSR delay. If the user wants to change the duty-cycle of the signal, the delay is also changed and vice-versa. In Figure 7.10 a measurement of the relative error rate as a result of a not well tuned duty-cycle as a function of the delay setting and the duty-cycle setting (MSR) is shown. The white area in this plot is the error-free region. For each delay setting there is a different optimal MSR setting. That results in a large effort with respect to tuning of this fine delay implementation. Therefore, other implementations have been developed.

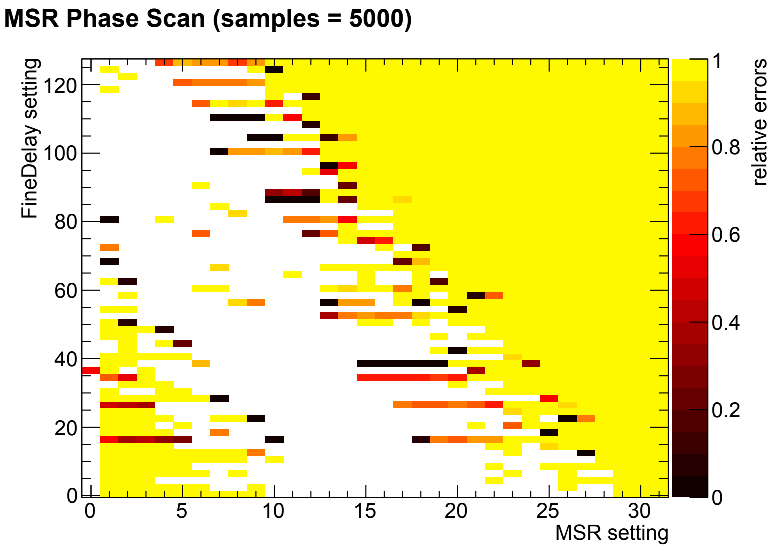


Figure 7.10: Measurement of the relative error rate of the MSR delay as a function of the duty-cycle and delay setting. The white area corresponds to the error-free region [77].

IODELAY2 primitive

The Xilinx Spartan-6 FPGAs provide a IODELAY2 primitive in their FPGA I/O logic, which was designed to allow compensation of the skew of incoming signals. It can be used as input and output delay with fixed or variable delay time. For the usage in the transmitter data path it would need to be configured to a variable output delay. But "the variable modes of the delay line do not apply when it is being used as an output" [76], so this mode is not possible for the IODELAY2 in the Spartan-6 FPGAs.

To cope with this limitation, Xilinx provides a solution [79]. In this workaround the IODELAY2 primitive of an unused I/O port is used as a variable input delay and the delayed input signal is then routed back to the output. Figure 7.11 shows a block diagram of this solution.

This workaround has been implemented into the BOC firmware and the performance of this solution has been measured. The BOC card does not provide many unused I/O pins and these pins are not in the same I/O region as the designated output pins. This results in a large overhead for the local routing (red wire in Figure 7.11) during implementation of this design.

Figure 7.12 shows the results of a duty-cycle measurement. To check the impact of the IODELAY2 implementation a firmware version without the IODELAY2 has

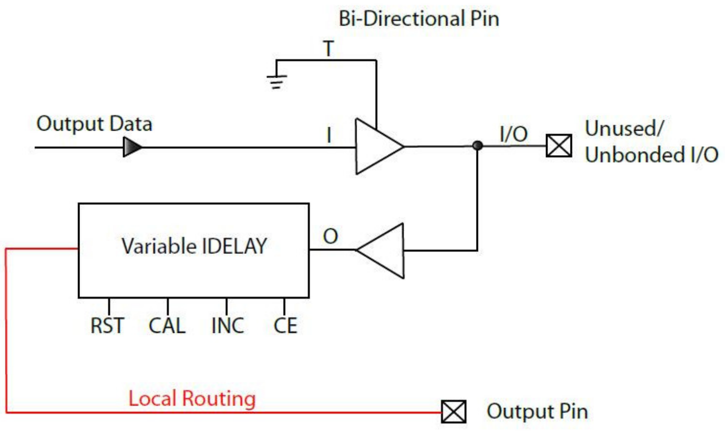
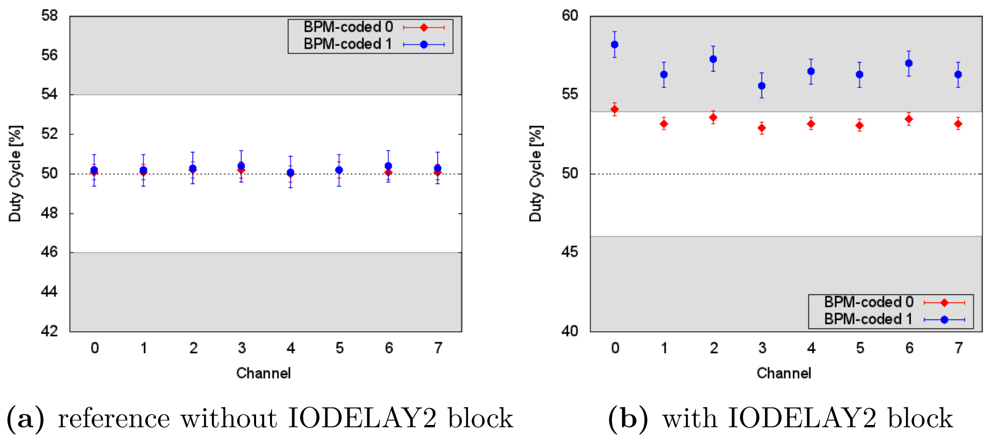


Figure 7.11: Usage of the IODELAY2 primitive as a variable input delay. The IODELAY2 primitive of an used I/O has to be routed back through the FPGA to the designated output [79].

been implemented and used as reference. It shows a clean 50 % duty-cycle. Instead when using the IODELAY2 primitive with the workaround, the duty-cycle of that signal is out of the allowed range. Also it depends on the signal that is sent out. As the IODELAY2 block is the last stage of processing in the TX path, the duty-cycle cannot be tuned anymore after this stage. The reason for the duty-cycle distortion in this case could be traced down to the local routing.



(a) reference without IODELAY2 block (b) with IODELAY2 block

Figure 7.12: Measurement of the duty-cycle for BPM encoded '1' (40 MHz) and '0' (20 MHz) signals with and without the IODELAY2 workaround solution. The white area is the operating range of the optoboard from 46 to 54 percent duty-cycle.

As the first version of the fine delay using the IODELAY2 primitive totally failed in the first measurement, another way of implementing the fine delay had to be found. The IODELAY2 primitive gave very good results in terms of linearity of the delay, so it was decided to still try to use it.

This lead to the new idea of using the fixed output delay of the IODELAY2 primitive, which is fully supported by the Spartan-6 architecture. This has the advantage that the delay can be added to the signal right in the I/O block and it can be registered with a flip-flop in the I/O region before the delay is applied. It eliminates the routing as the cause of the duty-cycle distortion in the previous implementation. The only problem which needs to be addressed is the configuration of the delay, because it is now a fixed delay. But as the delay can be changed during a synthesis, it is stored somewhere in the configuration memory of the FPGA. Therefore, it is only fixed from the FPGA fabric's point of view. The way to change the delay value during runtime is to change it in the FPGA configuration. The Spartan-6 architecture provides an internal configuration access port (ICAP), which can be used to read and write the configuration memory. Changing parts of the configuration memory during the operation of the card is commonly referred as "partial reconfiguration". Figure 7.13 shows the concept of this mechanism. Instead of directly accessing the delay settings, they are changed in the configuration memory of the FPGA (yellow) through the ICAP interface (blue).

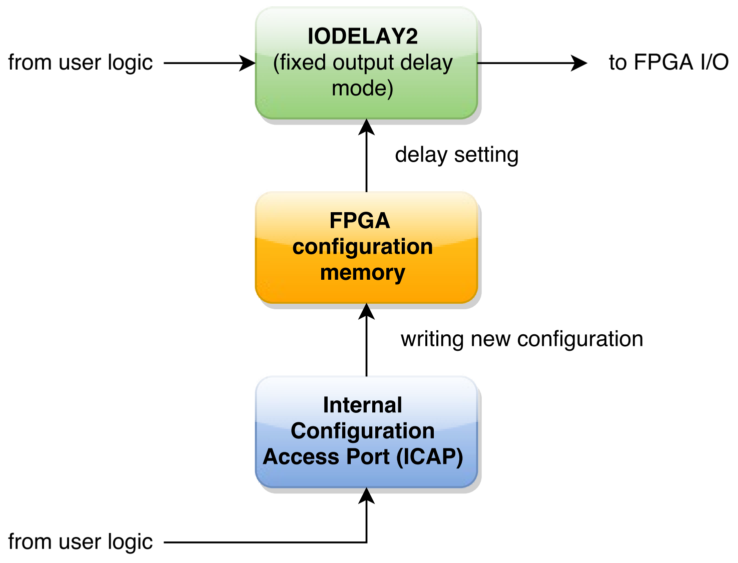

from user logic
IODELAY2
(fixed output delay mode)
to FPGA I/O
delay setting
FPGA configuration memory
writing new configuration
Internal Configuration Access Port (ICAP)
from user logic

Figure 7.13: Mechanism how to change the fixed output delay setting using partial reconfiguration. The IODELAY2 primitive is used as fixed output delay and its setting will be changed in the FPGA configuration memory through the ICAP interface.

As the mapping of the configuration bits to functions in the FPGA is intellectual property of the FPGA vendor, it is not publicly documented. This makes it difficult to find out which of the configuration bits are influencing the delay setting. The easiest way to change the delay is on the netlist level. In the netlist all FPGA elements and their configurations are stored. A value changed there will result in a change of configuration bits. The netlist is the result of the place and route process during the synthesis of the firmware (see Section 7.2.2). Xilinx provides a tool called "FPGA editor" to edit these netlist files. From the edited netlist a differential configuration bit file can be generated. It only contains the changed configuration bits and has a size of some kilobytes. Using the Makefile-based build flow, these partial bit files are automatically generated. Each TX channel has its own set of partial bit files, one for each delay setting.

To load these partial bit files, an ICAP configuration interface was implemented into the BMF firmware. This configuration interface consists of a FIFO buffer to hold the data of one partial bit file. The FIFO buffer can be filled through the normal register interface. When the partial bit file has been loaded into the FIFO buffer, the programming into the FPGA configuration memory can be started. An internal state machine feeds the content of the partial bit file into the ICAP interface. When the loading has finished, the configuration interface will assert a done signal in the status register. Then a new bit file can be loaded into the FIFO buffer.

Using this partial reconfiguration, new measurements of the delay and duty-cycle behaviour were made. Figure 7.14 shows the result of such a measurement. The delay of the signal is linear, and also the duty-cycle is nearly independent of the delay setting. It is always in the accepted range of the optoboard. The range of the delay is about 7 ns which is enough to cover the step size of the coarse delay.

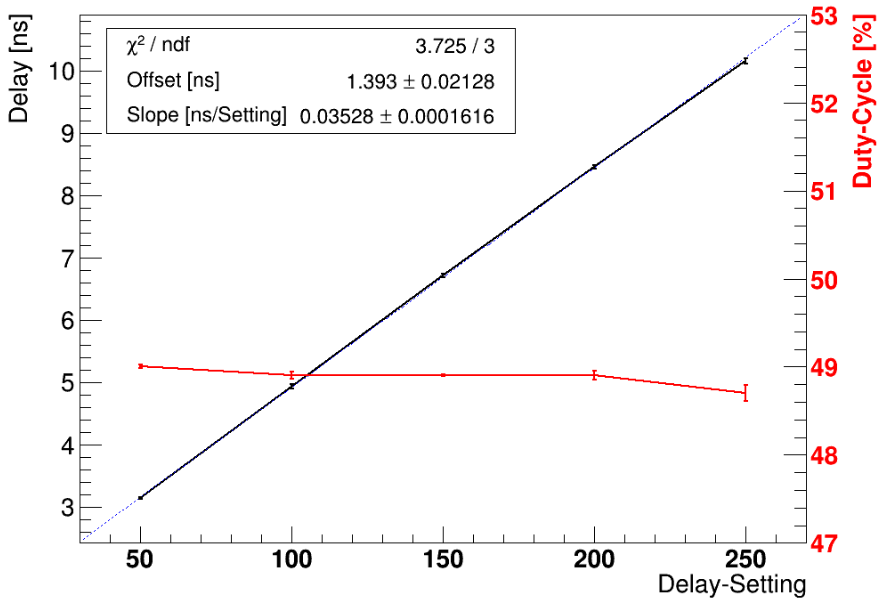


Figure 7.14: Measurement of the delay and duty-cycle using the partial reconfiguration method for changing the IODELAY2 value [80].

7.5.7 Debugging and monitoring tools

Additional to the required modules, the transmitter data path also utilizes some special debugging and monitoring tools. These tools were mainly added to check the card functionality and to use it in standalone operation without a ROD connected.

A central part of the debugging functionality in the transmitter data path belongs to an internal serial port that can be used to inject a serial data stream to be sent through the optical interface. A FIFO buffer in front of the serial port can store up to 16 kBit of data which can be sent out as a coherent serial data stream. This allows the communication to a front-end chip in standalone operation and was widely used to test the functionality of the card. The FIFO buffer can be filled from the internal register bank. After the user has stored the data into the FIFO buffer, the sending of the data can be started by enabling the transmission. Another operation mode of the serial port is the 8b10b mode. It can be used to generate a 160 Mbit/s 8b10b encoded data stream. In this mode, the data in the FIFO buffer is encoded using a Xilinx 8b10b encoder core [81]. The data is then serialised out at 160 Mbit/s. This operation mode is helpful for testing and debugging the functionality of the RX data path. If an optical fibre is connected between the TX plugins and the RX plugins, a loopback can be performed and the data will look like coming directly from the detector.

Besides sending test data out, the serial data stream from the ROD is also monitored. A simple decoding unit checks the different front-end commands, like triggers, calibration pulses, etc. Each command will be counted separately allowing to track the number of commands. Also invalid commands will be counted. Optionally, they can also be stored into a FIFO buffer. This is helpful for debugging the full chain between ROD and front-end chip. It allows checking if the data is already corrupt at the BOC card level. When it has been confirmed to be already corrupted in the BOC card, the data in the FIFO buffer allows one to analyze how the commands get corrupted.

On top of that channel specific monitoring which looks only on the command data format, a special spying utility was developed, which can additionally spy on contents of commands. It is based on the command decoding block used in the FE-I4 emulator (see Section 7.7). The user can select one of the 16 TX-channels and the monitoring will collect a snapshot of up to 512 FE-I4 commands into a FIFO buffer.

For low-level monitoring a Chipscope logic analyser core was added to the transmitter firmware. It is connected to the serial data lines and the error detection flags. This allows one to trigger on erroneous data words and directly view the waveform to analyse the type of corruption.

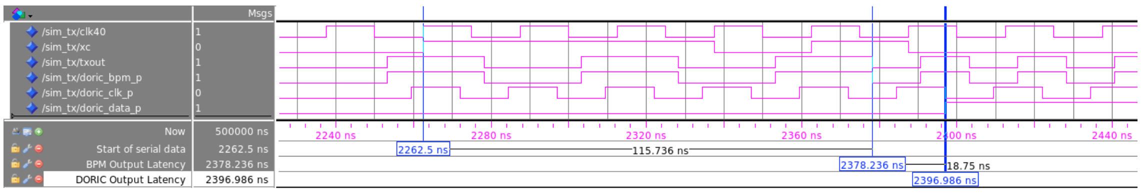
7.5.8 Simulation results of the transmitter firmware

To ensure that the transmitter firmware is working as expected, several simulation tests were created. These simulations test different parts of the transmitter. First, the basic functionality, like the BPM encoding has been tested. Later also the debugging and monitoring utilities were simulated.

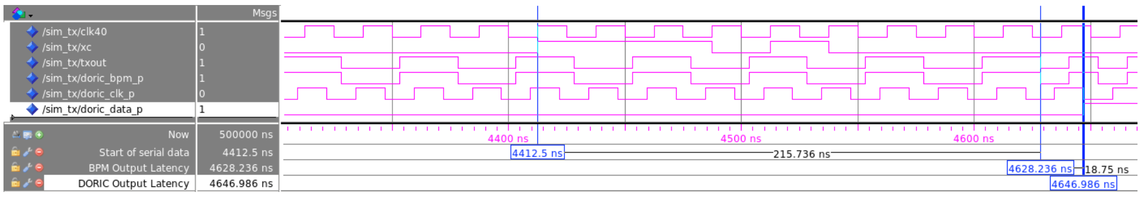
TX data path latency & BPM encoding

An important part of the simulation is the timing of the TX data path and the verification of the BPM encoding as this is the central part of the transmitter firmware. A testbench including one channel of the transmitter firmware together with a DORIC simulation model has been created. The test data is injected at the serial lines coming from the ROD. A closer look has been taken on the latency, which the TX data path is adding to these serial signals and if the BPM encoding is properly working together with the optoboard simulation model for the DORIC. Figure 7.15 shows a latency simulation with two different test cases. In both cases the ROD is generating a trigger datastream (“11101”). Being measured is the time when the trigger command arrives on the BPM encoded output signal. In Figure 7.15a the latency with a coarse delay setting of 0 is shown. The latency between input and output in the simulation is about 116 ns. Then the coarse delay is increased to 32, which adds $32 \times 3.125\,\text{ns} = 100\,\text{ns}$ of latency. As shown in Figure 7.15b the latency is exactly increased by this amount of time, to about 216 ns. In these diagrams it can also be seen that the DORIC can decode the clock and data signal from the BPM encoded data stream. The latency of this decoding is 18.75 ns regardless of the setting of the coarse delay. So it can be concluded that the basic functionality of the transmitter firmware is working as expected. Also the latency is less than the required 150 ns.

What is not tested in simulation is the fine delay implementation, as the partial reconfiguration cannot be simulated. Therefore, the fine delay performance will be presented with measurements in Chapter 8.



(a) without delay, resulting latency 115.736 ns



(b) coarse delay = 32, resulting latency 215.736 ns

Figure 7.15: Simulated waveforms showing the latency in the TX data path. The latency is measured from serial input coming from the ROD (“xc”) to the output of the BPM-encoded data (“doric_bpm_p”). It is shown by the first two cursors. An additional cursor shows the latency of the optoboard simulation model.

Monitoring

To test the monitoring implemented into each TX channel, test signals containing a known number of front-end commands were sent through the transmitter. The monitoring should detect and count all of these commands correctly. By reading the counters and comparing to the number of sent commands, the functionality of the monitoring can be verified:

```
# ** Note: #triggers = 500 expected 500
# ** Note: #ECR = 10 expected 10
# ** Note: #BCR = 100 expected 100
# ** Note: #CAL = 50 expected 50
# ** Note: #Slow = 30 expected 30
# ** Note: #Corrupt = 10 expected 10
```

The monitoring for errors in the data stream has also been tested. Figure 7.16 is showing two trigger commands being sent, of which one has a bitflip. After a decoding and detection latency, the erroneous command word is correctly flagged as corrupted.

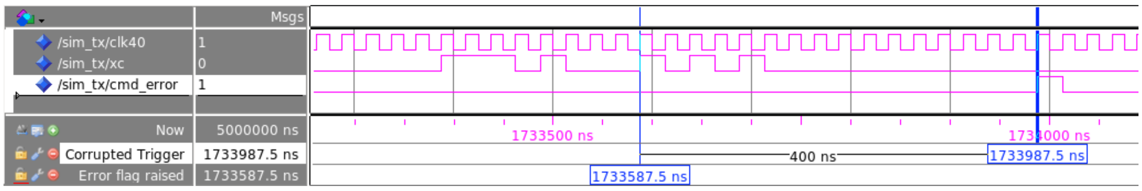


Figure 7.16: Simulation of the TX monitoring flagging bit-flip in a trigger command as corrupted word by raising the "cmd_error" flag 400 ns after the start of the corrupted command.

7.6 BMF receiver firmware

7.6.1 Overview

The firmware for the receiver data path is responsible for sampling the data from the detector and forward the decoded data to the ROD. A block diagram of the receiver path firmware is shown in Figure 7.17.

7.6.2 Clock and data recovery

The incoming data stream from the front-end chips needs to be sampled and phase aligned to the internal clock signal. This job is performed by the clock and data recovery. The phase of the incoming data signal is unknown due to differing cable lengths.

The clock and data recovery algorithm is based on a Xilinx application note [82]. Central part of this algorithm is the oversampling of the incoming data by a factor 4 in the FPGA-internal SERDES blocks. Then the algorithm searches for transitions in the sampled data. The optimal sampling point of the data is 2 samples off this transition right in the middle of the bit.

To be protected against jitter in the data signal, the optimal position of the sampling point is measured for a longer period. During this training phase an internal counter for each sampling point is incremented, whenever this sampling point is considered to be good. When one of the sampling point counters reaches its top value, the corresponding sampling point is chosen. The clock and data recovery will use this sampling point until a new training sequence is started.

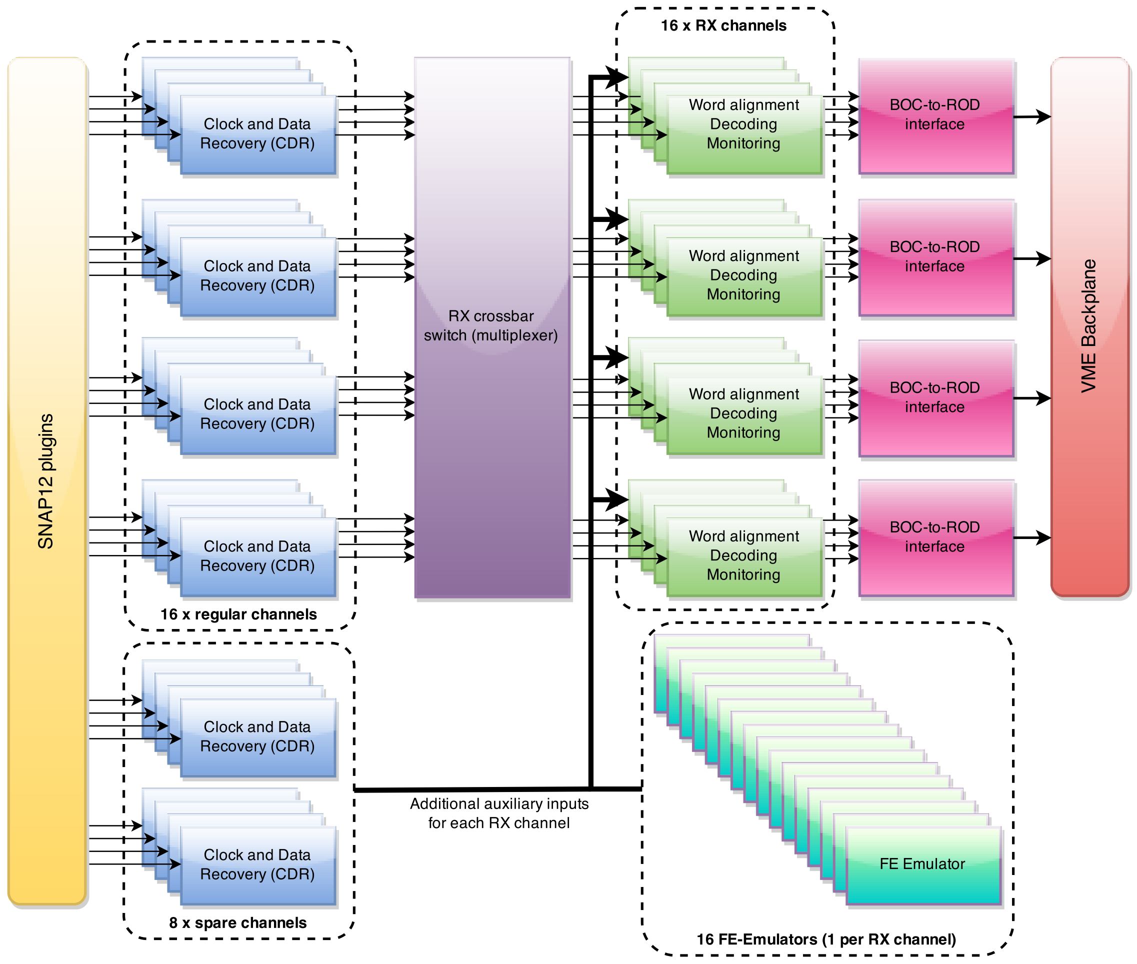

SNAP12 plugins
Clock and Data Recovery (CDR)
Clock and Data Recovery (CDR)
Clock and Data Recovery (CDR)
Clock and Data Recovery (CDR)
16 x regular channels
Clock and Data Recovery (CDR)
Clock and Data Recovery (CDR)
8 x spare channels
RX crossbar switch (multiplexer)
Additional auxiliary inputs for each RX channel
16 x RX channels
Word alignment Decoding Monitoring
Word alignment Decoding Monitoring
Word alignment Decoding Monitoring
Word alignment Decoding Monitoring
BOC-to-ROD interface
BOC-to-ROD interface
BOC-to-ROD interface
BOC-to-ROD interface
VME Backplane
FE Emulator
16 FE-Emulators (1 per RX channel)

Figure 7.17: Block schematic of the BMF receiver firmware

7.6.3 Channel multiplexer

After the clock and data recovery mechanism the data is fed into a crossbar switch. This allows one to route every optical input channel to every RX channel and thus changing the connectivity between the fibres and the connected RX channels freely. Therefore, the BOC card can be operated in non-standard environments from the connectivity point of view without changing anything of the overlaying systems. The BOC card can even “fake” a full detector connected to one input data stream to several RX channels. This is very helpful in environments which have only access to a limited amount of front-end chips, like lab environments.

7.6.4 Decoding

The decoding of the 8b10b data is performed using a standard Xilinx 8b10b decoding block [81]. Before the decoding can start, the serial data stream has to be matched to 8b10b encoded data words. To find the right word boundaries in the 8b10b data stream the k-words are important, as they are the only 8b10b encoded data words, which cannot be found across word boundaries. To decrease the probability that k-words are detected by mistake due to bit flips, only a sequence of two k-words is being identified. Table 7.4 shows the possible sequences of the two k-words. Only if one of these sequences is detected correctly the decoding is performed. Otherwise the decoder will stay out of synchronisation. Also a timeout logic has been implemented which needs to detect k-words on a regular basis at least once a second. If that is not the case, the synchronisation of the data link is lost until the next k-word sequence is detected.

Table 7.4: k-Word sequences used to detect the 8b10b word alignment

Bit sequence	k-Word
0011111010 1100000111	EOF & SOF
1100000101 0011111000	EOF & SOF
0011111001 1100000111	IDLE & SOF
1100000110 0011111000	IDLE & SOF
0011111001 1100000110	IDLE & IDLE
1100000110 0011111001	IDLE & IDLE

The correctly aligned words can then be directly fed into the 8b10b decoding block. This decoding block is a lookup table-based implementation which uses the FPGA internal memories for storage of the decoding table. The incoming data word is used as the address input of the lookup table. The decoder also keeps track of the running disparity and flags disparity errors in the data stream. The result of the decoding is the data word and a k-word indicator. Additionally, the decoding block can tag decoding and disparity errors in the data stream. Each decoded data word is then forwarded to the ROD through the ROD data bus.

7.6.5 ROD data bus

As described in the requirements for the BOC card, the data between BOC card and ROD is exchanged on a 12-bit wide data bus. Each data bus carries the data of 4 front-end chips. Figure 7.18 shows a block schematic of the interface handler. The data from the decoding block is first stored into FIFO buffers and then each of the input channels is poked for new data on a round-robin basis. The data is sent to the ROD without any check that the ROD is ready to take the data. By asserting a busy signal the ROD can pause the data-taking when its internal buffers are almost full. The busy signal will immediately stop the triggers to the detector and therefore the data coming from the detector also stops.

