Data Flow in the Mu3e Filter Farm

von

Marius Köppel

Masterarbeit in Physik
vorgelegt dem Fachbereich Physik, Mathematik und Informatik (FB 08)
der Johannes Gutenberg-Universität Mainz
am 26. September 2019

1. Gutachter: Prof. Dr. Niklaus Berger
2. Gutachter: Prof. Dr. Patrick Achenbach

Ich versichere, dass ich die Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Mainz, den 26. September 2019

Marius Köppel
Mu3e collaboration
Institut für Kernphysik
Johann-Joachim-Becher-Weg 45
Johannes Gutenberg-Universität D-55128 Mainz
makoeppe@students.uni-mainz.de

Abstract

The Mu3e experiment at the Paul Scherrer Institute searches for the decay $\mu^+ \to e^+e^+e^-$. This decay violates charged lepton flavour conservation - so an observation would be a clear indication for Physics Beyond the Standard Model of particle physics. The Mu3e experiment aims for an ultimate sensitivity of one in 10^{16} μ decays. To this end, more than one billion μ tracks per second need to be detected and reconstructed.

Since the corresponding data of about 1 TB/s cannot be saved to disk, a trigger-less online readout system needs to be designed which is able to analyze the data while running. A farm with PCs equipped with powerful graphics processing units will perform the data reduction. In this work the missing pieces of the Field Programmable Gate Array based system which is used to preprocess, sort and transport the data from the detector to the filter farm and also configures the detector are designed and tested.

Zusammenfassung

Das Mu3 Experiment am Paul Scherrer Institut sucht nach dem Zerfall $\mu^+ \rightarrow e^+e^+e^-$. Dieser Zerfall würde die geladene Leptonenfamilienzahlerhaltung brechen. Eine solche Beobachtung wäre ein klarer Indikator für Physik jenseits des Standardmodells der Elementarteilchenphysik. Das Mu3e Experiment strebt eine oberste Sensitivität von einem in 10^{16} μ Zerfällen an. Dafür müssen mehr als eine Milliarde Teilchenspuren pro Sekunde detektiert und rekonstruiert werden.

Da die dafür benötigten Daten von 1 TB/s nicht auf einer Festplatte gespeichert werden können, muss ein Trigger-loses online Auslesesystem entwickelt werden, welches in der Lage ist die Daten während der Laufzeit zu analysieren. Ein Rechenzentrum, dessen Computer mit leistungsstarken Grafikprozessoren ausgestattet sind, soll die Datenrekonstruktion durchführen. In dieser Arbeit wurden die fehlenden Teile des auf im Feld programmierbaren Logik-Gatter-Anordnung basierenden Systems, welches zur Vorverarbeitung, Sortierung und Transportierung der Daten vom Detektor zum Rechenzentrum und auch zur Konfiguration des Detektors verwendet wird, entwickelt und getestet.

Contents

I. Introduction **1**

1. Theory **3**
1.1. The standard model of particle physics . . . 3
1.2. muon (μ) decays . . . 5

2. Principles of Particle Detectors **9**
2.1. Silicon detectors . . . 9
2.2. Principle of tracking detectors . . . 10
2.3. Multiple Scattering . . . 11
2.4. High Voltage Monolithic Active Pixel Sensors . . . 12
2.5. Scintillating Fibres . . . 13

3. Field Programmable Gate Arrays **15**

4. Data Transmission **17**
4.1. Basics of data transmission . . . 17
4.2. Electrical data transmission . . . 19
4.3. Serial data links . . . 20
4.3.1. Physical layer . . . 20
4.3.2. Data link layer . . . 21
4.4. High speed links in field programmable gate arrays (FPGAs) . . . 23
4.4.1. GX Transceiver . . . 23
4.4.2. Intel Arria 10 Transceiver PHY Layer . . . 24
4.4.3. Transceiver design IP blocks . . . 25
4.4.4. Transceiver Channel datapath and clocking . . . 25
4.5. Optical data transmission . . . 27
4.6. Bit error rate tests (BERTs) . . . 27

II. The Data Acquisition of the Mu3e Experiment **29**

5. The Mu3e Experiment **31**
5.1. Signal and background processes . . . 32
5.2. Detector concept . . . 32
5.3. Pixel detector . . . 33
5.4. Fibre detector . . . 34
5.5. Tile detector . . . 35

6. Data Acquisition System **37**
6.1. Expected data rate . . . 37
6.2. Readout system . . . 38
6.2.1. Front-end board . . . 38
6.2.2. Switching board . . . 39
6.2.3. PC interface board . . . 40
6.2.4. Farm PC . . . 40
6.2.5. Clock and reset system . . . 40
6.3. Maximum Integrated Data Acquisition System (MIDAS) . . . 41

III. Test Setup **43**

7. Communication Protocols **45**
7.1. Communication front-end boards to switching boards . . . 45
7.1.1. MuPix communication protocol . . . 45
7.1.2. MuTRiG communication protocol . . . 46
7.1.3. Slow control communication protocol . . . 47
7.1.4. Run control signals . . . 47
7.1.5. Idle state . . . 48
7.2. Communication of clock and reset system . . . 48

8. Data Readout via PCIe **51**
8.1. Peripheral Component Interconnect Express . . . 51
8.2. Direct Memory Access . . . 53
8.3. Direct Memory Access block implementation . . . 54
8.4. Direct Memory Access rate measurements . . . 55

9. Data Flow and Firmware Design **57**
9.1. Sensor printed circuit boards (PCBs) . . . 57
9.2. Communication inside the front-end boards . . . 57
9.3. Front-end board to switching board . . . 58
9.4. Switching board to PCIe board . . . 59
9.5. Slow control from switching board to front-end board . . . 60

10. Test Setup **63**
10.1. Simulation of the slow control entities . . . 65
10.2. Simulation of the data readout entities . . . 66
10.3. Slow control tests . . . 67
10.4. Data readout tests . . . 69

11. Conclusion and Outlook **71**

A. Acronyms **75**

B. Simulations **79**

C. Register Transfer Level (RTL) Designs **81**

List of Figures **91**

List of Tables **93**

Bibliography **95**

Acknowledgement **101**

Part I.

Introduction

In the summer of 2012, the Higgs boson was discovered by the ATLAS [1] and the CMS [2] experiment at the CERN Large Hadron Collider (LHC) [3] completing the Standard Model of particle physics (SM) to a self consistent theory. The SM is able to describe the nature at subatomic scales with great precision. All models describing the physics on earth and everyday life can be derived from the SM and general relativity. The SM also explains most particle physics experiments of the last century.

However, there are a few things which are not explained by the SM, e.g. the existence of dark matter or the asymmetry between baryon and anti-baryons in our universe. Also some of the properties of the SM are not satisfying, e.g. there are a large number of free parameters which need to be measured and the theory of gravity is not includable in the SM. The observation of neutrino oscillations [4, 5, 6] arouses the suspicion that also also the charged lepton flavor can be violated. Observing a decay which breaks this symmetry would be a clear hint for physics beyond the SM. Mu3e is one of the experiments searching for the decay $\mu^+ \to e^- e^+ e^+$.

Since this process is quite rare in many theories beyond the SM, a high sensitivity is important. This kind of experiment has to observe an immense rate of physical events leading to the problem that the data acquisition system (DAQ) needs to be able to handle them. A key feature is to reduce the data during the readout process by the use of field programmable gate arrays (FPGAs).

In the following part a theoretical introduction to charged lepton flavor violation (LFV) is given. This is followed by a short explanation of the used detector principles in the Mu3e experiment and an overview of the FPGA architecture. At the end the data transmission based on electrodynamics and their implementation on the FPGA is described.

1 Theory

Particle physics is aiming to unravel the fundamental laws of nature on a microscopic scale. The current status of this effort is summarized in the SM. This theory is a relativistic quantum field theory (QFT). More specifically it is a U(1) × SU(2) × SU(3) gauge theory, where nature is described in terms of elementary particles and their interactions.

In section 1.1 a more detailed discussion about the Standard Model of particle physics (SM) based on [7] is given. Because of the relevance for this thesis, the muon (μ) decays in the SM and beyond the SM are discussed in section 1.2.

1.1. The standard model of particle physics

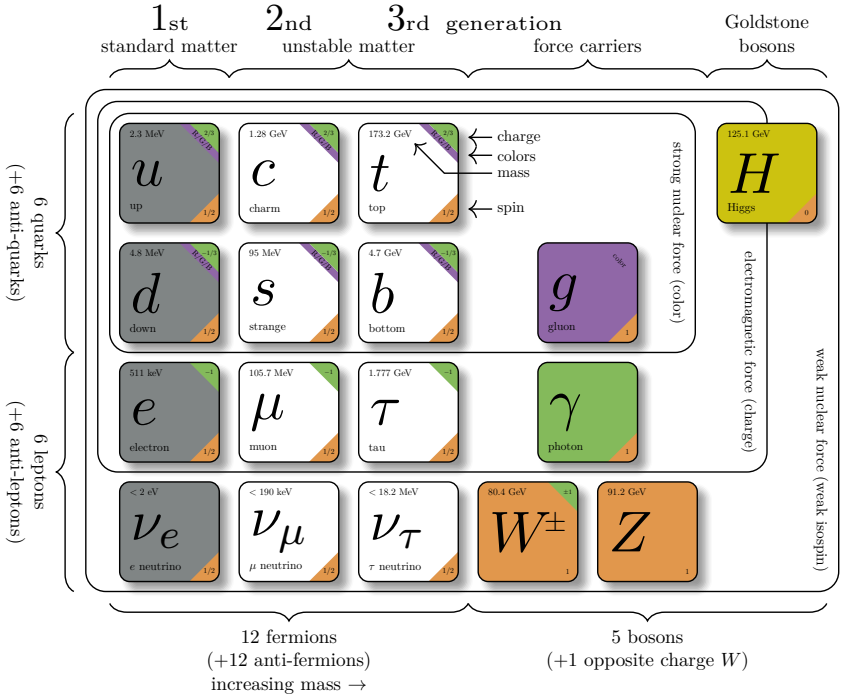


Figure 1.1.: Overview of the SM, taken from [8].

Every particle can be seen as a field in the SM. These particles are shown in fig. 1.1. They can be split into fermions and bosons. The SM contains 12 fermions and their antiparticles which are spin $1/2$ particles and build up matter and antimatter. On the other hand there are 4 vector bosons with spin 1 and a scalar boson with spin 0 called Higgs boson. The vector bosons are responsible for the interactions between the particles and the Higgs boson is the pseudo Goldstone-Boson of the spontaneous symmetry breaking of the Higgs mechanism, producing the mass of the particles.

The fermions are split into 3 generations and into quarks and leptons. The first generation of the quarks contains the up quark (u) and the down quark (d). With the electron, which is part of the first generation of leptons, they form the stable matter in our universe. The other generations are unstable. Each generation of quarks has one quark with electrical charge $q = 2/3$ and one with $q = -1/3$. The second generation contains the charm quark (c) and the strange strange quark (s) and the third one the top quark (t) and the bottom quark (b). Increasing the generation leads also to higher particle masses. The u-quark for example has a mass of 2.3 MeV and the t-quark comes with a mass of 173.2 GeV which is five orders of magnitude higher. Besides, all quarks can interact via the strong, weak and electromagnetic interactions, because they all have color charge, weak isospin and electric charge. The lepton generations are split into electron (e), muon (μ) and tau (τ), which all have an electric charge and three generations of electron neutrino (ν_e), muon neutrino (ν_μ) and tau neutrino (ν_τ), which have no electric charge. The neutrinos do not have a mass in the SM.

Interactions in the SM are described by 12 vector boson fields. There are 8 gluons (g's) mediating the strong interaction, 2 charged $W^\pm$ and the neutral Z boson which carry the weak interaction and the photon (γ) which is mediating the electromagnetic interaction. The g's and the γ do not have a mass, while the $W^\pm$ has a mass of $80.4\,\mathrm{GeV/c^2}$ and the Z bosons has a mass of $91.2\,\mathrm{GeV/c^2}$ [9]. Since the SM is a U(1) $\times$ SU(2) $\times$ SU(3) gauge theory the boson fields arise under the local gauge invariance of the SM particle fields. The strong interaction arises due to the symmetry of the local SU(3) transformation.

The weak and the electromagnetic force can be unified to the electroweak interaction because they contain a neutral gauge boson [10]. In the SM this unification is described in the Weinberg–Salam theory as a Yang–Mills field with an SU(2) $\times$ U(1) gauge group. The Higgs mechanism breaks this U(1) $\times$ SU(2) symmetry spontaneously and gives a non-zero mass to the $W^\pm$ and the Z bosons [11]. This mechanism also leads to another massive particle which is the Higgs boson and which was the last particle to be detected of the SM. This boson was discovered at the LHC in 2012 [1, 2]. Since fermion fields couple through Yukawa interactions to the Higgs field, the fermions are also getting their mass from the Higgs field.

With all there parts the SM is able to describe physical processes up to the scale of the electroweak interactions with high precision. However, there 18 free parameters need to be measured for achieving this.

Besides there theoretical issuses, there are a few experimental observations which can not be described by the SM. For examples, it is not known what the dark matter is. Beside the visible matter, the universe also contains invisible matter, the so called dark matter. This matter explains the measured velocity distributions of stars orbiting around the galaxies [12]. It was also observed that neutrinos mix between their generations [13]. This does break the lepton family conservation which is a symmetry in the SM and leads to the conclusion that

the neutrinos are not massless.

There are several other observations which can not be described by the SM. To develop a theory beyond the SM, particle physics experiments can search for observable deviations from the SM predictions. One clear hint for new physics would be charged lepton flavor violation (LFV). To observe this, the decay $\mu^+ \to e^- e^+ e^+$ can be studied.

1.2. μ decays

For understanding why this decay is interesting, a few words on μ decays in the SM and beyond the SM are given in the following.

μ decays in the Standard Model

In the SM, the leading order μ decay is the Michel decay $\mu^+ \to e^+ \, \overline{\nu}_\mu \, \nu_e$ which is shown in fig. 1.2. The next to leading order process is the radiative decay $\mu^+ \to \gamma e^+ \, \overline{\nu}_\mu \, \nu_e$ with a branching fraction (BF) of $1.4(4) \times 10^{-2}$ for photon energies above 10 MeV [14]. The γ which is emitted in the radiative decay can converted into an electron-positron pair $\mu^+ \to e^+ e^+ e^- \, \overline{\nu}_\mu \, \nu_e$. This decay is called internal conversion with a BF of $3.4(4) \times 10^{-5}$ [9].

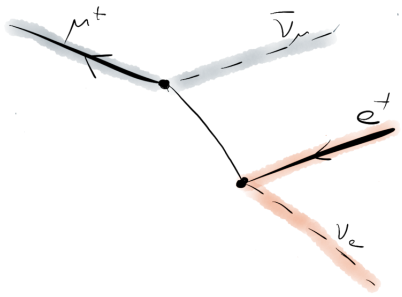


Figure 1.2.: Michel decay of the μ into an e, an ν_e and an anti ν_μ.

The decay $\mu^+ \to e^- e^+ e^+$ is forbidden in the SM since it does not conserve lepton flavor. By extending the SM with neutrino oscillations the decay can be possible. This decay is shown in fig. 1.3.

The BF of this decay is 2.1×10^{-55} in the extension of the SM by the neutrino oscillation [15]. Because of this tiny probability, the decay is nearly impossible to observe in finite time.

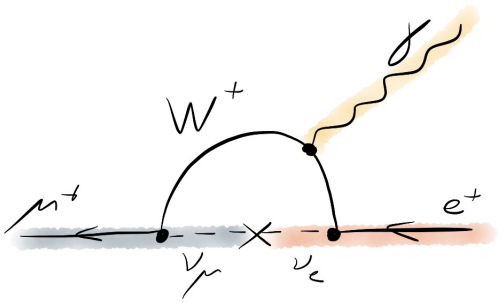


Figure 1.3.: μ decay over the exchange of an ν_e and ν_μ in the expansion of the SM by neutrino oscillation.

μ decays beyond the Standard Model

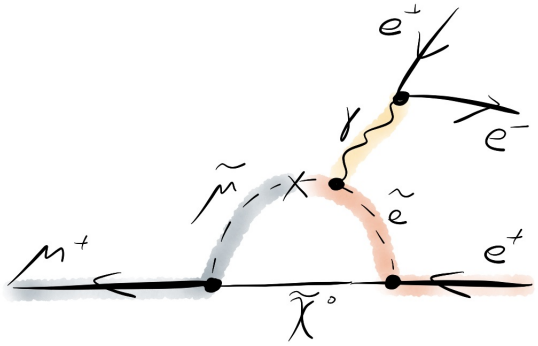


Figure 1.4.: μ decay via Super Symmetry (SUSY) particles.

Beyond the SM there are theories which predict LFV decays at BFs possible to reach experimentally. One of these theories is Super Symmetry (SUSY) [16]. In this theory the decay $\mu^+ \to e^- e^+ e^+$ could be possible via a quantum loop as seen in fig. 1.4 [17].

Another example is the Scotogenic Model which was first proposed by Ernest Ma [19]. This model could provide a dark matter candidate as well an explanation for the neutrino masses. This model adds three right-handed neutrinos (N_i) and a scalar $\mathrm{SU}(2)_L \times \mathrm{U}(1)_Y$ doublet to the SM. The SM neutrino masses are generated via a radiative seesaw machanism. This leads to small SM neutrino masses without requiring very massive sterile neutrinos. The general

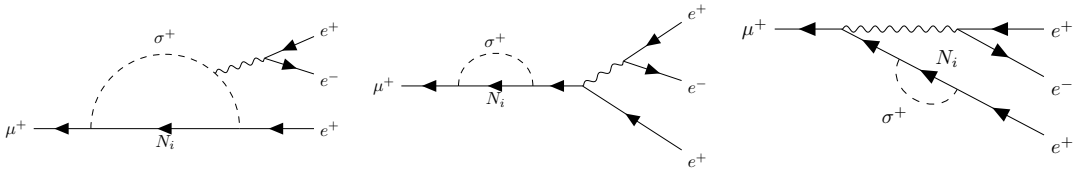


Figure 1.5.: Possible production channels for the decay $\mu^+ \to e^- e^+ e^+$ considering the Scotogenic Model (cf. [18]) by the loop of the scalar doublet σ^+.

radiative seesaw machanism is explained in [20]. Since these new neutrinos N_i are sterile, they do not carry any lepton number so they are able to break lepton flavor conservation. In fig. 1.5 possible Feynman diagrams for the decay $\mu^+ \to e^- e^+ e^+$ with this model are shown.

All these theories are constrained by experimental limits. The most important are $\mu \to e\gamma$ measured by the Mu to E Gamma (MEG) experiment with a BF, $\mathcal{B}(\mu \to e\gamma) < 4.2 \times 10^{-13}$ at 90 % confidence level (CL) [21], $\mu^+ \to e^- e^+ e^+$ measured by the SINDRUM experiment with a BF, $\mathcal{B}(\mu^+ \to e^- e^+ e^+) < 1.0 \times 10^{-12}$ at 90 % CL [22] and $\mu^- Au \to e^- Au$ measured by the SINDRUM II experiment with a BF, $\mathcal{B}(\mu^- Au \to e^- Au) < 7 \times 10^{-13}$at 90 % CL [23] . If one would observe one of these decays it would be a clear hint for physics beyond the SM.

2 Principles of Particle Detectors

In particle physics experiments, tracking detectors are a key part because they are able to determine the momentum and trajectory of charged particles. The ideal tracking detector should neither stop the particle nor strongly deflect is since one wants to reconstruct the trajectory from the positions where the particle hits the detector. For this the material budget needs to be as small as possible.

In the Mu3e experiment the μ will decay at rest leading to electrons and positrons with low momenta. For this it is specifically important that the detector material is minimal since multiple scattering increases with decreasing momenta.

With modern semiconductor technologies, it is possible to build very thin detectors. In section 2.1, section 2.2 and section 2.3 the basics of silicon detectors, tracking systems and multiple scattering are explained based on [24]. After that, the special type of pixel detector used in Mu3e is discussed in section 2.4. At the end, the principle of scintillating fibre detectors is introduced in section 2.5 based on [25].

2.1. Silicon detectors

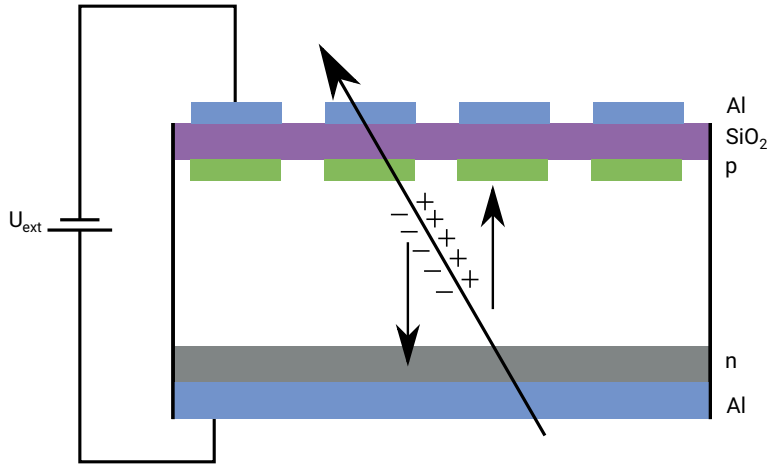


Figure 2.1.: Sketch of a particle travelling through a silicon pixel detector. By appling a external voltage the whole area between the p- and n-doped regions becomes active.

In fig. 2.1, a sketch of a silicon detector is given. The detector contains different p- and n-doped regions. It also contains n^+ doped implants. Because of the p- and n-doped areas, a pn-junction is formed. Within this region, the charge carriers can diffuse into the opposite region where they are recombine with their conterparts leading to a charge free region outside of the pn-junction called depletion zone. By applying an external reverse-bias voltage, the detector can be depleted and the full volume becomes active. If a charged particle travels through the detector, it creates free electron hole pairs. The bias voltage leads to a faster drift of the particles in the direction of the electric field creating an induction signal which can be processed using amplifiers and comparators. By using small n^+ doped implants it is possible to measure the spatial information precisely. If these sensors are extended in one dimension, one calls them strip detectors. If they have a fine segmentation in two dimensions, they are called pixel detectors.

2.2. Principle of tracking detectors

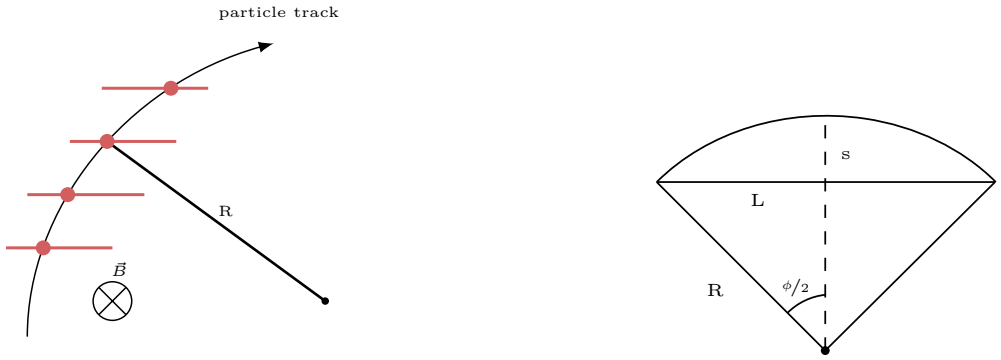


Figure 2.2.: Sketch of a particle traveling through a tracking detector and important variables of the particle track in the x-y plane.

In fig. 2.2 a schematic illustration of a particle trajectory traveling through a tracking detector in the x-y plane is shown. For the momentum determination of a particle passing a barrel-shaped tracking detector one needs to apply a magnetic field perpendicular to the particle track. Then the Lorentz force is equal to the centripetal force and for the momentum one finds the equation:

$$\frac{m \cdot v^2}{R} = e \cdot v \cdot B \rightarrow p = e \cdot B \cdot R. \tag{2.1}$$

Here m is the mass, v the velocity, e the electric charge and p the momentum of the particle. B is the magnetic field applied perpendicular to the particle track. The radius R of the track is obtained by the use of a circle fit through the measurement points of each tracking layer. The tracking layer can be made of multiple silicon detectors. To determine the momentum resolution, the sagitta s can be derived for small ϕ with:

$$s = R - R \cdot cos(\frac{\phi}{2}) \approx R \cdot \frac{\phi^2}{8} = R \cdot \frac{L^2}{8 \cdot R^2}. \tag{2.2}$$

Furthermore the momentum resolution is given by

$$\frac{\Delta p}{p} = \frac{\Delta R}{R} = \frac{L^2}{8 \cdot R \cdot s} \cdot \frac{\Delta s}{s} \sim p \cdot \frac{\Delta s}{B \cdot B \cdot L^2}. \tag{2.3}$$

In order to obtain a good momentum resolution a large path length L, a large magnetic field B and a good measurement of sagitta is needed. In this configuration the particle momentum is large and multiple scattering can be neglected. Since in the Mu3e experiment the decay products will not have such a large momenta, multiple scattering needs to be considered.

2.3. Multiple Scattering

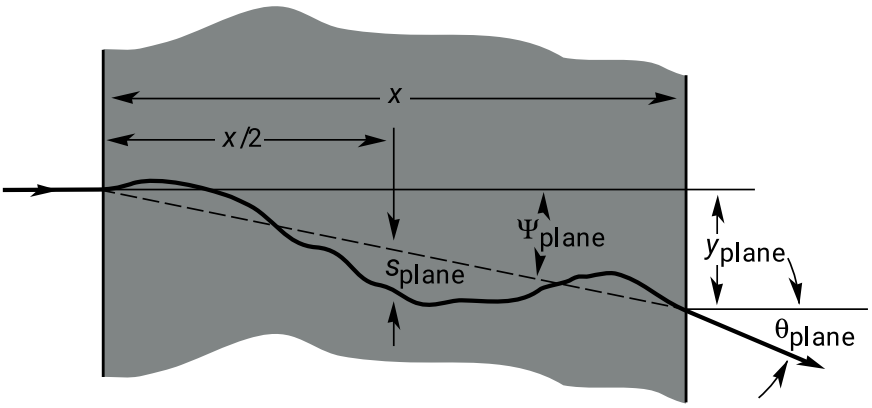


Figure 2.3.: Particle passing through matter, taken from [9].

