Department of Physics and Astronomy

University of Heidelberg

Master thesis

in Physics

submitted by

Jens Kröger

born in Lübbecke

2017

Readout Hardware for the MuPix8 Pixel Sensor Prototype and a Firmware-based MuPix8 Emulator

This Master thesis has been carried out by

Jens Kröger

at the

Physikalisches Institut Heidelberg

under the supervision of

Prof. Dr. André Schöning

Readout Hardware for the MuPix8 Pixel Sensor Prototype and a Firmware-based MuPix8 Emulator

The Mu3e experiment is dedicated to search for the lepton flavour violating process $\mu^+ \rightarrow e^+e^-e^+$ with an unprecedented precision of 1 in 10^{16} muon decays. To reach this sensitivity, an excellent momentum resolution as well as a very precise vertex reconstruction are crucial.
The Mu3e detector will consist of an ultra-thin pixel tracker in combination with scintillating tiles and fibres for an improved timing resolution. The tracking detector will be equipped with novel High-Voltage Monolithic Active Pixel Sensors which are thinned to 50 µm. The first large-scale prototype of the MuPix family is the MuPix8. It is currently under characterisation.
Within the scope of this thesis, a number of hardware components for the readout chain of MuPix8 have been designed and tested successfully. Furthermore, a firmware-based chip emulator embedded in the readout FPGA has been developed in order to validate the functionality of the readout firmware and software prior to the delivery of the MuPix8. In a second step, the emulator has been migrated onto a separate FPGA such that the hardware of the entire data acquisition chain could be tested.
Besides that, contributions to the commissioning of the chip have been made. These comprise investigations of the analogue amplifier output, the hitbus signal, the data quality as well as the addressing scheme. In addition, a calibration for the on-chip temperature diode has been performed and the power consumption of the chip has been determined.

Auslese-Hardware für den MuPix8 Pixelsensor-Prototypen und ein Firmware-basierter MuPix8 Emulator

Das Mu3e-Experiment wird mit einer bisher unerreichten Genauigkeit von 1 in 10^{16} Myonzerfällen nach dem leptonzahlverletzenden Prozess $\mu^+ \rightarrow e^+e^-e^+$ suchen. Eine exzellente Impulsauflösung sowie eine sehr präzise Vertexrekonstruktion sind entscheidend, um diese Sensitivität zu erreichen.
Der Mu3e-Detektor wird aus einem ultradünnen Pixel-Spurdetektor bestehen, der für bessere Zeitmessungen um szintillierende Fasern und Kacheln ergänzt wird. Der Spurdetektor wird mit neuartigen hochspannungsbetriebenen monolithischen aktiven Pixelsensoren ausgestattet sein, die bis auf 50 µm gedünnt werden. Der erste großflächige Prototyp der MuPix-Familie ist der MuPix8. Er wird derzeit charakterisiert.
Für diese Arbeit wurde eine Reihe von Hardware-Komponenten für die Auslesekette des MuPix8 entworfen und erfolgreich getestet. Außerdem wurde ein Firmware-basierter Chip-Emulator entwickelt, der im Auslese-FPGA integriert ist. Er hat ermöglicht, vor der Lieferung des MuPix8 die Funktionalität der Auslese-Firmware und Software zu testen. In einem zweiten Schritt wurde der Emulator auf einen separaten FPGA migriert, um die Hardware der gesamten Auslesekette testen zu können.
Darüber hinaus wurden Beträge zur Inbetriebnahme des Chips geleistet. Diese umfassen eine Untersuchung des analogen Verstärkerausgangs, des Hitbus-Signals, der Datenqualität sowie des Adressschemas. Zusätzlich wurde eine Kalibration für die Temperaturdiode auf dem MuPix8 durchgeführt und die Leistungsaufnahme des Chips wurde bestimmt.

Contents

1 Introduction

The smallest constituents of matter – the elementary particles – as well as the interactions between them are described by the Standard Model of Particle Physics. To date, it has withstood countless experimental tests with an enormous precision.