7.6.6 Data Monitoring

Data monitoring is an important aspect in the receiver firmware, as it allows one to track and debug failures in the system at the earliest possible point in the off-detector readout system. Several status flags allow the monitoring of the data quality. The clock and data recovery unit reports if it could lock on an incoming signal at all, which means there is at least some activity on the data line. In the next step

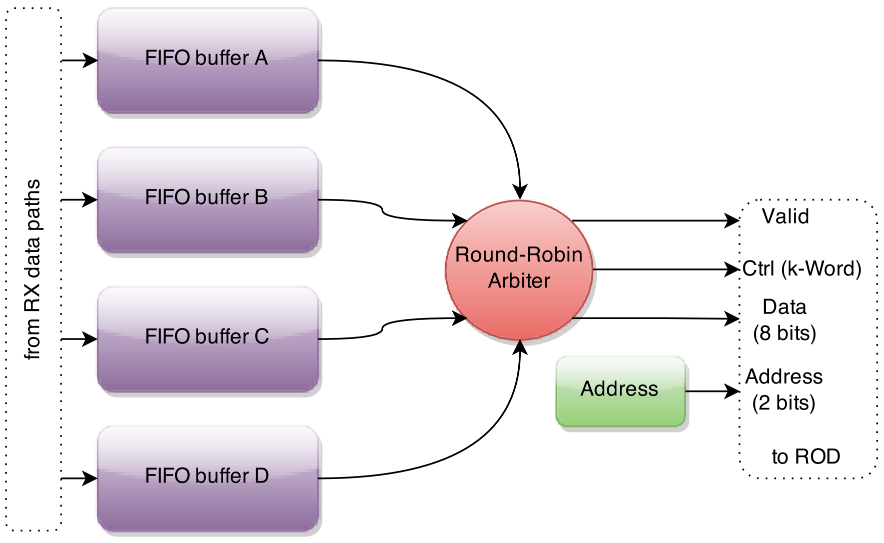


Figure 7.18: Round-Robin arbiter for the ROD data bus interface

the word alignment reports about the synchronisation to 8b10b word boundaries. Then the 8b10b decoder reports gives information data words with errors and if the disparity of the incoming data is good. A final check against the frame and data format is done on the decoded data. It checks if the sequence of bytes fits into the protocol the front-end chip should send out. All status bits are accumulated, which means the user always reads back the worst status and needs to manually reset the status bits. A reset is performed by writing a '1' to the corresponding bit in the status register.

Additionally, there is the possibility to directly spy on the data using an internal FIFO buffer. On request a certain amount of data can be stored in the buffer until it is full. The user can choose if the encoded or the decoded data should be stored in the buffer and if IDLE words are suppressed or not. The data can later be read out through the register interface. Together with the FIFO buffer in the transmitter path, this buffer allows the full standalone operation of the card, as commands can be injected into the TX FIFO buffer and the response can be monitored with the RX FIFO buffer.

It was requested during the development of the firmware that DCS data can be filtered out of the data stream to be forwarded to the DCS. Therefore, another monitoring tool for this data has been implemented. The front-end chip can send measured values from an internal ADC on request through the optical interface. This data needs to be filtered in the BOC card. This filtering is also performed in the RX path, as it is spying for register readbacks of the ADC value register (register number 40). The value in the register readback is then stored in a FIFO buffer and can be read out by the BCF, which can then forward this data to a DCS server PC. Currently, only the filtering is implemented in the firmware, whereas the control of the readback and the forwarding to the DCS has still to be implemented.

Another value to monitor is the link occupancy. The link occupancy monitoring is performed by counting the number of non-IDLE words behind the 8b10b decoder and divide that by the total number of data words. In the firmware the measurement window is 65k words. The live occupancy is showing the occupancy in the last window, whereas the mean occupancy is a low-pass filtering of the live occupancy:

$$\mathrm{Occ}_{mean}[n] = \frac{252}{256}\mathrm{Occ}_{mean}[n-1] + \frac{4}{256}\mathrm{Occ}_{live}[n] \tag{7.1}$$

The link occupancy can be read out through the register interface. A value of 0 equals 0 % occupancy and 65535 corresponds to 100 % occupancy. The ROD can use this occupancy information to monitor for saturating data links.

7.7 FE-I4 emulator firmware

The front-end emulator in the BMF firmware plays an important role as debugging platform. It was developed from the need to have some kind of detector interface always available, even if no real detector is connected to the card. Each BMF contains 16 individual front-end emulators. This is the same number as front-end chips in the real detector. Figure 7.19 shows a block diagram of the most important emulator modules.

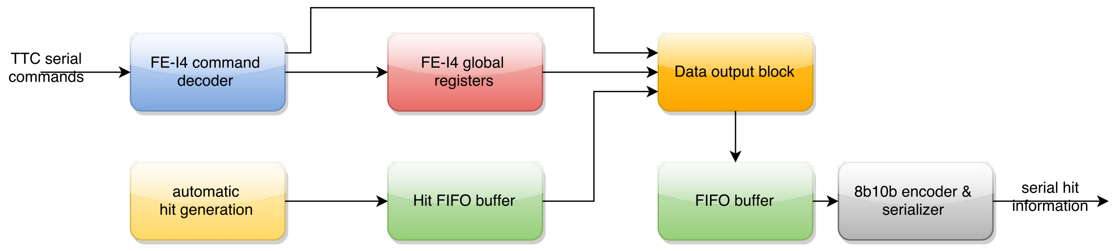


Figure 7.19: Block schematic of the FE-I4 emulator in the BOC firmware. The TTC commands are decoded and a response is generated. The response will be 8b10b encoded and sent out of the emulator.

The emulator consists of two parts. The first part is decoding all front-end commands. The decoded commands are transferred to a data generator. It generates the response of the front-end chip to the command. Two response types can be generated by the emulator. First, a readback of the global registers can be done, which returns the address and value.

The second and more important response is hit information after a trigger command has been received. The emulator provides two possible modes for generating hit information. The first mode is a total manual hit injection, where the user can provide a pattern to be sent out when the next trigger arrives. This can be useful to test the behaviour of the readout chain to all possible forms of incoming data frames. The second mode is an automatic hit generation using a pseudo-random number generator. The hits are generated constantly during operation and will be stored into a FIFO. As the pseuro-random number generator can also generate hits outside the row- and column boundaries of the chip, a check on the generated hits is performed and hits outside the boundaries are rejected. If a trigger command is received by the emulator, a certain number of hits is taken from the FIFO and sent out by the emulator. Figure 7.20 shows an occupancy histogram of the hits during a scan with 10000 triggers and 200 hits per trigger. The distribution of hits is not absolutely uniform due to the used pseudo-random number generator, but it is sufficient for debugging purposes of the system.

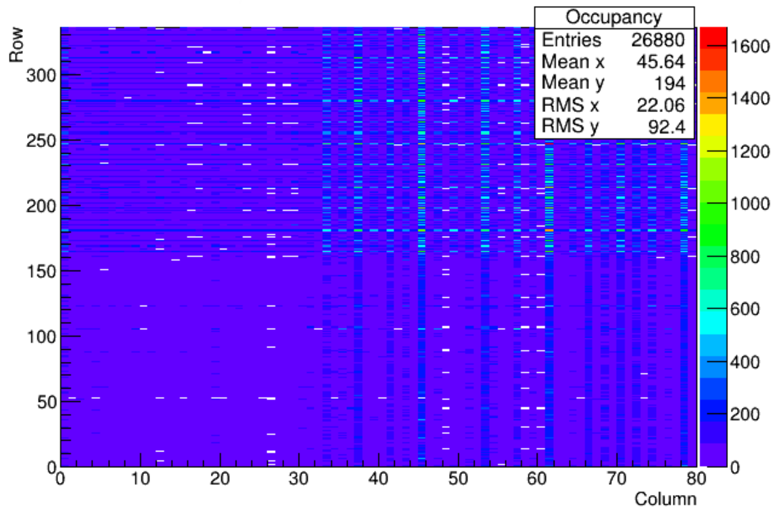


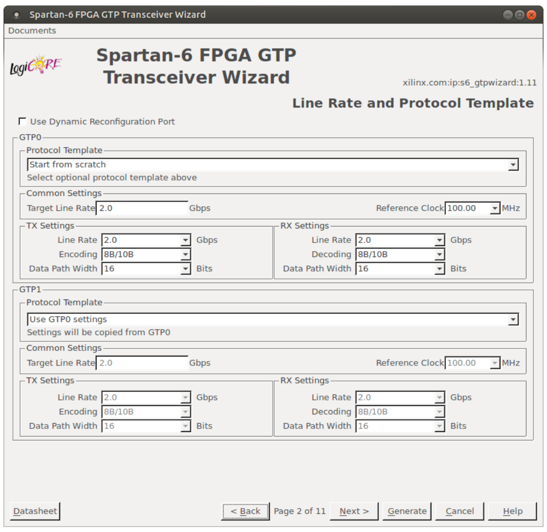
Figure 7.20: Occupancy histogram of an emulator scan using 10000 triggers with 200 hits per trigger. The colour-grade shows the number of hits per pixel [80].

7.8 BMF S-LINK firmware

The S-LINK implementation in the BMF firmware was developed by Andreas Kugel from the University of Heidelberg. As it was foreseen to handle all the data processing in the FPGAs, also the S-LINK is completely handled in the BMF firmware. IBL uses a dual-HOLA [83] implementation. The HOLA was already used in the Pixel Detector readout as a dedicated daughter card on the old Pixel BOC cards. The original HOLA implementation uses 2.5 Gbit/s TLK2501 transceiver chips [84]. For the IBL readout system the TLK2501 transceiver chips have been replaced by the FPGA-internal MGTs.

As the protocol and also the HOLA implementation itself did not change between the Pixel Detector and IBL readout system, the firmware of the HOLA was taken over. To cope with different interfaces between the MGTs and the TLK2501 transceiver, a wrapper module has been developed. This wrapper allows one to use the MGTs in the same way, as the TLK2501 chip. This has the huge advantage of having a HOLA firmware fully compatible with the old Pixel Detector readout and only change the interface to the high-speed transceivers.

The configuration of the MGT cores in the FPGA is shown in Figure 7.21. The transceivers run at a line rate of 2 Gbit/s using a 100 MHz reference clock. As the IBL uses a dual-HOLA implementation, also two MGTs are configured. The link encoding uses 8b10b, which was already used in the data path from the detector. The IDLE word of the S-LINK interface consists of the k-word 0xBC followed by the data word 0xC5. This is used for correct operation of the word alignment of the 8b10b encoding which looks on the k-word 0xBC in a similar way as it was already presented in Section 7.6.4. The clock correction is needed, as sender and receiver use different reference clocks and therefore the transmission frequency is slightly different. To compensate the differences in frequency clock correction sequences are injected or removed in the receiver. That means they must not carry any information, as it would be lost or duplicated. For the S-LINK interface the IDLE word was chosen, as it can be injected or removed from the datastream without any interference.


Spartan-6 FPGA GTP Transceiver Wizard
Documents
LogiCORE
Spartan-6 FPGA GTP Transceiver Wizard
xilinx.com:ip:s6_gtpwizard:1.11
Datasheet
< Back
Page 2 of 11
Next >
Generate
Cancel
Help

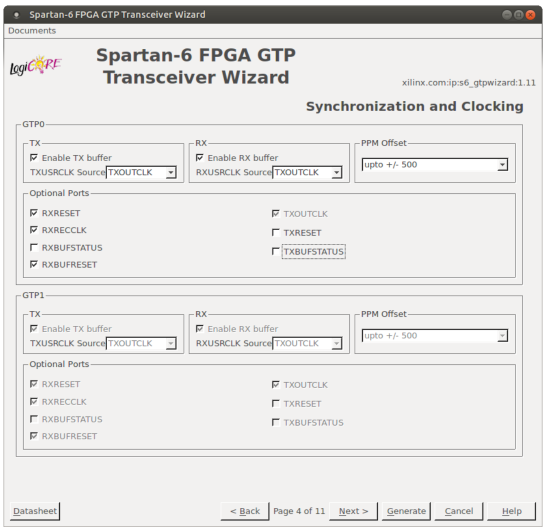
Line Rate and Protocol Template

[ ] Use Dynamic Reconfiguration Port

GTP0
- Protocol Template: Start from scratch — Select optional protocol template above
- Common Settings: Target Line Rate 2.0 Gbps; Reference Clock 100.00 MHz
- TX Settings: Line Rate 2.0 Gbps; Encoding 8B/10B; Data Path Width 16 Bits
- RX Settings: Line Rate 2.0 Gbps; Decoding 8B/10B; Data Path Width 16 Bits

GTP1
- Protocol Template: Use GTP0 settings — Settings will be copied from GTP0
- Common Settings: Target Line Rate 2.0 Gbps; Reference Clock 100.00 MHz
- TX Settings: Line Rate 2.0 Gbps; Encoding 8B/10B; Data Path Width 16 Bits
- RX Settings: Line Rate 2.0 Gbps; Decoding 8B/10B; Data Path Width 16 Bits

(a) line rate & protocol

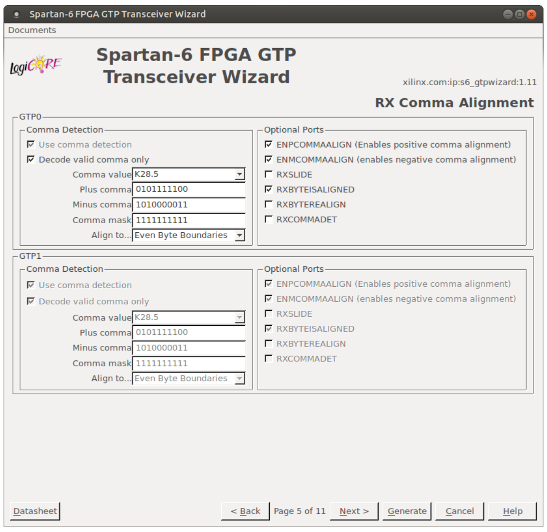

Spartan-6 FPGA GTP Transceiver Wizard
Documents
LogiCORE
Spartan-6 FPGA GTP Transceiver Wizard
xilinx.com:ip:s6_gtpwizard:1.11
Datasheet
< Back
Page 4 of 11
Next >
Generate
Cancel
Help

Synchronization and Clocking

GTP0
- TX: [✓] Enable TX buffer; TXUSRCLK Source TXOUTCLK
- RX: [✓] Enable RX buffer; RXUSRCLK Source TXOUTCLK
- PPM Offset: upto +/- 500
- Optional Ports: [✓] RXRESET; [✓] RXRECCLK; [ ] RXBUFSTATUS; [✓] RXBUFRESET; [✓] TXOUTCLK; [ ] TXRESET; [ ] TXBUFSTATUS

GTP1
- TX: [✓] Enable TX buffer; TXUSRCLK Source TXOUTCLK
- RX: [✓] Enable RX buffer; RXUSRCLK Source TXOUTCLK
- PPM Offset: upto +/- 500
- Optional Ports: [✓] RXRESET; [✓] RXRECCLK; [ ] RXBUFSTATUS; [✓] RXBUFRESET; [✓] TXOUTCLK; [ ] TXRESET; [ ] TXBUFSTATUS

(b) clocking & synchronisation

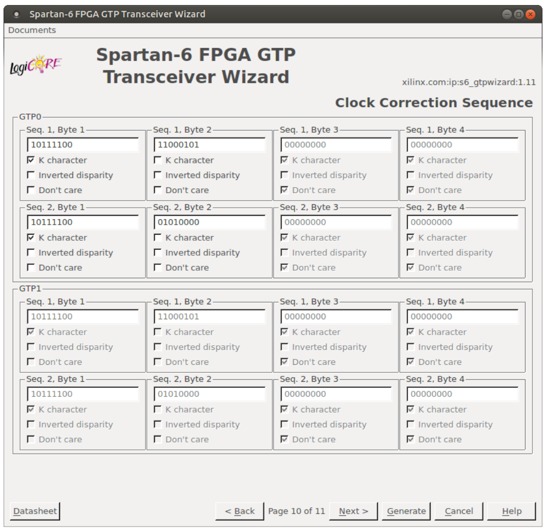

Spartan-6 FPGA GTP Transceiver Wizard
Documents
LogiCORE
Spartan-6 FPGA GTP Transceiver Wizard
xilinx.com:ip:s6_gtpwizard:1.11
Datasheet
< Back
Page 5 of 11
Next >
Generate
Cancel
Help

RX Comma Alignment

GTP0
- Comma Detection: [✓] Use comma detection; [✓] Decode valid comma only; Comma value K28.5; Plus comma 0101111100; Minus comma 1010000011; Comma mask 1111111111; Align to... Even Byte Boundaries
- Optional Ports: [✓] ENPCOMMAALIGN (Enables positive comma alignment); [✓] ENMCOMMAALIGN (enables negative comma alignment); [ ] RXSLIDE; [✓] RXBYTEISALIGNED; [ ] RXBYTEREALIGN; [ ] RXCOMMADET

GTP1
- Comma Detection: [✓] Use comma detection; [✓] Decode valid comma only; Comma value K28.5; Plus comma 0101111100; Minus comma 1010000011; Comma mask 1111111111; Align to... Even Byte Boundaries
- Optional Ports: [✓] ENPCOMMAALIGN (Enables positive comma alignment); [✓] ENMCOMMAALIGN (enables negative comma alignment); [ ] RXSLIDE; [✓] RXBYTEISALIGNED; [ ] RXBYTEREALIGN; [ ] RXCOMMADET

(c) comma alignment


Spartan-6 FPGA GTP Transceiver Wizard
Documents
LogiCORE
Spartan-6 FPGA GTP Transceiver Wizard
xilinx.com:ip:s6_gtpwizard:1.11
Datasheet
< Back
Page 10 of 11
Next >
Generate
Cancel
Help

Clock Correction Sequence

	Value	K character	Inverted disparity	Don't care
GTP0 Seq. 1, Byte 1	10111100	✓		
GTP0 Seq. 1, Byte 2	11000101			
GTP0 Seq. 1, Byte 3	00000000	✓		✓
GTP0 Seq. 1, Byte 4	00000000	✓		✓
GTP0 Seq. 2, Byte 1	10111100	✓		
GTP0 Seq. 2, Byte 2	01010000			
GTP0 Seq. 2, Byte 3	00000000	✓		✓
GTP0 Seq. 2, Byte 4	00000000	✓		✓
GTP1 Seq. 1, Byte 1	10111100	✓		
GTP1 Seq. 1, Byte 2	11000101			
GTP1 Seq. 1, Byte 3	00000000	✓		✓
GTP1 Seq. 1, Byte 4	00000000	✓		✓
GTP1 Seq. 2, Byte 1	10111100	✓		
GTP1 Seq. 2, Byte 2	01010000			
GTP1 Seq. 2, Byte 3	00000000	✓		✓
GTP1 Seq. 2, Byte 4	00000000	✓		✓

(d) clock correction

Figure 7.21: MGT IP core settings

Table 7.5: Description of some important boclib utilities

Utility	Description
ShowVersion	show version information for all FPGAs
DumpRegisters	register dump of all important registers
FlashBcf	BCF firmware upgrade
FlashBmf	BMF firmware upgrade
LaserOn	disable software laser interlock
LaserOff	enable software laser interlock
BocMon	BOC monitoring tool

7.9 IBL BOC software library

Additional to the firmware and software running on the card, a library to access all BOC card registers from a standard PC connected to the card via network has been developed. This software library is called “boclib” and was fully developed in C++. It provides abstract functions for using all the implemented firmware functionality.

The central part of of this library is the “Boc” class. In this class all the low-level communications functions are handled. It is responsible for the communication with the IPBus server in the BCF and provides functions for reading and writing registers. All higher-level classes rely on this communication class. This also allows one to exchange this class by a different one implementing other ways of reading and writing registers (e.g. from the ROD).

On top of the “Boc” class, two classes representing the functionality of the FPGAs exist. The “Bcf” class is mainly responsible for providing global functions, like enabling or disabling the laser interlock or functions for programming of the BMF firmware. The BMF functionality is provided by the “Bmf” class. It provides access to all the TX and RX channels as well as BMF global functions, like the partial reconfiguration. On top of the “Bmf” class, the classes for TX and RX channels are positioned. These classes provide all the functionality of the firmware, which is available for controlling a TX/RX channel.

Beside the pure card functionality the boclib also provides basic classes for stand-alone operation of the card. Especially, a class for controlling a front-end chip is provided. This class takes functions from the TX and RX channels to communicate with the front-end chip using only the BOC card.

Around the software library a lot of utilities were developed, each performing a special operation on the card. Table 7.5 gives a short list of the most important tools that are delivered together with the library.

As the boclib is also heavily used for testing the card functionality, it is maintained together with the firmware. If a new firmware function is developed, also boclib functions for interacting with the new firmware are added. This ensures that the boclib and the firmware are always compatible to each other.

8 Production of the IBL BOC card

8.1 IBL BOC production

After the testing of the prototypes, the revision D of the IBL BOC card was produced to equip CERN and various laboratories. The PCBs were produced by KSG Leiterplatten GmbH in Gornsdorf, and later the mounting of the components was done by Vistatronic in Mainhausen.

The production was split in batches. Table 8.1 shows the batch number, the delivery date by Visatronic and the number of delivered PCBs. Each batch has been tested in a production test after the reception. The testing and its results are the focus of this chapter.

Table 8.1: Overview of the IBL BOC production batches and their delivery date

Batch number	Number of cards	Delivery date	Comment
1	5	18.11.2013	IBL pre-production
2	15	12.12.2013	IBL
3	28	01.07.2014	Layer-2
4	46	N/A[1]	Layer-1

8.2 Overview of the production test

When the production of a batch is finished, the first test that is performed is a short production test at the manufacturer site. It should ensure that no obvious problems show up and in case something is wrong, it may allow one to repair the card immediately. During the short production test, the firmware for all FPGAs will be loaded and a short optical loopback test is performed. All cards that have passed this test successfully are delivered to Wuppertal, where they undergo the complete production test. During this production test all important card functions will be tested to ensure good quality of the cards, which can afterwards be delivered to CERN or other laboratories.

The production test is split into 12 parts:

1. Optical inspection
2. Electrical inspection & power consumption measurement
3. BCF firmware programming

[1]not yet delivered completely

4. BMF firmware programming
5. Clock test
6. Network configuration and serial number readout
7. Loopback test
8. Optical signal integrity test
9. Coarse delay measurement
10. Fine delay measurement
11. VME backplane testing
12. S-LINK testing

Most of above tests are automated tests being programmed in a production test utility, which heavily uses functions from the BOC software library (see Section 7.9). The software library is extended by software libraries to interface the measurement equipment (oscilloscope, optical switch). This allows most of the tests to run without any user interaction.

8.3 Production test setup

To perform the production test, a test stand was designed allowing all of the tests to be performed with minimal user interaction. The central part where the device-under-test is placed, is a VME backplane adapter for powering and clocking of the card together with a metal frame for holding the card during the test. Figure 8.1 shows an overview sketch of the setup. The powering of the card is done using an DC power supply with current limitation, to avoid any damage to the card in case of an electrical failure.

8.4 Production test steps and their results

8.4.1 Optical & electrical inspection

The first test after the reception is the optical inspection. During the optical inspection, pictures of both PCB sides will be taken and it will be checked that all components are mounted correctly. If missing or wrongly mounted components are detected, the card will be sent back to the manufacturer for repair. The second step is an electrical inspection to check all power supply lines for shorts. If no shorts are detected, the power consumption will be measured in three different operation modes (bare with no firmware loaded, flashed with firmware loaded and with optical plugins). Figure 8.2 shows a histogram of the current consumption of the 5 V supply in all the operation modes.

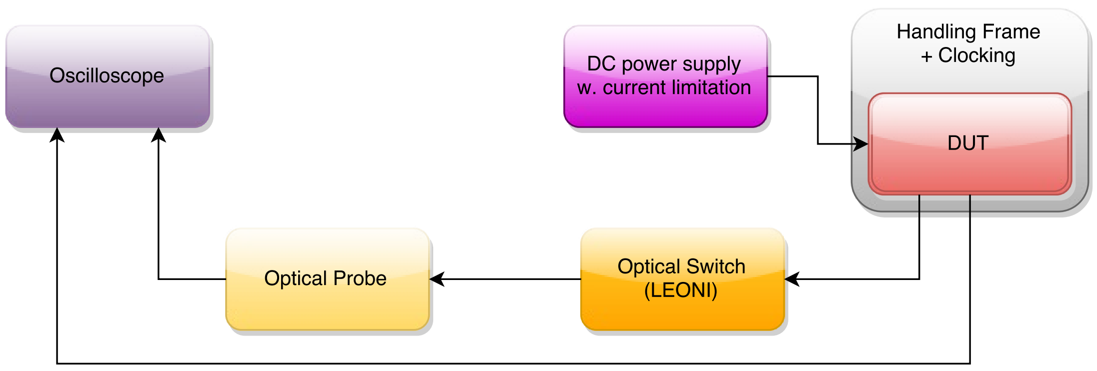


Figure 8.1: Overview of the production test setup. The device-under-test (DUT) is supplied by a DC power supply. All measurements are performed with an optical probe and an oscilloscope.

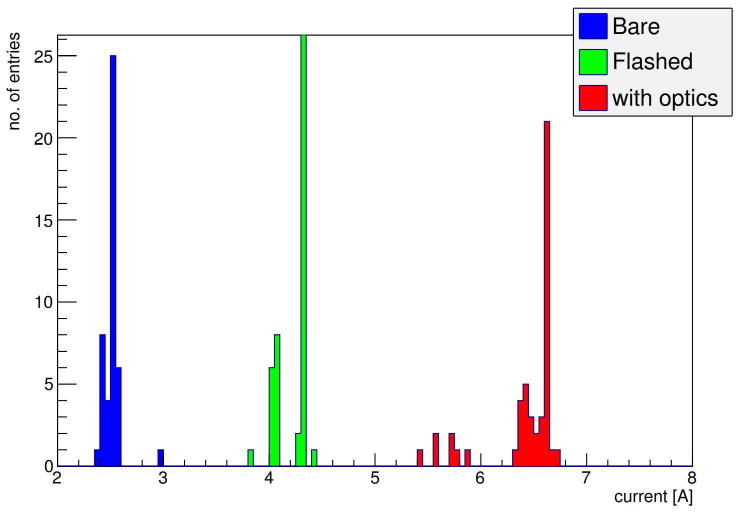


Figure 8.2: Histogram of the power consumption in the three different operation modes (bare without firmware, flashed with firmware, firmware and optical plugins).

Most important for the functionality is the bare current consumption without any firmware being loaded. It is a measure for the electrical integrity of the card. In addition, the power consumption strongly depends on the FPGA firmware. Therefore, the power consumption with loaded firmware has only limited informative value. Nearly all of the cards have a bare current consumption of about 2.5 A. From the testing of the first prototypes, the limit on the bare power consumption was set to 2.7 A. One card has a current consumption of nearly 3 A. The reason for the higher power consumption is not clear, as the card did not fail in any of the other production tests. This card has been separated and will not be used in the CERN setups. But, there are no arguments against the usage in a laboratory environment.

Additionally, the power consumptions with the firmware loaded (green histogram) and the optical plugins mounted (red histogram) were measured. The spread of the current consumption after loading the firmware can be explained by statistical fluctuations. These statistical fluctuations of about 100 mA already exist in the bare current consumption. After flashing the firmware they get amplified resulting in the double-peak histogram. The large spread in the power consumption with optics mounted on the card can be explained by two sources. In addition to the statistical fluctuations, there is a 1 A difference in current as the result of the laser interlock. If the lasers are turned off, the current consumption drops by about 1 A.

Overall, the current consumption does not show any surprises, except the single card with a high power consumption. This card will not be delivered to CERN, but can be used in laboratory environments.

8.4.2 Firmware loading & Network configuration

If the first basic tests are finished, the firmware will be installed to the card. This checks the connectivity between the FPGAs and the Ethernet connection is working. For programming the BCF a Xilinx JTAG programming cable will be used, whereas the BMF firmware is programmed using the regular in-system loader, which was described in Section 7.3.2.

As the firmware installation of the BMFs is performed over the Ethernet, this tests also the Ethernet connection of the BCF. The size of the BMF firmware images is around 9 MB and as the received image will be verified by calculating a CRC32 checksum this also is an indicator for Ethernet communication problems. Some of the cards in the Layer-1 production batch failed this test and had a non-working Ethernet connection or BCF memory problems. These issues will be described in more detail in Section 8.5.

After all FPGAs have loaded their firmware, the serial number can be read out and will be labeled to the PCB as an unique identifier for each card. Also the default IP address of the card is configured. Finally, the power consumption is measured again to check the electrical behaviour after firmware loading and after plugging in all optical plugins (see previous section).

8.4.3 Clock test

The clock test verifies the functionality of the clock distribution network (see Section 6.3.2). Therefore, the card is supplied with an external clock signal through the regular clock path. During this test the offset of the internal FPGA clock is measured against the input clock signal. This is an important parameter for the deterministic timing. It has to be stable and predictable in all situations. Also the behaviour of the global clock delay circuit is analysed during this test.