In fig. 2.3 the path of a particle traveling through matter is shown. One can see that the track gets deflected because the particle will scatter on the Coulomb fields of the nuclei resulting in an offset from the undisturbed track and a change in the angle. If the detector is very thin, the offset can be neglected and only the change in the angle remains. The root mean square (RMS) of the angle is well approximated by the Highland equation [9]:

$$\Theta_{rms} = \frac{13.6\,\text{MeV}}{\beta cp} z \sqrt{\frac{x}{X_0}} \left[1 + 0.038 \log\left(\frac{x}{X_0}\right)\right]. \tag{2.4}$$

Since the electron has a fixed charge, the only parameter in the equation which can be influenced is the radiation length x. For getting a good momentum resolution the multiple scattering angle needs to be small. This leads to tight constraints on the material budget.

2.4. High Voltage Monolithic Active Pixel Sensors

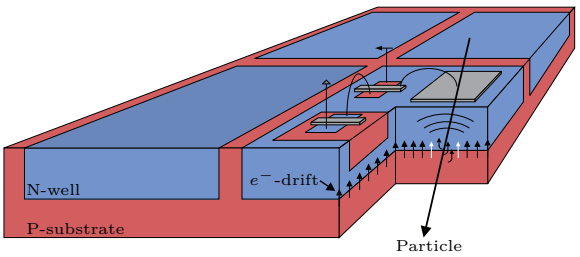


Figure 2.4.: Sketch of a High-Voltage Monolithic Active Pixel Sensor (HV-MAPS) sensor [26].

In section 2.1 the principle of silicon detectors was explained. For reading out the signal an amplification circuit can be integrated onto the pixels. This type of detectors are called Active Pixel Sensors (APS). Beside this the total readout of the signal can also be placed on the chip, leading to Monolithic Active Pixel Sensors (MAPS). These chips collect the signal via diffusion. The disadvantage of diffusion is the charge collection time of the size of μs compared to the typical drift time in a bias field of the order of ns.

In fig. 2.4 the sketch of a HV-MAPS is shown [26]. These sensors combine the monolithic part of the MAPS and the fast charge collection via drift. By the use of deep n-well located in the p-substrate, a diode is formed. The electronics of the pixel can be implemented inside the n-well. By the use of a HV-CMOS process, a high reverse bias voltage can be applied to the substrate which creates a large depletion zone. The p-substrate is only partially depleted leading to the possibility that the sensor can be thinned down from the back. This technology allows fast charge collection and guarantees a low material budget, which in turn ensures a good momentum resolution.

2.5. Scintillating Fibres

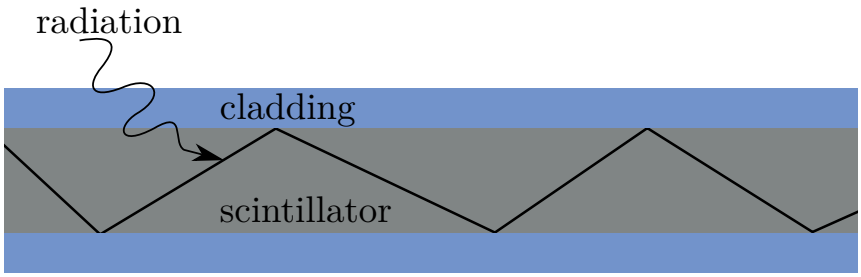


Figure 2.5.: Sketch of a particle travelling inside fibre scintillator.

Scintillators radiate light if they get excited by ionizing radiation. They come in various types and forms. Organic scintillators for example are made of an activator and a cladding substance. The energy of the particle is turned into light by emitting a photon via fluorescence inside the activator substance.

By forming these scintillators into a fine fiber, one can transport the light over a substantial distance by total internal reflection. This configuration is shown in fig. 2.5. Here the scintillator is in the core of the fiber which is surrounded by a carrier material. The light which is emitted inside the core arrives at the core-cladding border can have an angle that is greater than the critical angle for total reflection. With this configuration, the light can travel down the fiber. At the end of the fibres photomultipliers can be installed magnifying the signal by first converting the photons into electrical current which then gets amplified.

3
Field Programmable Gate Arrays

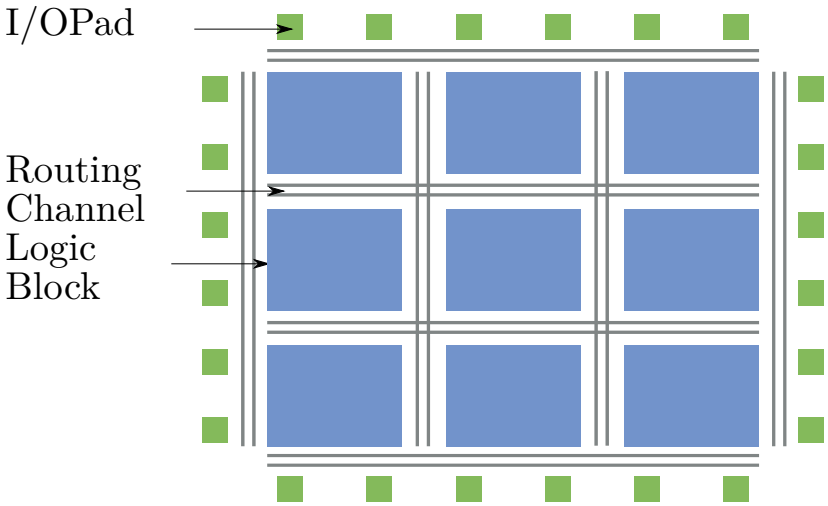


Figure 3.1.: Structure of a FPGA.

FPGAs are integrated circuits with reprogrammable hardware. The user is able to change the device after manufacturing by using reconfigurable interconnect lines and Logic Elementss (LEs). These LEs are made out of Look-Up Tables (LUTs) and registers. Between these LEs interconntect lines are placed. For programming the FPGAs, in the context of this work, the hardware description language VHDL is used. In fig. 3.1 a simple sketch of an FPGA is shown. It consists of a core of LEs connected via a global routing network and surrounded of a ring of input/output (I/O) pads. It is also possible to make use of memory blocks and hard intellectual property cores (IPs) which are fixed blocks inside the device provided by the FPGA manufacturer. This IPs are optimized for specific tasks for example high-speed serialization or deserialization.

With the use of the hardware description language, the data flow and the logic of the FPGA can be programmed. First this design is synthesised and a netlist is generated which can be used by the fitting tool to create the logic for the LEs and interconnects inside the FPGA. Beside this, also the desired operating frequency is taken into account. At the end, a configuration file can be uploaded to the device. All of this is done in the Intel Quartus Prime software [27] in the case of Intel (formerly Altera) FPGAs.

Logic Elements

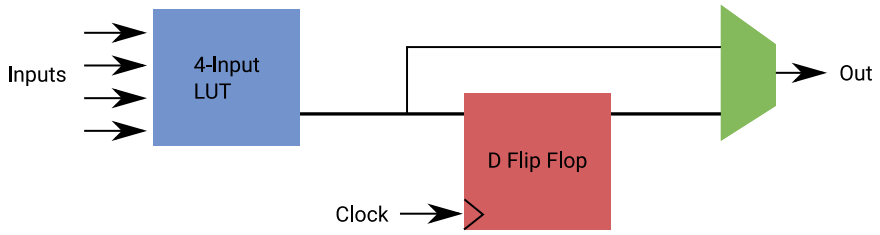
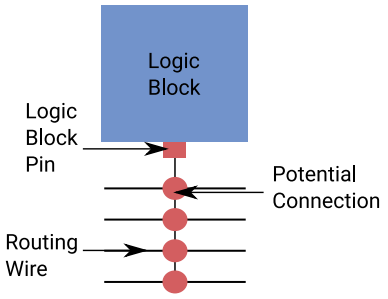


Figure 3.2.: Illustration of a simple LE containing a four input LUT and a D-flip-flop.

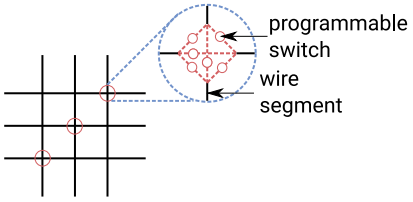
In fig. 3.2 a LE with four inputs and one output is shown. The output can be either directly connected to a LUT or driven by a flip-flop. A LUT is a simple array containing values which can be accessed by simple index operation. The flip-flop can be used for saving information. The D-flip-flop provides the data to the output, if there is a clock edge on the clock input. If there is no clock, a change of the input does not effect the output leading to a storage of the information inside the flip-flop until a clock signal is detected. By routing multiple LUT together different boolean functions can be expressed.

The FPGAs used in the Mu3e experiment are Static Random Access Memory (SRAM) based FPGAs. This FPGAs store the configuration of the LUT inside SRAM cells. Furthermore, Intel FPGAs are used since they have a good ratio of fast and slow I/Os, sufficient amount of LEs and they provide evaluations boards which can be partly used in the final detector.

Routing



(a) Connection of a LE to a routing wire.



(b) Sketch of a programmable switch sitting on the intersections of different routing wires.

For connecting different LEs together, the input and output pins can be connected to routing wires. In fig. 3.3a such a connetion is shown. Each of these routing wires terminates into a switch box sitting on the intersections. The switch boxes are shown in fig. 3.3b and can be programmed for connecting multiple LEs together.

4 Data Transmission

For finding a rare event, a lof of decays need to be studied. Therefore, a tremendous amount of data needs to be processed. For reading out such a data rate the use of high speed data links as well as FPGAs which are able to analyze the data during operation, is required.

In this chapter the basics of data transmission are described in section 4.1, followed by electrical data transmission in section 4.2 and the idea of serial data links in section 4.3. Also, the general implementations of high speed data transmission and the transmission inside FPGAs are shown in section 4.4. The optical transmission of data is described in section 4.5. At the end, in section 4.6, a test procedure of such data links is pointed out. The main parts of this section are taken from [28] and [29].

4.1. Basics of data transmission

Electrodynamics

The underlining theory of electrical or optical data transmission is the theory of classical electrodynamics [30]. This theory was derived by Maxwell and can be expressed in four linear equations:

$$\vec{\nabla} \cdot \vec{D} = \rho, \tag{4.1}$$

$$\vec{\nabla} \times \vec{H} - \frac{\partial \vec{D}}{\partial t} = \vec{j}, \tag{4.2}$$

$$\vec{\nabla} \times \vec{E} + \frac{\partial \vec{B}}{\partial t} = 0, \tag{4.3}$$

$$\vec{\nabla} \cdot \vec{B} = 0. \tag{4.4}$$

Here $\vec{E}$ describes the electric field, $\vec{B}$ the magnetic field, $\vec{D}$ the electric displacement, $\vec{H}$ the magnetizing field, ρ the charge density and $\vec{j}$ the current density. Between the four fields one can find the relations:

$$\vec{D} = \epsilon_r \epsilon_0 \vec{E}, \tag{4.5}$$

$$\vec{B} = \mu_r \mu_0 \vec{H}. \tag{4.6}$$

Here ϵ_0 is the vacuum permittivity, μ_0 is the vacuum permeability, ϵ_r and μ_r are the corresponding values depending on the medium. Also, the conductivity σ is given by Ohm's law,

$$\vec{j} = \sigma \vec{E}. \tag{4.7}$$

Plane waves

The following calculation can be found in [31]. In a non-conducting material σ and $\vec{j}$ are equal to zero. The equation for an electromagnatic wave traveling in such a medium is then given by:

$$\Delta \vec{E} = \epsilon\mu \frac{\partial^2 \vec{E}}{\partial t^2}. \tag{4.8}$$

In a conducting medium with $\sigma \neq 0$ and $\vec{j} \neq 0$ the wave equation is given by:

$$\Delta \vec{E} = \epsilon\mu \frac{\partial^2 \vec{E}}{\partial t^2} + \sigma\mu \frac{\partial \vec{E}}{\partial t}. \tag{4.9}$$

With the ansatz:

$$\vec{E}(\vec{x}, t) = E_0 e^{i\omega t - i\vec{k}\vec{x}}, \tag{4.10}$$

one finds

$$\vec{k}^2 = -i\omega\mu\sigma + \omega^2\mu\epsilon. \tag{4.11}$$

By splitting the wave vector $\vec{k}$ into a real and imaginary part with:

$$\vec{k} = \vec{\alpha} - i\vec{\beta} \tag{4.12}$$

the solution of the plane wave will be:

$$\vec{E}(\vec{x}, t) = E_0 e^{i\omega t - i\vec{\alpha}\vec{x}} e^{\vec{\beta}\vec{x}}. \tag{4.13}$$

So the length of the wave is given by α and the attenuation factor is given by β. The wave vector and the frequency are linked via the dispersion relation:

$$\vec{k}^2 = \omega^2\mu\epsilon. \tag{4.14}$$

Using this equation one finds for α and β:

$$\alpha = \omega\sqrt{\mu\epsilon}\left[\frac{1}{2} + \frac{1}{2}\sqrt{1 + \frac{\sigma^2}{\omega^2\epsilon^2}}\right]^{1/2}, \tag{4.15}$$

$$\beta = \frac{\omega\mu\sigma}{2\alpha}. \tag{4.16}$$

A similar procedure can be performed for real dielectrics, where ϵ is a complex number. This leads to a dispersion and an attenuation:

$$\epsilon = \epsilon' - i\epsilon''. \tag{4.17}$$

The wave equation, α and β are then:

$$\Delta\vec{E} = (\epsilon' - i\epsilon'')\mu\frac{\partial^2\vec{E}}{\partial t^2} + \sigma\mu\frac{\partial\vec{E}}{\partial t}, \tag{4.18}$$

$$\alpha = \omega\sqrt{\mu\epsilon'}\left[\frac{1}{2} + \frac{1}{2}\sqrt{1 + \frac{(\sigma + \omega\epsilon'')^2}{\omega^2\epsilon'^2}}\right]^{1/2}, \tag{4.19}$$

$$\beta = \frac{\omega\mu(\sigma + \omega\epsilon'')}{2\alpha}. \tag{4.20}$$

From these equations one can see, that the attenuation factor β depends on the conductivity σ and on the imaginary part of the permittivity ϵ''.

For the implementation of high speed electrical data links it is important to surround and separate the conductors with dielectrics. So the power loss of the wave inside the dielectric is given by:

$$P_{loss} \propto \omega \cdot \tan\delta. \tag{4.21}$$

Here $\tan\delta$ is the loss tangent and given by:

$$\tan\delta = \frac{\sigma + \omega \cdot \epsilon''}{\omega \cdot \epsilon'}. \tag{4.22}$$

The critical frequency, which defines the edges of conducting and insulating modes in a material is given by

$$\omega_c = \frac{\sigma}{\epsilon}. \tag{4.23}$$

For a lot of good insulators, the conductivity is to first order proportional to the frequency. Since the power loss is proportional to the frequency, the dielectric losses affect the high speed data transmission in electrical links.

4.2. Electrical data transmission

For transmitting high speed data, electrical links can be used. By modeling the electrical transmission line as an infinite cascade set of two-port systems, one can derive the telegrapher's equations. These systems are built out of segments with a series impedance Z, which is made out of a resistance R and an inductance L, and a parallel admittance y. The admittance is connected to ground and built of a capacitance C and a shunt conductance G. This configuration is shown in fig. 4.1.

With the telegrapher's equation one finds that the wave impedance Z_0 is given by

$$Z_0 = \sqrt{\frac{R + i\omega L}{G + i\omega C}}. \tag{4.24}$$

For a line without loss, $R = G = 0$, so Z_0 does not depend on the frequency ω.

For a real transmission line we have a non-zero resistance R and conductance G. One of the reasons for this is the above explained dielectric loss. Another one is the skin effect. The

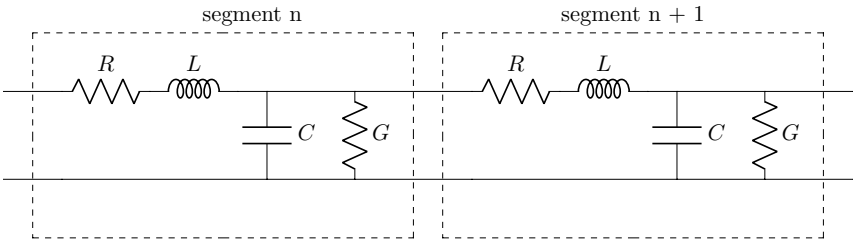


Figure 4.1.: Infinite Cascade model for deriving the telegrapher's equation, taken from [32].

coefficient β gives the distance a wave travels inside a conductor. The inverse of β is the distance over which the amplitude is reduced by the factor $1/e$ which is called skin depth δ. The skin effect then limits the area a high speed wave traveling inside a conductor to its outer surface. This can cause inter-symbol interference since these effects lead to attenuation and dispersion of high speed data signals. For a successful high speed data transmission these effects need to be corrected. In the following section serial data links are explained.

4.3. Serial data links

For communication between two devices which are not on the same circuit board, the reduction of connectors placed on the boards are important. Therefore serial data links can be used, since they only need one line instead of multiple ones used by parallel buses. On the other hand they need to operate at a higher frequency to transfer the same amount of data than parallel buses do.

Normally these high speed data links are built out of different layers. Each layer fulfills a special task. The minimum setup of links is the physical and data link layer. Therefore, the physical layer (PHY) maps the information to physical observables and transfers it to the data link. It also ensures electrical compatibility between the devices. The data layer is responsible for the successful communication and improves signal integrity. Therefore the data is encoded, aligned and clock recovery is performed if necessary. On top of these basic layers, multiple other layers for error correction or protocol dependent features can be used [33]. In the following the physical and data link layers are further discussed.

4.3.1. Physical layer

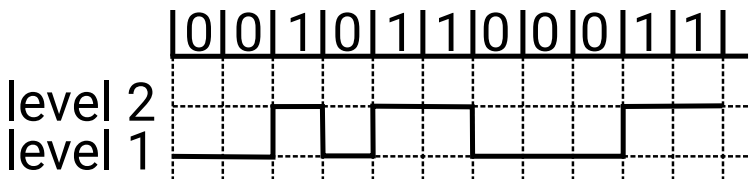


Figure 4.2.: Illustration of the non-return-to-zero (NRZ) code for data transmission.

Since the data need to be mapped to a physical observable most common electrical links use two voltage levels for representing a logical "1" and a "0". They do this with non-return-to-

zero (NRZ) binary coding. Figure 4.2 illustrates the two voltage levels which are applied to the cable for encoding a "1" and a "0". In the most common high speed links this encoding is implemented with the usage of a differential signal with low common mode. One of the standards for realizing this technology is low-voltage differential signaling (LVDS). In Intel FPGAs a proprietary standard called pseudo current mode logic (PCML) is used for very high speed links in the range of Gbit/s.

A fast clock for serializing the data on the transmitter side is used. This clock needs to be recovered on the receiver side. Otherwise, the bits can not be correctly recaptured. For this a clock and data recovery circuitry built out of phase interpolation and phase-locked loops (PLLs) is used. Beside the clock recovery method, it is possible to transfer the clock from the transmitter and recover the data on this one. In the Mu3e experiment the second option was chosen.

Phase locked loops

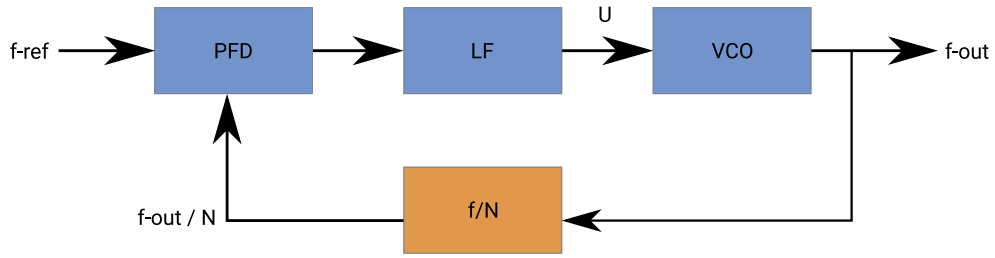


Figure 4.3.: Illustration of the principle of a PLL.

A PLL generates an output signal whose phase is related to the phase of an input signal. In fig. 4.3 the principle of a PLL is shown. The reference frequency f-ref goes into the phase frequency detector (PFD) generating the control voltage U via $U = f(f_{ref} - (f_{out}/N))$. The loop filter (LF) filters the signal and the voltage controlled oscillator (VCO) generates the output frequency f-out with U $f_{out} = f(U)$. The output frequency is divided by a integer value N. In the steady state, this loop generates an output frequency of $f_{out} = f_{ref} \cdot N$. Since N needs to be an integer, f-out can only be changed in steps of f-ref. This leads to very small f-ref if small steps are needed.

4.3.2. Data link layer

This section follows [34] and [32].

Previous RD	Next RD	Disparity choices	Disparity chosen
−1	−1	0	0
−1	+1	±2	+2
+1	+1	0	0
+1	−1	±2	−2

Table 4.1.: Choice of disparity, taken from [35]

The data link layer is used for encoding the data and handling of the data protocols. There are a few encoding schemes used. For example 8b/10b, 64b/66b or 128b/130b. Since 8b/10b converts 8 bits of user bits into 10 bits for line transmission, one can exclude some of the possible 1024 bits leading to a run-length limit of five consecutive equal bits. Furthermore, the sum of zeros and ones can be not more than two. By using this and also decoding some of the 256 possible 8 bit words in two different ways, the bit sequence can be made out of an equal number of ones and zeros guaranteeing DC balancing. Another advantage is the large attenuation length provided by the 8b/10b encoding [36]. The disparity, defined as the difference between ones and zeros in a word, is then 0 or ± 2. In a data stream the disparity is always measured and is called running disparity (RD) which can only have values of ± 1. So to guarantee DC-balance the disparity needs to maintain the running disparity with the limits shown in table 4.1.

	8 bit	10 bit	10 bit
Name	HEX	RD = −1	RD = +1
K.28.0	1C	001111 0100	110000 1011
K.28.1 †	3C	001111 1001	110000 0110
K.28.2	5C	001111 0101	110000 1010
K.28.3	7C	001111 0011	110000 1100
K.28.4	9C	001111 0010	110000 1101
K.28.5 †	BC	001111 1010	110000 0101
K.28.6	DC	001111 0110	110000 1001
K.28.7 †	FC	001111 1000	110000 0111
K.23.7	F7	111010 1000	000101 0111
K.27.7	FB	110110 1000	001001 0111
K.29.7	FD	101110 1000	010001 0111
K.30.7	FE	011110 1000	100001 0111

Table 4.2.: All symobles with † are called comma symbols and they are used for aliging the 8b/10b data stream. The bit order is from least significant bit to most significant bit taken from [35].

Besides the DC-balance, 8b/10b encoding provides so called K-symbols or control symbols. They can be used for recovering word boundaries or as idle characters. This is important since they guarantee bit transitions even when there is no data transmitted. The list of these characters is shown in table 4.2.

4.4. High speed links in FPGAs

In this part a high speed link inside a FPGA is explained by the example of an Intel Arria 10 FPGA, which is mainly used in this thesis. The Arria 10 GX Device Transceiver and the Intel Arria 10 Transceiver PHY are discussed. This part is adapted from the Intel Arria 10 Transceiver PHY User Guide [37].

4.4.1. GX Transceiver

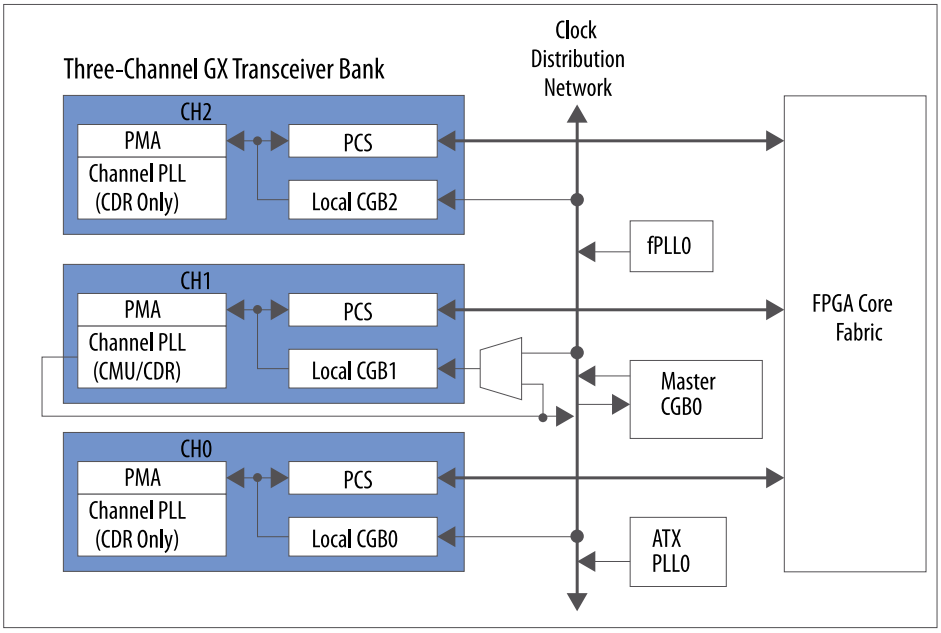


Figure 4.4.: High level overview of the transceiver bank architecture of a bank made out of 3 transceiver channels, adopted from [37].

The Intel Arria 10 FPGA has up to 96 GX transceiver channels which have integrated high speed analog signal conditioning and clock data recovery (CDR). Up to 17.4 Gbps can be transmitted via each of these channels. They come in banks of up to eight transceivers located in the left and right side periphery of the device. In fig. 4.4 one of there transceiver banks with 3 channels is shown. Each of the channels has an integrated PLL and a clock generation block (CGB). The CGB comes in two types, the Master CGB and the Local CGB. The Master CGB is connected to three channels and divides and distributes the clocks to the channels while the Local CGB divides and distributes the clocks to the Physical Coding Sublayer (PCS) and Physical Medium Attachment (PMA) blocks.

For encoding and decoding the data and also for alignment marker insertion and removal, the PCS is used. The alignment of the data stream is needed, since the transceivers does not know the bit position inside the stream. It is located on top of the PHY and it provides the interface between the PMA and the FPGA Core Fabric. Each channel is also connected

to the Clock Distribution Network and the FPGA Core Fabric. There are also fractional PLLs (fPLLs) implemented which are used for generating lower clock frequencies for data rates less the 12.5 Gbps and Advanced Transmit PLLs (ATXs) which can be used for the full range of supported data rates.

The advantage of the fPLLs is the possibility to generate frequencies which are $^N/_M$ times of the reference frequency. For this the fPLL first multiplies the reference frequency by an integer before it gets feed into the PFD. In the Mu3e experiment all transceiver PLLs are using the global 125 MHz clock, which is provided by the clock and reset system (cf. section 6.2.5), for their reference clock. Since the transceivers implemented on the front-end board operate with a data rate of 6250 Mbps and the data packages are in parts of 32 bits being 8b/10b encoded, the clock for the transceivers needs to be 156.25 MHz, leading to the use of a fPLLs.