However, many questions remain unanswered. The existence of dark matter, for instance, is a compelling indication for physics beyond the Standard Model. Furthermore, the Standard Model does not explain the existence of exactly three generations of particles or the observed matter-antimatter asymmetry in the universe. New physics can be probed for in three complementary realms:

- Experiments at the **high-energy frontier** allow for a direct search of new heavy particles. If the maximal centre-of-mass energy of collisions in an experiment is higher than the rest mass of a new particle, it can be produced and detected via its decay products. Examples of such experiments are ATLAS or CMS located at the Large Hadron Collider (LHC) at CERN.
- The second realm is the **high-precision frontier**. In this case, extremely large detector volumes are used to search for extremely weak interactions of conventional matter with unknown particles. A highly efficient background suppression is crucial as even a single event is a clear sign for new physics.
- At the **high-intensity frontier**, experiments search for decay modes which are in conflict with the predictions of the SM. To this end, high decay rates are required to keep the runtime of the experiments at a reasonable scale of a few years. Again, an extremely efficient background suppression is of crucial importance.

The Mu3e experiment faces the high-intensity frontier. It is dedicated to search for a lepton flavour violating decay in the charged sector, $\mu^+ \to e^+e^-e^+$. Such a decay has not been observed to date and would be a clear hint for new physics beyond the Standard Model.

In the first part of this thesis, the Standard Model of Particle Physics is presented with a particular focus on muon decays and the phenomenon of lepton flavour violation. In this context, possible muon decays based on physics Beyond the Standard Model (BSM) are discussed.

In the next chapter, a comprehensive review of the planned Mu3e experiment is given. The signal properties as well as existing background processes are discussed. From these, the experimental requirements on the Mu3e detector are derived. The Mu3e detector concept including all its subcomponents is presented.

Afterwards, a chapter is dedicated to the latest MuPix sensor prototype – the MuPix8 – covering the chip characteristics and features as well as the setups which are used to operate the chip. In this context, a detailed description of the required hardware and software is given.

The next chapter is dedicated to the firmware-based MuPix8 emulator. After a discussion of the concept, a precise description of the firmware implementation of the 'internal' as well as the 'external' emulator is given, followed by measurements which have been performed with these setups.

In chapter 6, measurements and findings from the process of commissioning the MuPix8 chip are shown.

Finally, the results and findings are summarised and an outlook is given on which future projects can make use of the hardware components and the MuPix8 emulator, which have both been developed within the scope of this thesis. Open questions and future improvements are pointed out.

2 Theory

In this chapter, which is inspired by [1], an introduction to the Standard Model of particle physics is given. A particular emphasis is put on muon physics in order to motivate the theoretical concepts behind the Mu3e experiment.

2.1 The Standard Model of Particle Physics

The **Standard Model of Particle Physics** (SM) provides a mathematical description of the fundamental constituents of our universe – the elementary particles – and the interactions between them. It has withstood countless experimental tests and is – until today – unprecedented in its precise agreement with experimental data.

Each interaction is described by a dedicated **Quantum Field Theory** (QFT) which is based on the concept of fields being the fundamental entities. Within this framework, particles can be understood as quantised excitations of these fields and interactions are described by the exchange of mediator particles called gauge bosons.

Fundamental Particles

The **fundamental particles** of the SM comprise six quarks and six leptons as well as their corresponding anti-particles. All these particles carry a spin of $\frac{1}{2}$ and are referred to as **fermions**. In addition, there are four types of gauge bosons which mediate the fundamental forces, and the Higgs boson. Figure 2.1 shows a summary of all particles forming the SM.

The fermions are grouped in three generations of two **quarks** and two **leptons** each. The first generation comprises the up- and the down-quark (u and d) as well as the electron (e) and the electron neutrino (ν_e).

This first generation is stable and makes up most of the known matter around us. u- and d-quarks make up protons and neutrons which are bound in atomic nuclei.