The results of the clock test are shown in Figure 8.3. The first plot in Figure 8.3a is analysing the behaviour of the clock circuit of a single card during many power-cycles. The other plot in Figure 8.3b is showing the behaviour between multiple cards. Both plots confirm that the clock circuit on the BOC card is working as expected. The offset of the clock signal of a single card during multiple power-cycles is (20.26 ± 0.038) ns. Also between multiple cards the behaviour is uniform with an offset of (20.25 ± 0.115) ns. This measurement is compatible with the measurements already seen during the first tests of the new clock circuit. With testing the fourth production batch, the statistics for this measurement will increase drastically. Both measurements also show that the slope of the global delay circuit is constant at about (10 ± 0.1) ps. This value matches the values specified in the datasheet of the delay chips [85].

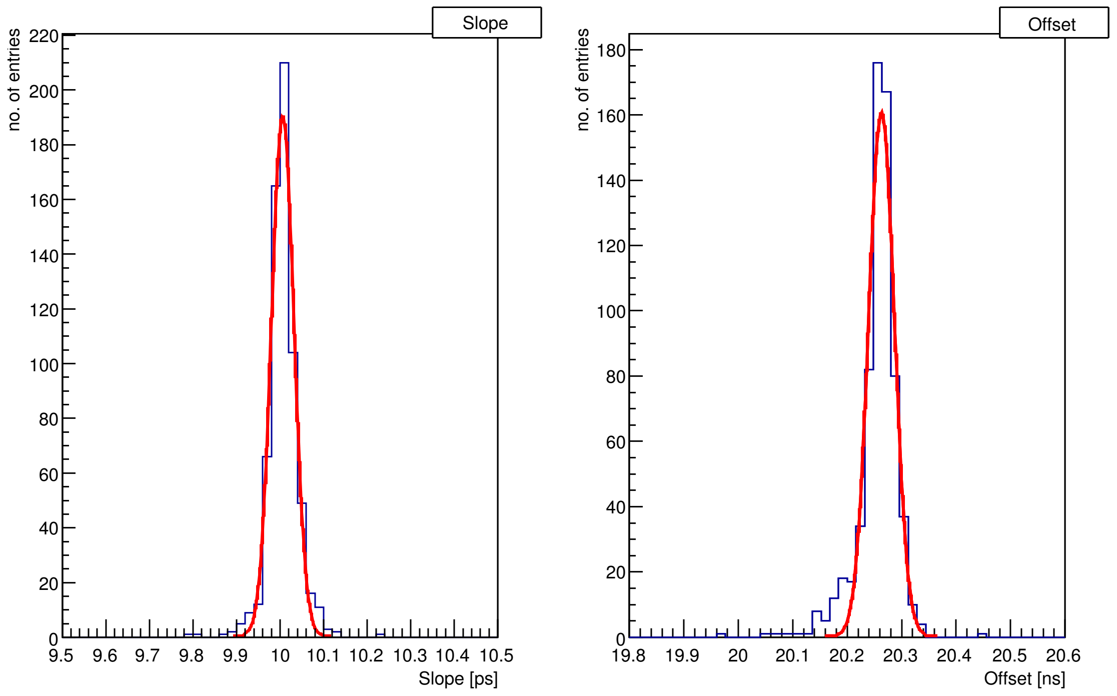
As a conclusion, this test confirms the expectations from the first tests shown in Section 6.3.2. Therefore, the clock distribution is fully deterministic and fulfils the requirements on the timing of the BOC card.

8.4.4 Loopback tests

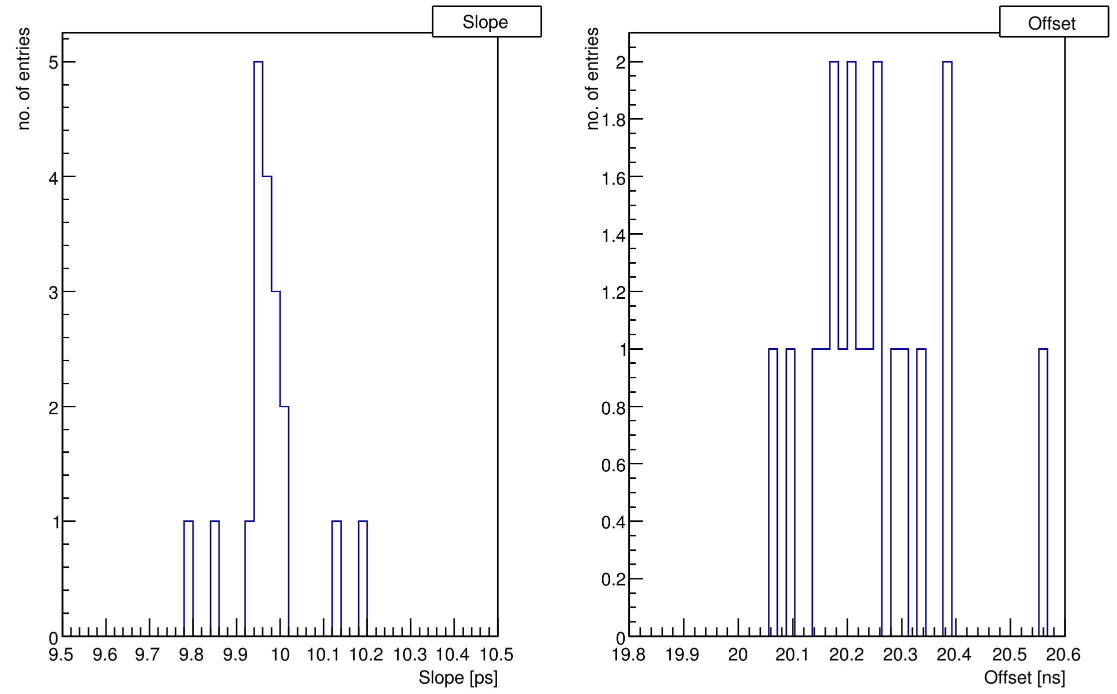
The next step tests the transmission and reception data path. This includes testing of the connection between the FPGA and the optical plugins. During this test, 8b10b encoded data at 160 MHz is generated in the transmission data path and then sent out through the optical interfaces. An optical loopback connection feeds the signal back into the receivers.

The first step of this loopback test is a pure functional test to ensure that all channels are basically working. The transmitters are sending IDLE words during this stage and it will be checked that all receiver channels can synchronize to the 8b10b encoding and no major transmission errors occur. When the synchronisation has finished, a short test pattern will be sent and compared. This test is also checking the functionality of the spare fibre input channels.

When the short loopback test has verified the basic functionality of the transmission path, also the bit error rate will be measured. In this test a pseudo-random pattern will be generated in the transmitters. In the receivers this pattern will be checked against the received data. From this a bit error rate can be measured by dividing the number of errors observed by the number of total bits received. The card is tested until a global (i.e. over all channels) bit error rate of 10^{-11} is reached. This typically takes about 20 s to complete. If an error in one of the channels is detected, the test will be marked as failed and the card has to undergo a manual inspection. During the production test of the first three batches no card failed this test.

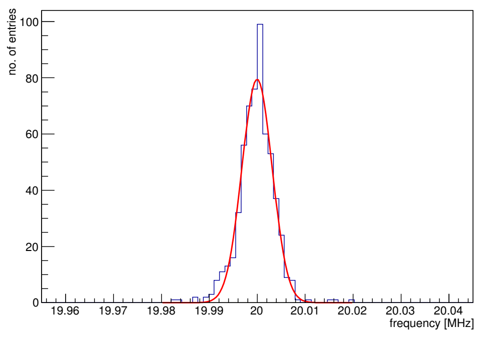


(a) single card, multiple power-cycles, slope = (10.01 ± 0.03) ps ($\chi^2/\text{ndf} = 3.37$), offset = (20.26 ± 0.02) ns ($\chi^2/\text{ndf} = 3.50$)

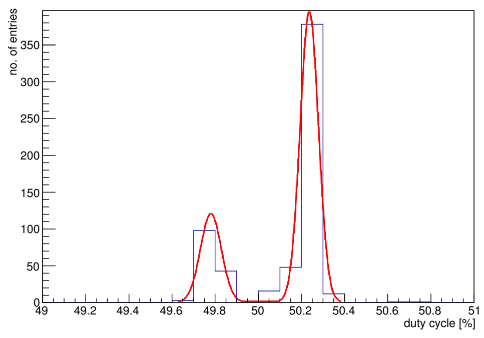


(b) multiple cards, , slope = (9.98 ± 0.08) ps, offset = (20.25 ± 0.12) ns

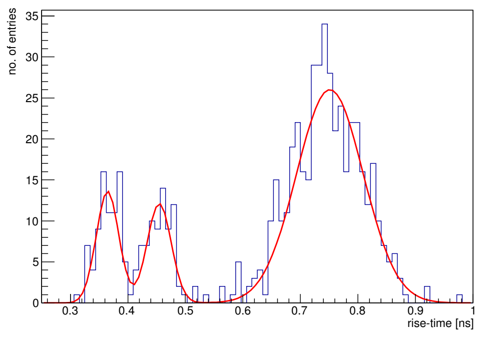
Figure 8.3: Measurement of the delay slope and offset of the global clock circuit.



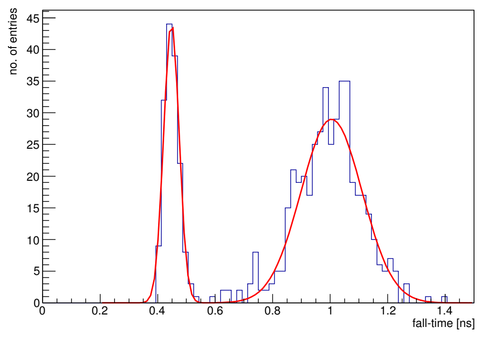
(a) frequency, peak at (20.000 ± 0.003) MHz, $\chi^2/\mathrm{ndf} = 1.35$



(b) duty-cycle, peaks at (49.78 ± 0.05) % and (50.24 ± 0.04) %, $\chi^2/\mathrm{ndf} = 5.98$



(c) rise-time, peaks at (0.365 ± 0.020) ns, (0.455 ± 0.021) ns and (0.751 ± 0.058) ns, $\chi^2/\mathrm{ndf} = 1.34$



(d) fall-time, peaks at (0.448 ± 0.026) ns and (1.005 ± 0.106) ns, $\chi^2/\mathrm{ndf} = 1.01$

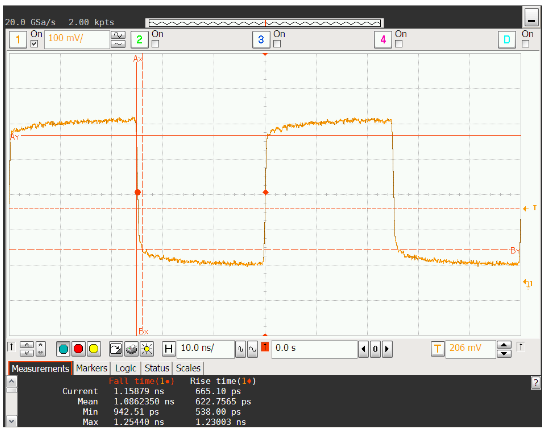
Figure 8.4: Measurements of the optical signal integrity during the production test.

8.4.5 Optical parameters

The transmission data path is also characterised in terms of optical parameters, like rise/fall-time, frequency, duty-cycle and the delay values for the coarse and fine delay in the TX path. These parameters are measured with an optical probe and an oscilloscope. The channel under test is selected by an optical switch. All measurements are fully automatised. The user only has to switch the fibres between the optical plugins. The results of the measurements for frequency, duty-cycle, rise- and fall-time of the optical signals are shown in Figure 8.4.

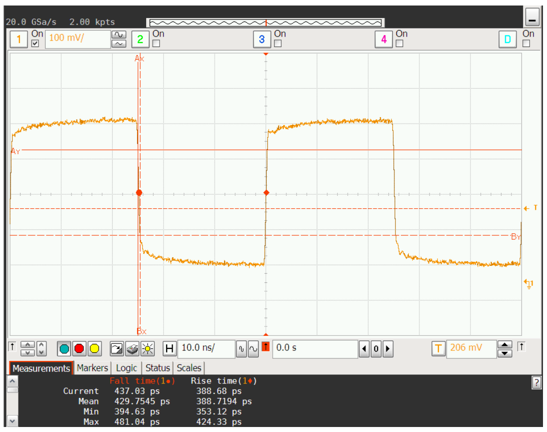
The frequency of the optical signals is (20 ± 0.004) MHz. During this test no data is transmitted. Therefore, the BPM encoding sends out only a 20 MHz clock signal (see description in Section 5.2.1). The width of the frequency distribution is in the range of the oscillator spread of the card. Therefore, all card have passed this test.

The duty-cycle of the optical signals is an important parameter for the optoboard. It requires the duty-cycle to be in the range between 46 and 54 %. During this measurement, no values outside this range have been observed. The histogram shows two peaks symmetric around the optimal duty-cycle value of 50 %. Most of the measurements have a duty-cycle of (50.2 ± 0.05) %. During the measurement of the fine delay, which will be shown later, the signals in the oscilloscope occasionally



```
              Fall time(1•)   Rise time(1♦)
     Current   1.15879 ns      665.10 ps
        Mean   1.0862350 ns    622.7565 ps
         Min   942.51 ps       538.00 ps
         Max   1.25440 ns      1.23003 ns
```

(a) 10-90 %



```
              Fall time(1•)   Rise time(1♦)
     Current   437.03 ps       388.68 ps
        Mean   429.7545 ps     388.7194 ps
         Min   394.63 ps       353.12 ps
         Max   481.04 ps       424.33 ps
```

(b) 20-80 %

Figure 8.5: Example of a rise- and fall-time measurement for the optical signals with different thresholds.

are getting inverted. The duty-cycle is mirrored at 50 % during this inversion. This explains the small peak at (49.8 ± 0.1) %. As all values are in the operating range of the optoboard, no cards failed in this test.

Unfortunately, during the measurement of the rise- and fall-time of the optical signals, odd effects shown up. It was required that the rise- and fall-time should be less than 1 ns. The first observation when analysing the measurement is that the fall-time exceeds this limit for some of the cards, and also the rise-time is close to that limit. In Figures 8.4c and 8.4d a multi-peak structure can be observed. After a detailed investigation of that issue, it turned out to be the threshold setting for the rise- and fall-time measurement in the oscilloscope, which had different settings for different measurements. The optical signal is shown in Figure 8.5. It has steep edges but a long tail when saturating. This is a known effect created by PiN diodes. The rise- and fall-time measurement in the oscilloscope takes two reference points and calculates the rise- or fall-time as the time distance between both points. The oscilloscope allows one to choose different points, either 10 % and 90 % (Figure 8.5a) of the signal amplitude, or 20 and 80 % (Figure 8.5b). Due to the long tail the threshold setting has large effects on the rise- and fall-time measurement. By changing the thresholds the rise-time is reduced by a factor of 1.7 and the fall-time is reduced by a factor of 2.6. Some of the rise- and fall-time measurements were performed using the 10-90 setting and some with the 20-80 setting and this explains the multi-peak distribution for the rise- and fall-time measurement. However, by correcting all measurements with the correction factors, they are all below the required 1 ns.

Another important parameter which needs to be verified during the production test is the coarse and fine delay implementation. This verification should ensure that all cards have a proper timing and the behaviour between the different cards is uniform. Especially, as the IODELAY2 primitives in the Spartan-6 are not calibrated, it needs to be tested that all cards behave in the same way.

The delay measurements were performed with an undelayed reference signal. Using the oscilloscope the time difference between the signal under test and the reference signal is measured. By applying different delay settings the slope can be measured. Also the duty-cycle is be monitored during the measurement. If the duty-cycle exceeds the operational range of the optoboard, the card will fail this test.

Figure 8.6 is showing the results of the delay measurements for the coarse and fine delay. In general it can be seen that both delay implementations are very uniform between all the channels and all the cards. Different to what has been described in the firmware chapter, the coarse delay step size is (6.25 ± 0.001) ns. The reason for the doubling of the step size is that the production test is running an old version of the BOC firmware, which has a different TX data path. More details about this old version and why it has been replaced will be given in Section 9.3.4. From the measurement the coarse delay step size is clearly confirmed, as the distribution of the coarse delay slope is narrow around the expected 6.25 ns per step.

The exact step size of the fine delay was not known until all cards had been tested. From prototyping tests it was clear that the implementation of the fine delay using the IODELAY2 primitive was working in general. But if all cards show the same behaviour in terms of the step size, could only be verified during the production test of the cards. The fine delay test was also originally designed to be a calibration of the card, where each channel on all the cards should be characterised and the individual step sizes for the fine delays are measured. When inspecting the results of the fine delay measurements, they were much better then expected before. All cards show a very uniform distribution in the histogram around (35 ± 0.7) ps per step. This result has shown that even if the IODELAY2 primitives are not calibrated the step size of the fine delay implementation can be assumed to be 35 ps per step for all cards and all channels. Also the requirement that the fine delay step size should be less than 100 ps per step is fulfilled by all tested cards. In addition, the duty-cycle has been monitored during the fine delay measurement and it did not exceed the operating range of the optoboard for all measurements.

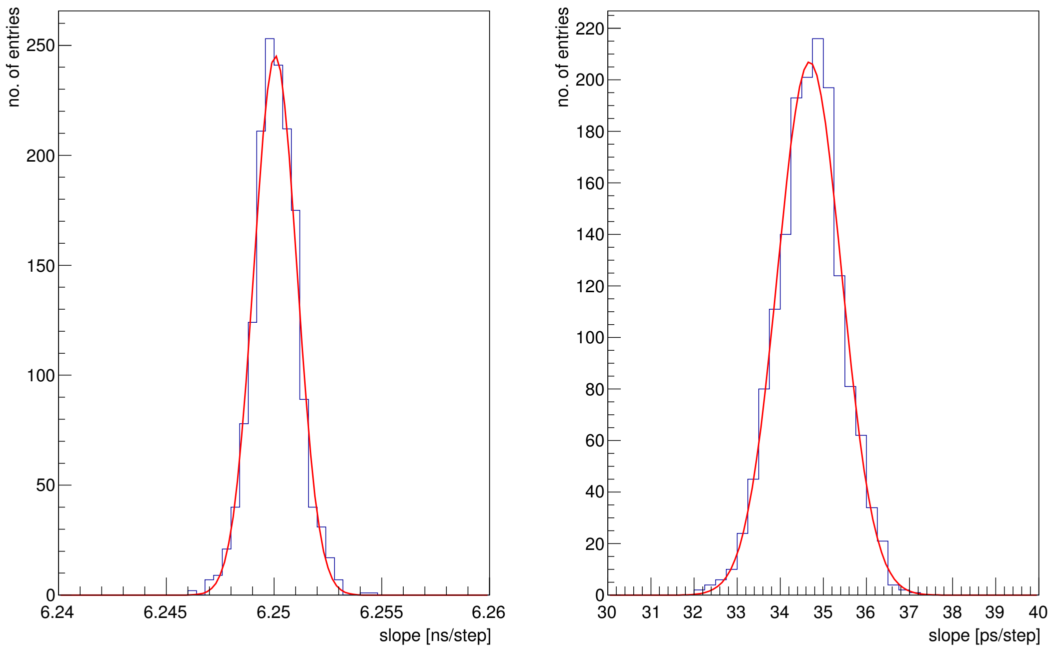
Concluding the results of the optical measurements it can be stated that all tested cards are fulfilling the requirements. Especially the delay measurement has shown very good results. The fine delay step size is about 35 ps per step and this improves the resolution by a factor of 10 compared to the old Pixel Detector readout system.

8.4.6 Backplane & S-LINK

When the signal integrity has been tested, a final test will be performed checking the VME backplane connections and the S-LINK interface. This test is performed in the VME crate and utilizes a special ROD firmware. This ROD firmware includes pattern generators and comparators for all important backplane signals. Usual test patterns are:

- all '1'/'0': testing for constant low/high signals
- walking '1'/'0': testing for shorted and swapped wires
- pseudo random numbers

The serial data lines for the TTC data are tested by with the all '1'/'0' and walking '1'/'0' patterns. The results of this test are viewed with internal Chipscope logic



(a) coarse delay, peak at (6.250 ± 0.001) ns, $\chi^2/\mathrm{ndf} = 1.65$

(b) fine delay, peak at (34.7 ± 0.7) ps, $\chi^2/\mathrm{ndf} = 0.89$

Figure 8.6: Measurement of the delay step size for the coarse and fine delay.

analysers in the BOC card. This test can find shorted or constant value signals, but does not perform a bit error rate measurement, as it is only a basic test that all serial lines are electrically ok. Figure 8.7 shows how the received test pattern looks like. The test operator has to compare this with the pattern sent out.

In the other direction, special test frames are generated in all RX channels to test the ROD data bus for the detector data. These frames contain a pseudo random byte pattern, which is received by the ROD through all 96 lines. The ROD then performs a bit error rate measurement of the incoming data. The measurement is presented to the operator also through the Chipscope logic analyser. Figure 8.8 shows the operator interface to this bit error rate measurement. Only if no errors

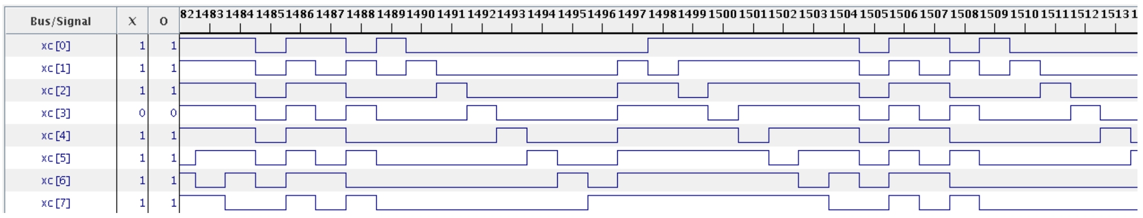
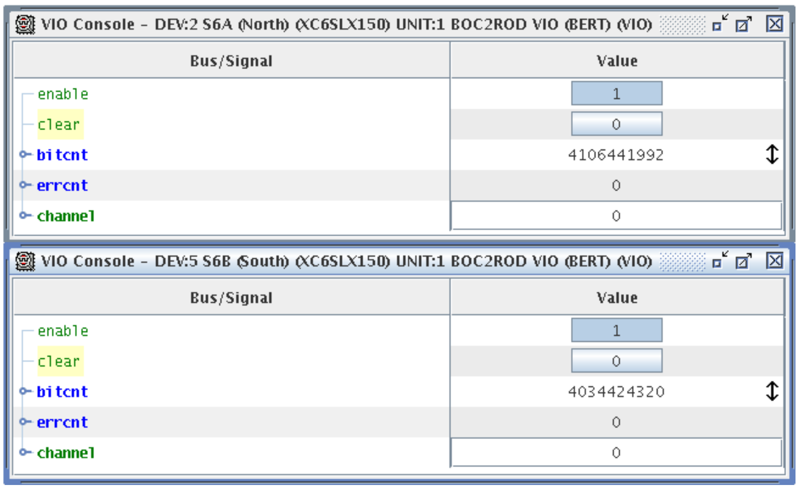


Figure 8.7: Chipscope interface with the test patterns received on the BOC card for a manual comparison. In this test, all 8 “XC” lines are connected and no shorts or swapped lines can be observed as the walking ’1’/’0’ patterns are good.



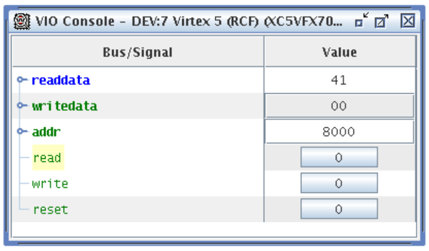
VIO Console – DEV:2 S6A (North) (XC6SLX150) UNIT:1 BOC2ROD VIO (BERT) (VIO)

Bus/Signal	Value
enable	1
clear	0
bitcnt	4106441992
errcnt	0
channel	0

VIO Console – DEV:5 S6B (South) (XC6SLX150) UNIT:1 BOC2ROD VIO (BERT) (VIO)

Bus/Signal	Value
enable	1
clear	0
bitcnt	4034424320
errcnt	0
channel	0

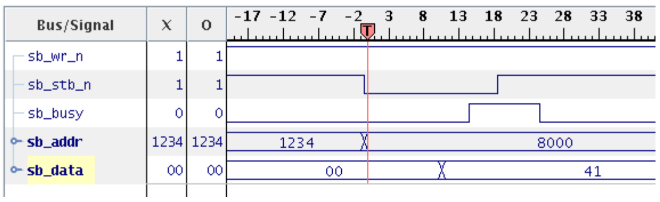
Figure 8.8: Chipscope interface to measure the bit error rate on the BOC-to-ROD interface



VIO Console – DEV:7 Virtex 5 (RCF) (XC5VFX70...

Bus/Signal	Value
readdata	41
writedata	00
addr	8000
read	0
write	0
reset	0

(a) user interface



Bus/Signal	X	O
sb_wr_n	1	1
sb_stb_n	1	1
sb_busy	0	0
sb_addr	1234	1234
sb_data	00	00

(b) waveform

Figure 8.9: Test of the Setup-Bus access during the production test of the BOC card

are observed, the card has passed this test. Otherwise, the card has to undergo a more detailed manual inspection.

The third important backplane interface, which has to be tested is the S-LINK interface. Again an all '1'/'0' and walking '1'/'0' test pattern is sent out through all four S-LINK interfaces on the backplane and then compared in the BOC card. The result of this comparison is an error flag, on which can be triggered in the Chipscope logic analyser. The error flag should remain always zero during the test time, which is in the order of some minutes. If the trigger is activated, this means the error flag was raised the test is failed and the card has to undergo the manual inspection.

Also the Setup-Bus connection is being verified in this test. By reading or writing registers and compare them to the expected value, it can be tested if the control interface between ROD and BOC card is working fine. Figure 8.9 shows a read-cycle for one of the BMF firmware version registers together with the corresponding waveform on the VME backplane.

When all backplane interfaces have been verified to be working, the last test the card has to pass is the S-LINK communication test. For this test the card is connected to a ROS-PC equipped with a ROBIN card. With this test a full S-LINK chain containing ROD, BOC card and ROBIN is established. First, it is checked that a stable connection can be established on all eight S-LINK connections. A stable connection is being reported by both sides with the “link down” status signal. This signal has to show the link up condition on both sides for all eight links. When all eight links are up a fragment generator is being activated in the ROD. That means the ROD is sending example event fragments through the BOC card to the ROBIN card. Shortly after activating the test, the S-LINK flow control should stop the sending and both sides should report that. Finally, the received fragments in the ROBIN card are being analysed and it is checked that no corrupted fragments are reported.

The backplane and S-LINK check has shown no errors on all cards, and so these cards have passed the complete production test and are ready to be sent to CERN.

8.4.7 Production test database

All measurement taken during the production test are written to a database and can be accessed for a detailed analysis. The database is hosted on a MySQL server and it can be accessed through a web interface. The measurements and analyses shown in this chapter are based on the dataset of the first three production batches.

8.5 Failures detected during the production test

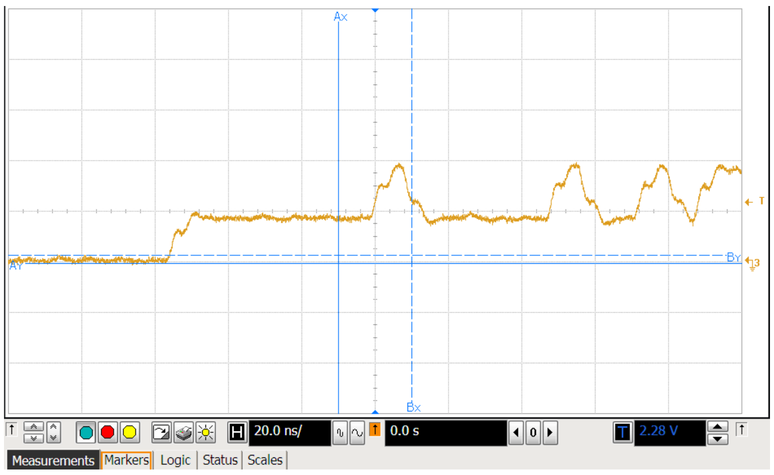
During the production test also some failures of cards were detected. This section will describe the two most important failure types and what has been done to repair these failures.