4.4.2. Intel Arria 10 Transceiver PHY Layer

The Intel Arria 10 Transceiver PHY Layer supports PMA and PCS functions. The PMA is the electrical interface to the physical medium and has standard blocks such as serializer or deserializer (SERDES), clock and data recovery PLL, analog front end transmit drivers and analog front end receive buffers.

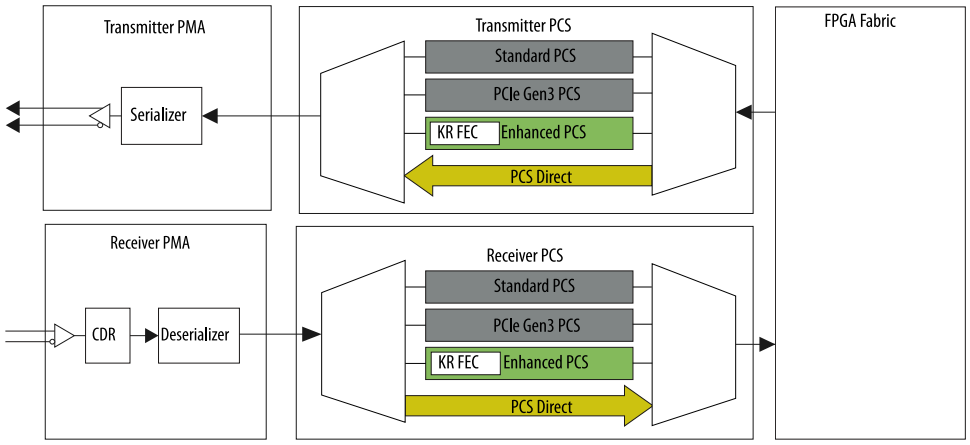


Figure 4.5.: High level overview of a transceiver channel with a transmitter and a receiver part, taken from [37].

PCS type	data rate
Standard PCS	1.0 Gbps to 10.813 44 Gbps
Enhanced PCS	1.0 Gbps to 17.4 Gbps
PCIe Gen3 PCS	8.0 Gbps

Table 4.3.: Supported PCS types.

In fig. 4.5 a transceiver channel in duplex mode is shown. In this configuration, three types of PCS are available. In table 4.3, the supported PCS types and their data rates are provided. Beside the normal PCS there is also the option for using a Peripheral Component Interconnect Express (PCIe) IP [38]. PCIe is a high speed serial bus interface which is used for connecting peripheral devices with the Central Processing Unit (CPU) inside of a personal computer (PC).

4.4.3. Transceiver design IP blocks

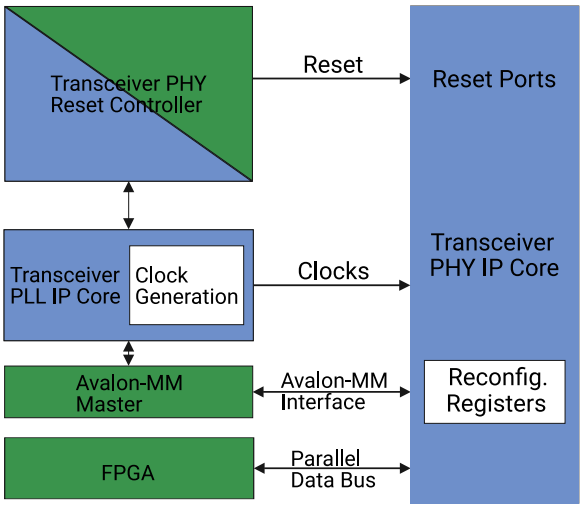


Figure 4.6.: Fundamental building blocks of an Intel Arria 10 Transceiver. The blue blocks are generated by Quartus the green blocks can be generated by the user, taken from [37].

In fig. 4.6 the fundamental building blocks of an Intel Arria 10 Transceiver are shown. Each Transceiver PHY IP Core comes with reset ports and reconfiguration registers and controls for the PCS and PMA configurations. The reset ports are connected to a Transceiver PHY Reset Controller IP Core which can be used for resetting each channel of the Transceiver PHY IP Core. This IP Core is connected to a Transceiver PLL IP Core and triggers the reset if the PLL is not locked to an input clock. The PLL provides a clock to the Transceiver PHY IP Core. There is also an option for an Avalon Memory Mapped Interface Master (Avalon-MM) which can be used for reading and writing the reconfiguration registers of the Transceiver PHY IP Core. In the Mu3e experiment this option is used for monitoring the current status of the transceivers and adjusting the settings during operation.

4.4.4. Transceiver Channel datapath and clocking

In fig. 4.7 the datapath and clocking of a transceiver operating with a data rate of 1250 Mbps is shown. The same type of transceiver is used in the Mu3e experiment to distribute the reset protocol (cf. section 6.2.5). All other transceivers used in this work only differ in the actual

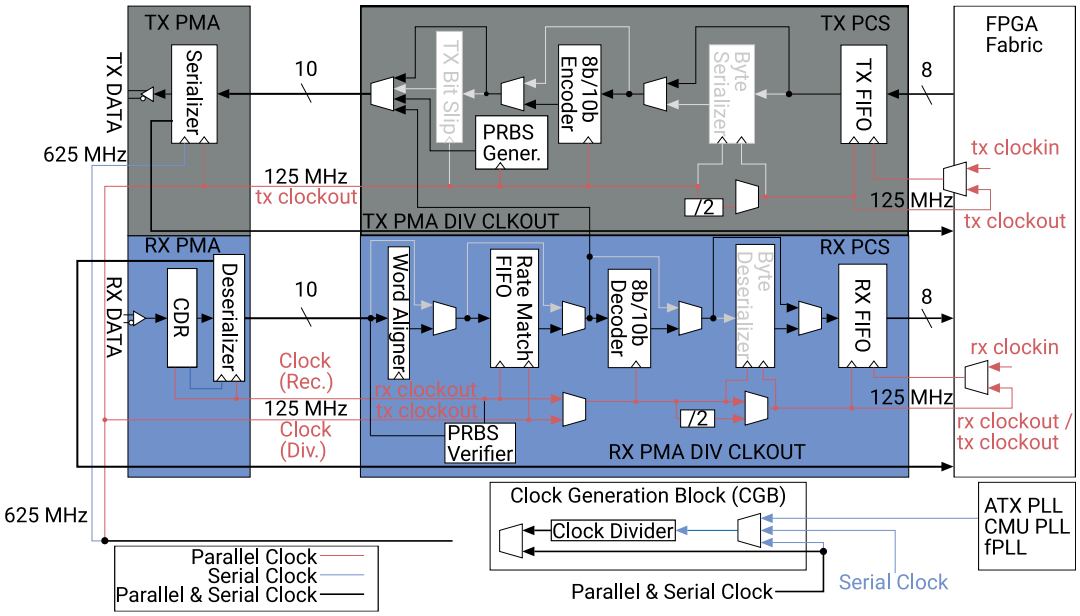


Figure 4.7.: Transceiver channel datapath and clocking at 1250 Mbps, taken from [37].

data rate and the parallel data width. This transceiver operates with an input data width of 8 Bit which is provided as a parallel stream from the FPGA core with a transceiver core clock (tx coreclkin) of 125 MHz. The data is sent to the Transceiver (TX) first in first out memory (FIFO). The FIFO is organizing the data buffer such that the oldest entry is served first on the output. The data on the FIFO output is sampled with a different clock (tx clkout) which is derived from the transceiver PLL. After the FIFO, the data is sent to the 8b/10b encoder. It is possible to enable a byte serializer in between. This serializer is able to double or quadruple the data width of the PMA serializer. With a higher width it is possible to run the PCS at a lower parallel clock frequency to accommodate a wider range of FPGA interface widths. The 10 Bit data is then sent to the PMA serializer which is implemented as a shift register. This register gets the parallel data with the low speed parallel clock and is shifted bit by bit with the high speed serial clock. With the use of the TX Bit Slip component it is possible to change the bit position of the 10b parallel data before it gets serialized.

The receiver part of the transceiver channel first recovers the high speed serial clock and the low speed parallel clock from the serial data via the CDR. After this, the serial data gets parallelized in the PMA deserializer. The parallel data is sent to the word aligner which aligns the bits by checking for the K-symbols inside the 10b data and starts the rate match process after it is synchronized. The rate match process can insert or delete these sets such that the rate match FIFO never gets into over- or underflowing. For compensating frequency part-per-million (ppm) differences between the transmitter and receiver clock, the rate match FIFO is used. This leads to a difference of 125 MHz $\pm$ 100 ppm. The rest of the receiver part is used for decoding the data back to 8 Bit and inserting it into the FPGA core.

4.5. Optical data transmission

Since electromagnetic effects can disturb electrical communication, fiber optical ones can be used. Here, pulsed light is sent through a fiber optical cable. For the transmitter part, light-emitting diodes (LEDs) are used. In high speed data applications lasers are used since they have a higher optical output power and a faster switching time. For the receiver part photodiodes convert the light back into electrical signals [39].

Since for the Mu3e experiment a lot of data needs to be read out, optical data transmission is preferred, because the bandwidth of optical communication exceeds that of copper based systems [32].

4.6. BERTs

For testing data links, pseudo-random data patterns can be generated and sent through the link. The receiver can predict the incoming data since pseudo-random data are a deterministic series. By comparing the incoming and predicted data, bit errors (BERs) can be found. Given a data rate r during a time interval t the total number of bits are $N_{bits} = r \cdot t$. With the total number BERs N_{err} the probability of a BER is given by:

$$P_{BER} = \frac{N_{err}}{N_{bits}}. \tag{4.25}$$

If no bit error is found in the data stream, only an upper limit can be calculated. This can be done by the use of the Bayesian approach with flat prior for a Poissonian distribution [40]. So the upper limit for a BER with a CL of 95% is given by:

$$P_{BER} \leq \frac{\ln(1 - \mathrm{CL})}{N_{bits}} \approx \frac{3}{N_{bits}}. \tag{4.26}$$

Part II.

The Data Acquisition of the Mu3e Experiment

The hunt for new in particle physics is done in multiple ways. One of it is to probe the SM for small deviations. Searching for rare decays is a possible way for achieving this. This investigations are challenging not only the SM but also state of the art technologies for processing the large amount of physical events which have to be studied.

The Mu3e experiment is one of these precision experiments exploring the decay $\mu^+ \rightarrow e^+e^+e^-$. In the following part, the detector concepts of the experiment are discussed as well as the DAQ which is the key part of handling the large amount of data flow.

5 The Mu3e Experiment

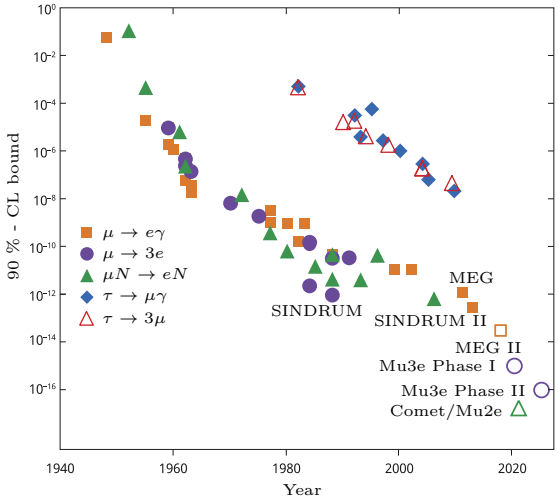


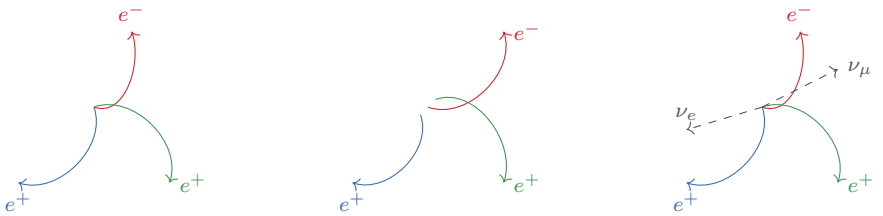
Figure 5.1.: History of searches for charged lepton flavour violation in μ and τ decays. Adapted form [41].

Since the observation of neutrino oscillation by the experiments Super-Kamioka Neutrino Detection Experiment (SK) [4], Sudbury Neutrino Observatory (SNO) [5], Kamioka Liquid Scintillator Antineutrino Detector (KamLand) [6] and others it is clear that the lepton flavour is not conserved in nature. As mentioned in chapter 1, the observation of lepton flavour violation in charged leptons would be a clear hint for new physics. In fig. 5.1 the history of searches for such decays is listed. The purple circles show experiments which search for the decay $\mu \to 3e$. The Mu3e experiment which will be built at the Paul Scherrer Institute (PSI) in Switzerland could set a further limit on the BF of the decay $\mu^+ \to e^+e^+e^-$. The data taking will be in two steps. Phase I will start at the end of 2021 and Phase II at the end of 2025 increasing the limit even further. The SINDRUM experiment limit is $\mathcal{B}(\mu^+ \to e^-e^+e^+) < 1.0 \times 10^{-12}$ at 90 % CL [22]. To improve the sensitivity for this BF by three orders of magnitude, a μ rate of 10^8 μ/s over one year of data taking needs to be achieved. The expected data rate of the detector is so high that it would be prohibitively expensive to permanently store all data.

For the second phase the sensitivity will be pushed down to 10^{-16} with a μ rate of 2×10^9 μ/s. In the work of this thesis only Phase I is considered.

For understanding how the data are taken in the Mu3e experiment first the signal and background processes are explained in section 5.1 followed by the detector concept in section 5.2 and subsequently the tracking detector in section 5.3, since the data flow of this part of the detector was mostly considered in this thesis.

5.1. Signal and background processes



(a) (b) (c)

Figure 5.2.: Overview of the signal process and the main background processes. In fig. 5.2a the signal process is drawn, fig. 5.2b shows the random combinatorial background and in fig. 5.2c the internal photon conversion is displayed.

The signal and the most common background processes are displayed in fig. 5.2. In fig. 5.2a the signal process $\mu^+ \to e^+e^+e^-$ is plotted. In fig. 5.2b the random combinatorial background is shown. Other processes happen at the same time and the particle tracks in the detector do not match up at the same vertex. The third picture fig. 5.2c shows the process of internal photon conversion $\mu^+ \to e^+e^+e^-\nu_e\nu_\mu$. In this process additional neutrinos are produced. Since they can not be detected in the detector the measured energies will not add up to the equivalent of the μ mass. For distinguishing this background process efficiently from the signal, the detector needs to have a momentum and time resolution better than 1.0 MeV/c and 1 ns.

5.2. Detector concept

A sketch of the detector is shown in fig. 5.3. The detector is built in a cylindrical shape and it has a central station in the middle and two recurl stations one up- and one downstream of the incoming beam. The 1.3 MW cyclotron at PSI is producing a 2.2 mA proton beam with an energy of 590 MeV. By shooting these protons onto a graphite target, secondary pions are produced and stopped in the target, they then decay into μ's with a momentum of 29.8 MeV/c. Other secondary particles (mostly positrons) are filtered out via the use of a Wien filter.

The μ beam is then hitting the detector target, which is a hollow double cone made of Mylar foil. This design was taken since it fulfills the requirements which are given by the beam size, the innermost layer of the pixel detector and having a high μ stopping rate. Further information on the detector design can be found in [43]. The stopped μ's decay at rest and the decay products get bent in the field of the 1 T solenoid magnet. By using a silicon pixel tracking detector, the trajectories of the electrons will be measured. At the recurl stations the trajectories hit the detector again, increasing the path length of the tracks. Since the vertex

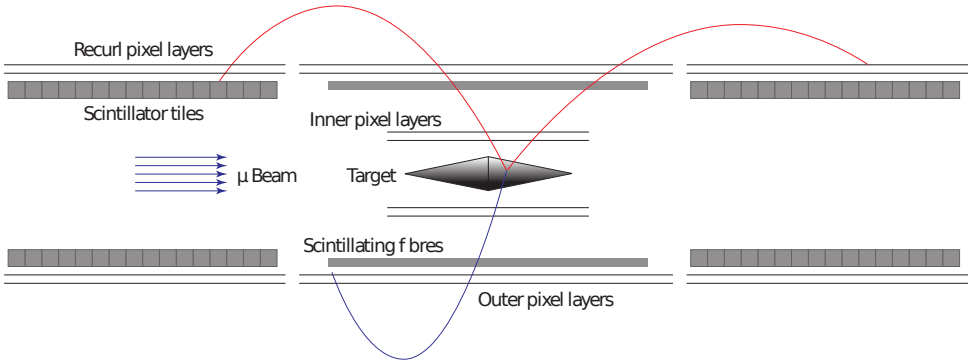


Figure 5.3.: Sketch of the Mu3e detector concept, taken from [42].

and momentum resolution is influenced by multiple scattering inside the detector material, the pixel layers need to be very thin. For this high-voltage monolithic active pixel sensors called MuPix are used. This chip is developed in different versions, the current one is called MuPix8. A detailed explanation of the MuPix is given in [44]. For time measurements in the central detector station a scintillating fibre detector is used and for the recurl stations scintillating tiles are taken [45]. Both detectors are read out via silicon photomultipliers (SiPMs) and an application specific integrated circuit (ASIC) which is called MuTRiG [46]. During the work of this thesis only the first version of this ASIC, STiC [47], was available. Nevertheless, in the following the actual design will be explained by the use of the MuTRiG chip.

5.3. Pixel detector

The MuPix is a HV-MAPS which can be thinned down to 50 μm and has a time resolution in the order of a few ns. In fig. 5.4 the schematic pixel electronics are shown. Here each pixel is equipped with a charge sensitive amplifier (CSA), a second amplification is performed in the periphery. All these amplifiers can be adjusted. Beside this, the threshold per pixel can be set by the tune digital-to-analog converter (DAC).

The data from the pixel is sent via a source follower to the periphery. There the analog signal will be digitized by a comparator. The Time-over-Threshold (ToT) signal is obtained from the comparator after the second amplifier. The threshold for this comparator can also be adjusted. So it is possible to suppress pixel to pixel variations. The ToT can be recorded for any pixel through a dedicated output line. If the comparator generated a signal, a time stamp is stored for this hit. Until the time stamp is not being read out, no other hit can be saved. The readout logic for each column is in a way that the hit with the lowest row index is read out first. This leads to a hit stream which is not time sorted. For the hit position the row and column address is generated and also send out. After this the data gets collected by a

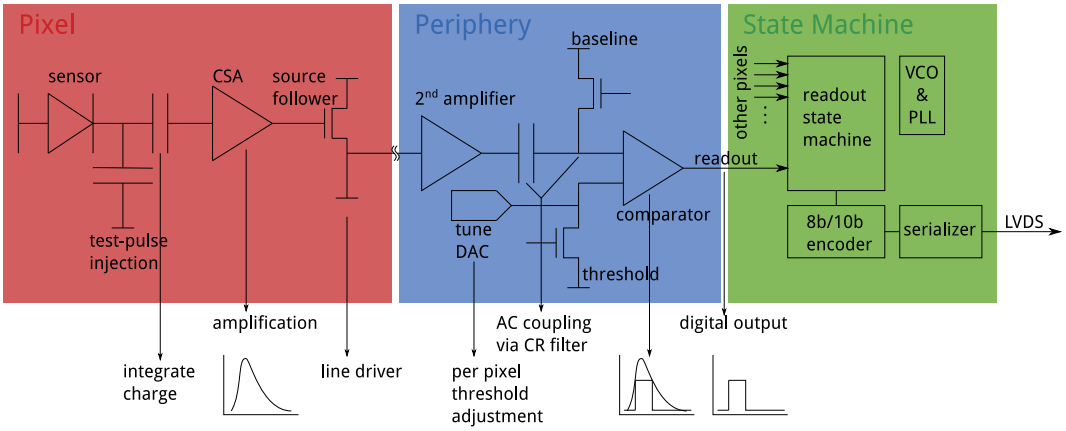


Figure 5.4.: Pixel electronics of the MuPix8 chip, taken from [44]. The red part shows the active part of the sensor, blue indicates the periphery which holds all readout electronics and the green section is the state machine reading out the pixel data.

readout state machine which produces zero-suppressed output. Before sending the data away via a 1.25 Gbit/s LVDS transmitter it gets 8b/10b encoded and serialized.

5.4. Fibre detector

(a) (b)

Figure 5.5.: computer-aided design (CAD) rendering of the fibre detector. In fig. 5.5b the colored parts are the positions of the fibre detector in the Mu3e experiment. In fig. 5.5a the connection between the fibre ribbons with the SiPMs via the MuTRiG PCB is shown, taken from [42].

The scintillating fibres are placed below the pixel detector. The fibres are grouped to 12 ribbons consisting of three layers (128 fibers) with a length of 300 mm and a width of 32.5 mm. They are connected to SiPM arrays at both ends. For the readout the MuTRiG ASIC is connected to a custom made PCB which is wired to the SiPMs via a flex print. In fig. 5.5 a CAD rendering of the placement of the fibre detector is given.

5.5. Tile detector

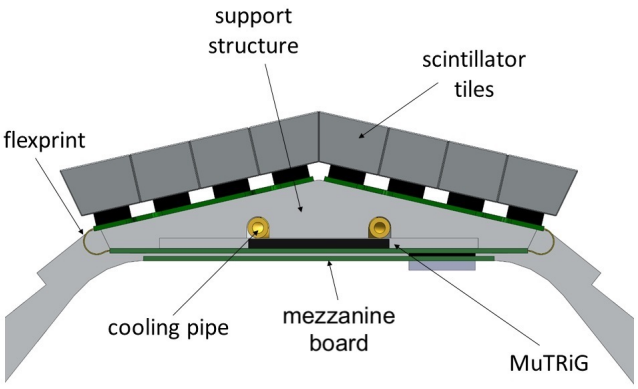


Figure 5.6.: Sketch of the tile detector. The tiles are connected to a PCB which holds the MuTRiG chip and embraces the cooling pips by the use of two flex prints. The PCB is then connected to a mezzanine board (MALIBU), taken from [42].

The tile detector is placed inside the two recurl stations and divided into segments with a length of 36.8 cm (in the beam direction). Each station holds 56 tiles in the beam direction and 56 tiles along the azimuthal angle. In fig. 5.6 a sketch of the tiles connected to their readout PCB is shown.

6 Data Acquisition System

As shown in chapter 5, the Mu3e experiment is searching for the decay $\mu^+ \rightarrow e^+e^+e^-$ which has a tiny BF. To detect this decay, the DAQ needs to read out all detector data. Since it would not be possible to save the total detector data, the DAQ needs to decide online which events should be saved to disk. In section 6.1, the expected data rate of the Mu3e experiment is calculated. In section 6.2 the readout system is explained. Finally, in section 6.3, the Maximum Integrated Data Acquisition System (MIDAS) is discussed. This system is used for controlling the DAQ of the Mu3e experiment.

6.1. Expected data rate

In Phase I of the Mu3e experiment 10^8 μ/s are stopped in the target leading to an average hit rate of all pixel detectors of 1057×10^6 hits/s. Starting from the noise rate of less than 1 Hz per pixel, measured for the current prototype (cf. [48]) the total noise rate for the pixel detector can be assumed to be less than 180 MHz taking into account the final MuPix sensor with 250 x 250 pixels and 2844 MuPix chips. Since each hit contains row and column address, time stamp and amplitude information and will be transmitted via 8b/10b encoding, the 32 bit hit will end up as 40 bit. So the total data rate from all pixel sensors will be 43 Gbit/s. Further information on the data rate can be found in [43].

The scintillating fibres are operating with a threshold of 0.5 photo electrons. This leads to the fact that the total SiPM dark count rate is read out on top of the actual physical signals. For reducing this rate, the front-end FPGAs, reading out the detector data, will form coincidences of signals. Further information on this can be read in [49]. In total a data rate of 26.3 Gbit/s from the scintillating fibres is expected.

The third detector part is the tile detector which sits at the recurl stations. Since there are much fewer particles hitting this part of the detector, the rate is less compared with the central station. The tiles have much more scintillation material than the fibres, which leads to a higher signal and allows for a much higher threshold. So the dark count rate is less than for the fibres. Further information on this can be read in [45]. The total data rate is expected to be 11.6 Gbit/s.

The total date rate of all detector systems will be 80.9 Gbit/s. This requires a readout system which is able to transport this amount of data out of the detector and reduce it to a manageable size.

6.2. Readout system

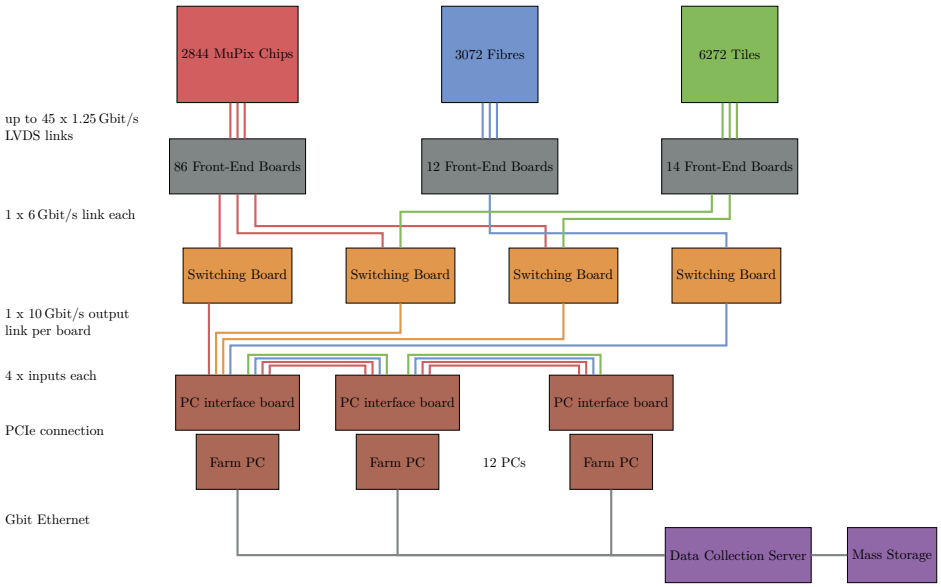


Figure 6.1.: Sketch of the Mu3e readout system.

In fig. 6.1, a sketch of the Mu3e readout system is given. For processing the data, the readout is build out of different slices which have different tasks. Each of them will be described in more detail in the next sections.