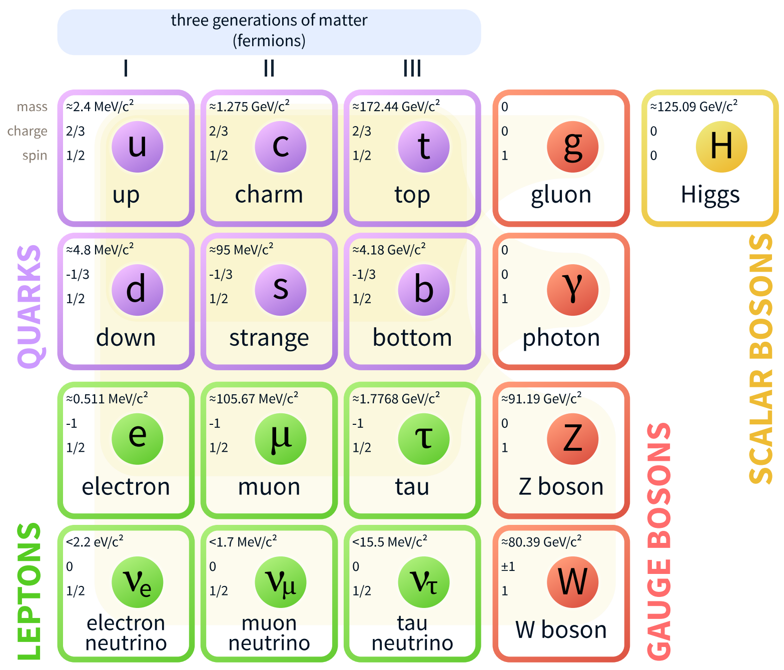

three generations of matter (fermions)
I
II
III
mass
charge
spin
≈2.4 MeV/c² 2/3 1/2 u up
≈1.275 GeV/c² 2/3 1/2 c charm
≈172.44 GeV/c² 2/3 1/2 t top
0 0 1 g gluon
≈125.09 GeV/c² 0 0 H Higgs
≈4.8 MeV/c² -1/3 1/2 d down
≈95 MeV/c² -1/3 1/2 s strange
≈4.18 GeV/c² -1/3 1/2 b bottom
0 0 1 γ photon
≈0.511 MeV/c² -1 1/2 e electron
≈105.67 MeV/c² -1 1/2 μ muon
≈1.7768 GeV/c² -1 1/2 τ tau
≈91.19 GeV/c² 0 1 Z Z boson
<2.2 eV/c² 0 1/2 νe electron neutrino
<1.7 MeV/c² 0 1/2 νμ muon neutrino
<15.5 MeV/c² 0 1/2 ντ tau neutrino
≈80.39 GeV/c² ±1 1 W W boson
QUARKS
LEPTONS
GAUGE BOSONS
SCALAR BOSONS

Figure 2.1: Fundamental particles of the Standard Model of Particle Physics. From [2], modified.

Together with electrons, the nuclei form atoms which in turn are the constituents of molecules.

The other two generations only show up when particles interact at higher energies. They correspond to heavier copies of the first generation. In contrast to the first generation, they are unstable and decay after a finite lifetime (except for the neutrinos). The second generation comprises the charm- and strange-quark (c and s), muon (μ^-) and muon neutrino ν_μ. The top- and bottom-quark (t and b), the tau τ and the tau neutrino ν_τ are grouped in the third generation. The reason for this pattern of three generations is not yet understood.

Up-type quarks – u, c and t – possess an electrical charge of $\frac{2}{3}$ in units of the elementary charge. Down-type quarks – d, s and b – have an electrical charge of $-\frac{1}{3}$. The three charged leptons – e, μ and τ – carry one negative elementary charge. For each of the charged leptons, there exists a corresponding neutrino which is named accordingly – ν_e, ν_μ and ν_τ – and is electrically neutral.

In addition to the fermions, there are four gauge bosons which carry a spin of 1. They mediate the fundamental forces of the SM which are described below. The Higgs boson is the only scalar particle in the SM carrying a spin of 0.

Fundamental Interactions

Interactions are described by the exchange of the gauge bosons between particles. Three of the four fundamental forces are incorporated in the SM: the electromagnetic, the weak, and the strong interaction. Only gravitation is not included. However, compared to the other forces it is extremely weak. Even though it is responsible for the formation of structures in the universe on large scales, it can be neglected on a particle interaction level.

The most familiar force known from everyday life is the **electromagnetic force**. It acts between all charged fermions, namely the three charged leptons (e, μ, τ) as well as all six quarks. The charged gauge bosons of the weak interaction take part in the electromagnetic interaction as well. The gauge boson mediating the electromagnetic force is the photon. The theory behind electromagnetic interactions is known as **Quantum Electrodynamics** (QED).