8.5.1 BCF memory problems

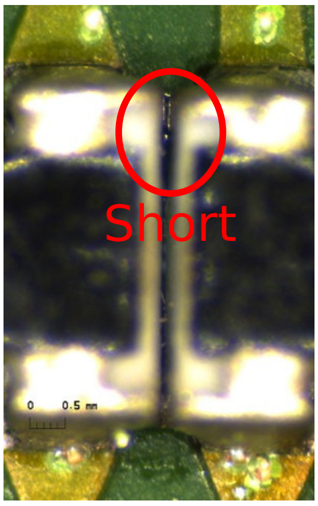
The testing of the cards unveiled some cards with serious memory problems in the BCF memory. The BCF memory is an essential part for the card operation. At the boot up of the card a memory test is performed and this memory test failed on some of the cards. During the failure analysis, two reasons for the memory failure have been detected.

For some cards the memory failure was the result of a not working voltage regulator. One of the regulators generating the supply voltage for the memory was not working properly on eight cards. When applying this voltage from an external power supply the cards operated normally. These cards were sent back to the manufacturer and the voltage regulators have been replaced. After the replacement all cards work as expected.

Another type of memory failure is more difficult to debug. The memory test of the BCF memory checks for data lines and address lines being shorted, open or constant on high or low. But none of these errors have been detected. One possibility is that one or more control signals between the memory and its controller interface are not working properly. From the software and firmware point of view these control signals are hidden behind the memory controller. Also they cannot be probed with an oscilloscope, because both chips are ball-grid-arrays and the pads are below the package. The only strategy now would be to replace components, in case of a serious



(a) intermediate signal level



(b) shorted wire

Figure 8.10: Reason for Ethernet problems on the BOC card in the fourth production batch

component failure. This has happened for two cards of the third production test, where the memory chip has been replaced by a new one. Unfortunately, this did not solve the memory problems. The next steps now will be a replacement of the FPGA which has been avoided until now due to cost issues. During the replacement of the components it is also foreseen to measure the resistivity of all PCB wires between both chips to check for open connections. This is only possible when both components are unmounted from the PCB. In summary, the reason for these memory problems is not yet clear and therefore no repair strategy is available.

8.5.2 Ethernet problems

During testing the Layer-1 batch of the cards, a large number of cards (8) failed the production test with Ethernet problems. What could be observed is that all the cards boot up correctly and also link and activity signals on the Ethernet chip are available, but no Ethernet connection could be established. Checking the signal levels between the FPGA and the Ethernet chip has shown that some of the signals have a strange intermediate level on some data lines from the Ethernet chip to the FPGA. Figure 8.10a shows a oscilloscope picture of this effect. After some investigation a short between two signals has been detected. Figure 8.10b shows a picture of the shorted signal taken with a microscope. After removing the short all Ethernet problems have been resolved.

8.6 Summary of the production test

To summarize the results of the production test for about 50 cards, including the prototype cards, most of the cards did pass the production test without any issues observed. The first 15 cards and additional cards in laboratory setups have been delivered and installed in the IBL readout setup. Up to today no hardware issues with these cards have been observed. The installation of another 26 cards for the Layer-2 readout is expected in the beginning of 2016.

During the production test also a lot of characterisation tests have been performed to better understand the behaviour of the cards. This includes the tests of the optical signal integrity, especially the measurements for the delay slope of the fine delay. Having a larger set of tested cards in the background the fine delay stepsize showed a very uniform distribution around 35 ps per step over all channels on all tested cards. This was not an expected results, as the FPGA manufacturer does not guarantee a certain calibration of the stepsize. The stepsize for the coarse delay implementation fitted also the expectations from the firmware development by having a very narrow distribution around the expected 6.25 ns per step. Even though this is not the final firmware version and the new firmware has a different implementation, the general behaviour should not change in this fully clock synchronous design.

The analysis of the production test also unveiled a general problem in the measurements of the rise- and fall-time. Due to different settings in the oscilloscope in different measurements the rise- and fall-time do not meet the requirement of being less than 1 ns in all cases. Further analysis showed, that the threshold setting is an important parameter when measuring optical signals received by PiN diodes, because of a long tail in the signal shape, even when the edges are clear. Using the correct setting of the measurements all cards can pass this test and have a rise- and fall-time meeting the requirements.

Finally, also some defect cards have been observed during the production test. Most of the cards had some soldering problems in voltage regulators or missing components, which could be fixed easily, but some cards had major issues. During the last production batch a major Ethernet problem showed up, which had its origin in a shorted signal due to solder brigdes, which need to be repaired by the manufacturer. After the repair the cards are expected to be working. Another major issue in the BCF memory is not yet understood completely and requires more analysis. The next step will be a replacement of all involved components (the memory chip and the FPGA) to see if that can solve the problems.

To conclude, all produced cards have been tested intensively to ensure the good quality when delivering the cards. The testing of the fourth batch dedicated for the Layer-1 readout is expected to start this summer after the complete delivery of the cards.

9 The IBL BOC card in the system test

9.1 Introduction

In the previous chapter the production test of the IBL BOC card has been presented. It ensures that all cards meet the specifications. Another very important test is the behaviour of the card in the full readout system.

The main motivation of this test is the interaction of all the components in the readout chain, also containing the real detector in the real environment. This gives the possibility to find issues which do not occur in a laboratory setup. Also it is the first and only possibility to test production versions of all components.

This chapter will describe the tests, which affected the BOC card in the system test. The testing itself is split into two parts. The basic communication as well as calibrating single front-end chips can be performed in a small local laboratory system in Wuppertal. Tests with the full detector and the data taking can only be done at CERN. There the readout system has been tested during the commissioning of the detector.

9.2 The Lab infrastructure

9.2.1 Overview

The readout setup in Wuppertal allows testing the cards on a low-level basis as well as using the high-level readout software, which is used at CERN. From the hardware point of view the Wuppertal setup is equipped with two ROD/BOC card pairs. Attached to these card pairs is an optical chain using the standard IBL optoboard and some detector modules. The higher-level readout system is being provided by two ROS-PCs. Missing in the Wuppertal readout setup is a TIM, so no data taking tests are possible. Figure 9.1 shows an image of the readout setup.

9.2.2 Front-end chips

The front-end chips, which are used in the system test setup in Wuppertal are FE-I4 single- and double-chip modules with sensors attached. An adapter PCB is being used to connect the data signals and supplying the low and high voltage to the chips. Figure 9.2 shows an IBL double-chip module with the adapter PCB. The RJ45 connectors are compatible with the ones on the optoboard adapter PCB and so a regular Ethernet cable can be used for this connection.

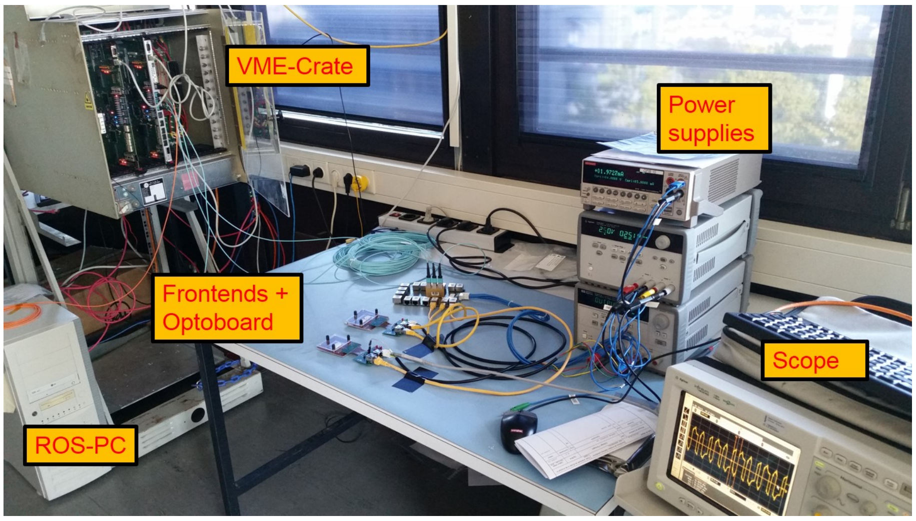


Figure 9.1: Picture of the IBL readout setup in Wuppertal

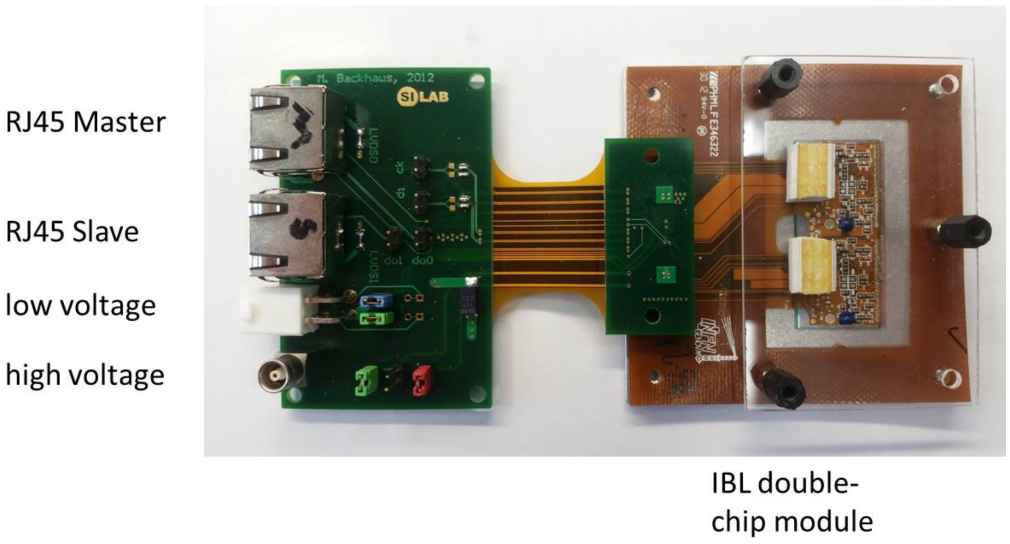


Figure 9.2: Picture of an IBL double-chip module with its adapter PCB

Table 9.1: Mapping of the RJ45 jacks on the optoboard adapter PCB to TX and RX channels on the BOC card

	BOC channel		**Optoboard-**
RJ45	**TX**	**RX**	**VCSEL**
M1	0	0	A
S1	0	2	A
M2	1	4	A
S2	1	6	A
M3	2	8	B
S3	2	10	B
M4	3	12	B
S4	3	14	B
M5	4	1	A
S5	4	3	A
M6	5	5	A
S6	5	7	A
M7	6	9	B
S7	6	11	B
M8	7	13	B
S8	7	15	B

The powering of the front-end chips is done using standard power supplies. As the front-end chips have their own shunting regulators for analog and digital voltage, they only have to be powered by a single voltage around 2 V with a current limitation of 1 A. For the high voltage power supply of the pixel sensor a HV sourcemeter was used to provide about 80 V bias voltage with a current limit of 8-10 μA.

9.2.3 Optoboard & adapter PCB

To integrate the IBL optoboard into the readout setup, an adapter PCB has been used. This adapter PCB carries the optoboard with its 100-pin hermaphrodite connector and gives access to the data lines through standard RJ45 connectors. These RJ45 connectors are used to connect to the front-end chips. Two of them are paired, whereas the master carries clock, command and data signals, and the slave only carries data signals. One RJ45 pair corresponds to one IBL module being connected. The adapter board allows 8 IBL modules to be connected. Table 9.1 shows the connectivity between the RJ45 connectors and the optical inputs on the BOC card. As this mapping does not match the connectivity of the CERN setup, where one IBL module is connected to two neighbouring RX channels, the mapping has to be adapted by the crossbar switch, which was described in Section 7.6.3.

Table 9.2: List of the RODs and BOC cards in the Wuppertal readout setup.

ROD name	VME slot	type	version	ROD-IP	BOC version	IP
ROD_C1_S8	8	dummy				
ROD_C1_S11	11	IBL ROD	revC	192.168.1.110	revD	192.168.0.76
ROD_C1_S15	15	IBL ROD	revD	192.168.1.150	revD	192.168.0.83

9.2.4 ROD/BOC readout hardware

The ROD/BOC readout hardware in the lab setup consists of two card pairs hosted in a VME crate. Table 9.2 shows a list of all RODs in the readout system. In the system test the card pair in slot 15 are used, as they are both the final card versions. The ROD is also connected to a PC using a USB serial port and a Xilinx JTAG programming cable. The UART output from the ROD PowerPC is being logged into a file and the programming cable can be used for reloading the ROD firmware.

9.2.5 ROS PCs

For testing also the connection to the higher-level readout two ROS-PCs are available in the readout setup. One ROS-PC is equipped with an old version of the ROBIN card. It allows testing the S-LINK connection between the BOC card and the ROS system. It was already heavily used during the S-LINK test of the production test.

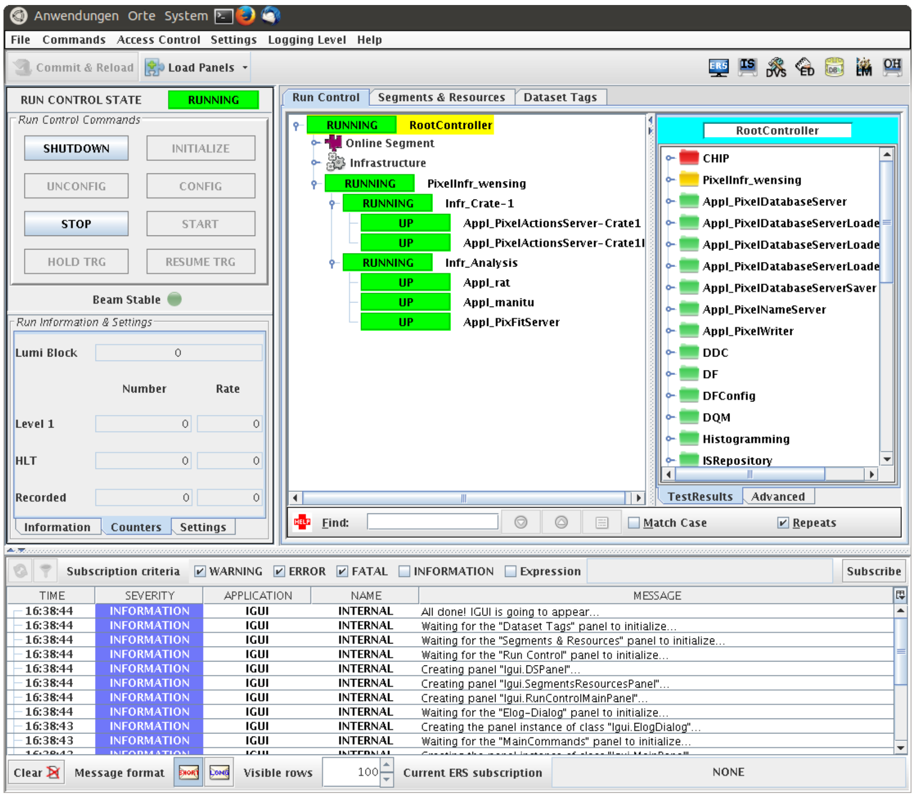
After the observation of S-LINK errors, which will be described later in Section 9.3.3, the local readout setup has been extended with a new version of the ROBIN card. It has the same version, as being used at CERN for the IBL readout and allows testing the cards using the real IBL ROS system.

Currently both ROS setups are not integrated into the software infrastructure and therefore can only be used with the low-level tools in a standalone operation. It is foreseen to integrate them into the system for testing data taking in the local setup as well.

9.2.6 Software Infrastructure

The software infrastructure for the readout setup consists of two main parts. The basis of the software is built by the TDAQ Run-Control software. TDAQ is a software framework, which hosts all the detector specific software packages and common utilities and libraries for all detectors. Figure 9.3 shows a screenshot of the main TDAQ user interface.

On top of the TDAQ Run-Control software the specific software for the Pixel detector is running. It is split into several parts. The central part is a software library called “PixLib”. On top of PixLib the Calibration Console and the Calibration Analysis packages are running. A screenshot of the Calibration Console is shown in Figure 9.4. It is the main software interface to the detector during calibration. Scan procedures can be executed from the Console and the results are also presented.

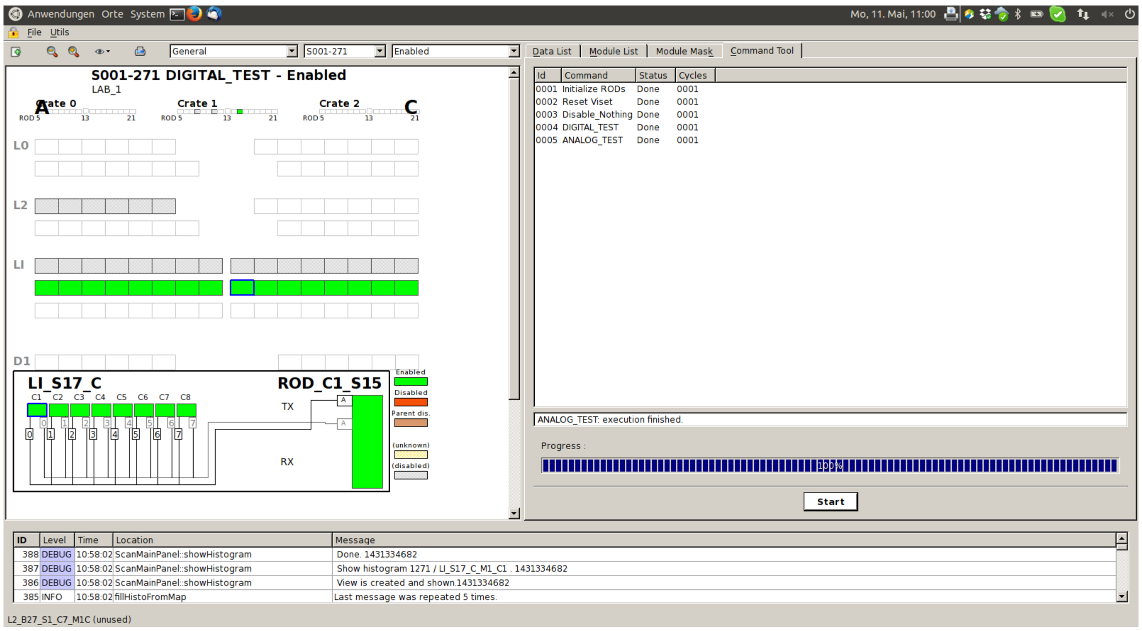


```
RUNNING  RootController
  Online Segment
  Infrastructure
  RUNNING  PixelInfr_wensing
    RUNNING  Infr_Crate-1
      UP  Appl_PixelActionsServer-Crate1
      UP  Appl_PixelActionsServer-Crate1I
    RUNNING  Infr_Analysis
      UP  Appl_rat
      UP  Appl_manitu
      UP  Appl_PixFitServer
```

```
RootController
CHIP
PixelInfr_wensing
Appl_PixelDatabaseServer
Appl_PixelDatabaseServerLoade
Appl_PixelDatabaseServerLoade
Appl_PixelDatabaseServerLoade
Appl_PixelDatabaseServerSaver
Appl_PixelNameServer
Appl_PixelWriter
DDC
DF
DFConfig
DQM
Histogramming
ISRepository
```

TIME	SEVERITY	APPLICATION	NAME	MESSAGE
16:38:44	INFORMATION	IGUI	INTERNAL	All done! IGUI is going to appear...
16:38:44	INFORMATION	IGUI	INTERNAL	Waiting for the "Dataset Tags" panel to initialize...
16:38:44	INFORMATION	IGUI	INTERNAL	Waiting for the "Segments & Resources" panel to initialize...
16:38:44	INFORMATION	IGUI	INTERNAL	Waiting for the "Run Control" panel to initialize...
16:38:44	INFORMATION	IGUI	INTERNAL	Creating panel "Igui.DSPanel"...
16:38:44	INFORMATION	IGUI	INTERNAL	Creating panel "Igui.SegmentsResourcesPanel"...
16:38:44	INFORMATION	IGUI	INTERNAL	Creating panel "Igui.RunControlMainPanel"...
16:38:44	INFORMATION	IGUI	INTERNAL	Waiting for the "Elog-Dialog" panel to initialize...
16:38:43	INFORMATION	IGUI	INTERNAL	Creating the panel instance of class "Igui.ElogDialog"...
16:38:43	INFORMATION	IGUI	INTERNAL	Waiting for the "MainCommands" panel to initialize...

Figure 9.3: Screenshot of the TDAQ software when running the local lab software infrastructure



Id	Command	Status	Cycles
0001	Initialize RODs	Done	0001
0002	Reset Viset	Done	0001
0003	Disable_Nothing	Done	0001
0004	DIGITAL_TEST	Done	0001
0005	ANALOG_TEST	Done	0001

ANALOG_TEST: execution finished.

ID	Level	Time	Location	Message
388	DEBUG	10:58:02	ScanMainPanel::showHistogram	Done. 1431334682
387	DEBUG	10:58:02	ScanMainPanel::showHistogram	Show histogram 1271 / LI_S17_C_M1_C1 . 1431334682
386	DEBUG	10:58:02	ScanMainPanel::showHistogram	View is created and shown.1431334682
385	INFO	10:58:02	fillHistoFromMap	Last message was repeated 5 times.

L2_B27_S1_C7_M1C (unused)

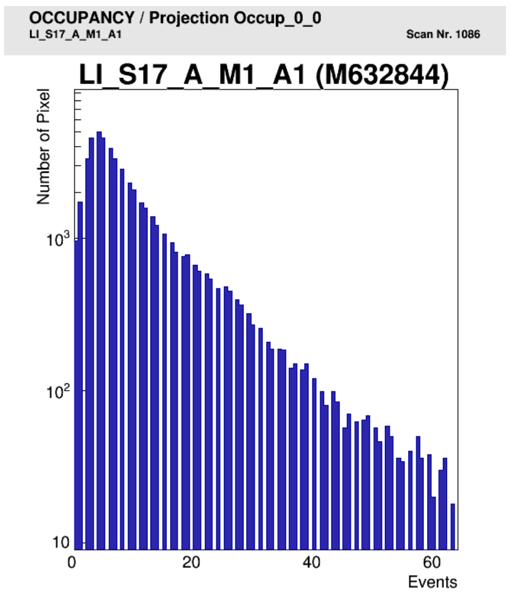
Figure 9.4: Screenshot of the calibration console window

PixLib is also used for hosting the connectivity database for the DAQ. In order to use PixLib the database has to be set up, including a mapping of all the available ROD/BOC pairs with their input and output links to the detector modules. For the Wuppertal readout setup, the three RODs from Table 9.2 have been implemented into the connectivity database. Each ROD is connected to a full stave. By using the front-end emulator in the BOC card, the readout for this full stave can be tested.

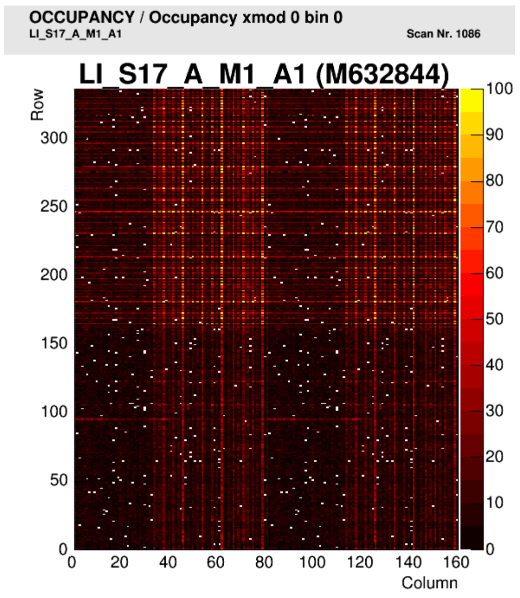
9.2.7 Scanning in the local infrastructure

Emulator

The first scans in the local infrastructure were performed using the front-end emulator in the BOC card. The front-end emulator returns random hits for every received trigger. These hits are collected into a hit occupancy histogram in the ROD. With the emulator it is possible, to perform scans without having real front-ends connected to the BOC card. Therefore, also the readout of a full IBL stave can be tested. Figure 9.5 shows the result of an analog injection scan for one DAQ module consisting of two emulated front-end chips. The occupancy histogram shows the same behaviour in the hit distribution as already observed during the testing of the emulator described in Section 7.7. Therefore, it can be concluded that the full DAQ readout chain is working.



(a) 1D projection



(b) Raw histogram

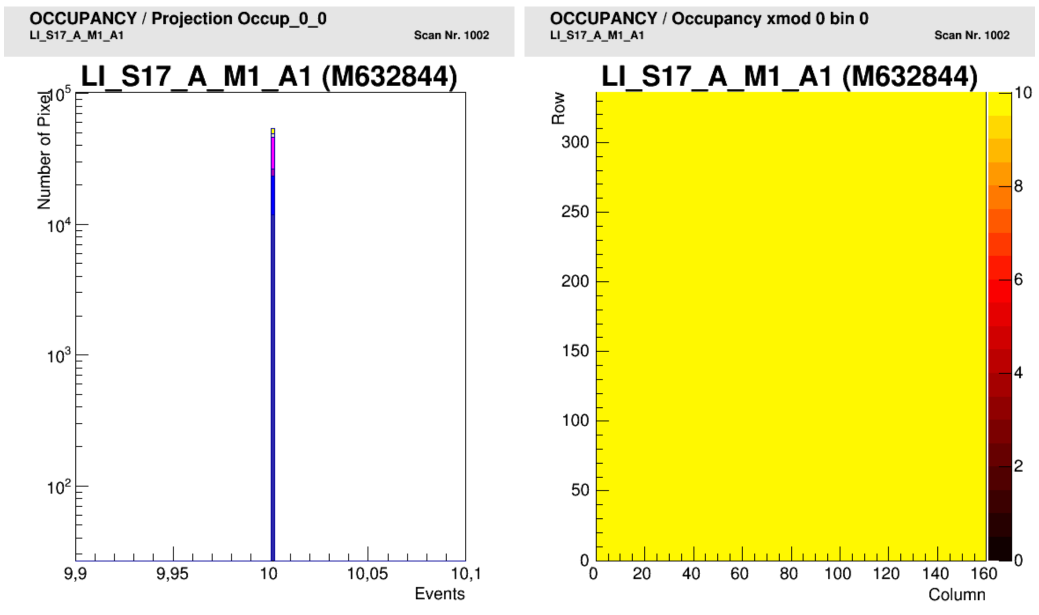
Figure 9.5: Occupancy histogram of a scan with the emulator in the BOC card.

Real front-end chip

After verifying that the DAQ readout chain is operational, the setup was extended with real front-end chips. The front-end chips are connected through the optoboard and optical fibres to the BOC card. This conforms to the setup used at CERN.

The first test with a real front-end chip was a digital injection scan. This scan is a test for all the digital logic in the readout chain. During the scan procedure, all pixels will be enabled and a hit is being injected into the digital region of the pixel. This hit is then being detected by the front-end chip and can be read out with the ROD/BOC system. Figure 9.6 shows the result of a perfect digital injection scan. The scan injected 10 hits into the digital region of each pixel, which is confirmed by the 1D projection. It shows that all pixels in the double chip module have seen exactly 10 hits.

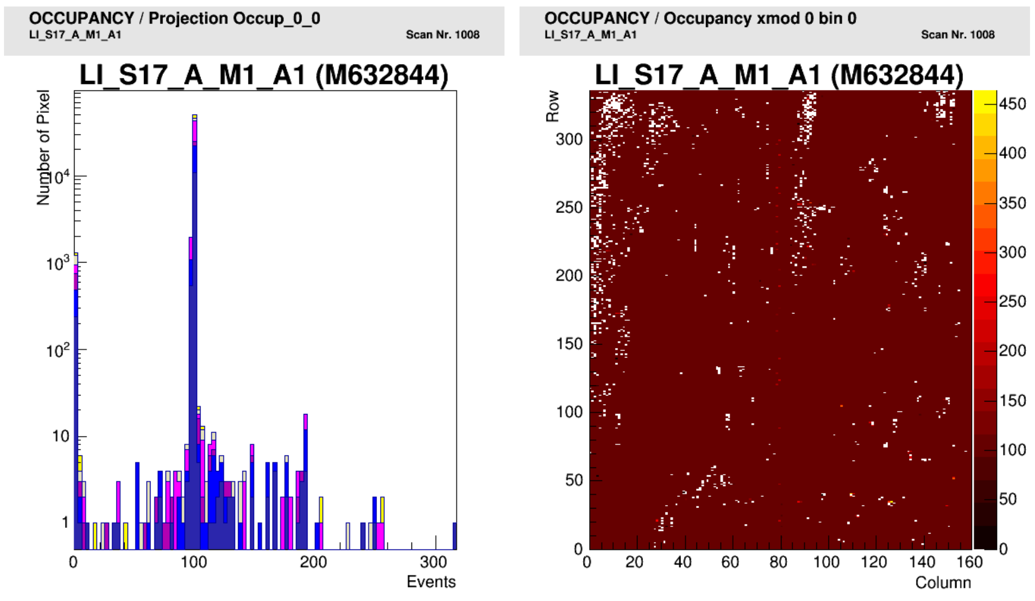
The same can be seen in the colour grade of the occupancy histogram. That means both front-end chips on this module no digitally bad pixels have been found.