6.2.1. Front-end board

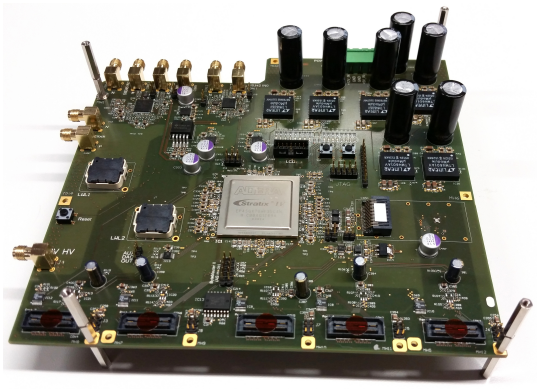


Figure 6.2.: Front-end board prototype with the Stratix IV FPGA.

As seen in fig. 6.1 every detector part is connected to a FPGA board. Since these boards are sitting directly at the detector, they are called front-end boards. These boards are custom developed for Mu3e, because they have to fulfill special requirements concerning form factor and used parts, since they will be placed inside the magnet in a helium atmosphere. The helium atmosphere is needed for cooling the different detector parts. The current prototype holds an Intel Stratix IV FPGA [50]. As discussed in section 2.4, the data stream of the pixel sensors is not sorted in time. Therefore, a time sorting algorithm is implemented inside the front-end boards connected to the pixel detectors [51]. After the sorting, clusters of hits are formed for the fibre events. The front-end boards which are connected to the fibre and tile detectors will in principle do the same for the fibre and tile hits. Then packets are build and send of via a 6 Gbit/s optical links to the switching boards.

Beside the data taking, the front-end boards also distribute the clock to the detector parts and send slow control information to the switching board. All of this is done over the same optical fibre where the data is sent through. Detector configurations are also shared via the front-end board to the detector parts. Since the configuration data for all sensors is not of the size of ”normal” slow control, the same optical fibre from the switching board is used for sending this configuration data. With ”normal” slow control humidity, temperature and other state informations of the detector is meant. Nevertheless, in the following both kinds are called slow control.

During development it turned out that the current version of the front-end board can not fulfill all of these tasks. For example the memory of the FPGA is not sufficient. Therefore, the second prototype, currently being designed, will be hosting an Arria V FPGA which fulfills these requirements.

6.2.2. Switching board

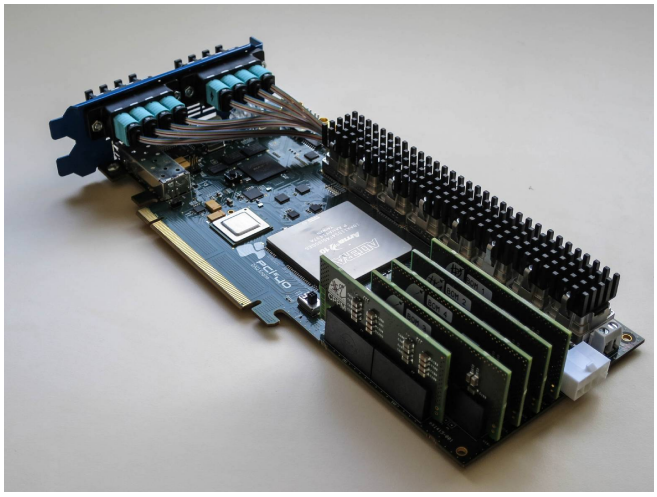


Figure 6.3.: Picture of the LHCb PCIe40 board used as the switching board, taken from [52].

The switching boards receive the data according to the geometric regions of the detector. There will be four boards. One will get data from the inner and outer central pixel detector, one from the fibre detector and two from the tile and pixel recurl stations.

The data from these regions are merged and time aligned and then transmitted to the PC interface board (PCIe board) via many 10 Gbit/s optical links. The used switching boards are the PCIe40 boards, see fig. 6.3 [53]. They are developed for the LHCb and ALICE upgrade. They come with 48 optical links, two PCIe 3.0 interfaces which have eight lanes each and two optical receivers for feeding in an external clock. The FPGA sitting on these boards is the Intel Arria 10. The optical links are connected to Avago Minipods [54]. The boards are placed in a PC and the slow control packets to the front-end boards can be sent via the PCIe interface directly to the board.

6.2.3. PC interface board

All four accumulated data streams from the switching boards are sent to one PCIe board sitting in a PC together with a graphics processing unit (GPU). The board is the Terasic DE5a-Net Arria X Development Kit which has four quad small-form-factor pluggable ports (QSFP) for receiving data from an optical link [55]. It also has eight lanes of PCIe 3.0 interface and a few gigabytes of Double Data Rate Synchronous Dynamic Random-Access Memory (DDR SDRAM). The task of this board is to merge the data from all four switching boards and store them in the off-chip memory. The hits from the central pixel detector of these stored packages are sent via a Direct Memory Access (DMA) interface to the GPU which runs a reconstruction algorithm for selecting potential signal events. These events will be then read out via a second DMA interface from the host PC. These so called farm PCs are daisy chained and each PC takes a fraction of the whole data stream and sends the other part to the next PC. In [56] it was shown that 12 of these PCs are sufficient for processing all data expected in the Phase I experiment.

6.2.4. Farm PC

As mentioned in the previous section, the farm PC hosts the PCIe board and a GPU on his motherboard. The GPU needs to perform many floating-point operations per second for executing the reconstruction algorithm which is a track fitting and vertex finding algorithm. Because of cost and usability reasons GPUs from Nvidia were chosen for this task since Nvidia provides one of the most well developed and user-friendly GPU computing interfaces on the market. For the host PC gaming hardware was chosen since it is has a lot of computing power and on board memory for doing the necessary operations.

6.2.5. Clock and reset system

For fulfilling the sub-ns timing precision requirements, the whole detector needs to be synchronized to an external clock and needs to get a synchronous reset at that precision. It is also important to minimize the clock jitter and the clock skew between the sensors. These demands are fulfilled by a custom-made clock and reset system. The chosen frequency for the Mu3e experiment is 125 MHz.

The whole system, built into the readout setup, which was built as part of this work, can be seen in fig. 6.4. The main part is the Genesys 2 Kintex-7 FPGA Development Board which

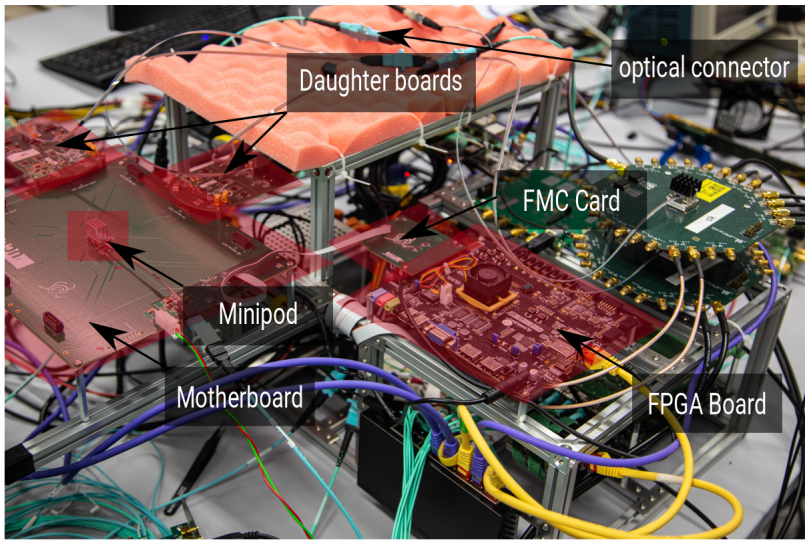


Figure 6.4.: Picture of the clock and reset system built into the readout test setup.

provides the 125 MHz clock and 1.25 Gb/s reset link [57]. They are sent off via a custom board connected with a FPGA Mezzanine Card (FMC) to the FPGA. Via two FireFly's the clock and reset line are sent to the mother board which copies them and distributs them electrically to 8 daughter boards hosting 3 FireFly's [58]. In total 14 clock and 14 reset lines are produced and sent out via an optical fiber connectors. The reset link is 8b/10b encoded and is able to send out a global synchronous reset to all connected front-end boards as well as performing other operations. Because the FPGAs operate with high speed data links, which can lead to data losses if the received data is sampled with a different clock, the global clock also needs to be fed into them. For the switching boards there will be a dedicated optical clock input. For the PCIe board a custom made board (clock transmission board (CTB) [59]) can be used. In chapter 7 a more detailed description of the used protocol is given. The whole specification of the clock and reset system can be found in [60, 61].

Since everything is intended to sit inside a 19-inch rack, a sufficient cooling system needs to be applied to it. This is done by mounting three fans which are controlled by a microcontroller connected to the FPGA. It is connected via Ethernet to the control system of the detector [62].

6.3. Maximum Integrated Data Acquisition System (MIDAS)

Beside the actual data taking the DAQ needs to be able to control and monitor the current status of the detector. For Mu3e this is done on multiple levels and accumulated in one single system. Maximum Integrated Data Acquisition System (MIDAS) is the main system for controlling the DAQ [64]. It is able to control a run, build the final events which are saved to disc, distribute slow control commands to all devices and has the possibility of on line monitoring. All of this is done in a web interface. Beside this MIDAS comes with its own serial bus Midas Slow Control Bus (MSCB) which will be used as a fall back solution if one of the front-boards needs to be reprogrammed, monitoring the environment inside detector,

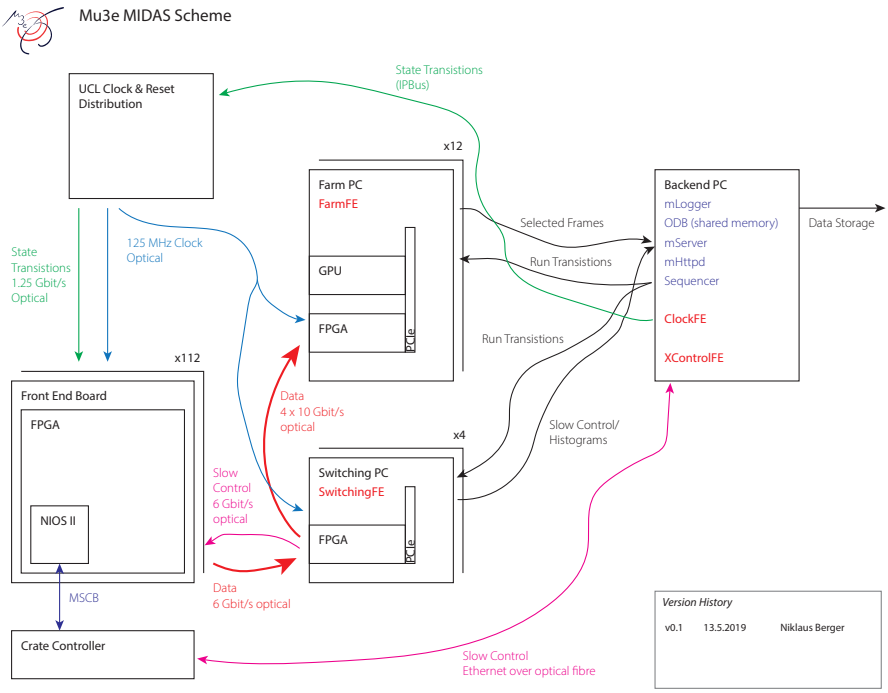


Figure 6.5.: Current version of the MIDAS scheme for the Mu3e experiment, taken from [63].

controlling the power distribution, etc. [65]. For this an independent data line is connected to each front-end board.

In fig. 6.5 the current version of the MIDAS scheme of the Mu3e experiment is shown. The back-end PC hosts the MIDAS server (mServer) which connects all the main MIDAS parts running on the other PCs of the system via Gbit Ethernet. Beside this the MIDAS logging system (mLogger), a shared database with the current detector settings, a Hypertext Transfer Protocol (HTTP) server instance called mHttpd for transferring the web data and a run sequencer (Sequencer) are running on this PC. The MIDAS instances are called front-ends (FEs) and they can interact with the different detector parts. The back-end PC hosts the ClockFE which controls the clock and reset system and the XControlFE which sends and receives slow control commands via an optical cable from the crate controller which converts this commands to MSCB and sends it to the front-end boards. This crate controller will be placed on the backplane of the crate where the front-end boards will be mounted to. On the front-end boards a NIOS II soft core processor is running which is controlled via this slow control line [66]. The back-end PC also sends slow control commands to the switching PC. There, the MIDAS FE for the switching boards is running, SwitchingFE, sending these commands to the front-end boards. In chapter 7 a detailed description on the slow control distribution is given. Via the IPbus protocol the state transitions are sent to the clock and reset system and are then distributed to the front-end boards [67]. Also, the run transitions are distributed to all detector parts over the back-end PC. With the use of Gbit Ethernet the selected event frames from the farm PC are sent to the back-end PC where they are sent of to the data storage.

Part III.

Test Setup

For building a functional detector it is necessary to test all individual parts of the system. This was done for most of the pieces of the Mu3e data acquisition system. Beside this integration tests which are close to the final system are crucial during the development of such a system. During the work of this thesis, such an integration was performed.

Important points of this design are the communication protocol formats which are explained in chapter 7, the data readout via PCIe (cf. chapter 8) and the firmware parts of the FPGAs for the data flow through the DAQ and the slow control system as described in chapter 9.

7 Communication Protocols

Communication protocols define the rule set for transmitting data between different deceives. They are import for a successful data transmission. In the following sections the ones used and developed in this thesis are described. This is the communication between the front-end board and the switching board described in section 7.1 and the communication with the clock and reset system discussed in section 7.2.

7.1. Communication front-end boards to switching boards

Since all detector data and detector configuration will be merged at some point during the readout process, a common communication protocol helps to design a dynamic firmware. For an experiment with a lot of data like the Mu3e experiment it is also important that the protocol needs to be designed in a way that there is no unnecessary overhead limiting the data rate. The specification of this common protocol between the front-end and switching boards which was developed during the work on this thesis is shown in the following.

Each data word contains 32 bit and the bits inside the firmware of the FPGAs are sent out via the link with the least significant bit first. All packets start with a preamble containing the type of the packet and the K-symbol K28.5. Also, all packets end with the K-symbol K28.4. In section 7.1.1 the protocol for MuPix data is described. After this the format for MuTRiG data is shown in section 7.1.2 followed by the slow control protocol in section 7.1.3 and the run control signals in section 7.1.4 at the end the idle state is explained in section 7.1.5. In the following, the bit counting is done by setting the least significant to 0 and the most significant one to 31.

7.1.1. MuPix communication protocol

Inside the preamble the comma word needs to be the least significant byte. So only the bits 31 down to 8 are available. In table 7.1 the communication protocol is shown in a byte field. The structure of the communication is in the way that first the preamble is sent which contains an identification which type of package will be sent. For MuPix data this is the pattern (111010) located in the bits 31 down to 26. Also, the front-end FPGA ID is sent with the bits 23 down to 8. After this, the MuPix Data Header is sent. This contains the 48 bit FPGA time stamp which should be sufficient for a run time of around 3 h. After this the sub-Header is sent, which contains the indicator (111111), the bits 9 down to 4 of the FPGA time stamp and 16 bit of overflow indicator from the hit sorter. This overflow bits show which time stamps have an overflow of the hit counter. This counter counts the number of hits with the corresponding time stamp which is generated inside the front-end board during data taking. If this time stamp overflows, the last hit is overwritten.

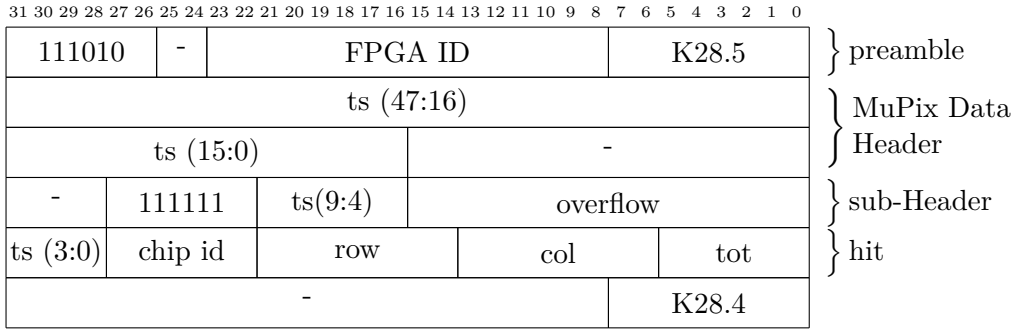


Table 7.1.: Structure of the MuPix Data.

Now the hits are sent until the lowest 4 bit of the 48 bit time stamp overflow. Then the sub-Header is sent again until all 10 bit overflow. After this a package is finished and the trailer will be sent. By not sending always the full 10 bit for every hit, it is possible to reach the desired data rate.

7.1.2. MuTRiG communication protocol

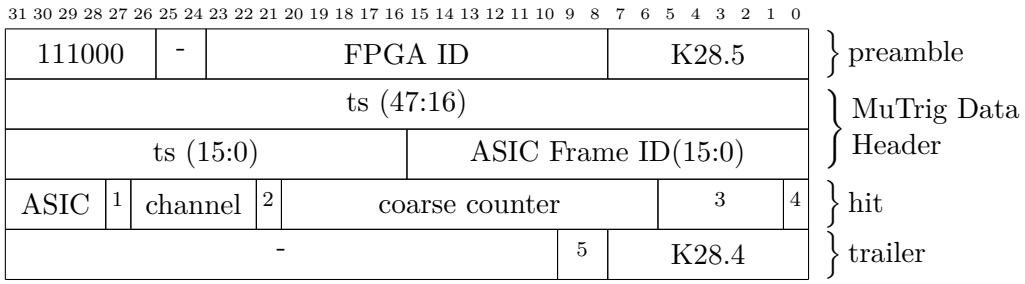


Table 7.2.: Structure of the MuTrig data packet. Here [1] is the hit type which is 1 for a T part and 0 for a E part, [2] indicates the time stamp of a bad hit, [3] is the energy flag, [4] is the fine counter, the first bit of [5] indicates if the L2 FIFO inside the ASIC has an overflow and the second bit indicates if there was an ASIC cyclic redundancy check (CRC) error.

Similar to the MuPix communication protocol, the MuTRiG one was defined and is shown in table 7.2. There the type in the preamble is given with the pattern (111000) followed by the MuTRiG data header. Since there is no hit sorter at the moment, the packet contains all hits from one frame received from the ASICs. The data header contains the 48 bit FPGA time stamp and the frame number followed by the hit data. In the end, the trailer with the same ASIC information is sent. An exact description of the MuTRiG data can be read in [46].

31 30 29 28 27 26	25 24	23 22 21 20 19 18 17 16 15 14 13 12 11 10	9 8 7 6 5 4 3 2 1 0	
000111	SC	FPGA ID	header K28.5	preamble
start address				
-		length		read
-		length		write
data				write
data				write
-			trailer (K28.4)	

Table 7.3.: Structure of the Slow Control protocol.

7.1.3. Slow control communication protocol

The preamble for the slow control data has the pattern (000111) and the type of slow control. Here (11) stands for the write-command and (10) for the read-command always from the position of a user sitting in the back-end PC and wanting to write or read to and from one front-end board. After this, the start address will be sent. The slow control settings for the measurements are stored in a Random-Access Memory (RAM) implemented on the FPGA. So the start address is the actual address of the first register inside this RAM where the user wants to read or write to. A more specific explanation how the slow control packets are processed inside the FPGAs are given in section 9.5. The next command will be the length indicating the number of registers after the start address which want to be read or written to. In case of a write-command the data for these registers will be sent afterwards. In the end, the package will be closed with the trailer word.

Every slow control request is acknowledged by the receiving FPGA. The response uses the same scheme of preamble and start address. If it was a read-command, the data in the registers are sent back. If it was a write-command, the 16 bit will be used as a write acknowledge.

7.1.4. Run control signals

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
run control signal	K30.7

Table 7.4.: Structure of run control signals.

Since the clock and reset system can only send a reset command to the front-end boards, there need to be a way to acknowledge this command. This is done by sending run control signals from the front-end boards to the switching boards. They get read out there and back played to MIDAS. The comma word K30.7 is reserved for these signals. In table 7.4 the run control signals are shown. Only one word is send with the according run command. The different run

commands explained more in detail in [68].

7.1.5. Idle state

If there are no data, idle packages are sent. They contain only the K-symbol K28.5. With these words, byte alignment is performed. They also change the states of the state machines running on the different FPGAs such that they do not perform any calculations. The byte alignment is done by a entity, implemented after the transceiver, which shifts the bytes of the 32 bit word by adjusting the position of the K-symbol K28.5 according to the defined communication protocol.

7.2. Communication of clock and reset system

Command	Code	Payload
Run Prepare	0x10	32 bit run number
Sync	0x11	-
Start Run	0x12	-
End Run	0x13	-
Abort Run	0x14	-
Start Link Test	0x20	-
Stop Link Test	0x21	-
Start Sync Test	0x24	-
Stop Sync Test	0x25	-
Test Sync	0x26	-
Reset	0x30	16 bit mask
Stop Reset	0x31	16 bit mask
Enable	0x32	-
Disable	0x33	-
Address	0x40	16 bit address

Table 7.5.: Reset link protocol.

As described in section 6.2.5 the Mu3e DAQ has a special clock and reset system. This system communicates with all front-end boards via a separate 1.25 Gbit/s optical reset link for changing the run states. This link works with 8b/10b decoded 8 bit words. For the idle state the K-symbol K28.5 is sent. The commands for the different runstates are given in table 7.5. For starting a normal run sequence, starting from the idle state, the following steps are necessary:

1. By sending **Run Prepare** the run number is distributed to all front-end boards. The command also triggers the transition to **Run Prepare**. In this state, FIFOs, buffers, etc. will be cleared. Also an active signal is transfered via the switching board to the MIDAS system. If all active signals are received, the next transition can be sent off.
2. The next signal is **Sync**, where all detector ASICs go into reset for at least a full time-stamp cycle.

3. With **Start Run** the reset gets released, the time-stamp starts to count from 0 and data are send off to the switching boards.
4. For ending a run, the **End Run** signal is sent. Then the system is going into the termination state and no new data is sent off from the front-end board. After the last data package, the run trailer is sent.

For aborting a run, the **Run abort** command can be sent. This triggers the transition to the idle state. Besides these states, there are also the options to start BERTs over all links (except the reset link) with **Start Link Test** and stop BERTs with **Stop Link Test**. With the **Start Sync Test** signal logic on the front-end boards starts to measure the phase of test signals coming over the reset link. This can be stopped by sending **Stop Sync Test**. Via the **Reset** signal and mask bits in the payload, parts of the system can be turned into reset. With **Stop Reset** they can be released. With the combination of **Enable** or **Disable** and **Address** parts of the system can be enabled or disabled for the DAQ. From the firmware side, all of this is implemented into the state controller entity located on the front-end board. A detailed description how this is implemented into the firmware can be found in [68]. The specification of the whole system is given in [60, 62, 69]. The software part for controlling these transitions is implemented into MIDAS by [70].

8 Data Readout via PCIe

Since the switching boards and the PCIe boards are placed inside PCs and all parts of the DAQ of the Mu3e experiment are controlled via MIDAS which is also running on these PCs, it is necessary to provide a communication system between them. As both FPGAs are equipped with PCIe interfaces, this high-speed serial bus standard is used. In the following, an overview of the data readout based on PCIe is given. First the PCIe bus and DMA are explained in section 8.1 and section 8.2. Then the current firmware implementation of the DMA block is shown in section 8.3. At the end, in section 8.4, a variation of the DMA block is shown which guarantees a deterministic data rate. Section 8.1, section 8.2 and section 8.3 are based on [56] and [71].

8.1. Peripheral Component Interconnect Express

Version	Introduced	Line code	Transfer rate	Throughput for 8 lines
1.0 [72]	2003	8b/10b	2.5 GT/s	2.0 GB/s
2.0 [73]	2007	8b/10b	5.0 GT/s	4.0 GB/s
3.0 [38]	2010	128b/130b	8.0 GT/s	7.88 GB/s
4.0 [74]	2017	128b/130b	16.0 GT/s	15.75 GB/s

Table 8.1.: PCIe link performance, taken from [75].

PCIe [72, 73, 38, 74] is a high-speed serial computer expansion bus standard which is specified by the PCIe Special Interest Group (PCI-SIG) [76]. It is commonly used for connecting GPUs, hard disk drives (HDDs) or Solid-State-Drives (SSDs) with the CPU over the motherboard of a PC. In contrast to the older bus standards like Peripheral Component Interconnect (PCI), PCIe is not really a bus. It is a point-to-point connection where each device has its own connection to a network switch. Each of these lanes has four wires for receiving and transmitting data (two differential pairs each) and a pair of wires connecting the device to the reference clock. The PCIe link is able to provide a full-duplex communication between two devices. For this one to 32 lanes can be used. Since it is a network based communication, the transaction takes place in the form of package transmissions. Comparable to a local Ethernet network, flow control, error detection and retransmissions take place. For the Media Access Controller (MAC), which is the physical address of a device inside a network connection mostly stored in hardware by the manufacturer, the physical position of the device inside the motherboard is used and later translated to a high-level address. In table 8.1 the data rates of different PCIe generations are shown.

For the communication via a PCIe link, three layers are involved. The Transaction Layer,

the Data Link Layer and the Physical Layer. For sending a write, read or the response of a read packet (completion packet), Transaction Layer Packets (TLPs) are used. For ensuring that every TLP arrives at the right destination, the Data Link Layer is used. This is done by wrapping each TLP together with its own header and a link CRC. For checking that no TLP gets lost on the way each one gets acknowledged. As mentioned earlier, a flow control mechanism takes care that the package is only sent if the link partner is ready to receive data. This ensures that every package which is transmitted to the Data Link Layer will arrive with a slight uncertainty on the arrival time.

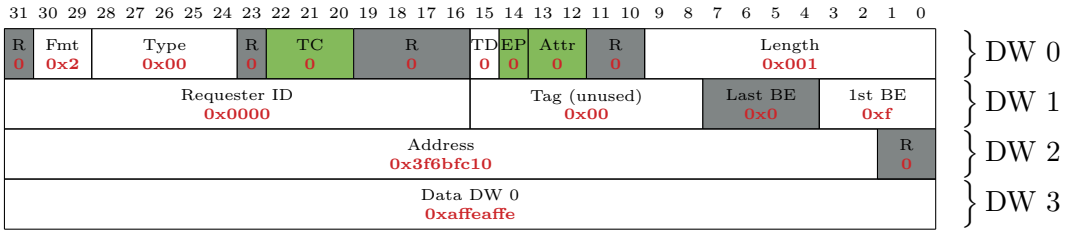


Table 8.2.: PCIe write request. Taken from [71]

Concerning complexity, a write TLP is the easiest transaction which can take place, because it does not require an answer. It contains only the destination device, the write address and the data. In table 8.2 an example of a write package is shown. In this example, the CPU wrote the value 0xaffeaffe to the address 0xfdaff040 by using 32 Bit addressing. Therefore, the package needs to consist out of four 32 Bit double words (DWs). The first three DWs are the header of the package and the last one is the data which will be written. The gray fields marked with an R are not used in any package. If they are only marked gray, they are not used in this specific write package. Fields colored in green can be nonzero and the values of this example package are marked red.