The **weak interaction** is mediated by the $W^{\pm}$, which carry electric charge, and the electrically neutral Z boson and acts between all quarks and leptons. It is the only interaction in which neutrinos take part and explains the β-decay of particular radioactive nuclei as well as the origin of the solar energy – nuclear fusion.

The electromagnetic and the weak interaction converge at high energy scales and can be described as a unified theory called **electroweak interaction**.

The **strong force** is responsible for the binding forces in neutrons and protons. It is described by **Quantum Chromodynamics** (QCD). The corresponding gauge bosons are gluons which are able to exchange colour charge (red, green, blue) between quarks. Unlike for the electromagnetic interaction, there are only colour-neutral objects in nature. Quarks cannot be observed freely but are always confined in baryons. There are mesons consisting of quark-antiquark pairs and hadrons being made up of three quarks. Only at very high energy scales, quarks become asymptotically free.

The Higgs Mechanism

For many years, the Higgs bosons was the last missing piece of the puzzle of the SM. It was discovered in 2012 by the ATLAS and CMS experiments at the Large Hadron Collider (LHC) at CERN [3, 4].

The Higgs boson corresponds to the excitation of the Higgs field which – unlike the fields associated with all the other SM particles – has a non-vanishing vacuum expectation value. Only with the Higgs mechanism, it can be understood how other fundamental particles acquire a mass.

2.2 LFV Processes

In the original SM, lepton flavour is conserved, i.e. the number of leptons per generation is constant. However, **lepton flavour violation** (LFV) has been observed in the neutrino sector. The results of several experiments such as Super-Kamiokande [5], SNO [6], KamLAND [7], and the Daya Bay Neutrino Experiment [8, 9], lead to the conclusion that neutrinos oscillate between the three flavour eigenstates when propagating in space. This is only possible when the neutrinos carry a non-vanishing mass which is not included in the SM.

Therefore, extensions of the SM have been developed in which right-handed massive neutrinos are included. Such theories, e.g. the so-called νMSM [10], yield consistent results with the experiments named above.

2.3 BSM Physics

Despite the fact that the SM has withstood countless experimental tests with an enormous precision, there are phenomena in nature which cannot be explained by the SM.

Astronomical observations show that only a small fraction of the matter in the universe is baryonic matter as described by the SM. For instance, the investigation of the rotation curves of galaxies (see Figure 2.2) shows that there must be dark matter, i.e. matter that interacts with 'classical' matter only through gravitation [11]. Also observations of gravitational lensing [12] imply the existence of such matter.

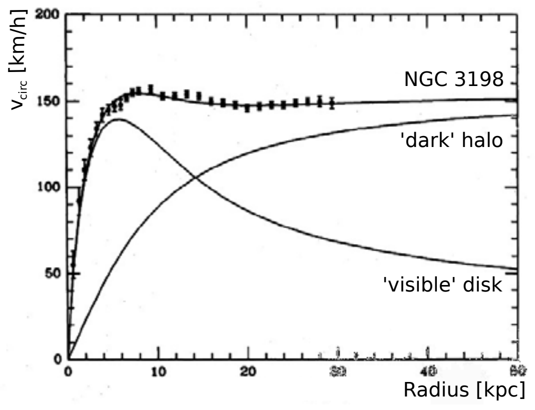


Figure 2.2: Rotation curve of the galaxy *NGC 3198*. In addition to the data points, the contributions from the *disk* of visible matter and the *halo* of dark matter are shown. From [13], modified.

Further open questions are related to the origin of the observed matter-antimatter asymmetry in the universe or the inclusion of gravitation into a more comprehensive model.

Theories Beyond the Standard Model (BSM) like **Supersymmetry** (SUSY) or **String Theory** provide new approaches of explaining these open questions, for instance by predicting new particles. It is up to experimental particle physicists to detect such particles or exclude their existence for particular mass ranges and coupling strengths. By pushing the limits to higher energies one might get a direct grasp of very heavy new particles. High rates are required to explore very rare decays and thus allow for an indirect access to higher mass regions.