(a) 1D projection **(b)** Occupancy histogram

Figure 9.6: Digital scan result using a real front-end chip connected through the optical chain to the BOC card

Having confirmed that the front-end chip and the readout chain is working, more advanced calibration scans can be performed. To test the analog region of each pixel cell, an analog injection scan can be performed. This scan uses a set of capacitors in each pixel cell. They can be charged up to a certain voltage. The charge stored on these capacitors can then be injected into the analog part of the pixel cell. During the scan, each pixel is injected 100 times. An ideal front-end chip would therefore show 100 hits for each pixel. Figure 9.7 shows the result of an analog injection scan with a real front-end chip. The occupancy histogram shows 100 hits for most of the pixels, but there are many pixels without hits (white spots). Also noisy pixels with more than 100 hits can be seen (red/yellow pixels). Several effects cause this behaviour. The front-end chips used in the lab setup do not have the same quality as the chips in the detector. Also, they are not optimally tuned and their configuration is a default configuration. Therefore, it is expected to not have an ideal result. It is foreseen to perform a tuning of the front-end configuration to improve this behaviour.



(a) 1D projection **(b)** Occupancy histogram

Figure 9.7: Analog injection scan result using a real front-end chip connected through the optical chain to the BOC card

Conclusion

In summary, all performed tests have shown good results. No instabilities have been observed and the interaction between ROD, BOC card, front-end chips and the software infrastructure has been tested with calibration scans. These results give a good basis for testing the BOC card in the readout setup at CERN.

9.3 System test at CERN

9.3.1 Overview of the SR1 and Pit setup

Having verified the basic functionality in the system test setup in Wuppertal, the tests could be extended to the test setups at CERN which are even more relevant, because they are the only test setups having the final versions of all hardware components. The readout setup at CERN splits up into two independent systems. The real detector in the cavern is commonly referred to as “Pit setup”. This setup should not be used for development tests to not affect the physics data taking. However, during commissioning of the detector without physics data taking, it was also used for verifying developments. The main development setup at CERN is placed in SR1, which is the main testing area for the ATLAS Detector. It is equipped with two

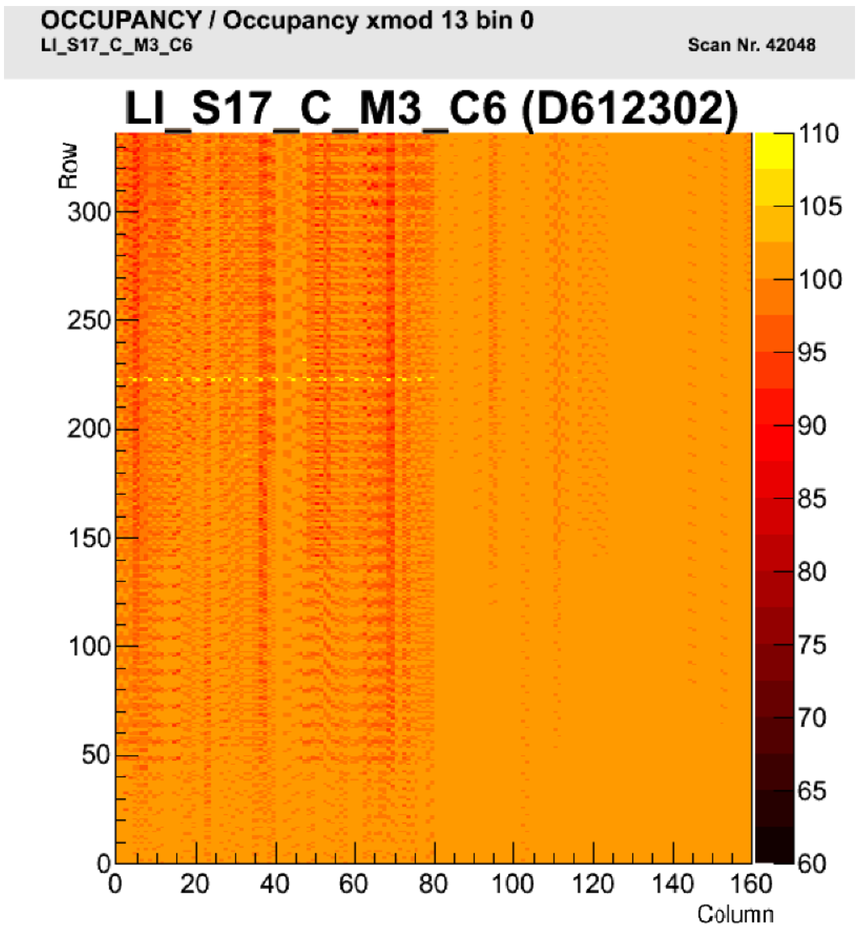


Figure 9.8: Picture of a corrupted digital injection scan in SR1 with many missing (red) or additional (yellow) hits.

IBL staves and the corresponding readout hardware. Usually, all developments are tested there, before they are deployed to the Pit.

The next sections will describe, how the testing of the detector has been performed and what problems with the BOC card have shown up during these tests. Many tests had also a large impact on the firmware design and these changes will also be presented.

9.3.2 FE-I4 corruption issue

During testing of the detector in the SR1 and also in the Pit readout setups, signs of data corruption between the front-end chip and the ROD have been observed. These digital injection scans showed corrupted pixels. This has not been observed in the local readout setup. The corruption can be seen in the occupancy histogram of a digital scan (see Figure 9.8). During the scan 100 hits for every pixel were injected. But in the occupancy histogram many pixels with less hits (red) and some with more hits (yellow) can be observed. Both front-end chips in this scan are flagged as digitally good so one would expect a clean histogram like it was shown already in Figure 9.6.

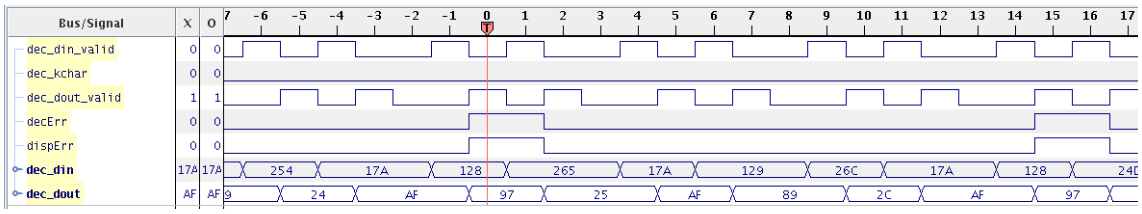
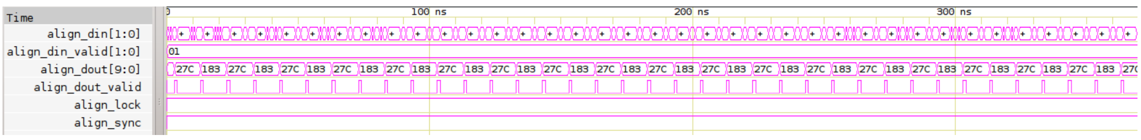


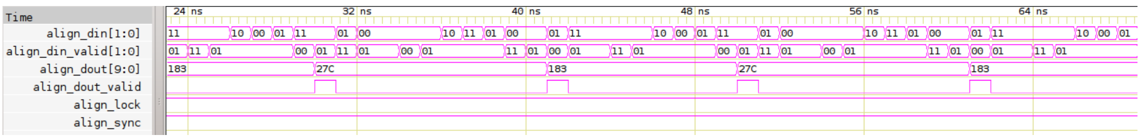
Figure 9.9: Waveform of the 8b10b decoder input and output signals with signs of FE-I4 data corruption as the decoding (decErr) and disparity (dispErr) error flags are being raised by the decoder as 0x128 is not a valid 8b10b symbol.

To check in which part of the readout chain the corruption is happening, more debugging was performed including some low-level checks in the BOC card. The RX data path allows the monitoring of all important signals of a readout channel using the FPGA-internal Chipscope logic analyser. With this logic analyser core the problem was investigated further. Figure 9.9 shows internal signals on an RX channel with signs of corruption, because decoding and disparity error flags are being raised. These decoding errors lead to a corruption of row and column information in the data and therefore the occupancy histogram is not filled properly. That is the reason why the occupancy histogram of this channel shows missing and additional hits.

To find the source of the corruption, the signals of the word alignment and clock and data recovery unit in front of the 8b10b decoder were analysed. They build the first stage of data reception in the firmware. Figure 9.10 shows the comparison of a good channel without corruption, and a bad channel with corruption. Both channels see valid 8b10b idle words (0x27C and 0x183). That means the communication in general is working. Different between the channels is the input valid signal to the word alignment (“align_din_valid”). This signal is generated by the clock and data recovery unit.



(a) good channel, the clock and data recovery output “align_din_valid” is stable



(b) bad channel, the clock and data recovery output “align_din_valid” is fluctuating

Figure 9.10: Comparison of the input and output signals of the word alignment. Both channel show good 8b10b idle words (“align_dout” is 0x27C or 0x183), but instabilities of the clock and data recovery can be observed in an unstable “align_din_valid” signal.

To understand what happens in the waveform of the bad channel, a closer look on the former version of the clock and data recovery unit has to be taken. It is based on the Xilinx Application Note XAPP224 [86]. Like the current version described in Section 7.6.2, it oversamples the incoming data signal with an oversampling rate of 4 and performs an edge-detection. But, different to the current version, it is automatically adjusting the sampling point based on the edge-detection and constantly tracking the position of the optimal sampling point. This is important if the sender frequency and the receiver frequency are not fully equal to each other. In this case the ideal sampling point is moving and sometimes the clock and data recovery mechanism does not output one bit in each clock cycle, but 0 or 2 to match the data rate.

In the waveform of the good channel (Figure 9.10a) the clock and data recovery mechanism generates one bit in each clock cycle. That means the data rate is fixed and the clock and data recovery mechanism has locked to it. The bad channel is constantly moving the optimal sampling phase. This can be observed by looking on the "align_din_valid" signal. It is constantly switching between 0, 1 and 2 bits valid in each clock cycle. Even if the output of the word alignment does not show any corruption, this is a clear indication of an instability in the data stream.

This observation lead to some changes in the BOC firmware. First, the clock and data recovery algorithm was changed. Instead of constantly tracking the phase, the sampling point will be chosen during a training phase and is kept fixed afterwards. A description of this algorithm was already given in Section 7.6.2. Also the word alignment was changed slightly. Instead of only detecting single k-words, it was adapted to detect only sequences of two consecutive k-words. This makes the word alignment more robust to data corruption resulting in k-words.

These firmware changes have reduced the rate of the corruption to some extent, but not all of the corruption could be cured by these changes. For further debugging the readout of the current sampling phase and a manual overwrite mechanism has been implemented into the firmware. Having these monitoring and control features available in the firmware, the debugging capabilites improved a lot.

With the new debugging tools it was observed that starting a new training sequence for channels showing corruption cured the corruption in some cases. In these cases also the sampling point moved to a more stable value and no decoding or disparity errors were observed anymore. But not for all channels a training sequence cured the corruption. In these cases adjusting the sampling point manually could also solve the problem. The result of this test is that the algorithm choosing the sampling point is not working correctly in all possible conditions. To improve the behaviour of this algorithm, it is foreseen to increase the number of sampling points per bit from four to six. This will increase the granularity of the sampling points and might stabilize the algorithm.

In case of corruption during the data taking, also some DAQ actions will be implemented into the monitoring software. If a channel showing corruption is identified, a new phase training will be started. In case that does not cure the corruption, a manual phase setting will be applied to this channel. Another workaround is a scan over all possible sampling points. By counting the number of decoding errors in a

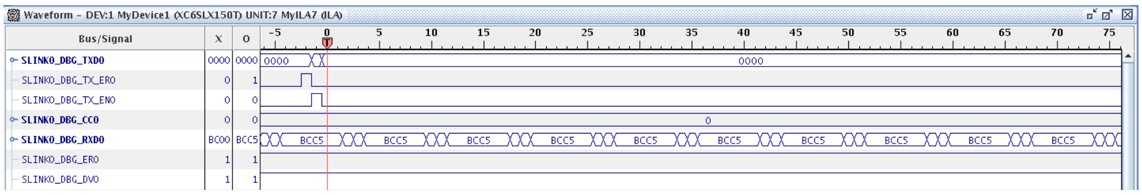


Figure 9.11: Chipscope waveform of a non-working S-LINK channel. The received data (“SLINK0_DBG_RXD0”) shows bad IDLE words (0xBCC5 instead of 0xC5BC) due to a wrong word alignment. Also the error flag “SLINK0_DBG_ER0” is constantly raised.

certain time interval, the optimal sampling point can be figured out by picking the point with the least amount of decoding errors.

Finally, it can be summarized that the source of the corruption is identified. The new algorithm for choosing the sampling point is working fine for most of the channels, but there are some cases in which it is not picked correctly. For these situations manual interventions are foreseen to cure the corruption. However, it is planned to fix the clock and data recovery algorithm to be stable under all conditions.

9.3.3 S-LINK issue

While commissioning the detector in the Pit system, instable S-LINK connections were observed. The expected behaviour of the S-LINK is that all links are up after powering on the VME crate. During the testing most of the links come up fine, but some are dysfunctional. These tests were the first tests with also the new ROBIN cards in the ROS system.

By analysing the S-LINK signals more in detail it could be observed that parts of the word alignment were not working properly. Figure 9.11 shows the Chipscope waveform of a non-working S-LINK channel. This channel receives unusual IDLE words. In the S-LINK protocol IDLE words are defined to be 0xC5BC. In this specific case the bytes in the IDLE words are swapped to 0xBCC5. Also the error flag is constantly raised. The swap is a result of wrong word boundaries. The BOC firmware has been modified to detect these failures. If a swapped IDLE word is being recognised, the incoming data will be delayed by half a word, resulting in proper IDLE words afterwards.

Unfortunately, the stability of the S-LINK did not improve much. As this kind of failure was not observed with the old ROBIN cards, the difference between the TLK transceivers and the MGTs in the new FPGA-implementation has been analysed. The main difference between both implementations is the reset sequence. The clock and data recovery unit of the MGTs needs to be reset whenever the frequency of the RX PLL changes. This has to happen on both sides to establish a working link between the BOC card and the ROBIN. The old ROBIN card with external transceiver chips performed this reset automatically. In the FPGAs of the BOC card and the ROBIN this was not the case. After adding additional logic to reset the clock and data recovery unit, the stability of the S-LINK connection improved much. With new firmware versions deployed on the BOC card and the new ROBIN card no instabilities could be observed anymore.

In summary, the issues with the S-LINK connection were created by a wrong usage of the FPGA-internal high-speed transceivers. After fixing the reset of the clock and data recovery unit, all issues were finally solved and the cards are ready for data taking.

9.3.4 IBL detector timing

The commissioning of the detector also includes a verification of the detector timing in the Pit. This can be done by checking the timing of the hits in the front-end chip with cosmic or collision data. During the analysis of this data a critical bug in the firmware of the BOC card was uncovered. The bug could result in a non-deterministic timing of the TX data path. That means the complete timing of the BOC card could change after a power-cycle or reset of the card.

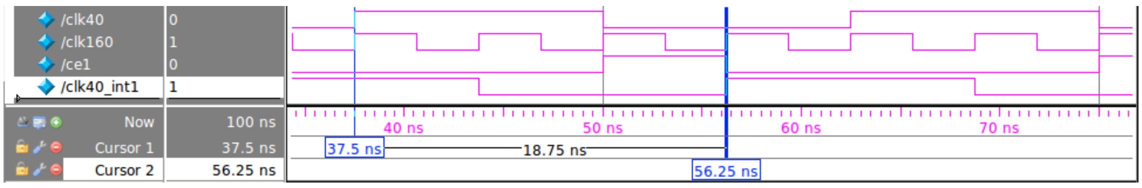
To understand the cause of this bug, the former version of the transmitter data path has to be described. In former firmware versions the transmitters were running at a frequency of 160 MHz. This allowed multi-frequency transmission at 40, 80 and 160 Mbit/s. The decision for this implementation was driven by the fact to also generate 8b10b data at 160 Mbit/s for verifying the receivers. The slow frequencies were derived from the 160 MHz by using a “clock-enable” signal, which is only active once every one, two or four clock cycles. With that mechanism flip-flops connected to this clock enable signal only see a fraction of the clock pulses.

In the 160 MHz clock domain the alignment to the external 40 MHz clock signal is lost. There are four different possibilities for the position of a rising edge in the 160 MHz clock domain with respect to the 40 MHz clock: 0°, 90°, 180°, 270°. It is not deterministic which of these four will carry the clock enable pulse for generating the internal 40 MHz. Therefore, the phase and the latency of internal 40 MHz is unknown. Figure 9.12 shows the four possible positions of the clock enable pulses and the effect on the latency of the internal 40 MHz clock.

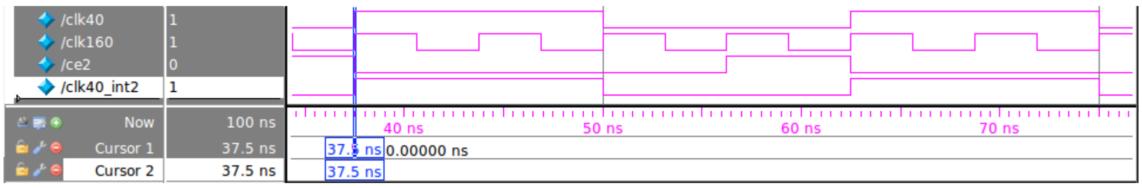
To change this unwanted behaviour, significant changes were applied to the firmware. The clocking of the TX path changed from 160 MHz to the 40 MHz LHC clock. Therefore, all signals are synchronous to the LHC clock signal and the non-deterministic phase shift cannot happen anymore. A detailed description of the firmware for the TX data path was already given in Section 7.5. The feature of generating an 80 Mbit/s data stream has been dropped, because it was not used often.

With the new firmware, timing scans for the full IBL detector were done. These scans are using hit information derived from cosmic or collision events with beam. The result of such a timing scan is an information about the optimal delay setting (coarse and fine delay) of the TX data path. The correct timing is important for the efficiency of the readout (see Section 4.7).

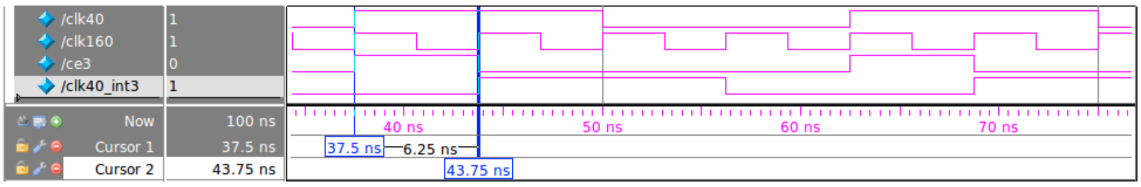
The first scans of the IBL timing with the new firmware were performed during cosmic data taking in the milestone run number 9 (M9) in March 2015. During M9 the IBL detector has been read out with a readout window size of four bunch crossing (100 ns). It means all hits of four consecutive bunch crossings are collected per trigger. The timing of the hits has been analysed after the reconstruction of



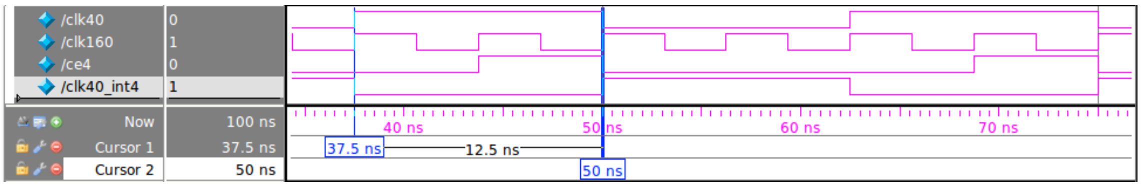
(a) possibility 1



(b) possibility 2



(c) possibility 3



(d) possibility 4

Figure 9.12: Example of the four different alignments for the clock enable pulses in the old TX data path and the corresponding internal 40 MHz clock signal with its latency to the external reference clock.

the data. During reconstruction the tracks of particles are calculated based on the hit information of the full ATLAS Inner Detector. Therefore, it provides an independent source for the timing. Each IBL hit on track could be assigned to one of the four bunch crossings. The hits of all other sub-detectors are inside one bunch crossing as they are already properly timed in. By comparing these two numbers one can estimate the coarse and fine delay which needs to be applied. Figure 9.13 shows the mean hit arrival time along stave 9 during the M9 cosmic run. The y-error corresponds to the RMS for the hit arrival time distribution. Double-chip modules are combined into one entry, whereas all 3D single-chip modules have their own entry. This analysis shows an unexpected behaviour in the timing of the two 3D modules “LI_S09_C_M4_C8_1” and “LI_S09_C_M4_C8_2”. Both modules share the same TTC command line. That means they get the same LHC clock and the command data signals. However, their mean hit arrival time is differing by more than half a bunch crossing. It was expected that the hit arrival is almost equal.

On the BOC card the delay can only be adjusted per TTC line. Front-end chips sharing the same TTC line are required to have the same hit arrival timing. The

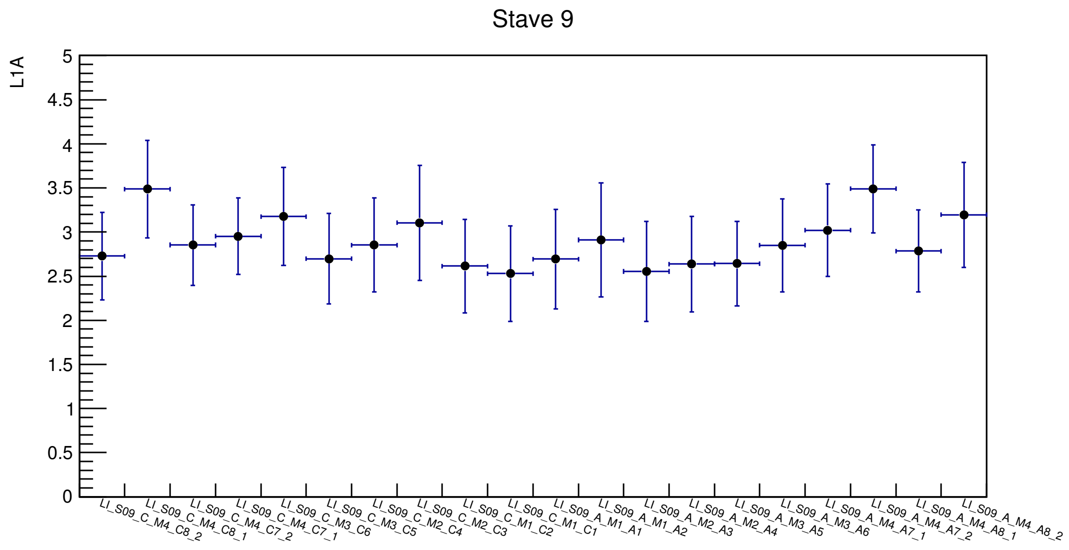


Figure 9.13: Mean hit arrival time (L1A) during the M9 cosmic run for IBL-stave 9. The y-error corresponds to the RMS for the hit arrival time distribution.The readout window has a size of four bunch crossings. Double-chip modules are combined into one entry, whereas all 3D single-chip modules have their own entry.

result of the M9 timing scan is that they do not have the same timing. This is a major issue in the IBL timing.

Further analysis and communication with the front-end designers revealed that the bias voltage of the internal discriminator (DisVbn) has a large impact on the hit timing inside the front-end chip. The behaviour of the hit timing was be analysed with a calibration scan. This scan measures the time between hit injection and detection of the hit. This time is usually referred as T_0 and the scan is called a "T0-Scan". Figure 9.14 shows the T_0-dependency as a function of the DisVbn setting. This plot confirms the strong dependency of the hit timing on the DisVbn setting. Also the hit timing differs between front-end chips due to variances in the production of the chips.

To compensate the different timing of the front-end chips a tuning algorithm has been developed. This tuning algorithm keeps the DisVbn setting of one front-end chip in a module fixed to a specific setting. The DisVbn setting of the second chip is then adjusted to achieve the same T_0. The tuning can be verified with the T0-Scan. The parameter to be looked at is the difference between the T_0 of chips sharing the same TTC line:

$$\Delta T_0 = |T_0(\text{Chip}\,1) - T_0(\text{Chip}\,2)| \tag{9.1}$$

If both chips have the same hit timing, this value is zero and large values correspond to a bad timing situation.

Figure 9.15a shows the timing situation before the DisVbn tuning was applied. Each entry in the 2D histogram corresponds to two chips sharing the same TTC line. The x-axis shows the module along the stave and the y-axis is the stave. The plot shows a large discrepancy in timing. The timing difference between two front-end

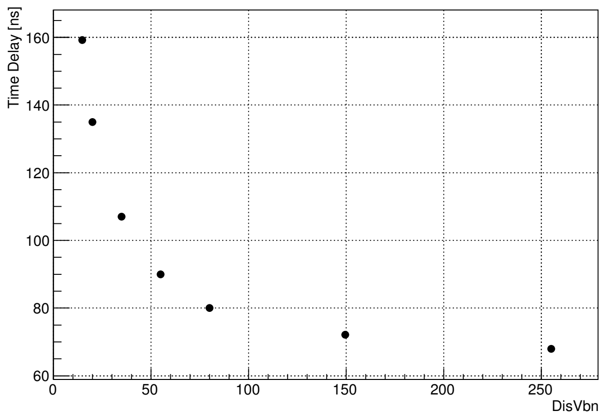
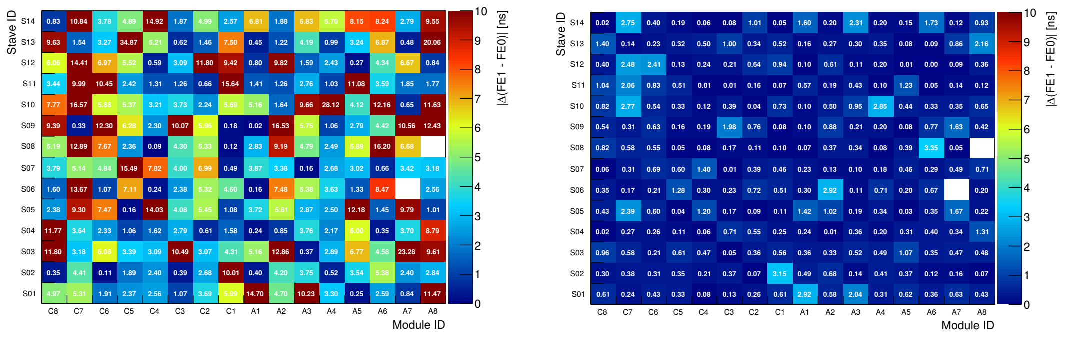


Figure 9.14: Time between hit injection and detection (T_0) as function of the discriminator bias voltage (DisVbn) setting [87]

chips on the same TTC line is up to 35 ns for module C5 on stave 13. The modules A8 on stave 8 and A7 on stave 6 were disabled during the scan.

After tuning the DisVbn setting for all modules, the timing situation is much better. Figure 9.15b shows the timing difference after applying the new DisVbn settings from the tuning. The discrepancy in timing could be reduced to 3.3 ns at maximum. Most of the modules have a timing difference of less than 1 ns. With this new DisVbn settings the delay settings for all the TTC lines in the BOC card could be applied.