Here Format (Fmt) and Type indicate that this package is a write command. Having the TD bit set to zero shows that there is no extra CRC for the TLP data. Since the Data Link Layer comes with its own CRC, no extra one is needed, if one assumes that the TLPs do not get changed by the device. The next field is the Length, which shows how many 32 Bit words of data are sent. Followed by the Requester ID, which shows zeros. This says that the packet comes form the PCIe port closest to the CPU which is called the Root Complex. The Tag field can be fill with anything the sender wants. In this example it is set to zero. With setting the 1st Byte Enable (BE) to 0xf all four bytes in the first data DW are valid. Since in this case the length is one and only one DW data is sent, the Last BE needs to be zero. Before the data DW, the write address for the first data DW is sent. Since the last significant bits (LSBs) of the second DW are zero the write address is written by itself. For getting the right address one needs to multiply 0x3f6bfc10 by four which is 0xfdaff040. Concerning the data which is transferred via PCIe it is important to know that it is big Endian and Intel processors which are used in the Mu3e experiment are using little Endian. So the data for the CPU in this case will be 0xfeaffeaf. For addressing more then 4 GB four DWs are sent for the header using two 32 Bit words. According to the specification [72] the receiver can not address 32 Bit addresses with four DWs. So it is important to distinguish between 32 Bit and 64 Bit addresses. The payload for a write command can be much longer than a single 32 Bit word. The maximum size is limited by the configuration of the peripheral. For Mu3e, the FPGAs which are connected via PCIe have a maximum payload of 2048 Byte [77].

Since in Mu3e only write commands are used for sending the data to the PC only a short comment for the other two packages: The read command needs in addition the read address and the length. It then gets a completion package as reply from the device which contains the requested data.

8.2. Direct Memory Access

Since any device inside the PCIe network is able to write or read TLPs to and from any other device, it is possible to perform Direct Memory Access (DMA) via this network. DMA is a feature for accessing the memory of a system independent of the CPU. Normally, this is done by programmed I/O. The advantage of DMA is that the CPU is not busy with performing the read and write operations like in I/O. Since a lot of other programs are running on the PCs of the Mu3e experiment, the CPU is more or less always busy so this behavior is desired. Nevertheless, it is possible to send an interrupt from the DMA controller to the CPU, if one wants to do operations on the received data. Sending this interrupt however stops the DMA processes and leads to a loss of data rate.

For achieving DMA via a PCIe device it needs to know about the physical buffer address to which the data needs to be transfered. This can be done by writing a software driver which informs the peripheral about the buffer size and address. In a Unix system like Linux this driver is loaded into the Linux kernel, which is the central part of the operating system providing interfaces between various devices. The task of the driver is to allow the kernel to communicate with specific hardware. Beside this, the memory in a Linux system is organized in two different parts, user and kernel space. Normal programs run in the user space and only the kernel code is stored inside the kernel space. For accessing different parts of memory, the Linux memory manager can map physical memory from RAM or HDD to virtual memory in such a way that they appear as a block of user space memory, seen from a program. This is done by using units of pages which are placed in a page table. Pages are continuous blocks of virtual memory with a fixed length. Furthermore, they are the smallest units inside the memory management. The size of these pages is set by the used memory management unit of the CPU. For Intel x86 systems they are set to 4 kB. When performing DMA, the driver needs to allocate page-locked memory meaning that this memory will not be changed by other programs during the DMA operation.

For informing the FPGA about the mapped address regions, the CPU can use dedicated memory of the PCIe device via Memory-mapped input/output (MMIO). The PCIe device is providing this by the use of Base Address Registers (BARs). The driver can share the page addresses and the size of the DMA memory by sending them via a PCIe write command to one of the BARs. Also the FPGA can do this in the other direction. For the Mu3e experiment four BARs are implemented on the FPGA. Two of them are used for sharing static informations like DMA memory size or status flags of the FPGA. They are called write and read registers, seen from the PC. Additionally there are two BARs (write and read memory BARs) for dynamically informations like the slow control communication between the switching board and MIDAS. They also come in a write and read direction, seen from the PC.

8.3. Direct Memory Access block implementation

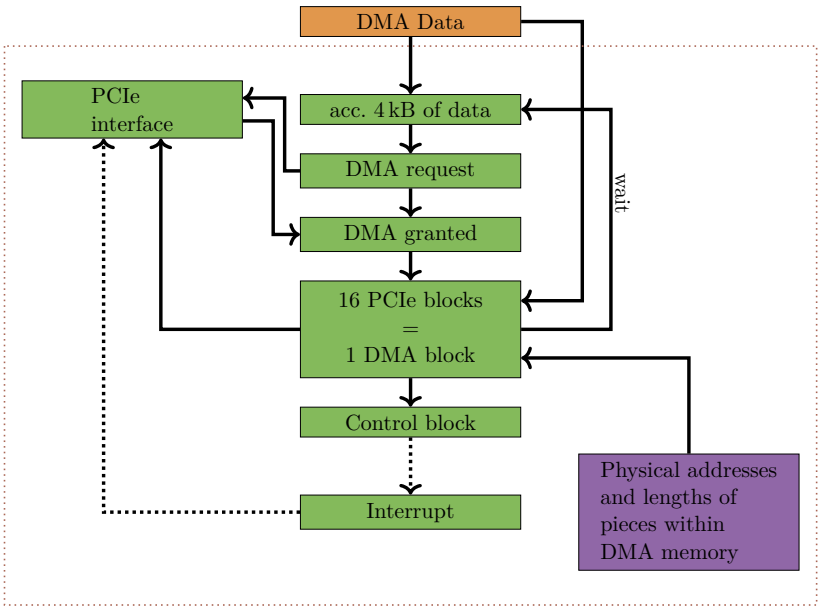


Figure 8.1.: Design of the firmware block for data readout via DMA. Partially taken from [56].

In fig. 8.1 the firmware implementation for sending data from the FPGA via DMA done in [56] is shown. In the Mu3e experiment, the PCIe board will have two of these DMA blocks. One will be used for sending the inner pixel data to the GPU which is doing the on line tracking algorithm and the other one will send the selected data from the GPU to MIDAS.

The data which should be sent via DMA will be first stored in a buffer until 4 kB is accumulated. Then the firmware is requesting a DMA transmission. This request is processed by the PCIe IP, provided by Intel, which takes care of the communication with the PCIe link. As mentioned in section 8.2, the conversation with the CPU is done via BARs. For the DMA requests, a priority list is used and if no read or write request is pending on the link, the PCIe interface releases the link for the request. After granting the requests the so called DMA block is split into 16 256 byte PCIe packets. These packets are sent by the use of the TLP to a memory address inside the RAM. The size of the DMA block is chosen to be 4 kB, because of the trade-off between not using the link for too long and sending a big block of data at once which does not require too much flow control. This leads to requesting always a new DMA transmission after sending one DMA block. In section 8.1 it was discussed that this sequence guarantees that the package will arrive but also comes with a nondeterministic waiting time until the DMA request is granted. Therefore, the buffer, which is constantly filled with data, can overflow. Since the DAQ in the Mu3e experiment is always sending data, it is important to have a deterministic data rate at any given point of the system. Otherwise data could get lost during the readout process. In section 8.4 a solution of this problem is shown.

After sending the DMA block, a control block is sent to another address inside the RAM, which can be polled by the CPU informing about the transmitted data. Besides this an interrupt can be sent as Message Signalled Interrupt (MSI). The interrupt feature is done by sending a write request to a specific address which is known by the device driver for triggering the interrupt handler. This way every 64 DMA blocks one interrupt is sent. However, sending an interrupt decreases the data rate which can be transmitted via the DMA. In [56] the data rate with interrupts and no errors was measured to be 1.5 GB/s, by sending data from the Altera Stratix IV [50] and the Stratix V [78] with the PCIe 2.0 standard from the FPGA to the GPU. According to table 8.1 the maximum speed for 8 lanes of PCIe 2.0 is 4 GB/s. The errors comes from handling the interrupts and sending housekeeping signals over the link which leads to an overflow of the memory inside the DMA block which accumulates the 4 kB of data.

8.4. Direct Memory Access rate measurements

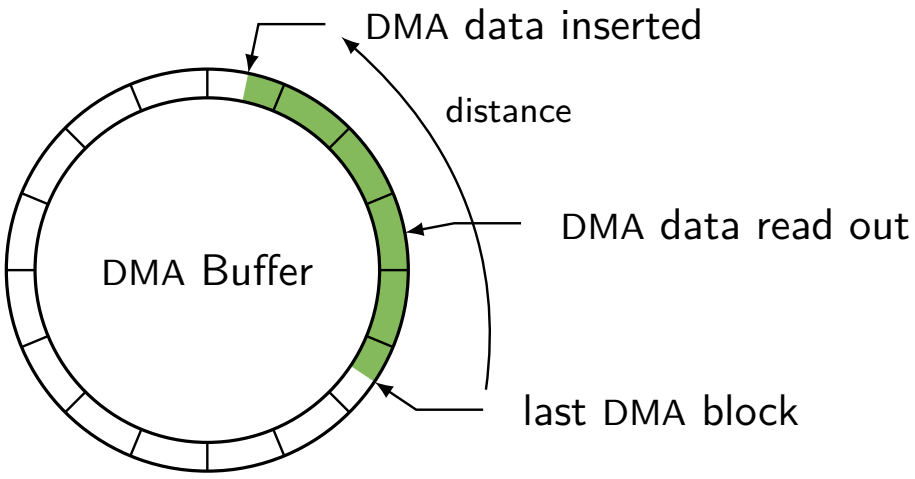


Figure 8.2.: Sketch of the half-full flag of the ring buffer inside the DMA block.

To guarantee a deterministic and sufficient data rate the interrupts where disabled and a half-full flag was implemented. In fig. 8.2 the work of this flag is illustrated. For the accumulation of the data a buffer was inserted inside the DMA block. For measuring if this functionality is working the buffer is written from the FPGA with a changeable data rate. The current write pointer of this data is marked as **DMA data inserted**. On the other side the DMA block reads from this data and accumulates 4 kB of data. The end of the last block is marked with **last DMA block** and the current read pointer from this buffer to the accumulation process is marked with **DMA data read out**. If the distance between the **last DMA block** and the **DMA data inserted** pointer is greater than half the total buffer size, the half-full flag is set to one. By checking this flag, the data which gets inserted into the DMA block can be stopped until it gets back to zero.

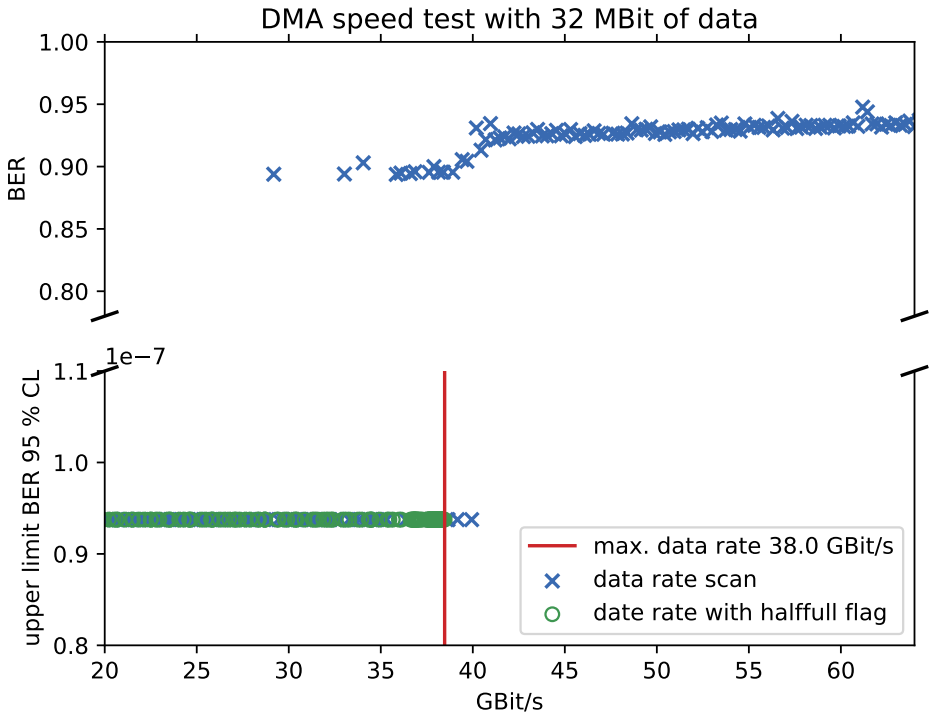


Figure 8.3.: Data rate scans with the use of the half-full flag marked in green and without marked in blue. This was done by sending for every point 32 MB of data.

In fig. 8.3 a rate scan was performed by changing the rate of a process inside the FPGA which sends data to the DMA block and then to the RAM of the PC. On the PC software checks for wrong data patterns. This scan was done once without stopping to write to the buffer and once with using the half-full flag. The measurement was done by using the Terasic DE5a-Net Arria X Development Kit which uses PCIe 3.0. According to table 8.1 a theoretical data rate of 63.04 GBit/s for 8 lanes of PCIe is possible. By not using the half-full flag, one can clearly see the nondeterministic behavior of the PCIe housekeeping, because around 30 GB/s it is possible to get errors leading to a constant increase in the errors above 40 GBit/s. On the other hand, the use of the half-full flag is able to dynamically adjust the data rate to the current housekeeping load and guarantees no errors. For calculating the real data rate with the half-full flag, the clock cycles of having the flag set to one was subtracted from the scanned rate. Since PCIe 3.0 does 128b/130b line encoding, only 1.54 % of the theoretical data rate of 64 GBit/s comes from overhead. This leads to the loss of 25.04 GBit/s according to housekeeping signal. Nevertheless, the maximum data rate of 38 GBit/s is sufficient, since according to [56] only 32.8 GBit/s are needed for the Mu3e experiment.

9 Data Flow and Firmware Design

The data flow inside the DAQ of the Mu3e experiment is managed with the use of different FPGAs and PCBs installed in the detector. For forwarding the data between different devices of the system, the firmware of the FPGAs needs to be designed to be able to handle the expected rate. Also controlling the whole system and configuring the detector parts need to be achieved by there circuits.

In the following, the firmware design of the data flow starting from the senors (cf. section 9.1), over the parts inside the front-end board (cf. section 9.2), followed by the communication between the front-end board and the switching board in section 9.3 and the readout via DMA done on the PCIe board in section 9.4 is explained. At the end of this chapter the slow control configuration between the switching board and the front-end board is shown in section 9.5.

9.1. Sensor printed circuit boards (PCBs)

For connecting the pixel sensors to the front-end board a custom PCB was designed [79]. The same purpose is fulfilled by the MALIBU board for the tiles and fibres detectors. Both supply the chips with the necessary voltages, clocks and other signals. At the MuPix PCB the chip is glued to a so called insert which can be connected to it. The baseline voltages and the thresholds for the MuPix8 are controlled by adjustable voltage dividers respectively on-board DACs. The MALIBU board can be configured to power up and down the STiC and start a readout test. All of there configurations can be set by the use of the slow control system designed during the work of this thesis.

9.2. Communication inside the front-end boards

The front-end boards should contain a common firmware part handling e.g. the state transitions or the data sending to the switching board. Beside this, the front-end boards are connected to different sub-detector parts which have all specific data readout systems. For merging these data streams together, the communication between the common firmware parts and the specific parts need to be merged together. Their packet assembling is explained in [68].

9.3. Front-end board to switching board

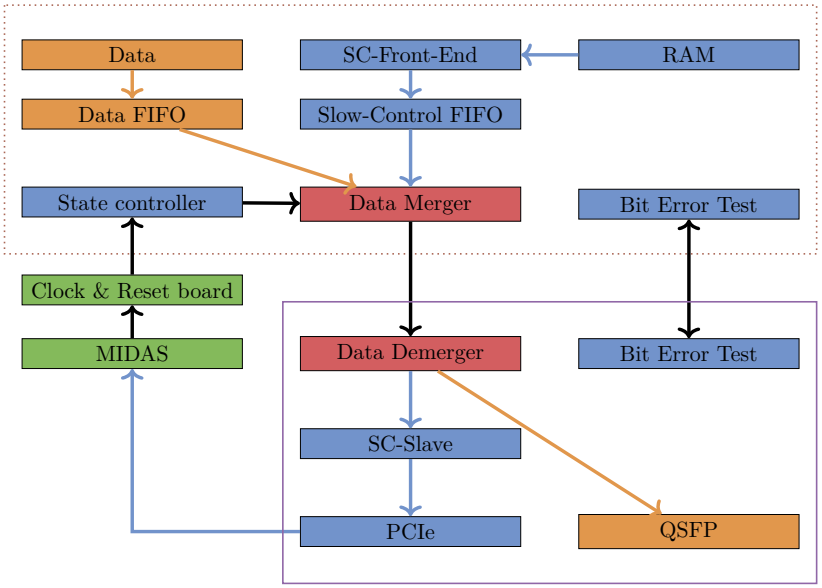


Figure 9.1.: Design blocks for the data and slow control communication from the front-end board to the switching board. The dotted line marks the firmware inside the ArriaV the normal line inside the ArriaX.

For generating random data in the format of the hit data from the MuPix communication protocol shown in section 7.1.1, an entity on the front-end board was designed. This data is buffered in a FIFO which is connected to an entity which acts as a multiplexer for slow control data and data according to the current run state. The internal RAM on the front-end board is used for storing slow control data coming eighter from the NIOS processor or from a detector. This RAM is read out by an entity called SC-Front-End. After this, the slow control data is stored in a FIFO which is connected to the data merger. A more detailed description on the slow control communication between the two boards is given in section 9.5.

The data merger sends the data in a parallel stream to an optical transceiver which then sends it to the switching board. On the switching board, a demultiplexer is receiving the data from an optical receiver and splits it into slow control and data. The slow control is then sent off via PCIe to the MIDAS front-end while the data is sent again via an optical transceiver to the PCIe board.

The clock and reset board is controlled via MIDAS and signals the current run state to the front-end board with an optical cable. Here the protocol described in section 7.2 is used.

Beside this, a bypass for bit error tests is implemented. This is necessary for testing the optical link independently of other hardware components.

9.4. Switching board to PCIe board

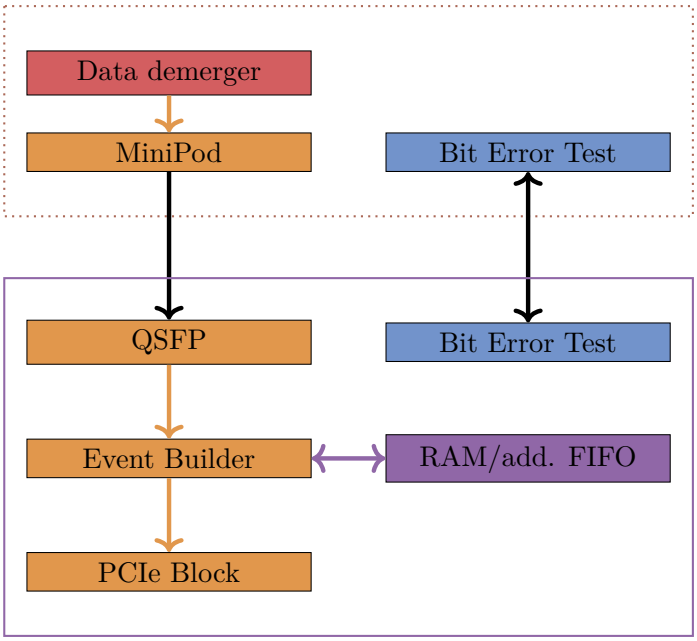


Figure 9.2.: Design blocks for data communication from the switching board to the PCIe board. The dotted line marks the firmware inside the switching board the normal line inside the PCIe board.

After the data was passed through the switching board it will be sent to the PCIe board where it will be read out by the use of the DMA block. In the final DAQ the hits from the detectors will be time aligned on the switching board. Since this part of the firmware is not implemented at the moment, the data will be just forwarded without any modification. On the PCIe board itself the packages will be counted by the event counter. This entity buffers the data stream in to a RAM and saves the RAM addresses of the trailers in a FIFO. By subtracting the addresses of two trailer words, the length of the package can be calculated and the DMA engine can be turned on, if a whole package arrived on the PCIe board. At the actual experiment this package building can be used with the DDR SDRAM and the GPU by saving the DDR SDRAM addresses inside the RAM and tacking the GPU selection as a trigger for enabling the DMA readout.

Another feature of this readout is that the PC knows how long one package is. This is important because checking every single word inside the mapped DMA memory leads to the problem, that the FPGA is writing faster to this memory than the PC can read from it. By knowing the length, the PC can climb up hand over hand through the memory.

Like for the communication between the front-end board and switching board entities for BERTs are implemented.

9.5. Slow control from switching board to front-end board

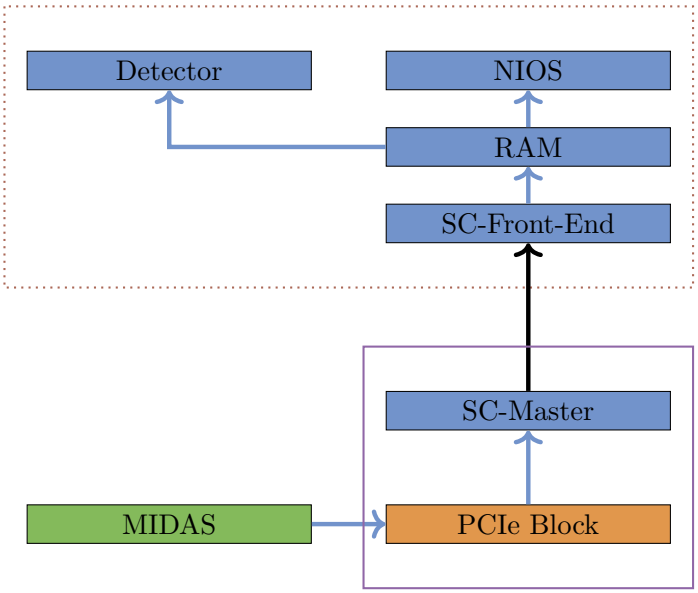


Figure 9.3.: Design blocks for the slow control communication from the switching board to the front-end board. The dotted line marks the firmware inside the ArriaV, the normal line inside the ArriaX.

The communication from the switching board to the front-end board is only slow control. Here the MIDAS front-end sends a slow control command via PCIe to the switching board. On the switching board the SC-Master entity is running which takes this command and sends it to the according front-end board via an optical link. The SC-Front-End entity unpacks this command on the front-end board and sends it to the RAM. The RAM addresses are mapped either to the NIOS or to the detector.

The design for sending a slow control commands from MIDAS to the front-end board is shown in fig. 9.4. First one has to write the slow control protocol into the write memory of the PCIe interface. After this a start marker has to be written in front of the preamble address. By seeing the start marker the SC Master starts to read out the slow control package and sends it to the transceiver which transfers it to the front-end board. On the front-end board, the SC-Front-End decodes the package. In the case of a write command the entity starts at the address given in the protocol and writes the data to a RAM. This RAM is split into three sections. The first one contains 64 k of 32 bit words. The second part has 256 32 bit words which are directly connected to peripheries like the MuPix for example. The last region also contains 256 32 bit words. The first address of this region is called **cmdlen** and writting to this address triggers an interrupt to the NIOS processor which will then read the second address **offset**. In the first half of the **cmdlen** word one can specify what the NIOS should do. For

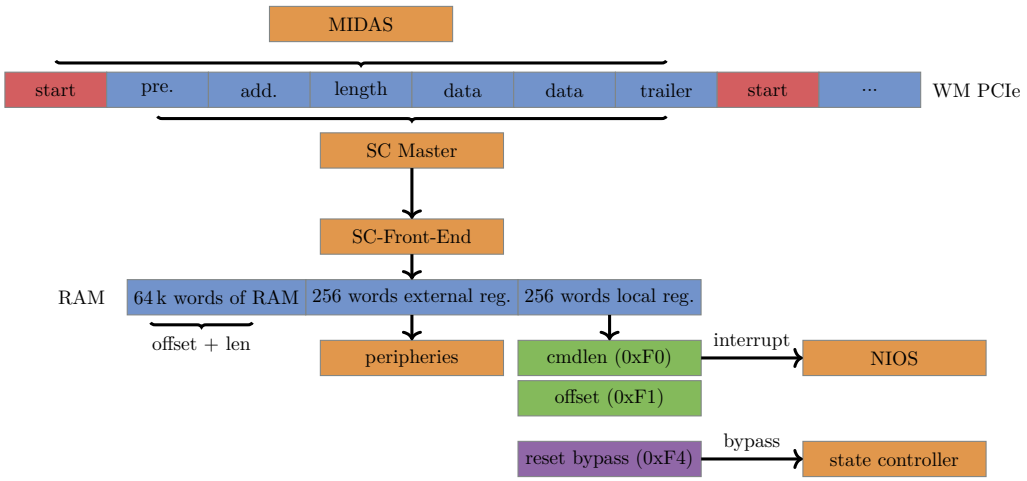


Figure 9.4.: Design blocks for writing a slow control command from MIDAS to the front-end board.

example by setting it to 0x0101 it will read from the first part of the RAM the number of words specified in the second half of **cmdlen** starting from the address set in **offset** and sends the data to the MALIBU board. By using this kind of protocol one can setup up different control sequences which are controlled from MIDAS. Beside this the state controller can also be bypassed by sending a reset command to the 4th address in the third part of the RAM. This function is important since one has a redundant option for changing the state of the front-end board in the case the clock and reset systems does not work or the link to the clock and reset system is broken.

After the slow control package is processed by the SC-Front-End entity an acknowledge is send back to the switching board. This acknowledge is decoded by the SC Slave and the package is written to the read memory of the PCIe interface. The MIDAS front-end is permanently looking into this memory and reads out new packages. Since this whole process is described over different devices, the software running on the PC needs to wait for this acknowledge to continue.