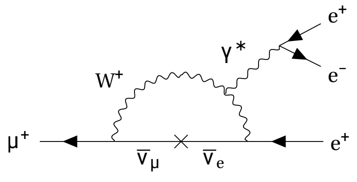
2.4 Muon Decays

Muons are unstable and have a mass of ~106 MeV/c^2 and a lifetime of ~2.2 µs. The dominant SM decay $\mu^+ \to e^+\nu_\mu\bar{\nu}_e$, the Michel decay, has a branching ratio of ~100 %. The second most probable decay is $\mu^+ \to e^+\nu_e\bar{\nu}_\mu\gamma$ with a branching ratio of 1.4 %, followed by the process $\mu^+ \to e^+e^-e^+\bar{\nu}_\mu\nu_e$ with a branching ratio of 3.4×10^{-5} [14].

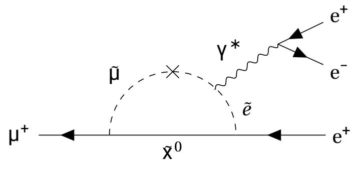
The process $\mu^+ \to e^+e^-e^+$ can in principle occur within the rules of an extended SM when the ν_μ changes its flavour to a ν_e in a loop as shown in the Feynman diagram in Figure 2.3a. This would correspond to a lepton flavour violating process as discussed in 2.2. However, this process is strongly suppressed due to the large mass of the W boson of 80.4 GeV/c^2 in the loop compared to the neutrino mass splitting of $\Delta m_{21}^2 \approx 8 \times 10^{-5}\,\mathrm{eV}^2$. Therefore, the branching ratio of $\mathcal{O}(10^{-54})$ is far beyond anything that could be accessed experimentally.

A SUSY process, for instance, might allow the decay $\mu^+ \to e^+e^-e^+$ in a loop as shown in Figure 2.3b involving new super-symmetric particles. Different BSM theories allow for the process $\mu^+ \to e^+e^-e^+$ at tree-level as shown in Figure 2.3c. This would only be possible via a new type of interaction which allows a direct coupling of a μ^+ to an e^+.

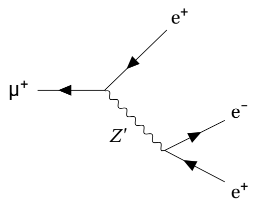
In such SUSY models or other BSM theories, much higher branching ratios are possible through new types of interactions or loop contributions with new sorts of particles – potentially up to a level which can be accessed by experiment. Therefore, any observation of the decay $\mu^+ \to e^+e^-e^+$ would be a clear sign for new physics [15].



(a) νMSM process.



(b) SUSY process.



(c) Tree-level LFV process.

Figure 2.3: Feyman diagrams for νMSM and BSM processes of the decay $\mu^+ \to e^+e^-e^+$.

3 The Mu3e Experiment

In this chapter, the concept of the Mu3e experiment and its detector components is discussed in detail. Furthermore, the signal and possible background processes are described. In the end, an overview of the situation of experiments dedicated for the search of lepton flavour violating processes in the charged sector is presented.

3.1 Signal & Background

The Mu3e experiment is dedicated to search for the signal $\mu^+ \to e^+e^-e^+$ as shown in Figure 3.1a. In the following, muons imply μ^+ and electrons as well as positrons will be referred to as electrons. Thus, the signal can be denoted $\mu \to eee$. The exact characteristics of the signal as well as the background processes are discussed in the following.

3.1.1 The Signal

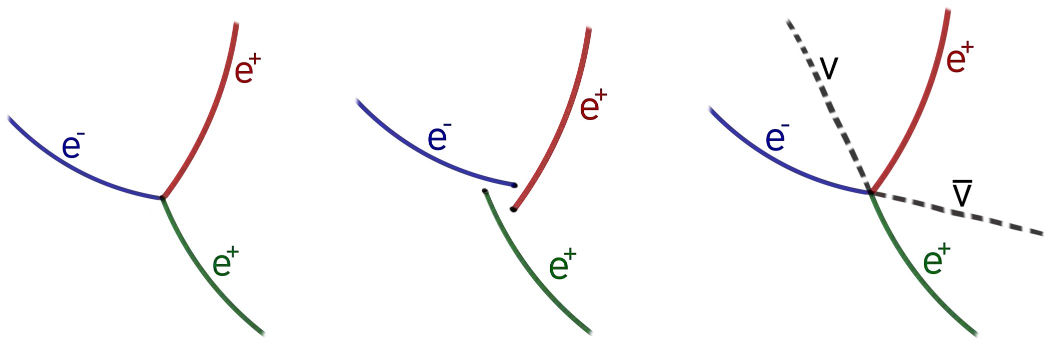
The **signal** consists of three electrons emerging from a common vertex as depicted in Figure 3.1a. As they stem from the decay of a single muon, they are coincident in time and the energy of all decay particles must sum up to the muon mass:

$$E_{\text{tot}} = \sum_i E_i = m_\mu c^2 \approx 105.7\,\text{MeV} \tag{3.1}$$

Since the muons are stopped before decaying, the momenta of the three electrons sum up to zero:

$$\vec{p}_\mu = \sum_i \vec{p}_i = 0. \tag{3.2}$$

Consequently, a single decay particle cannot have a momentum of more than half the muon mass $\approx 53\,\text{MeV/c}$.



(a) Signal. **(b)** Combinatorial. **(c)** Internal conversion.

Figure 3.1: Signal and background topologies.

With the help of these characteristics, the signal can clearly be distinguished from background processes as described in the following and summarised in Table 3.1.

	signal	combinatorial background	internal conversion
common vertex	yes	no	yes
coincidence in time	yes	no	yes
$\sum \vec{p}$	$= 0$	any	$\neq 0$
$\sum E$	$= m_\mu$	any	$< m_\mu$

Table 3.1: Characteristics of signal and background processes.

3.1.2 Background

The target sensitivity of the Mu3e experiment will be determined by the ability to suppress background processes. There are two main types of background which are described below.

Combinatorial background, as shown in Figure 3.1b, arises due to limited spatial and timing resolution. Three electrons from different processes like Michel decays and Bhabha scattering can be mistaken to stem from a common vertex and be coincident in time within the measurement uncertainties. This type of background is suppressible by an excellent momentum and vertex resolution of the detector.

In contrast, background from **internal conversion** (see Figure 3.1c) is irreducible. It arises from a fundamental physics process and corresponds to the third most frequent muon decay $\mu \to eee\nu\nu$ with a branching ratio of 3.4×10^{-5} [14]. It resembles the signal because the three detected electrons are also coincident in time and stem from a common vertex. Even though the neutrinos leave the detector unobserved, one can distinguish this decay from signal events as the sum of the three electron momenta does not vanish and the sum of the energies does not equal the muon mass. That is because small fractions of the momentum and the energy are carried away by the two neutrinos.

3.2 Experimental Requirements

The detector requirements can be derived from the signal and background properties as described above in consideration of the target sensitivity of one in 10^{16} decays.

An excellent vertex resolution is required to be able to suppress combinatorial background. A highly precise momentum resolution below 1 MeV/c and a very accurate timing measurement below 500 ps per track are needed to discriminate the signal against internal conversion decays and combinatorial background. As the electrons have very low momenta, the momentum resolution is limited by multiple Coulomb scattering in the detector material. Therefore, the material budget of the detector must be kept as low as $\mathcal{O}(1\,‰)$ of a radiation length per layer.

In addition, the Mu3e experiment requires a very high rate capability of $\mathcal{O}(10^9)$ muon decays per second to keep measurement time at a reasonable scale of one year. Finally, the highest possible geometric acceptance is desired to probe new physics in the largest possible phase space.

In Figure 3.2, the branching ratio of the internal conversion background is plotted against the undetected energy. At the target sensitivity of 1 in 10^{16} muon decays, the missing energy has to be determined at a level of 1 MeV.