(a) before DisVbn tuning

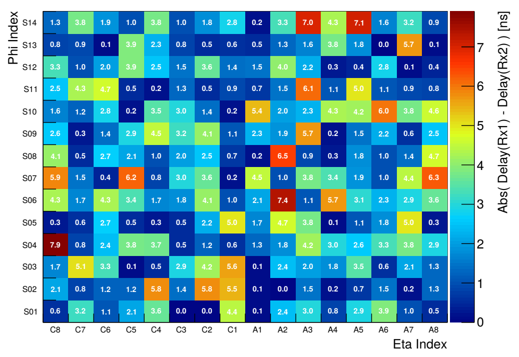
(b) after DisVbn tuning and manual adjustment

Figure 9.15: Time difference in the hit timing between two front-end chips sharing the same TTC line (ΔT_0). On the x-axis the module ID along the stave is shown, the y-axis is the stave ID (plot by Hideyuki Oide).

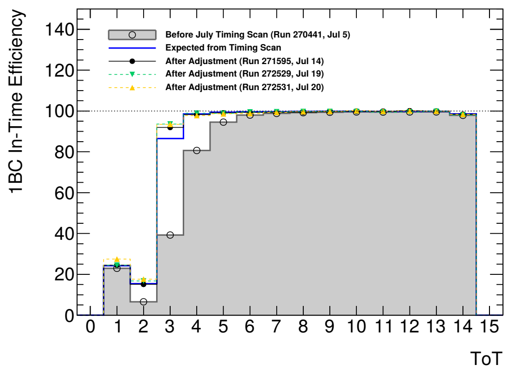
Before taking physics data the timing of the detector has been checked with beam data in July 2015. The beam is providing a timing-independent source for hits. Therefore, it can be used as a reference all the IBL modules. First, the DisVbn tuning has been verified. The result is shown in Figure 9.16a and it differs from of the T0-Scan in Figure 9.15b. This can be explained by the injection mechanism.

During a T0-Scan the hits are injected inside the front-end chip and their timing also depends on the characteristics of the chip. With beam the injection does not depend on the chip anymore. This difference can be observed in this plot. It is foreseen to retune the DisVbn settings in all front-end chips based on the $\Delta T0$ measurement with beam.

Also the in-time efficiency during data taking runs with beam has been analysed after applying all the coarse delay constants. The coarse delay values have been calculated from the results of the timing scan. The efficiency is measured with a four bunch crossing wide readout window and is counting the the number of hits in the target bunch crossing compared to the number of hits in all four bunch crossings. The efficiency is shown in Figure 9.16b. Hits with a ToT value above 5 have nearly 100 % efficiency. For low ToT hits the efficiency drops. The reason for this drop is the time walk effect. Low ToT hits are lost in the following bunch crossing and are not read out when using a one bunch crossing readout window. That means most of these hits are lost in the normal readout mode. By using the hit-doubling mechanism in the front-end chips these hits can be recovered. Also shown in this plot is the stability of the TX with respect to the timing. The efficiency is the same for several data taking runs and therefore no effects of a non-deterministic timing are observed.



(a) Verification of the DisVbn tuning with beam reference. Shown is the ΔT_0 for all IBL modules.



(b) In-time efficiency after application of different data taking runs in July 2015 after applying the coarse delay constants on the BOC card. The efficiency is calculated from the number of hits in the target bunch crossing divided by the total number of hits in all triggered bunch crossings

Figure 9.16: Verification of the DisVbn tuning and in-time efficiency (plots by Hideyuki Oide).

The result of all timing analyses show that the IBL timing is still an important topic. With the 25 ns bunch crossing frequency the readout window has to be reduced to one bunch crossing. To be able to do that, it is required to have a good timing. The latest timing scans during data taking runs in July 2015 have shown that the readout efficiency is sufficient to move to the one bunch crossing readout window. However, to recover low ToT hits, the hit duplication in the front-end chip has to be enabled. Also it is foreseen to apply also the fine delay values, to increase the efficiency of the low ToT hits.

Applying the fine delays has also been tested in the system test. After loading the partial bit files containing the fine delay values several severe errors in the firmware functionality showed up. For example, all S-LINK connections of the BOC card failed and the ROS reported bad event fragments. These issues were not observed in the Wuppertal readout setup, as not all parts of the firmware can be tested together. The result of this test is that loading the partial bit files affects major parts of the firmware. Changing the configuration of the IODELAY2 primitives seems to be impossible without side-effects on other parts of the firmware. As a workaround the integration of the fine delay values into the full firmware image of the card has been tested successfully. The fine delays are already applied after loading of the BMF firmware and no partial reconfiguration is needed. A disadvantage of this solution is that the firmware binaries for all cards are different and the settings which are loaded cannot be read out. To make tracking of the binaries easier, it is foreseen to tag all firmware binaries during their generation.

To summarize the behaviour of the BOC card in terms of timing, the new firmware version does not show the non-deterministic latency anymore. During the analysis of the IBL timing the large DisVbn dependency of the timing has been discovered. By tuning the DisVbn values in the front-end chips this situation has been improved drastically. However, after a verification with beam, the difference in timing between two front-end chips is still significant. It is foreseen to further adjust the DisVbn settings based on the timing scan with beam. Anyhow, after applying the coarse delay settings to the BOC cards, the timing of the detector is sufficient for the 25 ns bunch crossing frequency operation. By applying also the fine delay values in the firmware images, the efficiency of the readout system will be increased.

9.4 Summary of the system test

All in all, the system test of the card was an important test for the final card hardware and firmware. The system test unveiled some serious problems in the system and on the BOC card itself.

The first tests in the local laboratory system showed a very good performance of the BOC card during calibration scans. This setup is also very useful to test new firmware version before releasing them to CERN. But even with severe testing in the local lab setup, it could not be prevented that major issues with the card were observed in the system test at CERN. This system test showed three major problems related to the BOC card. The issue with instabilities on the S-LINK connection has been solved by adding a reset logic to the clock and data recovery mechanism of the MGTs.

Another issue is the corruption of data received from the front-end chips. On the BOC card several firmware changes were made to improve the robustness of the firmware against the corruption. Also more monitoring has been added. With the monitoring available, the source of the corruption could be traced down to issues in the clock and data recovery algorithm for choosing the optimal sampling point in the data. For some channels this algorithm seems to be unreliable. Improvements to increase the reliability are under development. Until these improvements are deployed, the DAQ system can bypass the algorithm for choosing the sampling point by setting a fixed sampling point for problematic channels.

The last of the three issues is the timing of the detector. The bug of having a non-deterministic latency in the TX data path is fixed by a new firmware and measurements of the detector timing show good results. Results from the timing scans will be fed back to the DisVbn tuning algorithm to further improve the readout efficiency. Also the deployment of the fine delay values is underway. Even if the partial reconfiguration showed serious side-effects and will not be used until these side-effects are understood and resolved, this is not a show-stopper for the IBL timing.

Overall, the performance of the BOC card in the system is good. Major issues have been resolved and the card is ready for the physics data taking starting in summer 2015.

10 Future readout concepts

10.1 LHC and ATLAS upgrade plans

In the next years the LHC will be upgraded to increase the luminosity. Figure 10.1 shows the major developments. It is planned to increase the luminosity of the machine by a factor 5, to $5 \cdot 10^{34}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$. The upgrade will take place in steps, each marked by a long shutdown period (LS1-LS3). In these periods also the LHC experiments can deploy upgrades.

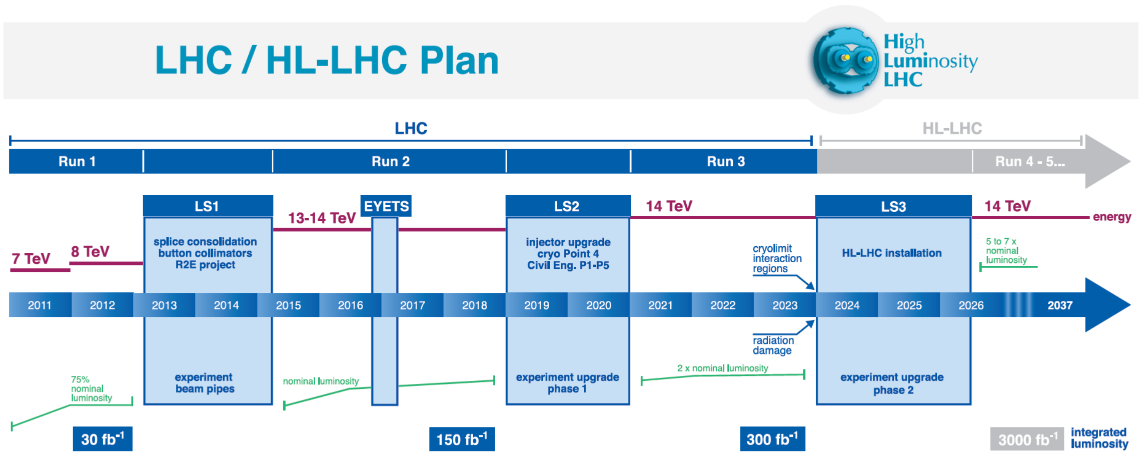


Figure 10.1: LHC upgrade plans [88]

Also ATLAS uses these shutdown to upgrade parts of the detector [89]. During LS1 from 2011-2013 the IBL and the Diamond Beam Monitor were installed into ATLAS. Also the muon system has been completed. The ATLAS upgrades during LS2 in 2019/20 will tackle the increased luminosity of $2-3 \cdot 10^{34}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$. The muon tracking system will be extended by the New Small Wheels and the ATLAS trigger system will be upgraded with the Fast TracKer. Finally, in LS3 starting 2024 the complete Inner Detector will be replaced by the new Inner Tracker (ITk). Also a new trigger architecture will be deployed in this upgrade.

10.2 The ATLAS ITk project

As stated in the last section, ATLAS will replace its Inner Detector (Pixel, SCT and TRT) by the new ITk. The main reasons for the replacement are the radiation damage after more than 10 years of operation, the bandwidth saturation, and a higher than supported hit occupancy of the current Inner Detector. The ITk will consist of a pixel detector and a silicon strip detector. Its baseline layout from the

Letter of Intent (LoI) [90] is shown in Figure 10.2. The innermost part consists of a 4-layer pixel detector with 8.2 m^2 of silicon area followed by a 6-layer silicon strip detector with 193 m^2 of silicon area. The forward region of the detector is covered by 6 pixel and 7 strip discs. Table 10.1 shows the distribution of the readout modules in the 4-layer pixel detector. In total 5532 modules need to be read out.

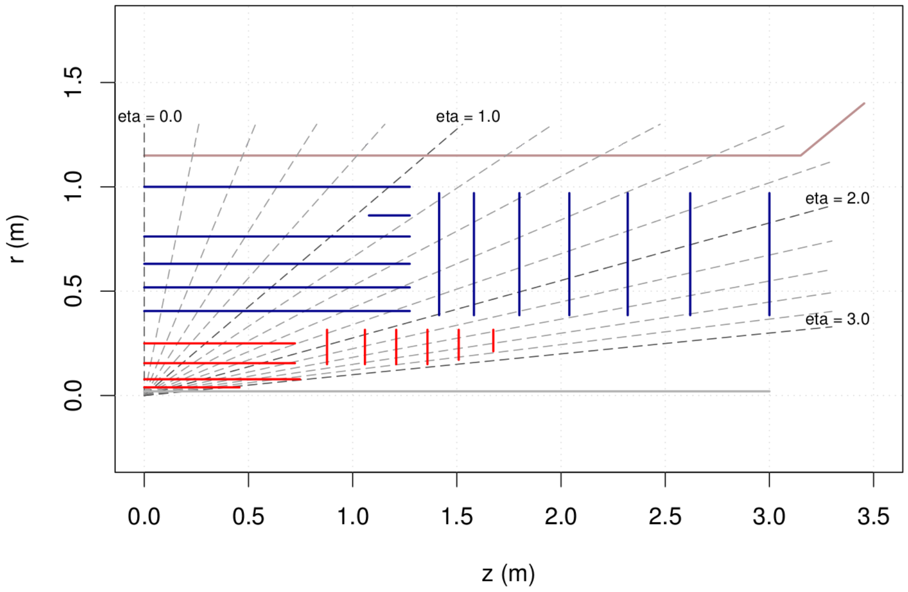


Figure 10.2: Sectional drawing of 1/4th of the ITk baseline layout with the interaction point in the lower left edge. [90]

During ongoing discussions also ideas for a 6-layer ITk pixel detector have been presented. This detector would cover an area of $|\eta| = 4$. Table 10.2 shows the distribution of staves and modules in this design. It would have a total silicon area of 17.33 m^2 and the total number of modules to be read out is 11880.

The trigger architecture of the ATLAS detector will also change, as a new trigger level in front of the LVL1 trigger is added to the system. Figure 10.3 shows how this level 0 (LVL0) and level 1 (LVL1) trigger system could look like. The impact on the ITk project is large due to the fact that also tracking information from the ITk should be part of the LVL1 trigger decision. This trigger processor is called “L1 Track” and has to get its trigger decision after a maximum latency of 6 microseconds. Therefore, the ITk readout system should not introduce too much latency in the data processing. The trigger rate for the LVL0 trigger is 500 kHz in the LoI design, but currently trigger rates of 1 MHz and more are under discussion.

10.3 Future readout architectures

As it is foreseen to read out the ITk pixel detector at the LVL0 trigger rate, a high readout bandwidth is required between the front-end chips and the off-detector readout system. For the innermost layer of the pixel detector the data rate per module will be in the region of about 5 Gbit/s. Compared to the IBL readout, which is running at 160 Mbit/s, this is an increase by a factor 31. Also the number

Table 10.1: Distribution of staves and modules in a 4-layer ITk pixel detector layout [90]

	Radius [cm] / z-position [cm]	**Number staves/disks**	**Modules per stave/disk**	**Number of modules**
Layer 1	3.9	16	22	352
Layer 2	7.8	16	36	576
Layer 3	15.5	32	35	1120
Layer 4	25.0	52	35	1820
Disk 1	±87.7	2	144	288
Disk 2	±105.9	2	144	288
Disk 3	±120.9	2	144	288
Disk 4	±135.9	2	144	288
Disk 4	±150.9	2	148	296
Disk 5	±167.5	2	108	216
Total number of modules				**5532**
Total surface [m^2]				**8.2**

of modules to be read out in total is much higher than in the current Pixel Detector which has 1968 modules including the IBL modules.

The current readout system cannot cope with such a high bandwidth and also with the high number of modules. Therefore, a totally new readout system needs to be designed. This readout system has to fulfil the requirements of the new trigger system as well. The latency of the L1 Track trigger system is planned to be 6 μs in total. It is foreseen that the delay introduced by an ITk readout system needs to be less than 1 μs.

This chapter will give a brief outlook on current ideas for the ITk readout. First it will present a concept for a fast optical link and then a totally new readout architecture based on switched network connections will be presented. Also ITk readout studies performed in Wuppertal will be shown.

10.4 GBT and Versatile Link

10.4.1 Overview

To cope with the high readout bandwidth that is needed to get the data out of the detector, CERN has started two projects related to detector readout: the Versatile Link project and the GigaBit Transceiver (GBT) project. Both were developed under the "Radiation Hard Optical Link Project". The Versatile Link project [91] is concentrating on the optical data transmission and develops a radiation hard bidirectional optical link for data rates of up to 5 Gbit/s. The GBT project on the other hand has its focus on the protocol driven over optical links and the corresponding radiation hard ASICs [92]. Figure 10.4 shows an overview of both projects and how they can be used to build up a readout chain. The GBT on the on-detector side is distributing and collecting all the data to and from the detector. The data

Table 10.2: Distribution of staves and modules in a 6-layer ITk pixel detector layout

	Radius [cm]	**Number staves/rings**	**Modules per stave/ring**	**Number of modules**
Layer 1	3.9	16	60	960
Layer 2	6.5	16	60	960
Layer 3	16	32	35	1120
Layer 4	20	40	35	1400
Layer 5	30	56	35	1960
Layer 6	34	64	35	2240
Ring Set 1	15-19	18	36	648
Ring Set 2	21-25	24	48	1152
Ring Set 3	27.5-31.5	24	60	1440
Total number of modules				**11880**
Total surface [m^2]				**17.33**

is split into several types, like TTC information, DAQ data (hits) and slow control data (DCS). On the off-detector side commercial hardware can be used to receive the GBT data. Between on- and off-detector the Versatile Link is responsible for the optical data transmission.

The GBT project from the architectural point of view is a link aggregator, which can combine a lot of slow electrical links into a high-speed link. Together with the Versatile Link an optical data transmission can be realised. The slow links in the GBT architecture are called "e-links". Each e-link corresponds to a serial input or output link. It can be used for all kinds of serial data transmissions with a data rate of up to 320 Mbit/s (future versions plan 1.28 or 2.56 Gbit/s). All e-links are then combined into a high-speed link running at a data rate of 4.8 Gbit/s per direction in the first version (a 9.6 Gbit/s version has already been announced).

The GBT also uses its own transmission protocol. In Figure 10.5 a typical GBT frame is shown. One frame has a length of 120 bits and a duration of one LHC clock cycle (25 ns), giving the data rate of 4.8 Gbit/s. Each frame starts with a header sequence of four bits, which allows the receiver to recover the frame boundaries. The header is followed by four slow control bits divided into two bits for the GBT slow control and two bits for general slow control. Then 80 bits of user data are following. They carry the data, which is transmitted and received through the e-links. Together with the two bits of slow control data the user can use 82 bits for data transfer. All data bits in the frame are finally secured by 32 bits of forward error correction, which is based on a Reed-Solomon code [93]. All together this makes a total line efficiency of 68 %.

The full encoding and decoding path for the GBT is shown in Figure 10.6. The 84 bits of user data (e-links + slow control) are first fed into a scrambling unit, which is used to maintain a DC balanced signal and prevent the occurrence of long constant bit sequences. Then the 4 bits header information is added and all 88 bits will be protected by error correction bits using a Reed-Solomon encoder. Finally, the

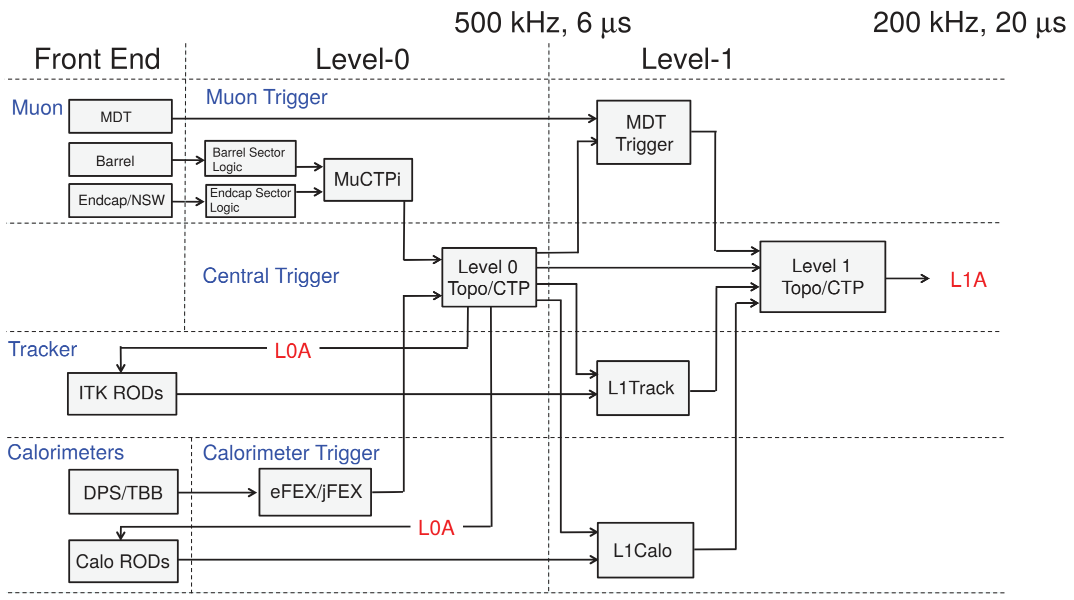


Figure 10.3: Overview of the ATLAS Level-0 and Level-1 trigger system after the phase II upgrade. The ITk system has to provide hits to the L1 Track system [90].

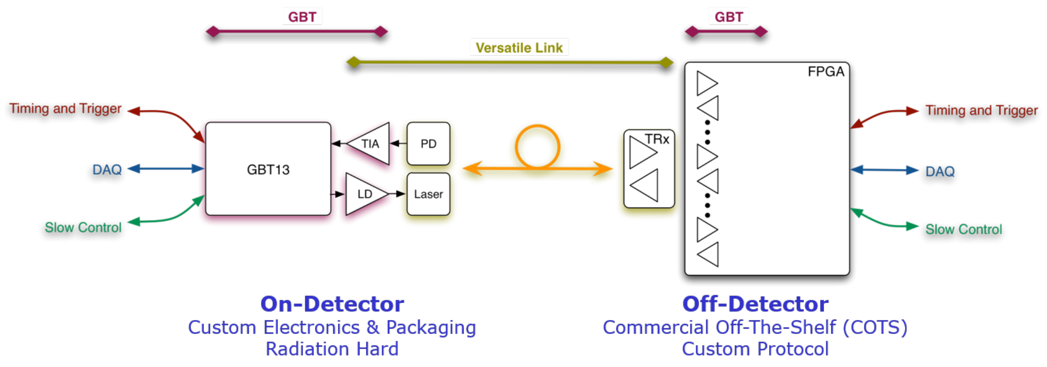


Figure 10.4: Overview of the GBT and Versatile Link project [92].

data including the error correction bits passes an interleaver stage. The interleaver shuffles bits in the data stream. This step is important to be robust against burst errors. The 120 bits are then sent out by the serialiser at a data rate of 4.8 Gbit/s. On the receiver side, all these steps are reversed. To recover the word alignment on the receiver side, the header bits in consecutive data frames are matched.

10.4.2 GBT studies in Wuppertal

To gain experience with the GBT firmware, which is provided by the GBT developers, a basic example design has been implemented. It utilizes the GBT Link Interface Board (GLIB) [95] which has been designed by the GBT group as an evaluation board for the GBT FPGA firmware. Central part of the GLIB is a Xilinx Virtex-6 FPGA with its 6.5 Gbit/s MGTs. The GLIB provides sockets for four full-duplex Small Form-factor Pluggables (SFPs) and two FMC expansion cards. Also it

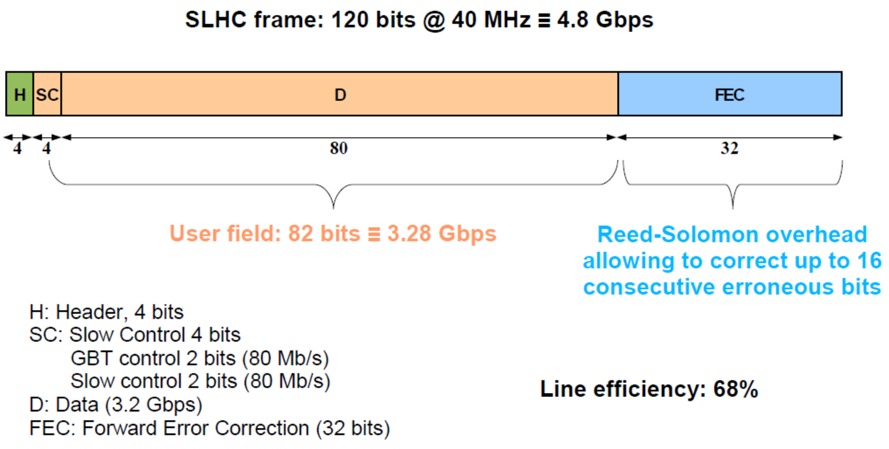

SLHC frame: 120 bits @ 40 MHz ≡ 4.8 Gbps
H
SC
D
FEC
4
4
80
32
User field: 82 bits ≡ 3.28 Gbps
Reed-Solomon overhead allowing to correct up to 16 consecutive erroneous bits
H: Header, 4 bits
SC: Slow Control 4 bits
GBT control 2 bits (80 Mb/s)
Slow control 2 bits (80 Mb/s)
D: Data (3.2 Gbps)
FEC: Forward Error Correction (32 bits)
Line efficiency: 68%

Figure 10.5: Structure of a GBT frame [94].

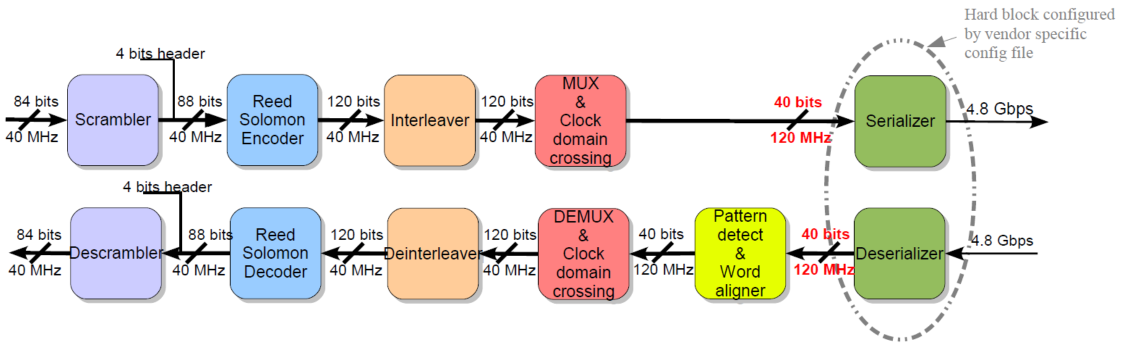


Figure 10.6: Block diagram of the GBT encoding and decoding data path [94].

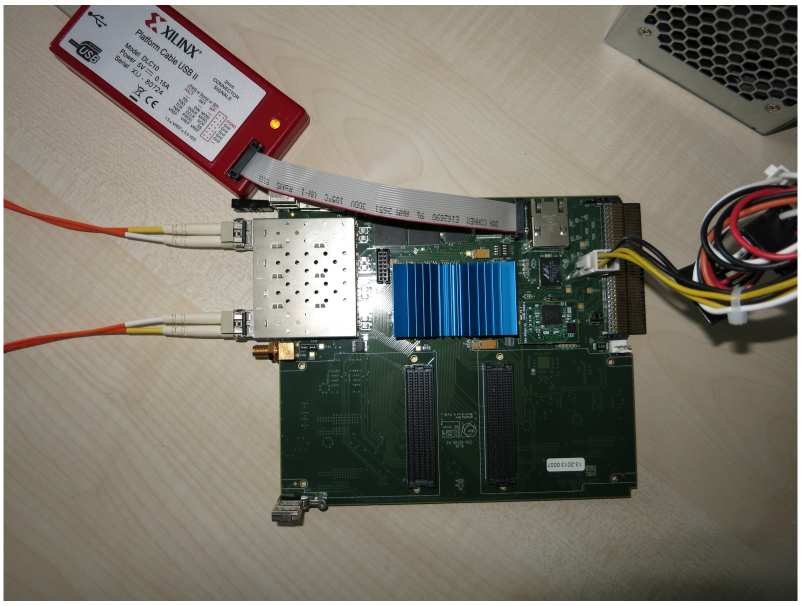


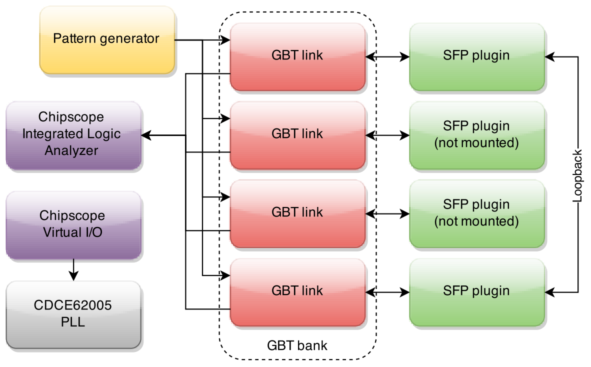
Figure 10.7: Picture of the GBT Link Interface Board for evaluating the GBT firmware.

has an Ethernet connection for configuration. The clock distribution on the board is provided by the CDCE62005 clock generator. It can be configured from the FPGA to provide different output frequencies. Figure 10.7 shows the GLIB with two SFP plugins mounted on it.