In fig. 9.5 the web page of this system running on the mServer is shown. By pushing the different buttons on this webpage one can trigger the above described protocol. For example a simple write command can be triggered by first specifying the front-end board one wants to write to by setting the **FPGA ID**. The values for **Start Address RAM** and **Data** are the RAM address and the data which will be written to this address. With the button **Send single data** every gets executed. It is also possible to load a file into the system which has the whole specification data for one of the detectors stored. This can be done by clicking on **Load file** and after setting the **FPGA ID** and **Start Address RAM** one can trigger the file writting with **Send file data**. The same can be done for a read command with **Send SC Read packet**. The write memory and read memory can be monitored by setting a start value and length and the click on either **Read RM PCIe** for reading the read memory or **Read WM PCIe** for reading from the write memory. Beside this the user can also reset

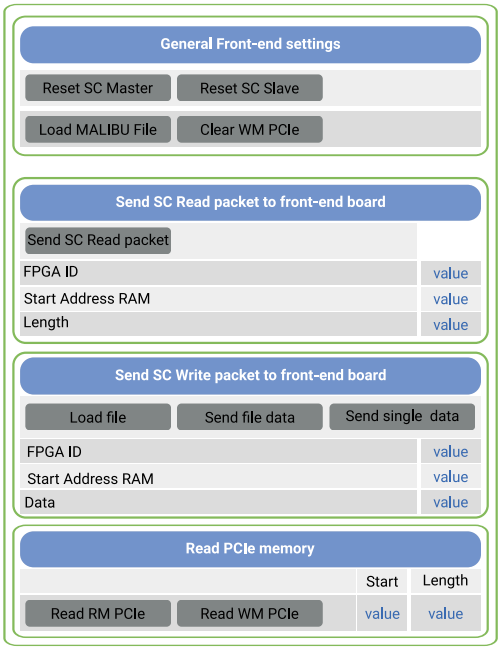


Figure 9.5.: MIDAS front-end panel of the basic slow control functions.

the SC Master and SC Slave by clicking on **Reset SC Master** or **Reset SC Slave**. This function is important since before starting a run all firmware parts need to be reset.

10 Test Setup

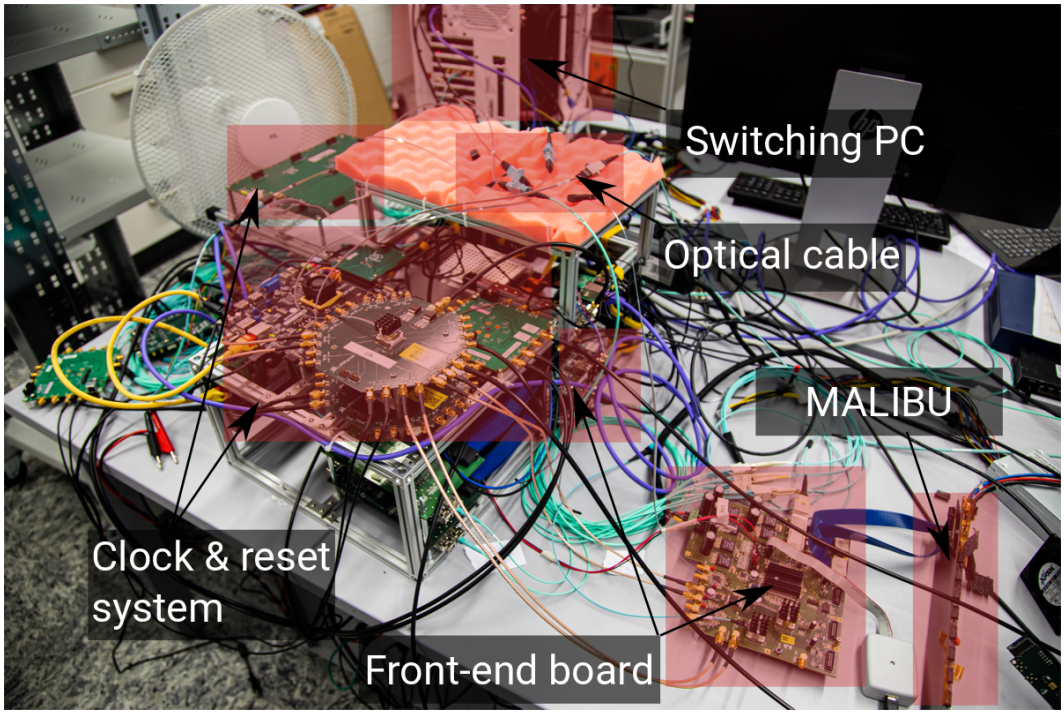


Figure 10.1.: Photo of the DAQ test setup. The main parts are colored in red. On the bottom right the MALIBU board is connected to the front-end board prototype. Additional Arria V boards are also marked as front-end boards. The clock and reset system is placed on the left side. All parts are connected via optical cables which can be manually changed for performing different tests. On the top the switching PC is located which is equipped with an Arria X board.

Figure 10.1 shows the test setup which was built during the work of this thesis. The setup allows different DAQ test and provides a whole vertical slice of the Mu3e DAQ.

In fig. 10.2 a sketch of this test setup is shown. The blue boxes are the used FPGAs and the clock and reset system. The green ones are the parts on which the MIDAS system is running

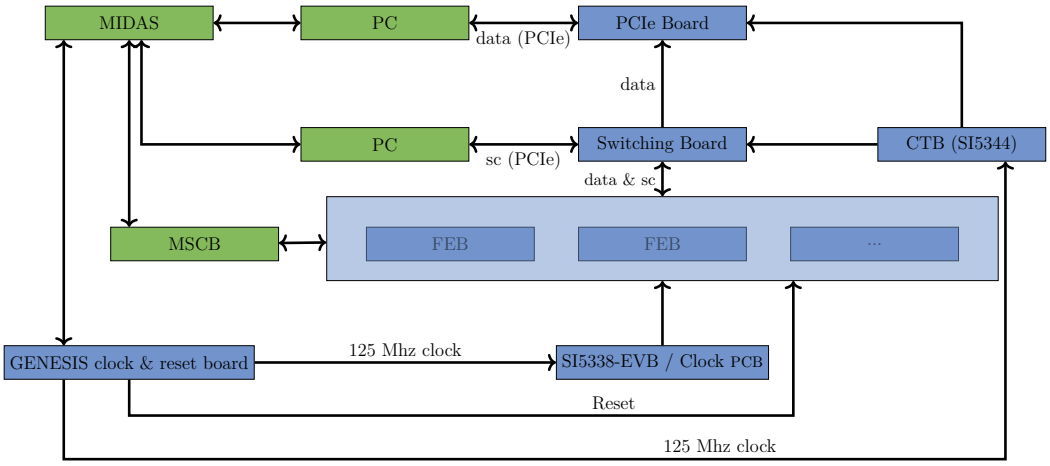


Figure 10.2.: Sketch of the DAQ test setup. The green marked boxes are the stations where the MIDAS front-ends are running. The blue boxes are the actual FPGAs and the clock and reset system.

on. Starting from the clock and reset system on the bottom left, the global 125 MHz clock is distributed to a CTB (cf. [59]) which converts the optical clock to a single ended SubMiniature version A (SMA) connection, since the Terasic DE5a-Net Arria X Development Kit, used for the switching board and PCIe board, has only this kind of clock input. Therefore, the CTB is synchronizing the optical input clock to an internal oscillator which is programmable via Serial Peripheral Interface (SPI). For configuring this chip the RS-422 Expansion Header of the Terasic DE5a-Net Arria X Development Kit was used and a custom SPI entity was developed which is controlled by the NIOS II soft core processor.

Besides this, the reset link is transferred optically to the front-end board. In the setup it is possible to use the actual front-end board prototype or multiple Altera Arria V GX Starter Kits [80]. If one uses the actual front-end board it is possible to connected the different detector PCBs to the setup. On the bottom right in fig. 10.1 the MALIBU board (cf. section 5.5) is connected to it. The reset link can be optically transversed to the boards while the 125 MHz clock needs to be converted to an electrical differential SMA connection. This was done by converting the clock via a Mini Pod to an electrical signal and then synchronizing it to an oscillator by the use of the SI5338 clock chip. On the front-end board this chip was already implemented and could be controlled via the NIOS II soft core processor. For the Altera Arria V GX Starter Kit the SI5338-EVB was used [81]. This board can be programmed via Inter-Integrated Circuit (I2C) (cf. [82]). Since this configuration is only need in the test setup, this was done by a Raspberry Pi 3 B+ board [83], because it is easy to setup and allows remote control via Ethernet. The clock and reset system was also connected via Ethernet which allows the control over MIDAS.

For performing DAQ tests different transceivers need to be implemented on the boards. Since the MALIBU or the MuPix PCB (cf. section 9.1) are connected via a 6 Gbit/s LVDS connection (for sending data) to the front-end board, this kind of data rate was configured on the front-end board transceiver. Furthermore, the same kind of transceiver is used for connecting the switching board to the front-end board. For the transceivers between the switching board

and the PCIe board the data rate should be 10 Gbit/s in the final detector. For the tests a data rate of 6 Gbit/s was taken, because the switching board was only forwarding the data to the PCIe board.

BERTs

Before all tests were performed the links between the boards were tested. For the links between the front-end board, the switching board and the PCIe board a BERT was performed. To this end 70.95 TB of data is sent between the boards and no error was detected which leads to an upper limit of BER of 5.23×10^{-15} bit.

Simulation

All firmware implementations were first simulated by the use of Modelsim [84] and GHDL [85] and then tested in hardware. For visibility reasons not the entire wave diagrams from these programs were converted into vector graphics and only the important parts of them are shown. Further parts of the firmware like some of the Register Transfer Level (RTL) designs can be seen in the appendix.

10.1. Simulation of the slow control entities

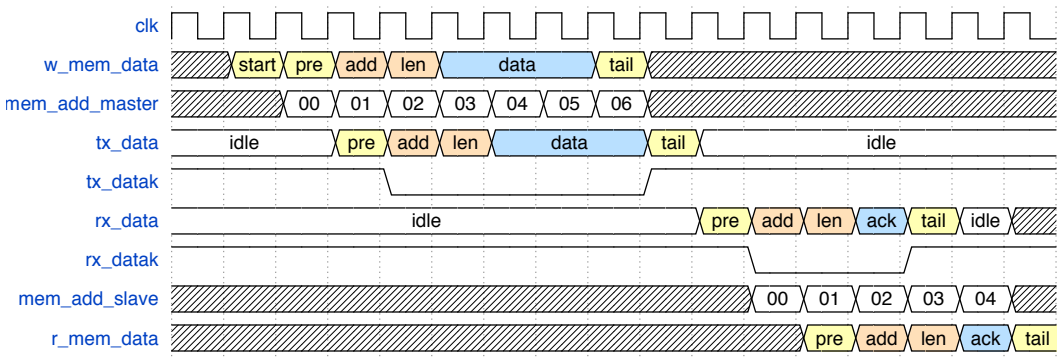


Figure 10.3.: Slow control wave simulation of sending and receiving slow control on the switching board. The entity was simulated with a frequency of 156.25 MHz and the software part of writing or reading data to the PCIe bars was done in VHDL. The colored parts show the different words of the slow control protocol.

For testing the configuration of detector parts from MIDAS via the switching board the PC was writing to the write memory BAR. In fig. 10.3 the simulated behavior (with no delays coming from data transfers) of the firmware is shown in a wave diagram. The data to the write memory is indicated as **w_mem_data** and contains a start condition and the whole slow control package. The start marker triggers the SC Master which reads out the data and sends it to the transceiver. If a comma word needs to be transfered, the **tx_datak** becomes "1" telling the transceiver to use a comma word. On the receiving side the acknowledge from the front-end board is fed into the SC Slave writing it to the read memory. MIDAS on the other

side waits until this acknowledge is inside the memory by polling from it. This behavior was successfully tested.

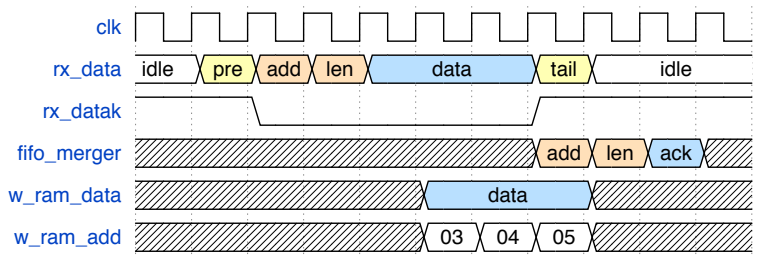


Figure 10.4.: Slow control wave simulation of a write command received on the front-end board. The colored parts show the different words of the slow control protocol. The entity was simulated with a frequency of 156.25 MHz.

Inside the front-end board a write command is received by the SC-Front-End as seen in the wave forms of fig. 10.4. There the link data is written by the SC-Front-End to the RAM starting from the address (**add**) until the total length (**len**) is reached. If all data is stored inside the RAM an acknowledge is generated and sent to a FIFO which is connected to the Data Merger. The Data Merger is reading from this FIFO and assembles the actual slow control package (cf. [68]).

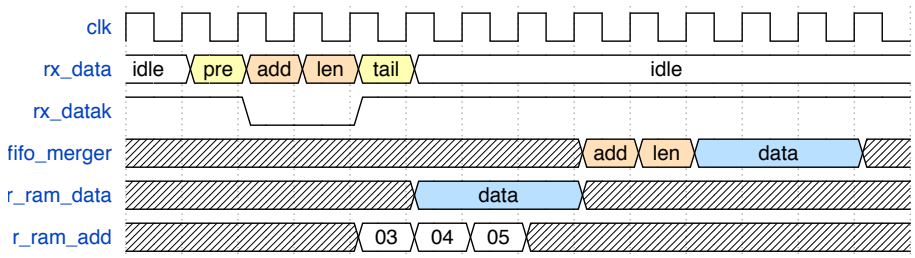


Figure 10.5.: Slow control wave simulation of a read command received on the front-end board. The colored parts show the different words of the slow control protocol. The entity was simulated with a frequency of 156.25 MHz.

The behavior for reading from the RAM is illustrated in fig. 10.5. Here the SC-Front-End only gets the start address and the length. The RAM data is also send to the Data Merger where it gets send back to the switching board. Both reading and writing are simulated successfully.

10.2. Simulation of the data readout entities

In fig. 10.6 the simulation of the event counter entity is given. Here one MuTRiG package is transfered to the entity. The data (**data**) is first buffered into the RAM until the trailer (**tail**) arrived. Then the trailer address is stored inside a FIFO and compared with the last trailer address from the package before. Everything is sent out via the DMA block, adding the length (**len**) of the package to the trailer (**tail-L**). By pulling up the **last_dma** signal the DMA block sends the trailer address of the DMA memory to the PC via PCIe which can be polled by the software. With the information of the trailer address and the length of the package, one can

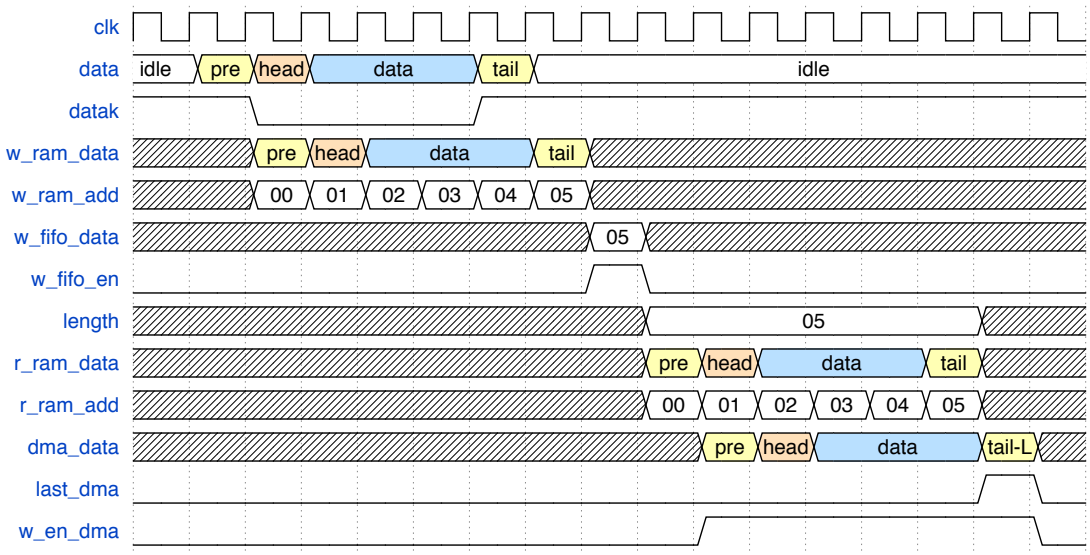


Figure 10.6.: Wave simulation of reading out data on the PCIe board received from the switching board. The colored parts show the different words of the MuTRiG data protocol. The entity was simulated with a frequency of 156.25 MHz.

read out package by package. This is needed since the software is not able to check all address of the DMA memory fast enough (cf. section 9.4).

10.3. Slow control tests

For testing the firmware in hardware the MALIBU and the MuPix PCB where connected to the front-end board as seen in fig. 10.1. The whole chain of things which are needed for configuring the PCBs are shown in fig. 10.7.

After powering up the boards the NIOS II soft core processor on the switching board configures the CTB that it synchronizes the global clock to its clock chip. After this the QSFP transceiver is configured and word alignment is done. The global clock from the CTB is converted into a 156.25 MHz clock on which all entities inside the dotted gray box are running.

On the front-end board the NIOS II soft core processor configures the SI Chip and the global clock gets synchronized and converted into the 156.25 MHz and the desired detector clock. The detector clock is feed out to the PCBs while the other firmware parts are running with the 156.25 MHz. After this the QSFP transceiver is configured and word alignment is done.

For starting the detector configuration the MIDAS software releases the global reset by configuring the clock and reset system. The state is fed into the Data Merger which enables the slow control path. The configuration is transversed according the explained protocol (cf. sec-

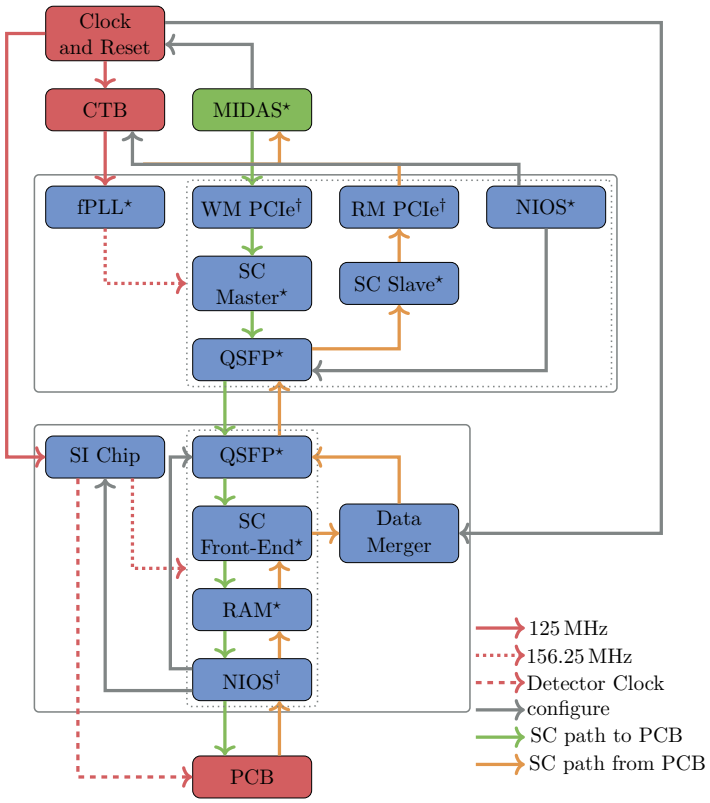


Figure 10.7.: Flow diagram of all needed entities and software parts for configuring the MALIBU board. The blue boxes are the entities inside the FPGAs. The first gray box is the switching board. The second one is the front-end board. Everything inside the dotted gray boxes is running at 156.25 MHz synchronized to the global clock operating at 125 MHz. The red blocks are the clock and reset system and the detector PCB. The green part is the MIDAS system running on the switching PC. Parts marked with * were implemented and designed during the work of this theses. Parts marked with † were adopted and partly changed. The rest was configured to work in the setup.

tion 9.5). After sending the data to the PCIe write memory BAR the MIDAS software polls the PCIe read memory BAR until the acknowledge arrived.

For testing the chain a configuration file which powers up the STiC chip connected to the MALIBU board and one which changes the threshold on the MuPix PCB were sent. For the STiC chip the NIOS II soft core processor on the front-end board is sending the file via SPI to the MALIBU board. For configuring the MuPix PCB the NIOS II soft core processor is replaced with a custom SPI entity designed in [86]. Both tests were successful and show that the whole procedure is working.

Parts of the slow control path are going through the NIOS soft core processor which runs with

a clock speed of 156.25 MHz leading to a limitation of the slow control speed. In fig. 9.4 it is indicated that this can be bypassed if needed. Specially for configuring the MuPix chips a higher speed is needed. Therefor the registers can be directly mapped to the PCB.

10.4. Data readout tests

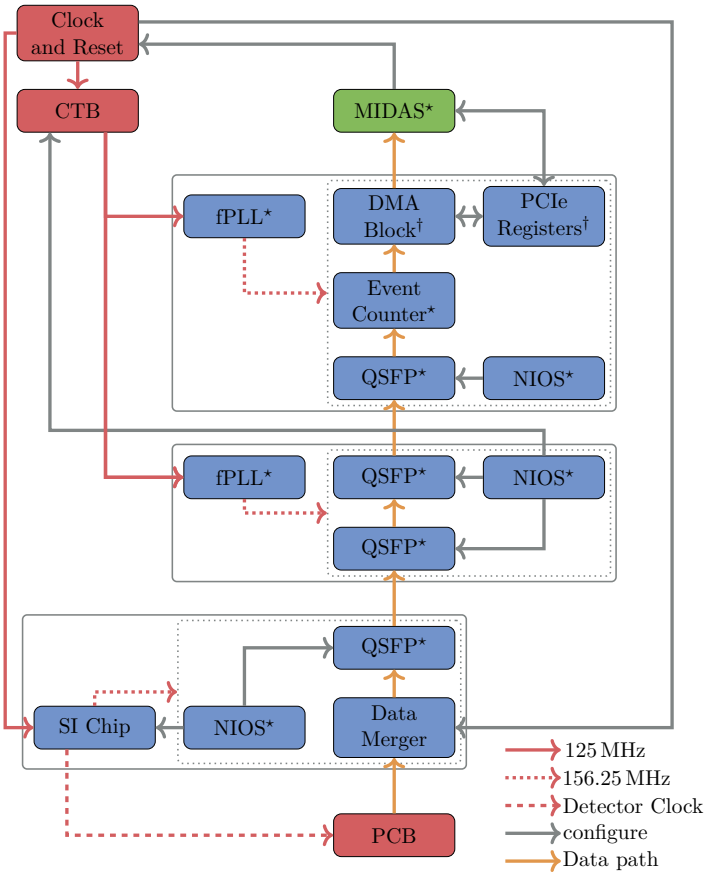


Figure 10.8.: Flow diagram of all needed entities and software parts for reading out data from the MALIBU board. The blue boxes are the entities inside the FPGAs. The gray box at the top is the PCIe board followed by the switching board and the front-end board. Everything inside the dotted gray boxes is running at 156.25 MHz synchronized to the global clock operating at 125 MHz. The red blocks are the clock and reset system and the MALIBU PCB. The green part is the MIDAS system running on the PCIe board PC. Parts marked with * were implemented and designed during the work of this theses. Parts marked with † were adopted and partly changed. The rest was configured to work in the setup.

For confirming if the readout works, data from the MALIBU board and pseudo random data was sent through the whole system. In fig. 10.8 the whole readout chain is shown.

For sending data from the MALIBU board the same configurations as done in section 10.3 need to be performed and the data merger needs to be changed into the running state by the use of the clock and reset system. After this the slow control system is used for configuring the STiC chip that it starts to send test data. The data is transfered throw the switching board to the PCIe board and then readout via DMA. The MIDAS software is able to configure the DMA block via the PCIe register BARs allowing to save the data to a file.

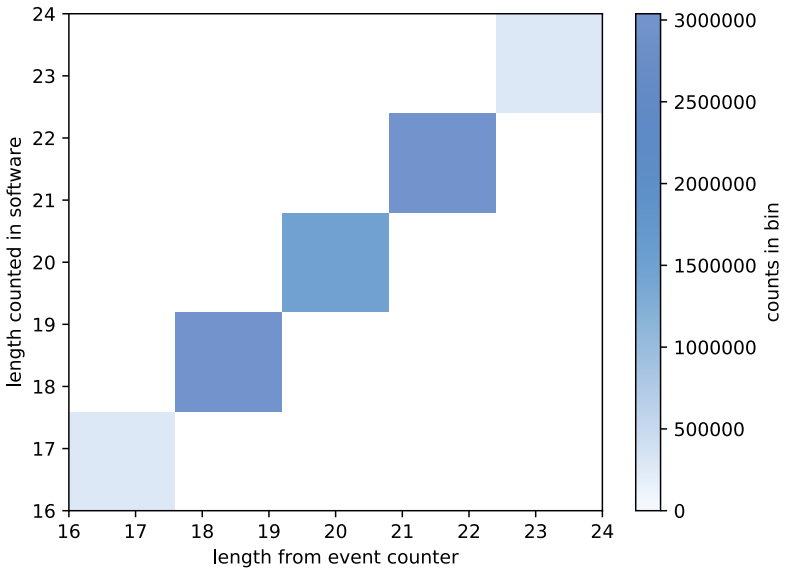


Figure 10.9.: Correlation of the generated event length done on the front-end board and the counted one of the MIDAS software.

By using pseudo random data, which was generated with a linear feedback shift register on the front-end board, the length of the events is known. This leads to the possibility to check, if all data was readout correctly by comparing the received length counted by the event counter with the generated one. In fig. 10.9 it is shown that the length from the generator counted in software and the length from the event counter perfectly correlate. In this test more than 3 million events according to the communication protocol were sent. This and the test with actual detector data indicates, that the event counting, the software implementation of the MIDAS front-end, the DMA block modifications and the other parts of the system works also with data rates which are expected in the Mu3e experiment.

11 Conclusion and Outlook

Testing the Standard Model of particle physics (SM) by performing high rate precision experiments is a promising way to search for indicators of physics beyond the SM. The Mu3e experiment searches for the decay $\mu^+ \rightarrow e^+e^+e^-$ pushing the sensitivity in its first phase down to 1 in 10×10^{15} muon (μ) decays. Since the experiment does not use a hardware trigger, on one side the data acquisition system (DAQ) needs to be able to readout all data and on the other side reduce the enormous rate to a manageably size saving only possible signal events after applying an online tracking algorithm. A lot of the different parts of the DAQ were tested and are fully working. Nevertheless an integration of all the submodules is needed to build the full system.

In this work such an integration on the side of the data flow and the slow control system was done. To this end the firmware for the different field programmable gate arrays (FPGAs) used in the DAQ was designed to read out the used sensors in a full vertical slice representing the whole DAQ system. For proving the functionality of the system, Bit error rate tests (BERTs) on the links were performed, pseudo random data was generated and checked for consistency after being readout, then the readout of the STiC chip was performed and finally also configuring the MuPix8 printed circuit board (PCB) and the MALIBU board was achieved.

By sending 70.95 TB of data over the whole links and detecting no error an upper limit of bit error was set to 5.23×10^{-15} bit. By changing the Direct Memory Access (DMA) block on the PC interface board (PCIe board) by the use of a half-full flag it is guaranteed to provide a deterministic data rate of up to 38 Gbit/s which is sufficient for the Mu3e experiment.

The whole system was only tested in a lab setup. Therefore, additional testing in test beams needs to be performed to guarantee that all functionalities are working properly. Concerning the whole experiment there are multiple steps to be done for finishing the DAQ. For example the time alignment of the data streams on the switching board is not yet implemented into the system. Furthermore the time sorting of the events of the fibres and tiles sensors needs to be designed.

Appendices

A

Acronyms

SM Standard Model of particle physics . . . 1
QFT quantum field theory . . . 3
u up quark . . . 4
d down quark . . . 4
c charm quark . . . 4
s strange quark . . . 4
b bottom quark . . . 4
t top quark . . . 4
e electron . . . 4
γ photon . . . 4
g gluon . . . 4
μ muon . . . v
SMA SubMiniature version A . . . 64
SPI Serial Peripheral Interface . . . 64
I2C Inter-Integrated Circuit . . . 64
τ tau . . . 4
ν_e electron neutrino . . . 4
ν_μ muon neutrino . . . 4
ν_τ tau neutrino . . . 4
SUSY Super Symmetry . . . 6
RMS root mean square . . . 11
MAPS Monolithic Active Pixel Sensors . . . 12
HV-MAPS High-Voltage Monolithic Active Pixel Sensor . . . 12
APS Active Pixel Sensors . . . 12
FPGA field programmable gate array . . . v
BERT Bit error rate test . . . v
BER bit error . . . 27
PHY physical layer . . . 20
NRZ non-return-to-zero . . . 20
LVDS low-voltage differential signaling . . . 21
PCML pseudo current mode logic . . . 21

PLL phase locked loop . . . 21
RD running disparity . . . 22
IP hard intellectual property core . . . 15
CDR clock data recovery . . . 23
PMA Physical Medium Attachment . . . 23
CGB clock generation block . . . 23
PMA Physical Medium Attachment . . . 23
PCS Physical Coding Sublayer . . . 23
fPLL fractional PLL . . . 24
ATX Advanced Transmit PLL . . . 24
SERDES serializer or deserializer . . . 24
CPU Central Processing Unit . . . 25
PCIe Peripheral Component Interconnect Express . . . 25
PC personal computer . . . 25
Avalon-MM Avalon Memory Mapped Interface Master . . . 25
MAC Media Access Controller . . . 51
TX Transceiver . . . 26
FIFO first in first out memory . . . 26
ppm part-per-million . . . 26
LED light-emitting diode . . . 27
CL confidence level . . . 7
DAQ data acquisition system . . . 1
PSI Paul Scherrer Institute . . . 31
ToT Time-over-Threshold . . . 33
ASIC application specific integrated circuit . . . 33
SiPM silicon photomultiplier . . . 33
CSA charge sensitive amplifier . . . 33
DAC digital-to-analog converter . . . 33
MIDAS Maximum Integrated Data Acquisition System . . . vi
PCIe board PC interface board . . . 40
GPU graphics processing unit . . . 40
QSFP quad small-form-factor pluggable ports . . . 40
DDR SDRAM Double Data Rate Synchronous Dynamic Random-Access Memory . . . 40
DMA Direct Memory Access . . . 40
FMC FPGA Mezzanine Card . . . 41
MSCB Midas Slow Control Bus . . . 41
mServer MIDAS server . . . 42
mLogger MIDAS logging system . . . 42
HTTP Hypertext Transfer Protocol . . . 42
FE front-end . . . 42

RTL Register Transfer Level vi
RAM Random-Access Memory 47
SK Super-Kamioka Neutrino Detection Experiment 31
SNO Sudbury Neutrino Observatory 31
KamLand Kamioka Liquid Scintillator Antineutrino Detector 31
PCI-SIG PCIe Special Interest Group 51
HDD hard disk drive 51
SSD Solid-State-Drive 51
TLP Transaction Layer Packet 52
CRC cyclic redundancy check 46
DW double words 52
LSB last significant bit 52
I/O input/output 15
MMIO Memory-mapped input/output 53
BAR Base Address Register 53
MSI Message Signalled Interrupt 55
LFV charged lepton flavor violation 1
BF branching fraction 5
MEG Mu to E Gamma 7
LE Logic Elements 15
LUT Look-Up Table 15
PFD phase frequency detector 21
LF loop filter 21
VCO voltage controlled oscillator 21
SRAM Static Random Access Memory 16
CAD computer-aided design 34
PCB printed circuit board vi
PCI Peripheral Component Interconnect 51
Fmt Format 52
BE Byte Enable 52
LHC Large Hadron Collider 1
CTB clock transmission board 41

B Simulations

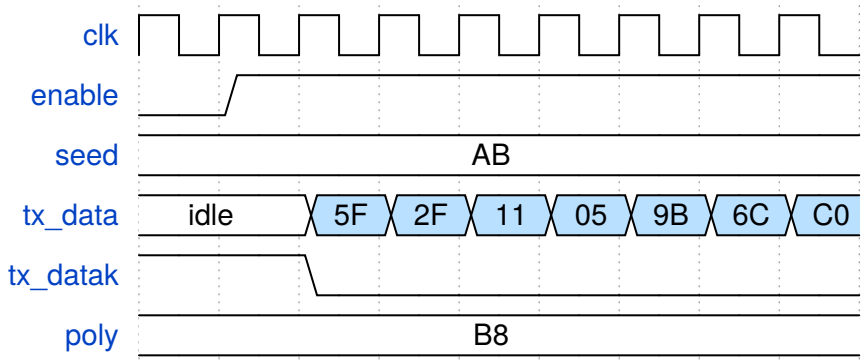


Figure B.1.: Linear shift register wave simulation.

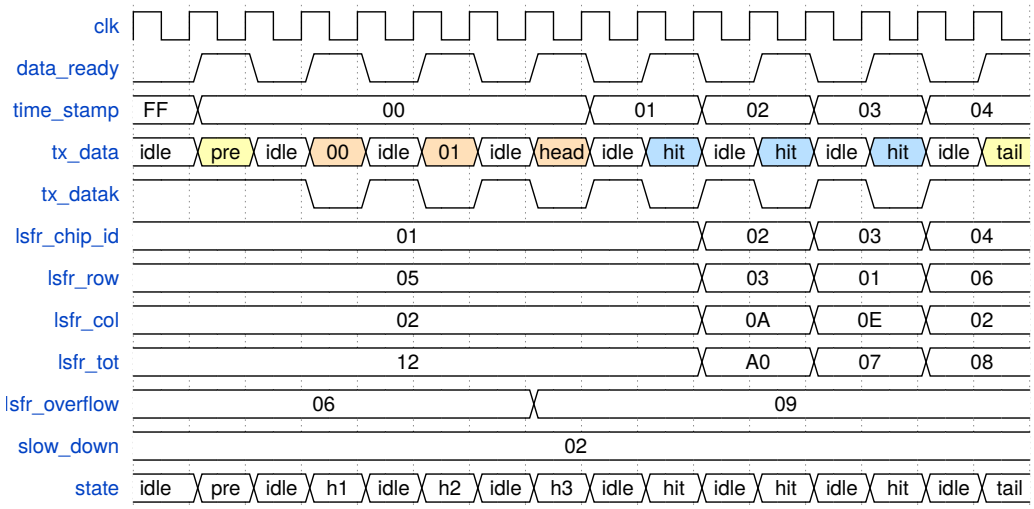


Figure B.2.: Data generator wave simulation.

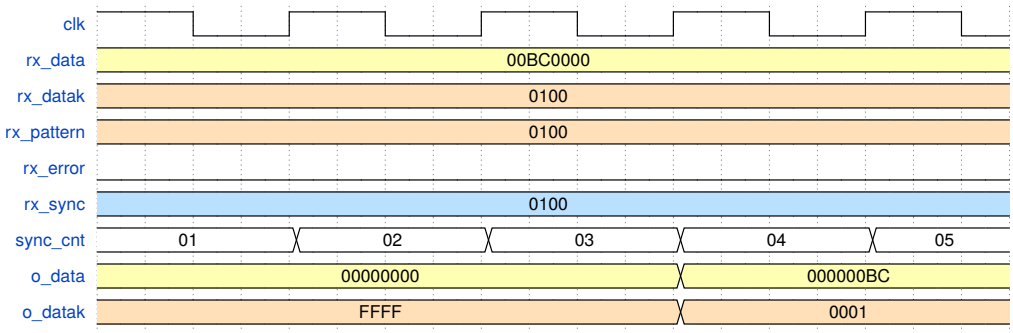


Figure B.3.: Word alignment wave simulation. For 32 bit input data this entity aligns the K-symbol to the least significant byte in the word. This is done if the following requirements are met: no error (rx_error) seen by the transceiver IP, rx_datak and rx_pattern are equal and synchronization of the transceiver for 3 clock cycles.

C

RTL Designs

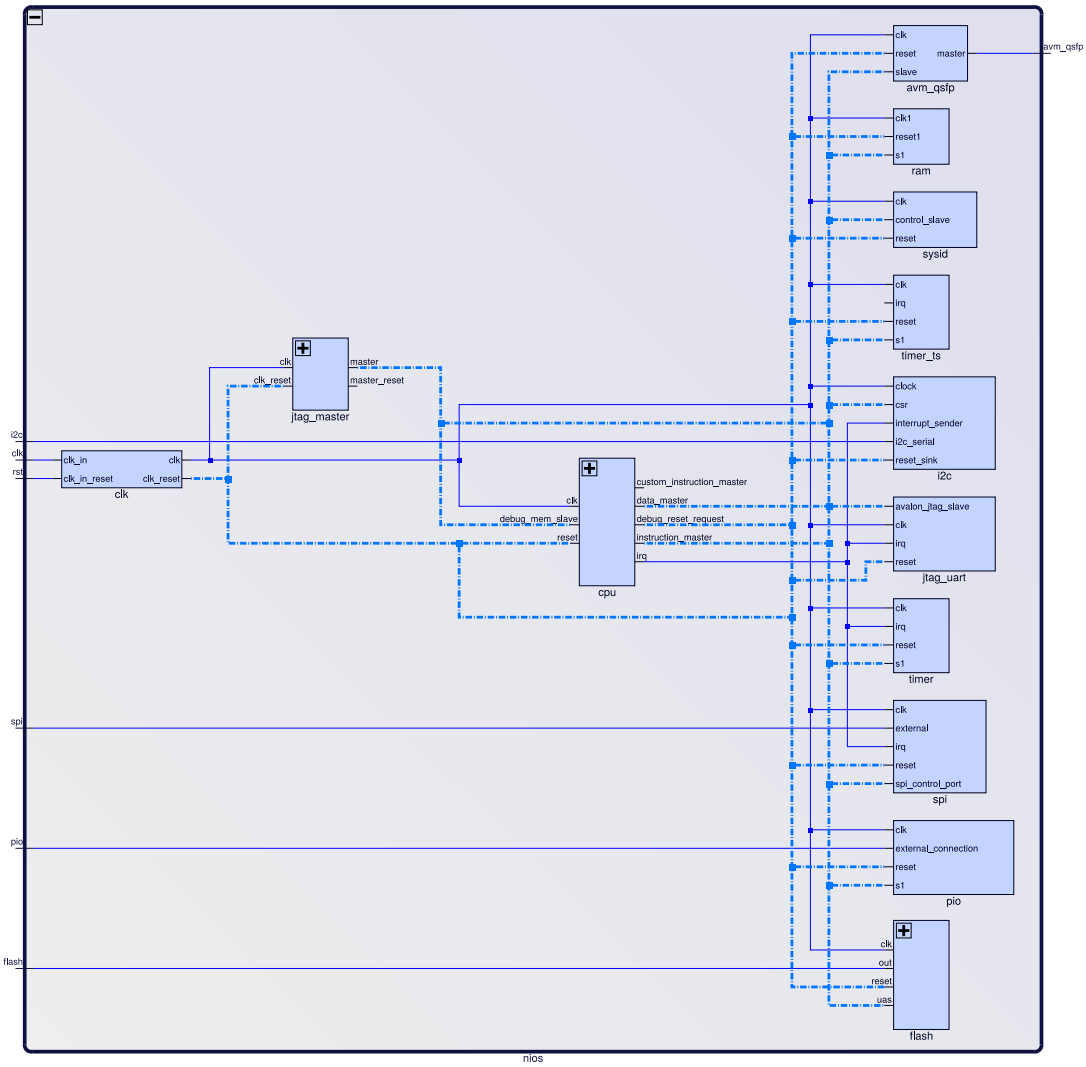


Figure C.1.: Overview of the NIOS II RTL design used inside the FPGA boards, generated with the Quartus software.

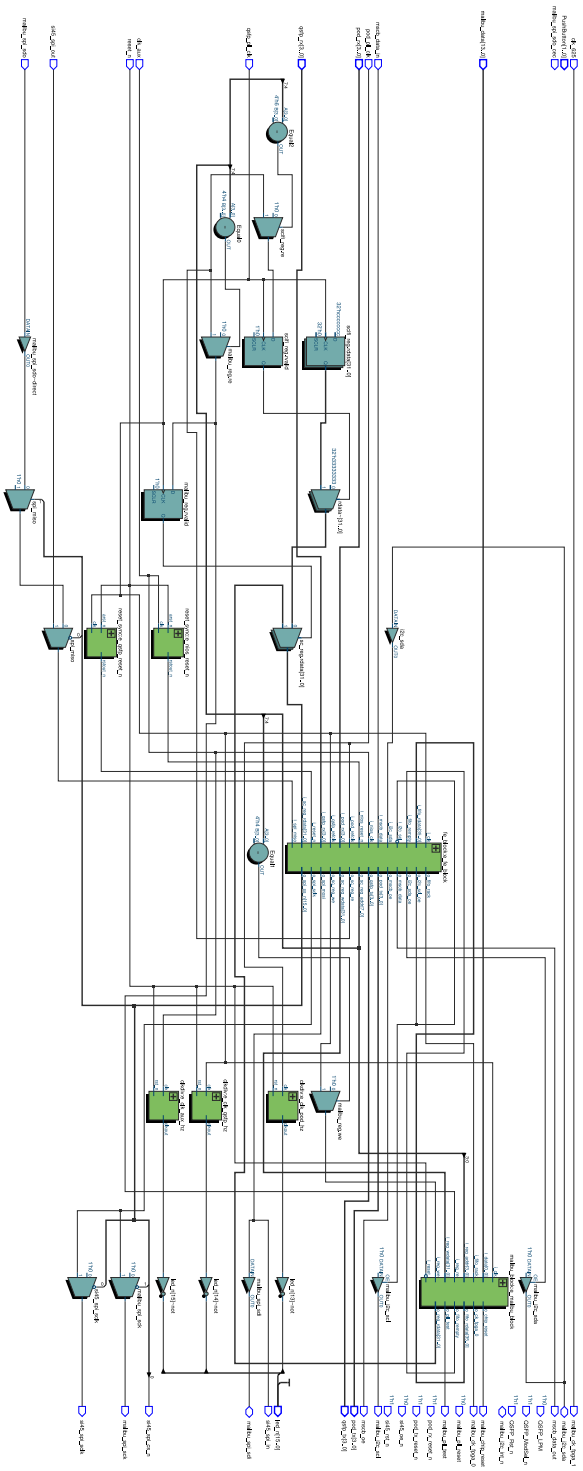


Figure C.2.: Overview of the front-end board RTL design, generated with the Quartus software.

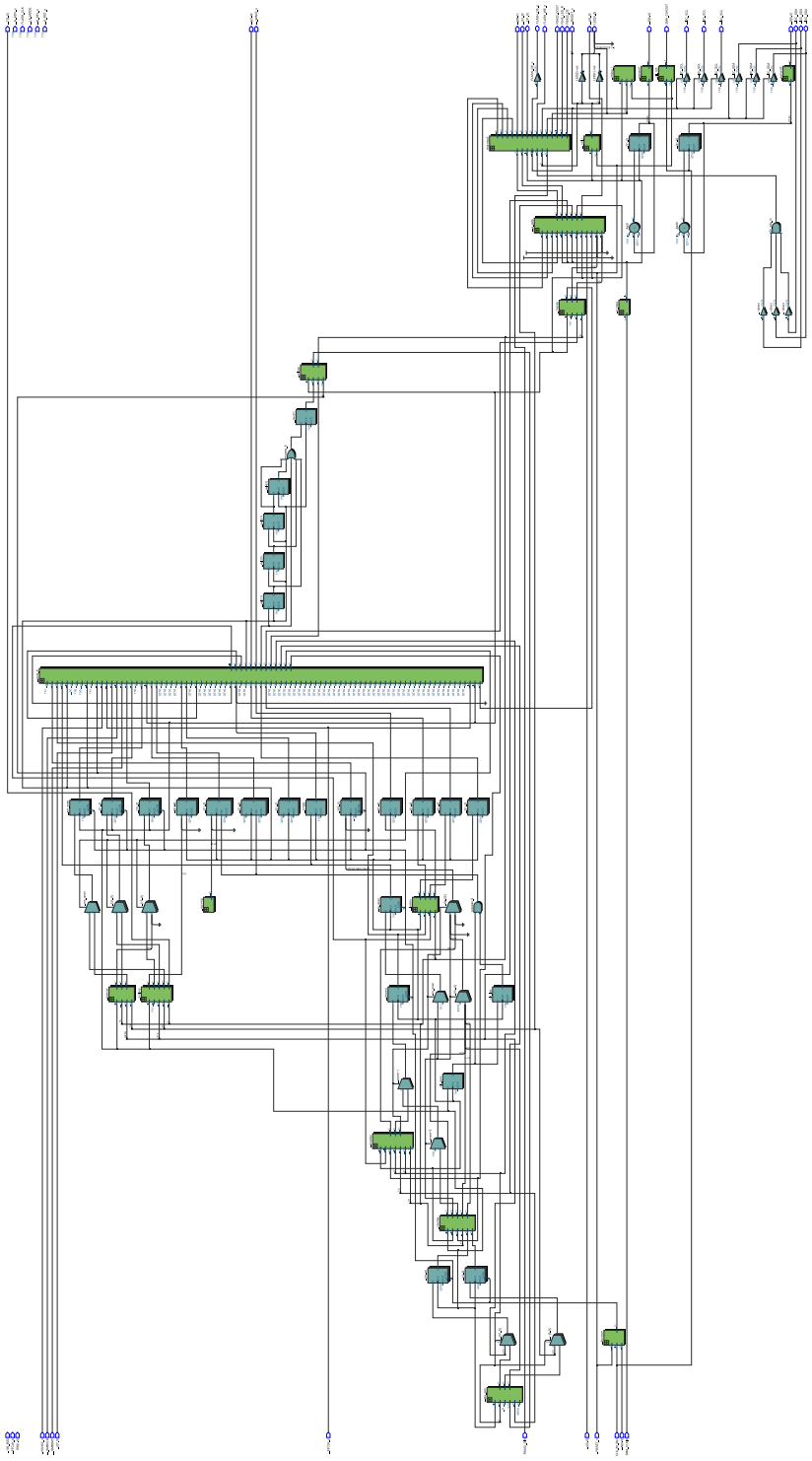


Figure C.3.: Overview of the Switching board RTL design, generated with the Quartus software.

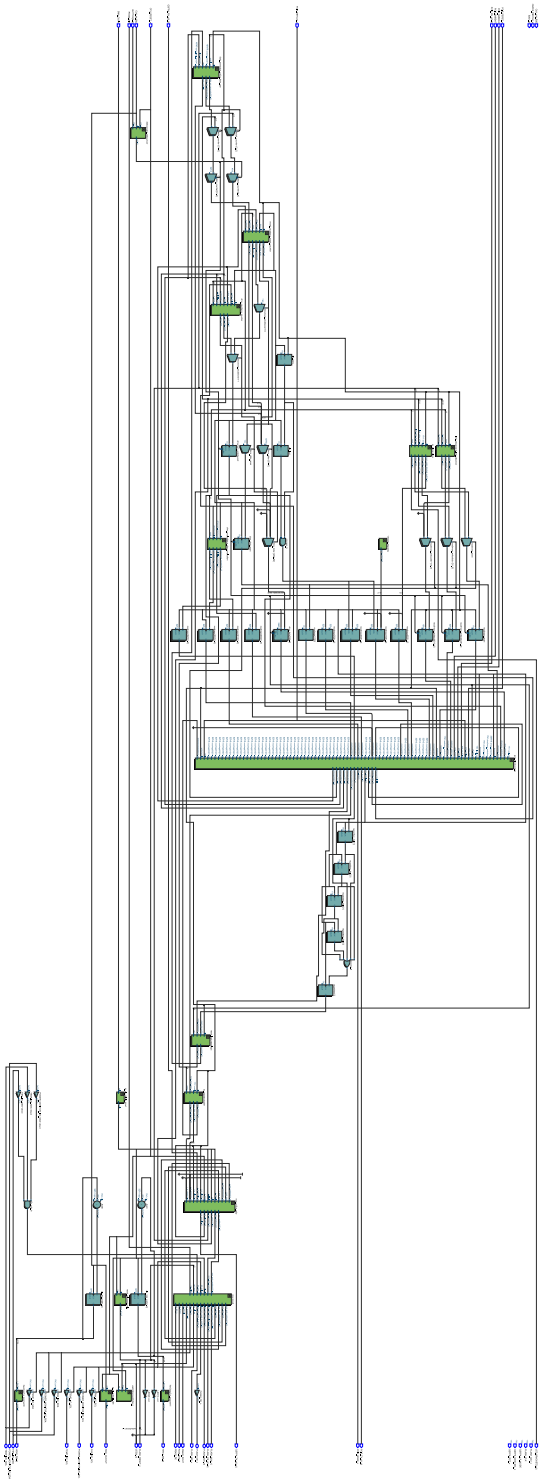


Figure C.4.: Overview of the PCIe board RTL design, generated with the Quartus software.

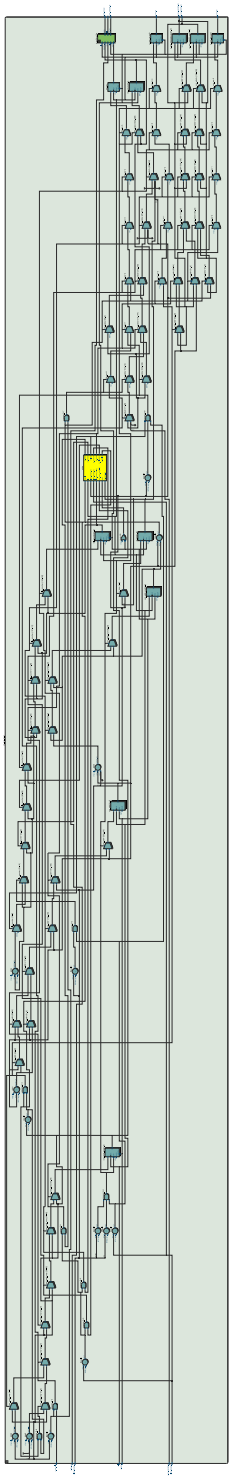


Figure C.5.: Overview of the SC-Front-End design, generated with the Quartus software.

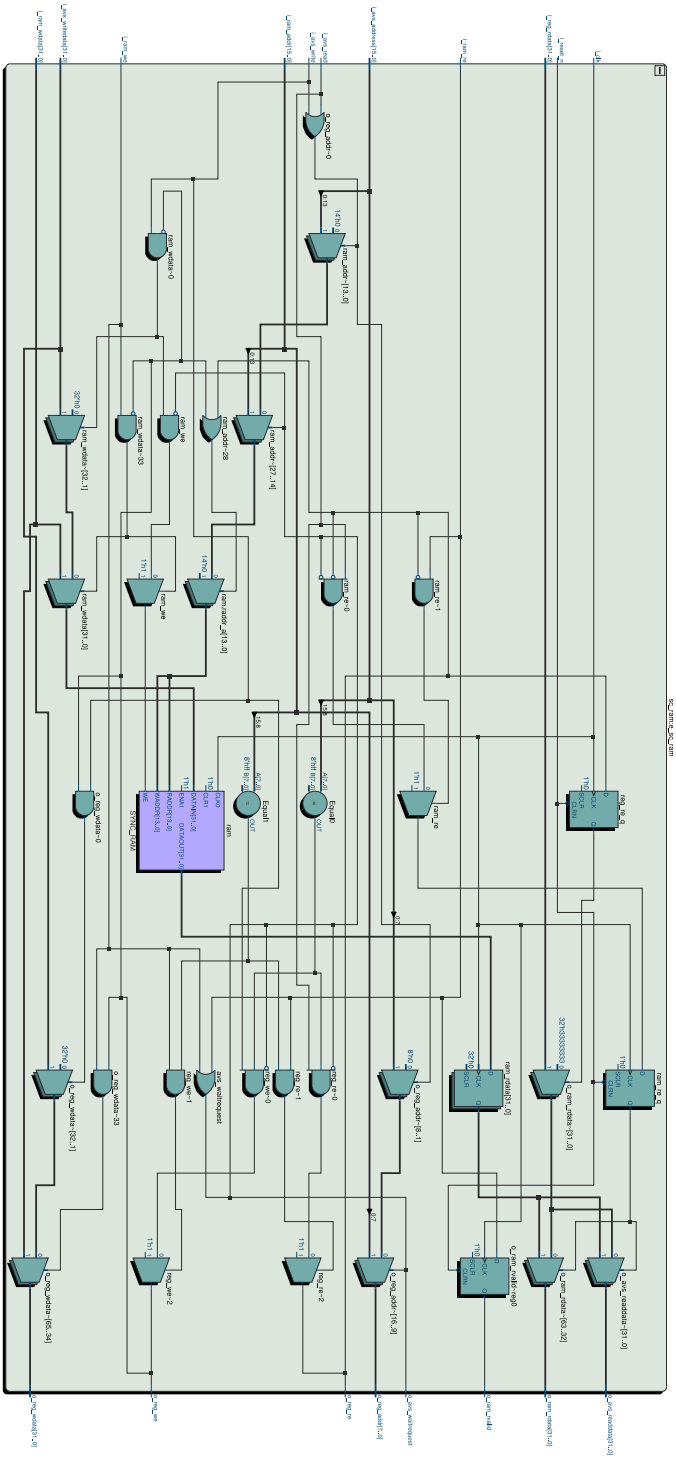


Figure C.6.: RTL design of slow control RAM implemented on the front-end board, generated with the Quartus software.