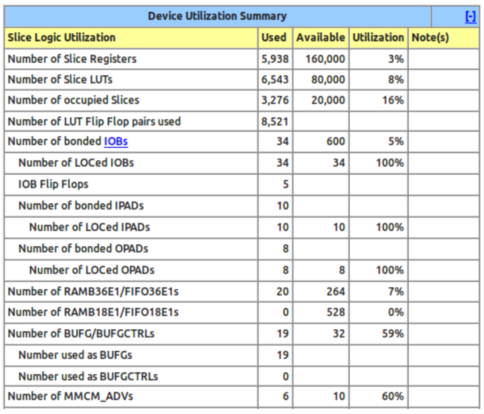
The firmware running on the GLIB board, is based on reference designs by the GBT developers and it uses the standard four link GBT bank. Figure 10.8a shows a block schematic of the firmware. Central part is the GBT bank with its four GBT links. Around the GBT bank several Chipscope debug components are placed. A pattern generator produces the user bits in the GBT protocol to be sent out in parallel through all links. The received data can then be viewed in the integrated logic analyser (ILA) block. For steering the board a virtual input/output (VIO) is also implemented. To allow modification of the board clocking, an interface to the CDCE62005 was designed, which can manipulate the PLL chip registers using the VIO interface.

After implementation of the firmware the resource usage has been checked. Figure 10.8b shows a summary of the resource usage for the GLIB. The most problematic number is the number of used clocking resources which is already 60 % of all available resources for that FPGA. The main reason for this high resource usage is the GBT firmware and its reception data path, which needs one clock buffer for each GBT link. As it is expected to have a high number (in the order of 20) of GBT links per off-detector card, this resource usage is unacceptable and would cause a not synthesizable firmware. It needs to be checked if future GBT firmware versions can reduce that number.

The testing of the firmware was performed on the GLIB with a local loopback connection over a short optical fibre. This verifies that the GBT implementation in the evaluation firmware works as expected. Figure 10.9 shows the output of a GBT receiver at the e-link level for a single GBT link during plugging of the fibre. First,



(a) Block schematic



Device Utilization Summary				[-]
Slice Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Registers	5,938	160,000	3%	
Number of Slice LUTs	6,543	80,000	8%	
Number of occupied Slices	3,276	20,000	16%	
Number of LUT Flip Flop pairs used	8,521			
Number of bonded IOBs	34	600	5%	
Number of LOCed IOBs	34	34	100%	
IOB Flip Flops	5			
Number of bonded IPADs	10			
Number of LOCed IPADs	10	10	100%	
Number of bonded OPADs	8			
Number of LOCed OPADs	8	8	100%	
Number of RAMB36E1/FIFO36E1s	20	264	7%	
Number of RAMB18E1/FIFO18E1s	0	528	0%	
Number of BUFG/BUFGCTRLs	19	32	59%	
Number used as BUFGs	19			
Number used as BUFGCTRLs	0			
Number of MMCM_ADVs	6	10	60%	

(b) Resource usage

Figure 10.8: Block schematic and resource usage of the GBT test firmware running on the GLIB with four GBT links.

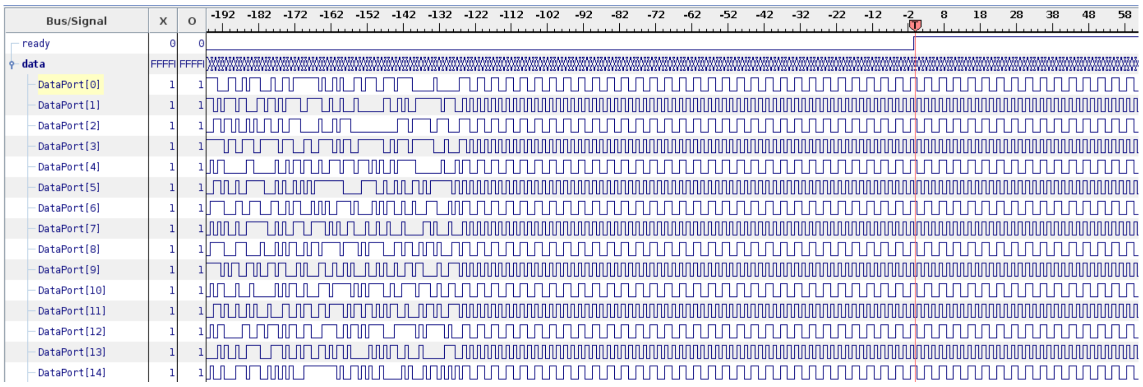


Figure 10.9: Chipscope waveform of a single GBT link during plugging of the optical fibre. The link training is finished when at the rising edge of the “ready” signal and the received pattern is stable.

the waveform shows noise being decoded by the GBT and the “ready” signal is low, meaning no link has been established. During the training sequence, the data signals change to the regular pattern which is sent out by the pattern generator. Finally, the “ready” signal is raised indicating that the link training has finished and the link is up.

In summary, the first steps using the GBT firmware have been done with the GLIB. The implementation of the links showed an unusual high usage of clocking resources. The next steps in the GBT evaluation would be to reduce the number of clocking resources and performing a bit error rate measurement, to check the reliability of the GBT link.

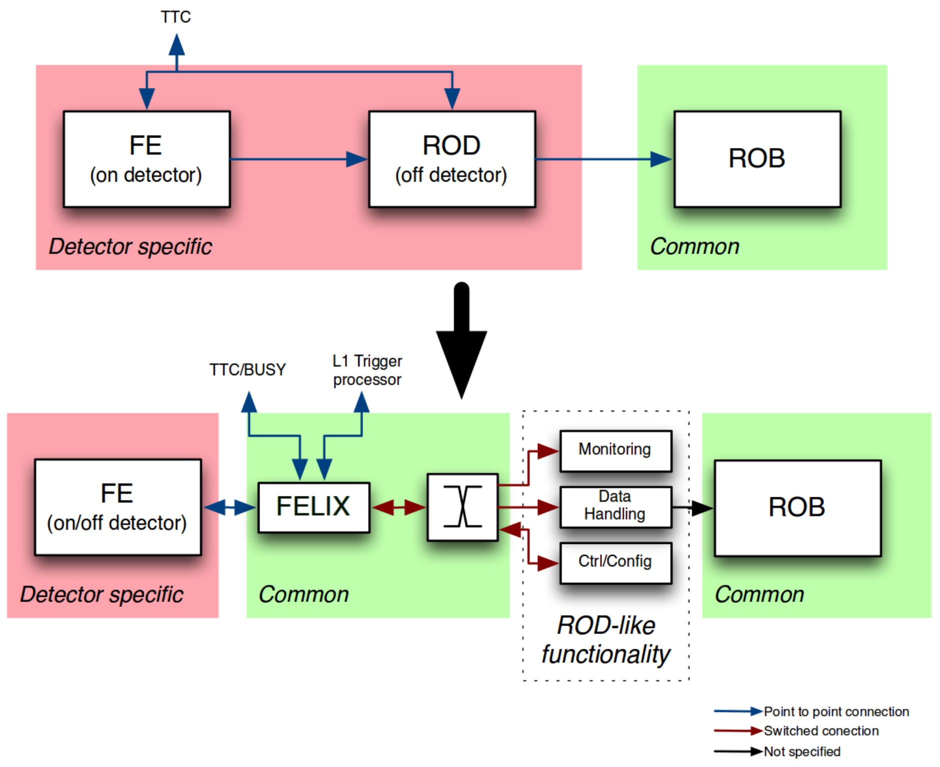


Figure 10.10: Evolution of the readout architecture with the introduction of the FELIX concept. The detector specific ROD is replaced by the FELIX box with a switched network connection [96].

10.5 The FELIX readout concept

10.5.1 Overview

The Front-End Link Interface eXchange (FELIX) concept is a new readout concept presented by the ATLAS TDAQ group [96]. Its main strategy is to convert detector- and user-specific protocols into common transmission protocols, like Ethernet. The big advantage of having common transmission protocols is the usage of commercial hardware. Figure 10.10 shows how the readout system architecture will change with the introduction of the FELIX concept. In the current readout architectures all ATLAS subdetectors have their specific readout processor (ROD) which is tightly coupled to the detector and forwarding all the data to the common readout subsystem (ROS). The new architecture introduces a common readout box which is called FELIX as the first off-detector stage.

The FELIX hardware will consist of a PC under the control of the TDAQ group. It is equipped with several FELIX PCIe cards and one or more 40 Gbit/s network interface cards. The general concept of the FELIX readout architecture is shown in Figure 10.11. The FELIX PCIe card will interface to a number of GBT links. Each GBT link is split up into its e-links. The data of these e-links is then received from and sent through the 40 Gbit/s network. For routing the data to the destination hosts, FELIX supports multiple addressing schemes. First, the complete data of an e-link can be routed to a destination host defined by an IP address and a network port. Another addressing scheme allows load balancing. Therefore, the e-link data has to be tagged with a stream identifier. This stream identifier is then used to address a machine in the switched network. This allows one to split up the event data from the front-end chips by using the trigger number. So the load of the event readout can be spread over multiple machines.

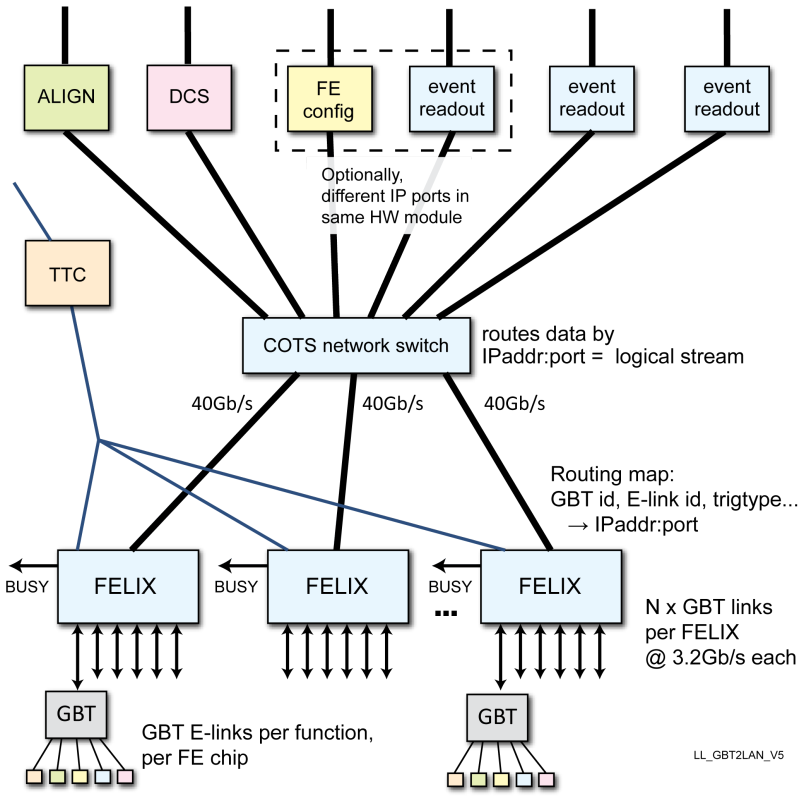

ALIGN
DCS
FE config
event readout
event readout
event readout
Optionally, different IP ports in same HW module
TTC
COTS network switch
routes data by IPaddr:port = logical stream
40Gb/s
40Gb/s
40Gb/s
Routing map: GBT id, E-link id, trigtype... → IPaddr:port
BUSY
FELIX
BUSY
FELIX
BUSY
FELIX
N x GBT links per FELIX @ 3.2Gb/s each
GBT
GBT E-links per function, per FE chip
GBT
LL_GBT2LAN_V5

Figure 10.11: Overview of the FELIX readout architecturen as a switch between GBT and Ethernet. All data processing nodes are connected to the commercial switched network. [97]

As FELIX is meant to be a detector independent system, it does no further data processing. In the TDAQ group it is commonly referred to be a switch between GBT and Ethernet. Therefore, all detector specific readout components are nodes in the switched network. This gives the possibility of a pure software readout system (software-ROD) where no custom off-detector hardware is needed. To ensure the low latency requirements of the TTC and L1 Track systems, special connections are available.

10.5.2 Concepts for ITk readout and calibration

As the ATLAS TDAQ group pushes towards the FELIX architecture, this has clearly an impact on the readout system for the ITk. ITk has special requirements for the detector readout due to high readout bandwidths in the inner layers. Also the calibration of the ITk is tightly coupled to the detector. The calibration method for pixel detectors has already been shown for the IBL in Section 4.6.3. A calibration is defined by the execution of nested loops. The innermost loop is the trigger loop and gathers all the data from the front-end chips. A trigger command may only be executed if all the data from the previous trigger has been gathered. Therefore, the calibration procedure needs feedback already in the innermost loop.

These requirements have an impact to the FELIX architecture. During a (calibration driven) discussion with one of the FELIX developers [98], four different options to tackle the ITk calibration in the FELIX architecture were evaluated. Figure 10.12 shows an overview of the four options.

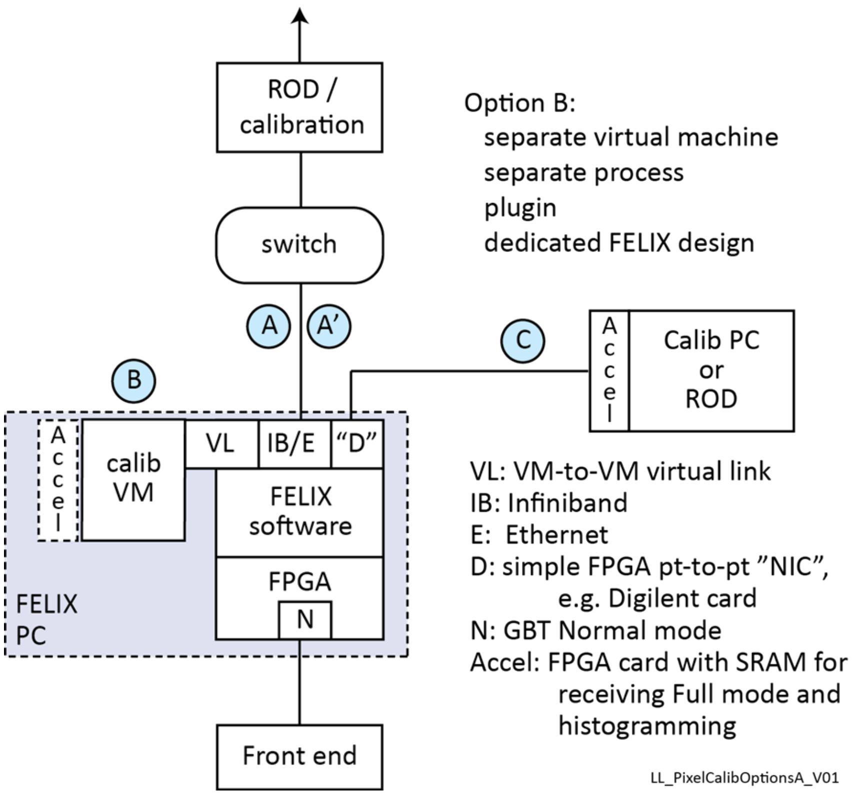


Figure 10.12: Different options for integration of the ITk calibration into the FELIX architecture [98].

Option A: standard FELIX data path

Option A in the concept of ITk calibration uses the standard FELIX data path. All data from the detector will be sent through a high-speed 40 Gbit/s Ethernet connection to the ROD. The ROD in this option would be a PC or FPGA card behind the switched network connection.

Option A': low-latency network

The second option is a sibling of the first option, but with a low-latency network connection. This network connection is served by the InfiniBand standard [99]. InfiniBand is a high-speed transmission protocol with very low overhead and a low latency, which is specified to be less than 2 microseconds. It combines a switched network connection with low latency.

Option B: Extension of the FELIX PC

The third option will add an extension to the FELIX PC, which is specialised for the calibration of the ITk. It is hosted in a virtual machine (VM), allowing the separation of responsibilities between the ITk and TDAQ community. As the virtual machine is executed in the FELIX PC, it is tightly coupled to it, and only a fast virtual network connection is established between the host and the virtual machine. All ITk calibration routines run inside the virtual machine and they may be supported by a fast FPGA-based accelerator card. This card could provide histogramming and fitting of data.

Option C: Tightly coupled calibration system

The last of the four options foresees a tightly coupled calibration system connected to the FELIX card over a direct non-switched connection. This connection has to be a high-speed and low-latency connection. Ideally, it gets a copy of all the data sent by the detector. The calibration system receives the data and performs the signal processing of the data, like histogramming.

Discussion of the four options

All four options have their advantages or disadvantages. Option A and A' are the two options, which are the most compatible options in the FELIX architecture. They both do not change anything in the architecture of FELIX. However, a disadvantage is that all ITk relevant components are hidden behind the FELIX system in a switched network. This breaks the tight coupling between the on-detector and the off-detector components. During calibration a trigger command has to pass the switched network and the FELIX card. The same happens to the hit response to the trigger command. Therefore, the network and FELIX data path has to be passed twice for each trigger command. As the readout system has to wait for the response in each trigger loop, the latency is multiplied by the number of trigger loops during the calibration. For complex calibration routines, this number can get huge increasing the total calibration time drastically. Also during data taking it is not clear, how the monitoring and data processing will look like. For event bookkeeping it is required to have access to the TTC information. The off-detector readout system does not have this access and it is not clear if the FELIX card can handle that.

The options B and C are the most difficult options. Option B would require an extension card sitting in the FELIX PC. This is a huge disadvantage as the FELIX PC is maintained by the TDAQ group. In case of issues during data taking it is not clear if ITk or TDAQ community will be in charge of fixing these issues. Also it is doubtful whether enough computing resources left in the FELIX PC to execute calibration routines even if the PC is extended with an accelerator card. For option C the number of available resources in the FELIX card is questionable. A dedicated high-speed link to an ITk calibration system would consume FPGA resources needed for regular FELIX operation.

Another issue of all four options is the handling of low-latency data to the L1 Track system. The allowed latency introduced by the off-detector readout systems including FELIX has to be less than 1 μs. Also it is still unclear if FELIX will be responsible for handing over the data, and if the latency introduced by FELIX is low enough. Therefore, the ITk readout working group has proposed two solutions. The first solution consists of an own readout card developed by the ITk community. It tries to fit into the FELIX concept by using a common hardware and firmware platform. In this solution the ITk card could look like a FELIX card from the outside, but it is extended with functionality for the ITk calibration. The second proposed solution would be a preprocessor card sitting in front of FELIX. This concept will also be used by other subdetectors, like the Tile calorimeter [100]. This preprocessor card will handle all the low-level interaction with the ITk detector. Also the calibration routines and the data histogramming will be handled in this card. During data taking the card is responsible for monitoring the detector data and forwarding it to the L1 Track system.

Overall, the FELIX discussion in the ITk community is in flow and no final decisions have been made. Also the ITk Pixel on-detector electronics are not finalised. Therefore, the requirements for the Pixel detector readout are not yet available, making a final decision impossible. The next steps in the evaluation of the FELIX concept for ITk will be calibration based studies on fast histogramming in FPGAs, which is anyhow needed for the ITk readout.

10.6 PCIe studies

Looking forward on future readout architectures and their interfaces, there is a trend to use common interfaces, like Ethernet or PCIe. Many current and future readout system rely on the PCIe interface. A recent development is the Yet Another Rapid Readout (YARR) system [101]. It was developed as cheap readout system for laboratory readout setups and can be connected to multiple front-end chips. The received data is fully processed in software making it very flexible.

To get familiar with the PCIe programming and interfacing with Xilinx FPGAs, PCIe studies were done in Wuppertal. These studies use the Spartan-6 FPGA family, which is already known from the IBL readout systems. The tests were performed using a Spartan-6 PCIe evaluation card, which is shown in Figure 10.13. The card provides a PCIe interface with one lane in version 1.0 and should give around 200 MB/s throughput.

For testing the throughput of the PCIe interface on the SP605 evaluation board, a special firmware has been developed. Figure 10.14 shows the block diagram of the firmware. Central part is an AXI interconnect, which can transfer up to 300 MB/s of data. Connected to this interconnect are several firmware blocks. The PCIe interface provides access between the host PC and all other blocks connected to the AXI interconnect. The 128 MB of DDR3 memory of the SP605 are connected through a

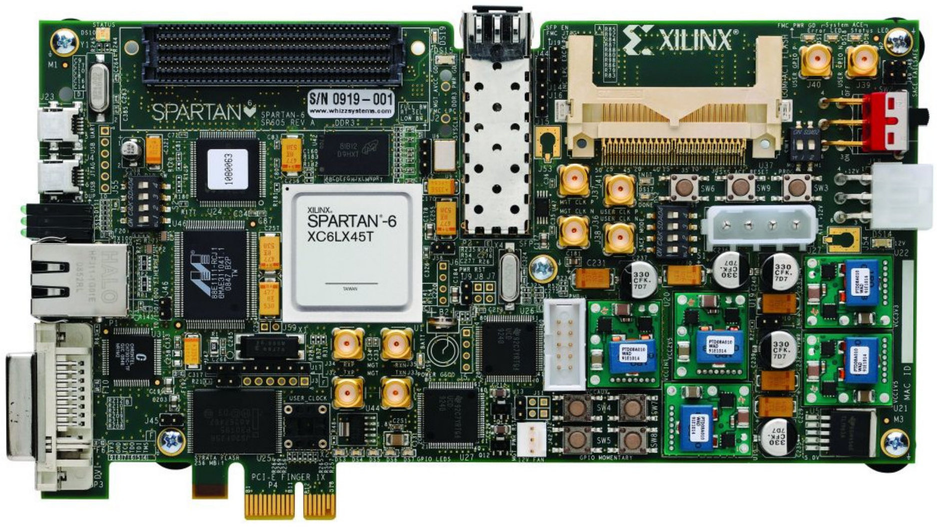


Figure 10.13: Spartan-6 FPGA evaluation card for PCIe development

DDR3 memory controller, which is using the Spartan-6 memory interface. A DMA[1] controller can be used for standalone data transfers between the host computer and the PCIe card. Finally some LEDs can be accessed over a general purpose I/O (GPIO) interface, which is useful for debugging.

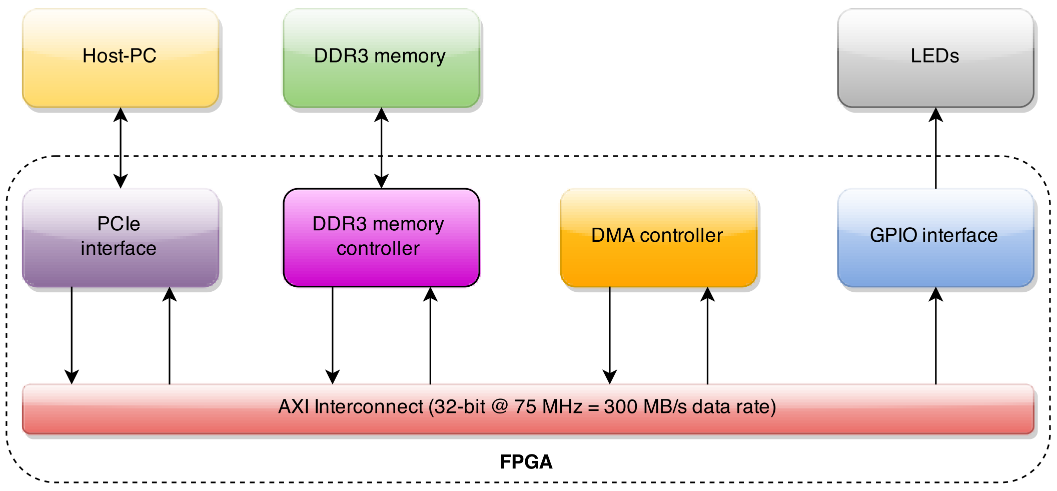


Figure 10.14: Block schematic of the PCIe test firmware for the SP605 evaluation card.

Using this firmware, the throughput between host and device was measured. It utilizes a Linux device driver controlling the DMA transfers. Figure 10.15 shows the transfer speed for read and write transactions depending on the transfer size. The transfer size is the size of contiguous memory transferred by the DMA controller in one transfer. The figure shows a clear dependency between the achievable speed and the transfer size. Small transfers are known to be slow, as there is a certain

[1]Direct Memory Access

overhead for starting a transfer. The effect of really large transfer sizes is not yet understood, as the transfer speed is decreasing slightly. The optimal transfer size is 256 kB. The maximum transfer speed has been 142 MB/s for read transactions and 108 MB/s for write transactions.

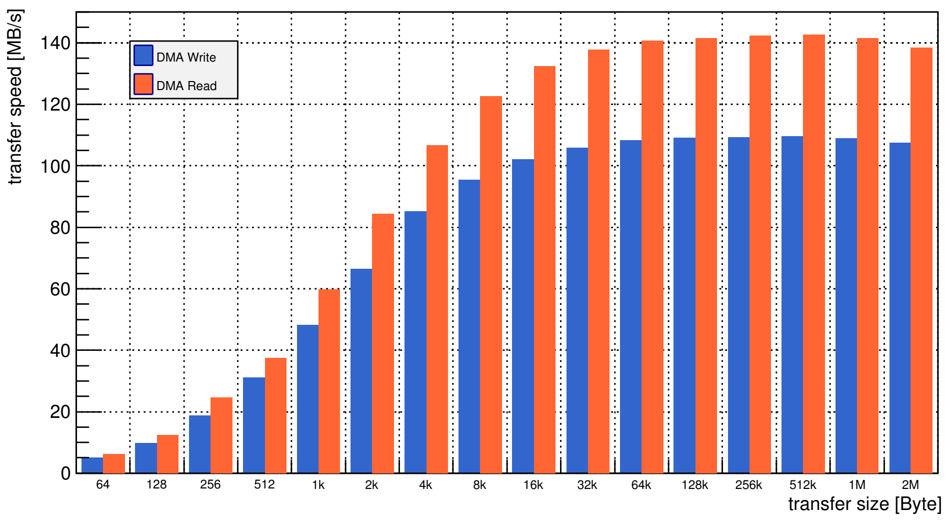


Figure 10.15: Measured transfer speed for read and write transactions with different block sizes.

In summary, the first results of the PCIe development in Wuppertal showed a good performance. Even if all the measurements were performed using a relatively old FPGA-family with a very limited PCIe performance, a scale up of the transfer speed is expected with modern FPGA-families. Therefore, new FPGA evaluation boards have already been ordered and the test design will be ported to these boards. All evaluation cards support PCIe version 2 with 8 PCIe lanes, which should give an increase in throughput of about a factor 16.

11 Summary and Outlook

During the LHC shutdown in 2013-2015 the ATLAS Pixel Detector has been upgraded with a new innermost layer, the IBL. It is equipped with 14 staves carrying 448 newly developed front-end chips. This upgrade adds 12 million pixels to the already installed 80 million pixels. These newly developed front-end chips put new requirements on the off-detector readout system consisting of the ROD and the BOC card.

To meet the new requirements, both off-detector readout cards have been completely redesigned using state-of-the-art FPGA technology. In this thesis studies on the development of the firmware for the IBL BOC card have been presented together with results from testing during the production and in the full readout system. The design of the firmware was driven by the requirements of the detector readout, i.e. the new front-end protocol, which is using the 8b10b encoding, and the timing constraints for the readout.

The hardware of the IBL BOC card uses three FPGAs. One is responsible for the control of the card. The two others handle the data transmission from the detector and to the higher-level readout system. The signal processing firmware in these FPGAs is divided into two major parts. The data path to the detector is sending the TTC information together with the LHC clock signal to the front-end chips using the BPM encoding. To adjust the timing of the detector with respect to the bunch crossing in a sub-ns range, delay lines have been included in the firmware. Different approaches for implementing these delays have been shown and compared. Finally, the fine delay has been implemented by using FPGA internal resources, the IODELAY2 primitives. This implementation provides a fine delay setting with about 35 ps per step. As these primitives do not allow a variable output delay the approach of partial reconfiguration has been used. In the data path from the detector the 8b10b encoding of the front-end chip running at 160 Mbit/s has to be tackled. Implementations for a clock and data recovery algorithm as well as for the decoding of the 8b10b data signals have been presented together with a lot of monitoring features to check the signal integrity.

Having the firmware in hand also the production of the cards could be started and all cards have been tested after the reception from the producer. Results from these production tests have been shown in this thesis as well. During the production test many parameters of the cards like power consumption, optical signal integrity and also the VME backplane have been verified and also first measurements of the fine delay implementation on a larger scale could be performed. By comparing the fine delay slopes of all cards and in all channels it was shown that the spread in the slope is small and all cards have a fine delay slope of about 35 ps per step. Also the other tests during production have shown a good quality of the cards. But some failures during this test were observed as well. These failures were mainly related to Ethernet connection problems and issues with the memory interface of the BCF.

While the Ethernet problems have been solved, the memory issue is still open and the next steps are exchanging the related components to see if the problems will be solved with new components.