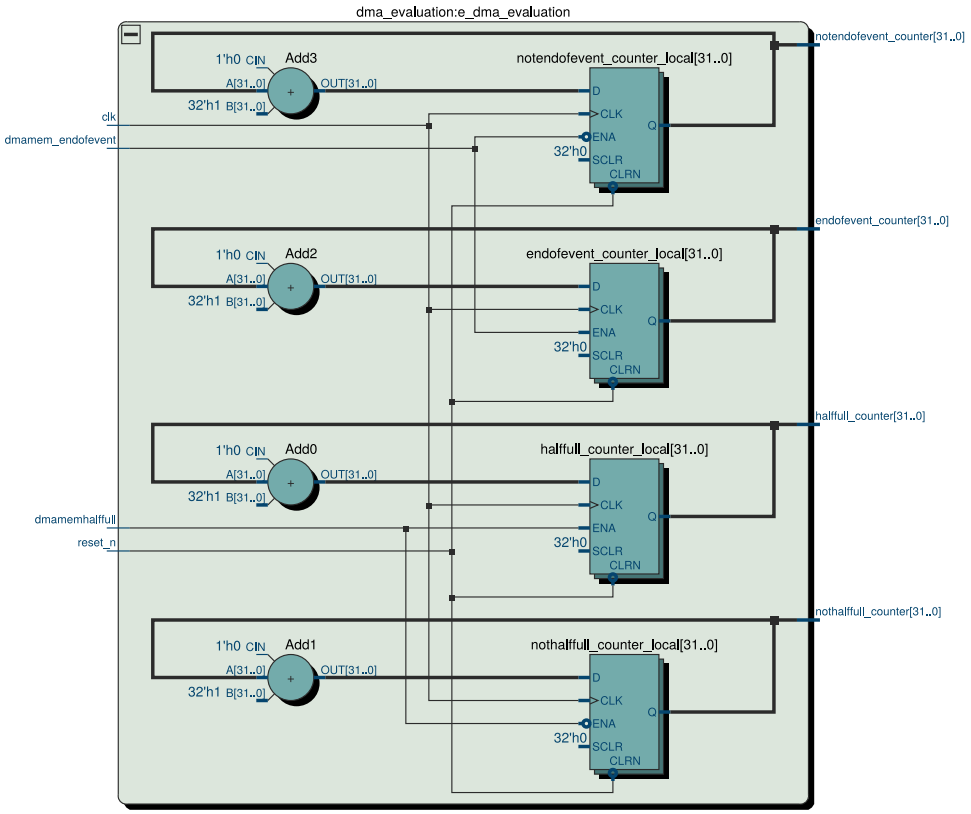


Figure C.7.: RTL design of the DMA evaluation entity used for performing the DMA tests, generated with the Quartus software.

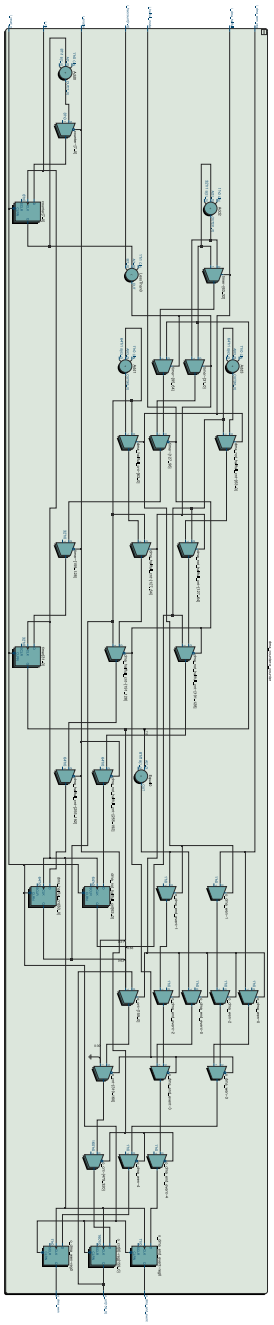


Figure C.8.: RTL design of the DMA counter entity used for performing the DMA tests, generated with the Quartus software.

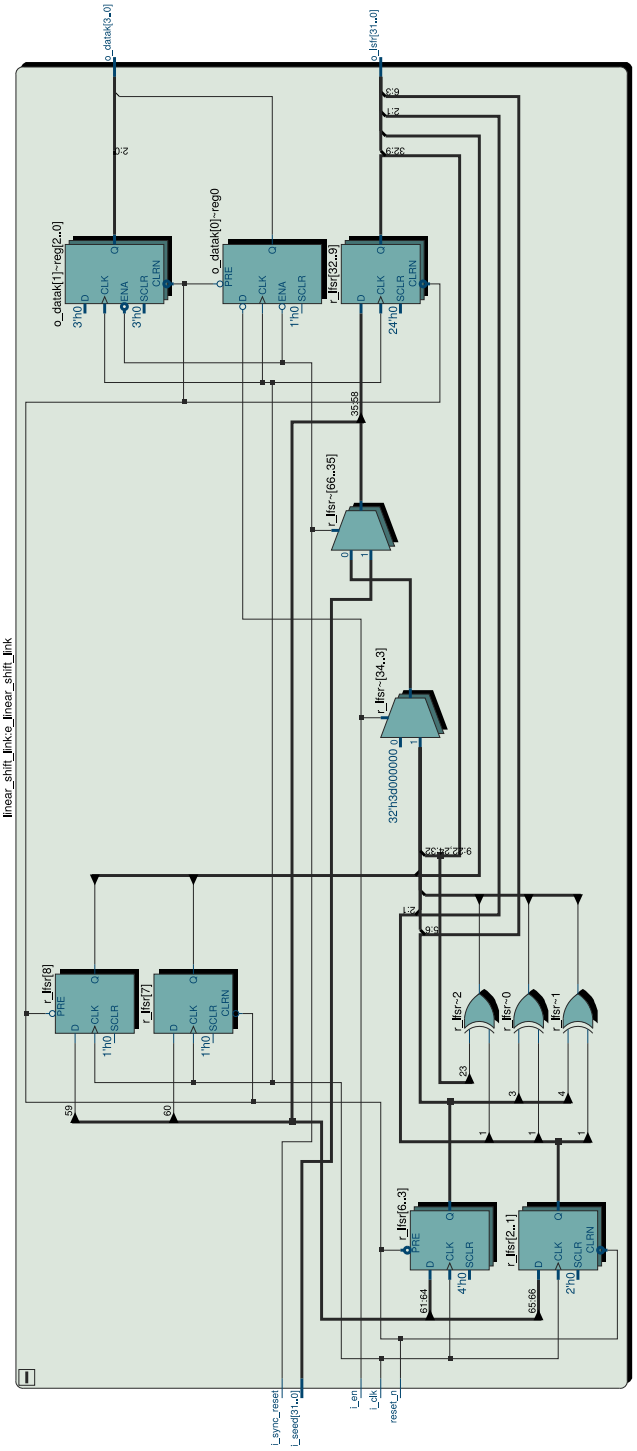


Figure C.9.: RTL design of the linear shift register used for generating pseudo random data, generated with the Quartus software.

List of Figures

1.1. Overview of the SM. . . . 3
1.2. Michel decay. . . . 5
1.3. μ decay with neutrino oscillation. . . . 6
1.4. μ decay via SUSY particles. . . . 6
1.5. Production channels of $\mu^+ \to e^- e^+ e^+$ in the Scotogenic Model. . . . 7

2.1. Particle travelling through a silicon detector. . . . 9
2.2. Sketch of a particle traveling through a tracking detector. . . . 10
2.3. Particle passing through matter. . . . 11
2.4. Sketch of a HV-MAPS sensor. . . . 12
2.5. Particle travelling inside fibre scintillator. . . . 13

3.1. Structure of a FPGA. . . . 15
3.2. Illustration of a simple LE. . . . 16

4.1. Infinite Cascade model. . . . 20
4.2. Illustration of the NRZ code. . . . 20
4.3. Illustration of the principle of a PLL. . . . 21
4.4. Overview of the transceiver bank architecture. . . . 23
4.5. Overview of a transceiver channel. . . . 24
4.6. Building blocks of an Intel Arria 10 Transceiver. . . . 25
4.7. Transceiver channel datapath and clocking. . . . 26

5.1. History of searches for charged lepton flavour violation. . . . 31
5.2. Overview of the signal process and the main background processes. . . . 32
5.3. Sketch of the Mu3e detector concept. . . . 33
5.4. Pixel electronics of the MuPix8 chip. . . . 34
5.5. CAD rendering of the fibre detector. . . . 34
5.6. Sketch of the tile detector. . . . 35

6.1. Sketch of the Mu3e readout system. . . . 38
6.2. Front-end board prototype with the Stratix IV FPGA. . . . 38
6.3. Picture of the LHCb PCIe40 board. . . . 39
6.4. Picture of the clock and reset system. . . . 41
6.5. MIDAS scheme of the Mu3e experiment. . . . 42

8.1. Firmware block for data readout via DMA. . . . 54
8.2. Ring buffer inside the DMA block. . . . 55
8.3. Data rate scans of the DMA block. . . . 56

9.1. Communication between the front-end board and the switching board. 58
9.2. Communication between the switching board and the PCIe board. 59
9.3. Slow control communication. 60
9.4. Writing slow control from MIDAS. 61
9.5. MIDAS front-end panel. 62

10.1. Photo of the DAQ test setup. 63
10.2. Sketch of the DAQ test setup. 64
10.3. Slow control wave simulation 65
10.4. Slow control write command wave simulation. 66
10.5. Slow control read command wave simulation. 66
10.6. Data readout simulation on the PCIe board. 67
10.7. Flow diagram for sending slow control commands. 68
10.8. Flow diagram for reading out data. 69
10.9. Correlation of the generated and counted event length. 70

B.1. Linear shift register wave simulation. 79
B.2. Data generator wave simulation. 79
B.3. Simulation of the word alginment. 80

C.1. Overview of the NIOS II RTL design. 81
C.2. Overview of the front-end board RTL design. 82
C.3. Overview of the Switching board RTL design. 83
C.4. Overview of the PCIe board RTL design. 84
C.5. Overview of the SC-Front-End design. 85
C.6. RTL design of slow control RAM. 86
C.7. RTL design of the DMA evaluation entity. 87
C.8. RTL design of the DMA counter entity. 88
C.9. RTL design of the linear shift register. 89

List of Tables

4.1. Choice of disparity. 21
4.2. Comma words in 8b/10b data stream. 22
4.3. Supported PCS types. 24

7.1. Structure of the MuPix Data. 46
7.2. Structure of the MuTrig data packet. 46
7.3. Structure of the Slow Control protocol. 47
7.4. Structure of run control signals. 47
7.5. Reset link protocol. 48

8.1. PCIe link performance. 51
8.2. PCIe write request. 52

Bibliography

[1] Georges Aad et al., *"Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC"*, Phys. Lett., **B716** 1–29, 2012, (arXiv:1207.7214 [hep-ex]).

[2] Serguei Chatrchyan et al., *"Observation of a New Boson at a Mass of 125 GeV with the CMS Experiment at the LHC"*, Phys. Lett., **B716** 30–61, 2012, (arXiv:1207.7235 [hep-ex]).

[3] Lyndon Evans and Philip Bryant, *"LHC Machine"*, Journal of Instrumentation, **3**(08) S08001–S08001, 2008.

[4] Y. Fukuda et al., *"Evidence for oscillation of atmospheric neutrinos"*, Phys. Rev. Lett., **81** 1562–1567, 1998, (arXiv:hep-ex/9807003).

[5] S. N. Ahmed et al., *"Measurement of the total active B-8 solar neutrino flux at the Sudbury Neutrino Observatory with enhanced neutral current sensitivity"*, Phys. Rev. Lett., **92** 181301, 2004, (arXiv:nucl-ex/0309004).

[6] K. Eguchi et al., *"First results from KamLAND: Evidence for reactor anti-neutrino disappearance"*, Phys. Rev. Lett., **90** 021802, 2003, (arXiv:hep-ex/0212021).

[7] David J Griffiths, *Introduction to elementary particles; 2nd rev. version*, Physics textbook. Wiley, New York, NY, 2008.

[8] David Galbraith and Carsten Burgard, Standard Model of the Standard Model, http://davidgalbraith.org/portfolio/ux-standard-model-of-the-standard-model/, [Online; accessed 23-September-2019].

[9] M. Tanabashi et al., *"Review of Particle Physics"*, Phys. Rev. D, **98** 030001, Aug 2018.

[10] S. L. Glashow, *"Partial Symmetries of Weak Interactions"*, Nucl. Phys., **22** 579–588, 1961.

[11] Peter W. Higgs, *"Broken Symmetries and the Masses of Gauge Bosons"*, Phys. Rev. Lett., **13** 508–509, 1964, [,160(1964)].

[12] F. Zwicky, *"Die Rotverschiebung von extragalaktischen Nebeln"*, Helv. Phys. Acta, **6** 110–127, 1933, [Gen. Rel. Grav.41,207(2009)].

[13] T. Kajita, E. Kearns and M. Shiozawa, *"Establishing atmospheric neutrino oscillations with Super-Kamiokande"*, Nucl. Phys., **B908** 14–29, 2016.

[14] R. R. Crittenden, W. D. Walker and J. Ballam, *"Radiative Decay Modes of the Muon"*, Phys. Rev., **121** 1823–1832, Mar 1961.

[15] G. Hernández-Tomé, G. López Castro and P. Roig, *"Flavor violating leptonic decays of τ and μ leptons in the Standard Model with massive neutrinos"*, The European Physical Journal C, **79**(1) 84, Jan 2019.

[16] Lorenzo Calibbi and Giovanni Signorelli, *"Charged Lepton Flavour Violation: An Experimental and Theoretical Introduction"*, Riv. Nuovo Cim., **41**(2) 71–174, 2018, (arXiv:1709.00294 [hep-ph]).

[17] J. Bernabeu, E. Nardi and D. Tommasini, *"μ - e conversion in nuclei and Z′ physics"*, Nucl. Phys., **B409** 69–86, 1993, (arXiv:hep-ph/9306251).

[18] Takashi Toma and Avelino Vicente, *"Lepton Flavor Violation in the Scotogenic Model"*, JHEP, **01** 160, 2014, (arXiv:1312.2840 [hep-ph]).

[19] Ernest Ma, *"Verifiable radiative seesaw mechanism of neutrino mass and dark matter"*, Phys. Rev., **D73** 077301, 2006, (arXiv:hep-ph/0601225).

[20] Steven Weinberg, *"Baryon and Lepton Nonconserving Processes"*, Phys. Rev. Lett., **43** 1566–1570, 1979.

[21] Toshinori Mori, *"Final Results of the MEG Experiment"*, Nuovo Cim., **C39**(4) 325, 2017, (arXiv:1606.08168 [hep-ex]).

[22] U. Bellgardt et al., *"Search for the decay $\mu^+ \to e^- e^+ e^+$ "*, Nuclear Physics B, **299**(1) 1 – 6, 1988.

[23] P. Wintz, *"Results of the SINDRUM-II experiment"*, Conf. Proc., **C980420** 534–546, 1998.

[24] Mark Thomson, *Modern particle physics*. Cambridge University Press, New York, 2013.

[25] Glenn F Knoll, *Radiation detection and measurement; 4th ed.* Wiley, New York, NY, 2010.

[26] Ivan Peric, *"A novel monolithic pixelated particle detector implemented in high-voltage CMOS technology"*, Nucl. Instrum. Meth., **A582** 876–885, 2007.

[27] Intel Corporation, Intel Quartus Prime Standard Edition User Guide, https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug-qps-getting-started.pdf.

[28] A. P. Clark, *Principles of Digital Data Transmission*. Halsted Press, New York, NY, USA, 1976.

[29] Sophocles J Orfanidis, Electromagnetic waves and antennas, http://eceweb1.rutgers.edu/~orfanidi/ewa/, 2016, [Online; accessed 23-September-2019].

[30] John David Jackson, *Classical electrodynamics*. Wiley, New York, NY, 3rd ed. edition, 1999.

[31] Wolfgang Demtröder, Experimentalphysik 2: Elektrizität und Optik, ISBN: 978-3-540-20210-3, 01 2004.

[32] Howard Johnson and Martin Graham, *High-speed Signal Propagation: Advanced Black Magic*. Prentice Hall Press, Upper Saddle River, NJ, USA, first edition, 2003.

[33] National Instruments, High-speed serial explained, https://www.ni.com/de-de/innovations/white-papers/15/high-speed-serial-explained.html, 2016, [Online; accessed 23-September-2019].

[34] Howard W. Johnson and Martin Graham, *High-speed Digital Design: A Handbook of Black Magic*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[35] Wikipedia contributors, 8b/10b encoding, https://en.wikipedia.org/w/index.php?title=8b/10b_encoding&oldid=910321009, 2019, [Online; accessed 15-August-2019].

[36] A. X. Widmer and P. A. Franaszek, *"A DC-balanced, Partitioned-block, 8B/10B Transmission Code"*, IBM J. Res. Dev., **27**(5) 440–451, 1983.

[37] Intel Corporation, Intel Arria 10 Transceiver PHY User Guide, `https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/arria-10/ug_arria10_xcvr_phy.pdf`, [Online; accessed 23-September-2019].

[38] PCI-SIG, PCI Express Base Specification Revision 3.0, `https://pcisig.com/specifications`, 2010, [Online; accessed 23-September-2019].

[39] National Instruments, Guide to fiber optics and premises cabling, `https://www.thefoa.org/tech/ref/appln/transceiver.html`, 2012, [Online; accessed 23-September-2019].

[40] Ilya V. Narsky, *"Estimation of upper limits using a Poisson statistic"*, Nucl. Instrum. Meth., **A450** 444–455, 2000, (`arXiv:hep-ex/9904025`).

[41] William J. Marciano et al., *"Charged Lepton Flavor Violation Experiments"*, Annual Review of Nuclear and Particle Science, **58**(1) 315–341, 2008.

[42] Mu3e collaboration, Technical design of the Phase I Mu3e Experiment, 2019, [Ongoing work; accessed 15-August-2019].

[43] Mu3e collaboration, Mu3e Technical Design Report, 2017.

[44] Heiko Augustin et al., *"The MuPix System-on-Chip for the Mu3e Experiment"*, Nucl. Instrum. Meth., **A845** 194–198, 2017, (`arXiv:1603.08751 [physics.ins-det]`).

[45] Hans Patrick Eckert, *The Mu3e Tile Detector*, PhD thesis, Ruprecht-Karls-Universität Heidelberg, 07 2015.

[46] H. Chen, K. Briggl, P. Eckert, T. Harion, Y. Munwes, W. Shen, V. Stankova and H.C. Schultz-Coulon, *"MuTRiG: a mixed signal Silicon Photomultiplier readout ASIC with high timing resolution and gigabit data link"*, Journal of Instrumentation, **12**(01) C01043–C01043, jan 2017.

[47] W. Shen et al., *"A Silicon Photomultiplier Readout ASIC for Time-of-Flight Applications Using a New Time-of-Recovery Method"*, IEEE Transactions on Nuclear Science, **65**(5), May 2018.

[48] H. Augustin et al., *"Performance of the large scale HV-CMOS pixel sensor MuPix8"*, 2019, (`arXiv:1905.09309 [physics.ins-det]`).

[49] S. Bravar et al., *"Scintillating fibre detector for the Mu3e experiment"*, Journal of Instrumentation, **12**(07) C07011–C07011, jul 2017.

[50] Intel Corporation, Stratix IV Device Handbook, `https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/stratix-iv/stratix4_handbook.pdf`, [Online; accessed 23-September-2019].

[51] Ann-Kathrin Perrevoort, *Sensitivity Studies on New Physics in the Mu3e Experiment and Development of Firmware for the Front-End of the Mu3e Pixel Detector*, PhD thesis, Ruprecht-Karls-Universität Heidelberg, 01 2018.

[52] LHCb collaboration, The LHCb PCIe Readout, `https://indico.cern.ch/event/468486/contributions/1144380/attachments/1241282/1825481/LHCb_PCIe_Readout.pdf`, 2016, [Online; accessed 23-September-2019].

[53] P. Durante et al., *"100 Gbps PCI-Express readout for the LHCb upgrade"*, Journal of Instrumentation, **10**(04) C04018–C04018, apr 2015.

[54] Broadcom, MiniPOD™12x10G Transmitter Module, `https://www.broadcom.com/products/fiber-optics/embedded-optical-module/minipod-embedded-optical-modules/afbr-811vxyz`, [Online; accessed 21-September-2019].

[55] Terasic Technologies Inc., DE5a-Net FPGA Development Kit User Manual, https://www.terasic.com.tw/cgi-bin/page/archive_download.pl?Language=English&No=970&FID=0bc2c05d074b8a05252d9b8e363d69d1, [Online; accessed 21-September-2019].

[56] Dorothea vom Bruch, *Pixel Sensor Evaluation and Online Event Selection for the Mu3e Experiment*, PhD thesis, Ruprecht-Karls-Universität Heidelberg, 11 2011.

[57] DIGILENT, Genesys 2 FPGA Board Reference Manual, https://reference.digilentinc.com/_media/reference/programmable-logic/genesys-2/genesys2_rm.pdf, [Online; accessed 21-September-2019].

[58] Samtec, FireFly, https://www.samtec.com/de/products/ecuo, [Online; accessed 21-September-2019].

[59] Tobias Wagner, Clock Transmission for the Data Acquisition System of the Mu3e Experiment, https://www.psi.ch/sites/default/files/2019-05/BachelorWagner.pdf, 2 2019, [Online; accessed 23-September-2019].

[60] Samer Kilani, Clock and Reset System Specification, 2017, [Mu3e internal note 0043].

[61] Niklaus Berger, Clock and Reset Distribution, 2017, [Mu3e internal note 0041].

[62] Samer Kilani, Mu3e Clock & Reset Distribution, 2019, [Mu3e internal talk 25.07.2019].

[63] Mu3e collaboration, The Mu3e spec book, 2019, [Ongoing work; accessed 14-September-2019].

[64] K. Olchanski et al., MIDAS, https://midas.triumf.ca, [Online; accessed 21-September-2019].

[65] K. Olchanski et al., Midas Slow Control Bus (MSCB), https://elog.psi.ch/mscb, [Online; accessed 21-September-2019].

[66] Intel Corporation, Nios II Processor Reference Guide, https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/nios2/n2cpu-nii5v1gen2.pdf, [Online; accessed 21-September-2019].

[67] C. Ghabrous Larrea, K. Harder, D. Newbold, D. Sankey, A. Rose, A. Thea and T. Williams, *"IPbus: a flexible Ethernet-based control system for xTCA hardware"*, Journal of Instrumentation, **10**(02) C02019–C02019, feb 2015, [Online; accessed 21-September-2019].

[68] Martin Müller, A Control System for the Mu3e DAQ, [Master thesis, in preperation].

[69] Niklaus Berger, Run Start and Reset Protocol, 2018, [Mu3e internal note 0046].

[70] Gavin Hesketh, Clock & Reset Protocols and MIDAS, 2018, [Mu3e internal talk 29.11.2018].

[71] Xillybus Ltd., How PCI express devices talk (Part I), http://xillybus.com/tutorials/pci-express-tlp-pcie-primer-tutorial-guide-1, [Online; accessed 15-August-2019].

[72] PCI-SIG, PCI Express Base Specification Revision 1.0, https://pcisig.com/specifications, 2002, [Online; accessed 23-September-2019].

[73] PCI-SIG, PCI Express Base Specification Revision 2.0, https://pcisig.com/specifications, 2006, [Online; accessed 23-September-2019].

[74] PCI-SIG, PCI Express Base Specification Revision 4.0, Version 1.0, https://pcisig.com/specifications, 2017, [Online; accessed 23-September-2019].

[75] Wikipedia contributors, PCI Express, https://en.wikipedia.org/w/index.php?title=PCI_Express&oldid=912780692, 2019, [Online; accessed 1-September-2019].

[76] PCIe Special Interest Group, PCIe Special Interest Group, https://pcisig.com/, [Online; accessed 23-September-2019].

[77] Intel Corporation, Intel Arria 10 and Intel Cyclone 10 GX Avalon-MM Interface for PCI Express User Guide, https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug_a10_pcie_avmm.pdf, [Online; accessed 23-September-2019].

[78] Intel Corporation, Stratix V Device Handbook, https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/stratix-v/stratix5_handbook.pdf, [Online; accessed 23-September-2019].

[79] Dirk Wiedner, The Mupix8 PCB, 2017, [Mu3e internal note 0042].

[80] Intel Corporation, Arria V GX Starter Kit User Guide, https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug_avgx_starter_dev_kit.pdf.

[81] Silicon Labs, Si5338-EVB, https://www.silabs.com/products/development-tools/timing/clock/si5338-development-kit, [Online; accessed 21-September-2019].

[82] Philips Semiconductors, I2C, https://www.nxp.com/docs/en/application-note/AN10216.pdf, [Online; accessed 23-September-2019].

[83] Raspberry Pi Foundation, Raspberry Pi Model B+, https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/, [Online; accessed 21-September-2019].

[84] Intel Corporation, ModelSim-Intel FPGA Edition Software, https://www.intel.de/content/www/de/de/software/programmable/quartus-prime/model-sim.html, [Online; accessed 23-September-2019].

[85] Tristan Gingold, GHDL, http://ghdl.free.fr/, 2017.

[86] Sebastian Dittmeier, *Fast data acquisition for silicon tracking detectors at high rates*, PhD thesis, Ruprecht-Karls-Universität Heidelberg, 11 2018.

Acknowledgement

I would like to thank my supervisor Prof. Dr. Niklaus Berger for giving me the chance to write my thesis in his group, for supporting me with helpful comments and for thoroughly reading my work. I am particularly thankful for the great collaboration with Dr. Alexandr Kozlinskiy and Martin Müller helping me to get the head around the challenging hardware development process. I would also like to thank all the members of the AG Berger, who contributed to the great atmosphere that allowed me to accomplish a lot over the last year while also having a lot of fun.

For helping me with my work on the slow control system for the MALIBU and MuPix PCB I would like to thank the members of the Mu3e collaboration and in particular Dr. Yonathan Munwes, Tiancheng Zhong, Dr. Sebastian Dittmeier and Dr. Konrad Briggl for explaining me the detector concepts.

Finally, I would like to thank my parents and my family who always supported me during my studies and Pia for her support during the time of writting this work.