The system test, which is verifying the card behaviour in the full readout system, has also been performed in a local readout system in Wuppertal as well as in the CERN setup during the commissioning of the detector. In the local setup mainly calibration scans, like digital or analog injections, have been used to verify the performance of the card. After these tests have been finished showing an overall good performance, the card was heavily tested reading out the detector at CERN. In the CERN setup three major issues have been observed. An instable S-LINK connection to the higher-level readout could be fixed by a firmware change in the BOC firmware and the ROBIN firmware. Also corrupted data signals from the detector have been observed in bad digital injection scans. The firmware has been hardened to reduce the corruption and further analyis lead to an instable algorithm for the clock and data recovery. This algorithm seems to be instable for some of the readout channels and needs manual intervention. But by adjusting the sampling phase manually, also this corruption could be cured. The last issue, which was observed at CERN is an instable timing behaviour of the BOC transmission data path. A serious bug resulted in a non-deterministic timing. This has been fixed by introducing a new transmitter firmware. Also an issue with the hit timing inside the front-end chip has been uncovered. After tuning the hit timing in the front-end chips, the timing of the detector is stable and also efficiency is reasonable after applying the coarse delay timing constants. However, the loading of the fine delays by partial reconfiguration has shown side effects on the firmware stability. The partial reconfiguration impacts other firmware parts and leaves them inoperational after the reconfiguration. To overcome these issues, all fine delay constants will be applied already at the firmware level and so the partial reconfiguration has been dropped for the time being. The next step in this development will be to understand the side-effects and find a way to apply the partial reconfiguration without them.

In summary, the performance of the IBL BOC card is good and it is ready for physics data taking in the next LHC run. In the coming winter break it is forseen to also exchange the readout cards for the Layer-2 of the Pixel Detector. The Layer-2 readout will then also move to the newly developed cards to allow doubling of the current readout bandwidth.

Finally, an outlook on future readout architectures for the next upgrade of the ATLAS Inner Detector has been presented. These architectures are mainly driven by ATLAS-wide developments. Also CERN is pushing towards a unification of the readout architectures. Two projects and additional studies to these projects have been shown: the GBT project and the FELIX project. Currently, the discussion in the ATLAS ITk community about perfect readout system are ongoing. Besides, most of the detector components are not yet chosen, so it is hard to define requirements for the readout. We are looking forward to exciting discussions and decisions about the upcoming ITk readout architecture. These discussion will be supplemented with upcoming studies for future readout architectures, like fast histogramming in FPGAs.

List of Tables

2.1 Overview of the three fundamental forces in the SM 6

4.1 Placement and parameters of the different pixel layers and disks [33] 20
4.2 List of all FE-I4 serial commands 25
4.3 Estimated link occupancy of the Pixel Detector readout links for a trigger rate of 100 kHz. 32

5.1 List of all possible k-words in the 8b10b encoding 38
5.2 Signal description of the S-LINK signals between ROD and BOC card 39

6.1 Mapping of the ELMB channels on the IBL BOC card 50

7.1 Mapping of the Wishbone slaves in the address decoding logic . . . 54
7.2 PLL clock output configuration and usage in the BMF firmware . . 59
7.3 Addressing scheme for the register banks in the BMF firmware. . . . 60
7.4 k-Word sequences used to detect the 8b10b word alignment 72
7.5 Description of some important boclib utilities 77

8.1 Overview of the IBL BOC production batches and their delivery date 79

9.1 Mapping of the RJ45 jacks on the optoboard adapter PCB to TX and RX channels on the BOC card 95
9.2 List of the RODs and BOC cards in the Wuppertal readout setup. . 96

10.1 Distribution of staves and modules in a 4-layer ITk pixel detector layout 115
10.2 Distribution of staves and modules in a 6-layer ITk pixel detector layout 116

List of Figures

2.1 Overview of all particles described by the Standard Model of particle physics. 4
2.2 Overview of fundamental interactions between particles of the SM. . 5
2.3 Feynman diagram of the β^- decay. 6
2.4 Potential energy between two quarks as a function of their distance. 7
2.5 Combined results of the Higgs search 9
2.6 Cross sections and event rates in proton-proton collisions as a function of the collision energy. 11

3.1 Overview of the LHC accelerator complex and the locations where the experiments and LHC services are placed. 14
3.2 Principle of a colliding beam experiment. 14
3.3 Computer-rendered image of the ATLAS Detector 15
3.4 Conceptual design of the ATLAS Detector as a general-purpose particle detector. 16
3.5 ATLAS coordinate system and the placement of the counting rooms. 17
3.6 Trigger and Data Acquisition system. 18

4.1 Computer-rendered image of the ATLAS Pixel Detector 19
4.2 IBL during its insertion into the ATLAS Pixel Detector in May 2014 20
4.3 Overview of the ATLAS IBL Detector Control System. 22
4.4 Overview of the IBL readout system. 23
4.5 Sectional drawing of the two pixel sensor types used in the IBL detector. 24
4.6 Picture of the first prototype IBL stave with placement of the sensor types and DAQ module names. 24
4.7 Overview of the analog part of a pixel cell in the FE-I4 front-end electronics. 25
4.8 Block schematic of the FE-I4 front-end chip. 26
4.9 Picture of the IBL optoboard. 27
4.10 Picture of the ATLAS IBL ROD. 28
4.11 Overview of the IBL scan procedure running on the ROD. 29
4.12 Picture of the latest ROBIN card receiving the detector data in the ROS-PCs 30
4.13 Visualisation of the time walk effect 31

5.1 Example waveform of the BPM encoding 36
5.2 S-LINK interface in the dual-HOLA implementation providing access to the IBL DAQ and the ATLAS FTK project 39
5.3 Setup-Bus communication state machine 40

6.1 Photography of the produced IBL BOC card 44
6.2 Overview of the FPGA connectivity on the IBL BOC card. 45
6.3 Overview of the clock circuit on the IBL BOC card. 46
6.4 First version of the clock circuit as it was used in revisions A, B and C with the CDCE62002 as jitter-cleaner 47
6.5 Production version of the clock circuit with the ISC8634-01 as zero-delay PLL buffer for jitter cleaning 47
6.6 Optical plugins used on the IBL BOC card 48
6.7 Schematics of the interlock circuit on the BOC card. 49
6.8 Picture of the Embedded Local Monitoring Board 49

7.1 Overview block schematic of the BCF firmware. 53
7.2 Finite state machine of the Wishbone arbiter logic for two masters . 54
7.3 Simulated and real waveform of a bus of a concurrent bus access by two masters. 56
7.4 Overview of the BMF firmware blocks. The register bank splits up into three segments for the RX, TX and common firmware parts. . . 58
7.5 SERDESSTROBE generation for clock alignment with an oversampling factor of 8. 59
7.6 Block schematic of the BMF transmitter firmware 60
7.7 Simulation of the BPM encoding example. 61
7.8 Connection of two cascaded SERDES primitives for a serialisation factor of 8. 62
7.9 Block schematic of the MSR delay 63
7.10 Measurement of the relative error rate of the MSR delay. 64
7.11 Usage of the IODELAY2 primitive as a variable input delay. 65
7.12 Measurement of the duty-cycle for BPM encoded '1' (40 MHz) and '0' (20 MHz) signals with and without the IODELAY2 workaround solution. The white area is the operating range of the optoboard from 46 to 54 percent duty-cycle. 65
7.13 Mechanism how to change the fixed output delay setting using partial reconfiguration. 66
7.14 Measurement of the delay and duty-cycle using the partial reconfiguration method for changing the IODELAY2 value. 67
7.15 Simulated waveforms showing the latency in the TX data path. . . . 69
7.16 Simulation of the TX monitoring flagging bit-flip in a trigger command as corrupted word. 70
7.17 Block schematic of the BMF receiver firmware 71
7.18 Round-Robin arbiter for the ROD data bus interface 73
7.19 Block schematic of the FE-I4 emulator in the BOC firmware 74
7.20 Occupancy histogram of an emulator scan. 75
7.21 MGT IP core settings 76

8.1 Overview of the production test setup. 81
8.2 Histogram of the power consumption in the three different operation modes. 81
8.3 Measurement of the delay slope and offset of the global clock circuit. 84

8.4 Measurements of the optical signal integrity during the production test. 85
8.5 Example of a rise- and fall-time measurement for the optical signals with different thresholds. 86
8.6 Measurement of the delay step size for the coarse and fine delay. . . 88
8.7 Chipscope interface with the test patterns received on the BOC card for a manual comparison. 88
8.8 Chipscope interface to measure the bit error rate on the BOC-to-ROD interface 89
8.9 Test of the Setup-Bus access during the production test of the BOC card 89
8.10 Reason for Ethernet problems on the BOC card in the fourth production batch 91

9.1 Picture of the IBL readout setup in Wuppertal 94
9.2 Picture of an IBL double-chip module with its adapter PCB 94
9.3 Screenshot of the TDAQ software when running the local lab software infrastructure 97
9.4 Screenshot of the calibration console window 97
9.5 Occupancy histogram of a scan with the emulator in the BOC card. 98
9.6 Digital scan result using a real front-end chip connected through the optical chain to the BOC card 99
9.7 Analog injection scan result using a real front-end chip connected through the optical chain to the BOC card 100
9.8 Picture of a corrupted digital injection scan. 101
9.9 Waveform of the 8b10b decoder input and output signals with signs of FE-I4 data corruption. 102
9.10 Comparison of the input and output signals of the word alignment. 102
9.11 Chipscope waveform of a non-working S-LINK channel. 104
9.12 Example of the four different alignments for the clock enable pulses in the old TX data path. 106
9.13 Mean hit arrival time (L1A) during the M9 cosmic run for IBL-stave 9.107
9.14 Time between hit injection and detection as function of the discriminator bias voltage setting. 108
9.15 Time difference in the hit timing between two front-end chips sharing the same TTC line (ΔT_0). 108
9.16 Verification of the DisVbn tuning and in-time efficiency. 109

10.1 Overview of the LHC upgrade plans until 2025. 113
10.2 Sectional drawing of 1/4th of the ITk baseline layout. 114
10.3 Overview of the ATLAS Level-0 and Level-1 trigger system after the phase II upgrade. 117
10.4 Overview of the GBT and Versatile Link project. 117
10.5 Structure of a GBT frame. 118
10.6 Block diagram of the GBT encoding and decoding data path. 118
10.7 Picture of the GBT Link Interface Board for evaluating the GBT firmware. 119

10.8 Block schematic and resource usage of the GBT test firmware running on the GLIB with four GBT links. 120
10.9 Chipscope waveform of a single GBT link during plugging of the optical fibre. 120
10.10 Evolution of the readout architecture with the introduction of the FELIX concept. 121
10.11 Overview of the FELIX readout architecturen as a switch between GBT and Ethernet. 122
10.12 Overview of different options for integration of the ITk calibration into the FELIX architecture. 123
10.13 Spartan-6 FPGA evaluation card for PCIe development 126
10.14 Block schematic of the PCIe test firmware for the SP605 evaluation card. 126
10.15 Measured transfer speed for read and write transactions with different block sizes. 127

List of Abbreviations

AXI	Advanced eXtensible Interface	57
BCF	BOC Control FPGA	43
BMF	BOC Main FPGA	43
BOC	Back-of-Crate	1
BPM	biphase-mark	27
CERN	European Organization for Nuclear Research	1
DAQ	Data Acquisition	1
DCS	Detector Control System	21
DORIC	Digital Optical Receiver IC	27
FELIX	Front-End Link Interface eXchange	121
FIFO	First-In First-Out	51
FMC	FPGA Mezzanine Card	45
FPGA	Field Programmable Gate Array	2
GBT	GigaBit Transceiver	115
HOLA	High-speed Optical Link for ATLAS	38
IBL	Insertable b-Layer	1
ITk	Inner Tracker	113
JTAG	Joint Test Action Group	45
LHC	Large Hadron Collider	1
MGT	Multi-Gigabit-Transceiver	43
PCIe	Peripheral Component Interconnect Express	30
PLL	Phase-Locked-Loop	40
QSFP	Quad Small Form-factor Pluggable	39
ROBIN	Readout-Buffer-Input	30
ROD	Readout-Driver	1
ROS	Readout-Subsystem	18
SERDES	Serialiser/Deserialiser	62
SFP	Small Form-factor Pluggable	117
SM	Standard Model	1
SPI	Serial Peripheral Interface	46

TDAQ	Trigger and Data Acquisition	17
TIM	Timing-Trigger-Control Interface Module	27
ToT	time-over-threshold	23
TTC	Timing-Trigger-Control	18
VCSEL	vertical-cavity surface-emitting laser	27
VME	Versa Module Eurocard	21

Bibliography

[1] P. Mohr et al. CODATA Recommended Values of the Fundamental Physical Constants: 2010. *Rev. Mod. Phys.*, 84, 2012. URL `http://arxiv.org/abs/1203.5425`.

[2] D. Perkins. *Introduction to High Energy Physics.* Cambridge University Press, 4th edition, 2000.

[3] F. Halzen and A. Martin. *Quarks and Leptons: An Introductory Course in Modern Particle Physics.* John Wiley & Sons, 1984.

[4] Wikimedia Commons. Standard Model of Elementary Particles. URL `http://commons.wikimedia.org/w/index.php?title=File:Standard_Model_of_Elementary_Particles.svg&oldid=137710639`.

[5] C.L. Cowan, F. Reines, et al. Detection of the Free Neutrino: a Confirmation. *Science Journal*, 124(3212), July 1956.

[6] L. Püllen. *Development of a Detector Control System for the serially powered ATLAS pixel detector at the HL-LHC.* PhD thesis, University of Wuppertal, February 2015.

[7] D. Meschede. *Gerthsen Physik.* Springer, 25th edition, 2015.

[8] S.L. Glashow. Partial Symmetries of Weak Interactions. *Nucl. Phys.*, 22, 1961.

[9] J. Goldstone, A. Salam, and S. Weinberg. Broken Symmetries. *Phys. Rev.*, 127, 1962.

[10] W. Demtröder. *Experimentalphysik 4. Kern-, Teilchen und Astrophysik.* Springer Spektrum, 4th edition, 2014.

[11] Z.Q. Liu et al. Study of $e^+e^- \to \pi^+\pi^- J/\psi$ and Observation of a Charged Charmoniumlike State at Belle. *Phys. Rev. Lett.*, 110, June 2013. doi: 10.1103/PhysRevLett.110.252002. URL `http://link.aps.org/doi/10.1103/PhysRevLett.110.252002`.

[12] LHCb collaboration. Observation of the resonant character of the $Z(4430)^-$ state. *Phys. Rev. Lett.*, 112, 2014. URL `http://arxiv.org/abs/1404.1903`.

[13] LHCb collaboration. Observation of J/ψ resonances consistent with pentaquark states in $\Lambda_b^0 \to J/\psi K^- p$ decays. July 2015. URL `http://arxiv.org/abs/1507.03414`.

[14] P. Higgs. Broken Symmetries and the Masses of Gauge Bosons. *Physical Review Letters*, 13(16), 1964.

[15] F. Englert and R. Brout. Broken Symmetry and the Mass of Gauge Vector Mesons. *Physical Review Letters*, 13(9), 1964.

[16] ATLAS Collaboration. Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC. *Phys. Lett. B*, 716, 2012. URL `http://arxiv.org/pdf/1207.7214`.

[17] CMS Collaboration. Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC. *Phys. Lett. B*, 716, 2012. URL `http://arxiv.org/abs/1207.7235`.

[18] Planck Collaboration. Planck 2013 results. I. Overview of products and scientific results. *Astronomy and Astrophysics*, 1303, 2013. URL `http://arxiv.org/abs/1303.5062`.

[19] B. Povh et al. *Particles and nuclei.* Springer, 6th edition, 2008.

[20] P. Jenni, M. Nessi, M. Nordberg, and K. Smith. *ATLAS high-level trigger, data-acquisition and controls: Technical Design Report.* Technical Design Report ATLAS. CERN, Geneva, 2003. URL `http://cds.cern.ch/record/616089`.

[21] L. Evans and P. Bryant. LHC Machine. *JINST*, 3(08):S08001, 2008. URL `http://stacks.iop.org/1748-0221/3/i=08/a=S08001`.

[22] Jean-Luc Caron. Layout of the LEP tunnel including future LHC infrastructures. AC Collection. Legacy of AC. Pictures from 1992 to 2002., Feb 1997. URL `http://cds.cern.ch/record/841560`.

[23] R. Alemany-Fernandez et al. Operation and Configuration of the LHC in Run 1. November 2013. ATLAS internal note.

[24] ATLAS Collaboration. ATLAS detector and physics performance: Technical Design Report, 1. Technical Report CERN-LHCC-99-014. ATLAS-TDR-14, CERN, 1999.

[25] J. Pequenao. Computer generated image of the ATLAS detector, March 2008.

[26] J. Pequenao. An computer generated image representing how ATLAS detects particles, January 2013.

[27] L. Bouvet et al. ATLAS Inner Detector Services: Parameters describing the orientation naming and layout of Inner Detector Services, 2005. EDMS: ATL-IC-EP-0017.

[28] C.-Y. Wong. *Introduction to High-energy Heavy-ion Collisions.* World Scientific, 1994.

[29] W. Buttinger. The ATLAS Level-1 Trigger System. *Jornal of Physics: Conference Series*, 396(1):012010, 2012. URL `http://stacks.iop.org/1742-6596/396/i=1/a=012010`.

[30] ATLAS Collaboration. ATLAS Fact Sheet. URL `http://www.atlas.ch/pdf/ATLAS_fact_sheets.pdf`.

[31] ATLAS Collaboration. *ATLAS pixel detector: Technical Design Report.* Technical Design Report ATLAS. CERN, Geneva, 1998. URL `https://cds.cern.ch/record/381263`.

[32] J. Pequenao. Computer generated image of the ATLAS pixel detector, March 2008.

[33] ATLAS Collaboration. ATLAS Insertable B-Layer Technical Design Report. Technical Report CERN-LHCC-2010-013. ATLAS-TDR-19, CERN, September 2010.

[34] ATLAS Collaboration. ATLAS Insertable B-Layer Technical Design Report Addendum. Technical Report CERN-LHCC-2012-009. ATLAS-TDR-19-ADD-1, CERN, May 2012.

[35] C. Marcelloni De Oliveira. Photography of the IBL insertion into ATLAS, May 2014.

[36] S. Kersten et al. Detector Control System of the ATLAS Insertable B-Layer. In *13th International Conference on Accelerator and Large Experimental Physics Control Systems*, October 2011.

[37] T. Flick. *Studies on the Optical Readout for the ATLAS Pixel Detector.* PhD thesis, University of Wuppertal, July 2006.

[38] C. Da Via. Silicon sensors go 3D. *CERN Courier*, May 2012. URL `http://cerncourier.com/cws/article/cern/49691`.

[39] ATLAS IBL collaboration. Prototype ATLAS IBL modules using the FE-I4A front-end readout chip. *JINST*, 7(11):P11010, 2012. URL `http://stacks.iop.org/1748-0221/7/i=11/a=P11010`.

[40] B. Mandelli. Final characterization of the ATLAS IBL detector modules with ^{224}Am during the construction phase. In *IEEE Nuclear Science Symposium and Medical Imaging Conference*, November 2012.

[41] K.E. Arms et al. ATLAS Pixel Opto-Electronics. *Nucl. Instr. Meth*, 2005. URL `http://arxiv.org/pdf/physics/0504142`.

[42] R. Kass et al. The new radiation-hard optical links for the ATLAS pixel detector. In *Meeting of the American Physical Society Division of Particles and Fields*, October 2013. URL `http://arxiv.org/pdf/1310.1062.pdf`.

[43] N. Krieger. *Development of the readout for the IBL Upgrade Project of the ATLAS Pixel Detector.* PhD thesis, University of Göttingen, 2012.

[44] M. Červ. The ATLAS Diamond Beam Monitor. *JINST*, 9(02):C02026, 2013. URL `http://iopscience.iop.org/1748-0221/9/02/C02026`.

[45] H. C. van der Bij et al. S-LINK, a data link interface specification for the LHC era. *IEEE Transactions on Nuclear Science*, 44:398–402, June 1997.

[46] I. Peric. *Design and Realisation of Integrated Circuits for the Readout of Pixel Sensors in High-Energy Physics and Biomedical Imaging.* PhD thesis, University of Bonn, 2004.

[47] S. Welch and J. Dopke. The ATLAS Pixel nSQP Readout Chain. October 2012.

[48] T. Flick. Back-end Electronics Status and Services. ATLAS Week Montreux, 2012. URL `https://indico.cern.ch/event/209427/session/0/contribution/5/attachments/324961/453173/Pixel_off-detector_Services.pdf`.

[49] B. Bergenthal. Firmware development for the IBL back of crate card to be used with the pixel-detector hardware. Bachelor thesis, University of Wuppertal, July 2015.

[50] *VMEbus Specification Manual*, October 1985. URL `https://www.mpp.mpg.de/~huber/vme/vmebus.pdf`. Third Printing.

[51] R. Forster. Manchester encoding: opposing definitions resolved. *Engineering Science and Education Journal*, 9, Dec 2000.

[52] S. Baron et al. Jitter impact on clock distribution in LHC experiments. *JINST*, 7:C12023, 2012. URL `http://iopscience.iop.org/1748-0221/7/12/C12023`.

[53] SNAP12 Specifications. Appendix to SNAP12 Multi-Source Agreement, May 2002. URL `http://www.physik.uzh.ch/~avollhar/snap12msa_051502.pdf`.

[54] S. Welch et al. IBL Optical Link Power Budget, 2013. Internal Document ATL-IP-ES-0200.

[55] A. X. Widmer and P. A. Franaszek. A DC-Balanced, Partitioned-Block, 8B/10B Transmission Code. *IBM Journal of Research and Development*, 27: 440–451, September 1983.

[56] M. Shochet et al. Fast TracKer (FTK) Technical Design Report. Technical Report CERN-LHCC-2013-007. ATLAS-TDR-021, CERN, June 2003.

[57] M. Goodrick. BOC - The Back of Crate interface for the SCT ROD. Technical report, Cambridge, 2003. Internal Report.

[58] M.S. Cooke. Studies of VCSEL Failures in the Optical Readout Systems of the ATLAS Silicon Trackers and Liquid Argon Calorimeters. *Proceedings of the DPF-2011 Conference*, August 2011. URL `http://arxiv.org/pdf/1109.6679.pdf`.

[59] N. Schroer. *Design and Evaluation of the IBL BOC for the ATLAS Experiment at CERN.* PhD thesis, University of Mannheim, December 2012.

[60] Xilinx Corp. *Spartan-6 Family Overview.* DS160. October 2011. URL `http://www.xilinx.com/support/documentation/data_sheets/ds160.pdf`.

[61] FMC: FPGA Mezzanine Cards Base Specification, 2008.

[62] Texas Instruments. CDCE62002 datasheet. URL `http://www.ti.com/lit/ds/symlink/cdce62002.pdf`.

[63] A. Cervelli et al. Report on test over SNAP12 plugins for the ATLAS Pixel IBL, 2012. Internal Note ATL-IP-TR-0009.

[64] B. Hallgren et al. The Embedded Local Monitor Board (ELMB) in the LHC Front-end I/O Control System. In *7th Workshop on Electronics for LHC Experiments*, September 2001.

[65] H. Burckhart and B. Hallgren. ATLAS DCS: Embedded Local Monitoring Board, December 2000. URL `http://elmb.web.cern.ch/ELMB/ELMB_5/elmb_5.html`.

[66] GHDL Homepage. URL `http://ghdl.free.fr/`. at the time of writing.

[67] Open Hardware Repository. URL `http://www.ohwr.org`.

[68] OpenCores. Wishbone B4: WISHBONE System-On-Chip (SoC) Interconnection Architecture for Portable IP Cores. Technical report, 2010. URL `http://cdn.opencores.org/downloads/wbspec_b4.pdf`.

[69] R. White and A. Khu, Xilinx Corp. *Embedded JTAG ACE Player.* XAPP424. April 2008.

[70] Xilinx Corp. MicroBlaze Soft Processor Core. URL `http://www.xilinx.com/tools/microblaze.htm`.

[71] ARM Ltd. AMBA Specifications. URL `http://www.arm.com/products/system-ip/amba-specifications.php`.

[72] Xilinx Corp. OS and Libraries Document Collection (UG643). URL `http://www.xilinx.com/support/documentation/sw_manuals/xilinx2015_2/oslib_rm.pdf`.

[73] lwIP Wiki. URL `http://lwip.wikia.com/wiki/LwIP_Wiki`.

[74] IPbus. URL `https://svnweb.cern.ch/trac/cactus`.

[75] Xilinx Corp. *Spartan-6 FPGA Clocking Resources. User Guide.* UG382. December 2013. URL `http://www.xilinx.com/support/documentation/user_guides/ug382.pdf`.

[76] Xilinx Corp. *Spartan-6 FPGA SelectIO Resources. User Guide.* UG381. February 2014. URL http://www.xilinx.com/support/documentation/user_guides/ug381.pdf.

[77] T. Heim. Design and development of the IBL-BOC firmware for the ATLAS Pixel IBL optical datalink system. Diploma thesis, University of Wuppertal, 2011.

[78] A. Weidberg et al. BPM12 - Project Specification. Technical report, November 1999. URL http://www-pnp.physics.ox.ac.uk/~weidberg/bpm12_specs.pdf.

[79] Xilinx Corp. AR 34276. Spartan-6 FPGA - Can the IODELAY2 be used to delay an output in Variable Mode, 2013. URL http://www.xilinx.com/support/answers/34276.html.

[80] M. Wensing et al. Firmware development and testing of the ATLAS IBL Back-Of-Crate card. *PoS(TIPP2014)216*, 2014.

[81] P. Vo, Xilinx Corp. *Parameterizable 8b/10b Encoder.* XAPP1122. November 2008.

[82] N. Sawyer, Xilinx Corp. *Data to Clock Phase Alignment.* XAPP225. February 2009.

[83] A. Ruiz and E. van der Bij. HOLA S-LINK. Hardware Specification. Technical report, June 2002. URL http://hsi.web.cern.ch/HSI/s-link/devices/hola/hw_spec.pdf.

[84] Texas Instruments. TLK2501: 1.5 to 2.5 Gbps transceiver, 2000. URL http://www.ti.com/lit/ds/symlink/tlk2501.pdf.

[85] ON Semiconductor. MC100EP195B datasheet. URL http://www.onsemi.com/pub_link/Collateral/MC100EP195B-D.PDF.

[86] N. Sawyer, Xilinx Corp. *Data Recovery.* XAPP224. July 2005.

[87] D. Elledge. The Relationship between Time Delays and DisVbn, Choosing a DisVbn, and Characterizing Time Walk on the FE-I4 Prot-1 Test Chip. Technical report. internal document.

[88] HL-LHC: High Luminosity Large Hadron Collider. URL http://hilumilhc.web.cern.ch/.

[89] *The ATLAS upgrade program*, 2014. URL http://arxiv.org/abs/1409.5002.

[90] ATLAS Collaboration. Letter of Intent for the Phase-II Upgrade of the ATLAS Experiment. Technical Report CERN-LHCC-2012-022. LHCC-I-023, CERN, Geneva, Dec 2012. URL https://cds.cern.ch/record/1502664.

[91] F. Vasey et al. The versatile link, a common project for super-LHC. *JINST*, 4:P12003, 2009.

[92] P. Moreira et al. The GBT Project. 2009. URL `https://cds.cern.ch/record/1235836`.

[93] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 1960.

[94] S Baron et al. Implementing the GBT data transmission protocol in FPGAs. 2009. URL `http://cds.cern.ch/record/1236361`.

[95] P. Vichoudis et al. The Gigabit Link Interface Board (GLIB) ecosystem. *JINST*, 8:C03012, 2012. URL `http://inspirehep.net/record/1226314?ln=de`.

[96] R. Bartoldus et al. Technical Design Report for the Phase-I Upgrade of the ATLAS TDAQ System. Technical Report CERN-LHCC-2013-018. ATLAS-TDR-023, CERN, Geneva, 2013. URL `https://cds.cern.ch/record/1602235`.

[97] L. Levinson. A new approach to interfacing on-detector electronics. Poster presentation, TWEPP2013.

[98] Private communication with Lorne Levinson.

[99] Infiniband Trade Association. URL `http://www.infinibandta.org/`.

[100] F. Carrio et al. The sROD module for the ATLAS Tile Calorimeter Phase-II Upgrade Demonstrator. *JINST*, page C02019, 2014.

[101] YARR: Yet Another Rapid Readout. URL `http://yarr.web.cern.ch/yarr/`.