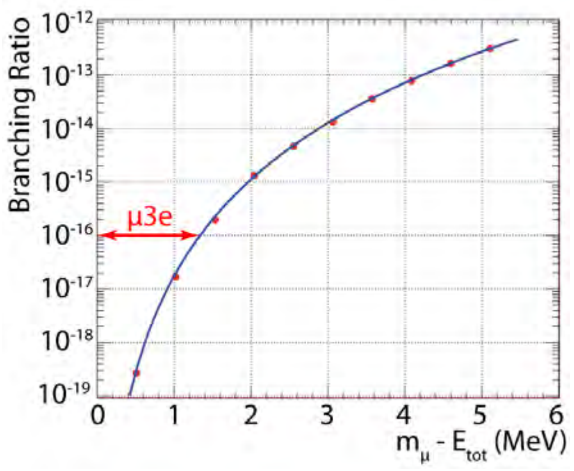


Figure 3.2: Plot showing the branching ratio of the decay $\mu \to eee\nu\nu$ versus the missing energy. The required missing energy resolution for Mu3e is shown in red [16].

3.3 The Mu3e Detector

In the following, the detector concept for phase I of the Mu3e experiment and the subcomponents of the detector are presented.

3.3.1 The Detector Concept

The underlying concept of the Mu3e experiment is to stop muons on a thin target. Because the entire detector is placed in a homogeneous 1 T magnetic field, the trajectories of the decay electrons are bent. This allows the electron momentum to be determined from the radius of the trajectory and the charge from the direction of curvature.

To keep the material budget low, the detector will be cooled with gaseous helium. The detector consists of three subsystems: the pixel tracker for an excellent vertex and track reconstruction, and the scintillating tiles and scintillating fibres for more precise timing measurements. The subdetectors as well as the muon beam and other components are presented in detail below.

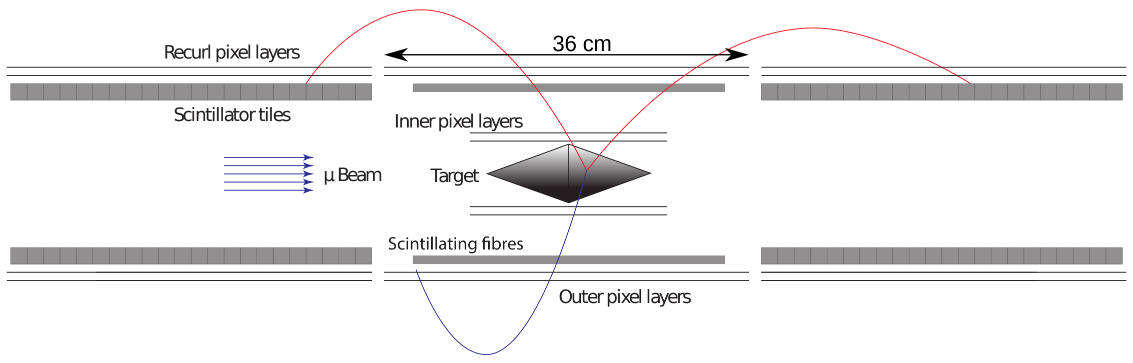


Figure 3.3: Schematic transverse view of the Mu3e detector in phase I.

3.3.2 The Muon Beam

The Mu3e experiment will be operated at the **Paul-Scherrer Institute** (PSI) in Villigen in Switzerland, where the world's most intense muon beamline – the πE5 [17] – is located. It provides low-momentum muons with rates up to $10^8\,1/\mathrm{s}$.

Currently, studies are carried out on the **high intensity muon beam** (HiMB) which would provide up to 10^{10} muons per second [18].

3.3.3 The Target

For the target of the Mu3e detector, a trade-off between a high stopping power, maximal vertex separation and the least possible amount of material in the transverse direction needs to be found. A double-cone target design out of Mylar with a thickness of $\mathcal{O}(200\,\mu\mathrm{m})$ meets these requirements. It has a length of 100 mm and a radius of 19 mm and will be held in place by thin nylon strings.

3.3.4 The Pixel Tracker

The Mu3e concept of an ultra-low material budget detector [19] is based on pixel chips which are thinned down to 50 µm. The entire support structure will be made from a very thin polyimide (Kapton® [20]) foil. Power supply, slowcontrol and readout of the pixel chips will be realized with the help of flexprints consisting of aluminium and polyimide. This way, the material budget is reduced to a radiation length of only 1.1 ‰ per layer.

The tracking detector will consist of two barrel-shaped double layers of pixel chips. The inner two layers will have a length of 12 cm and radii of $\sim$23.3 mm and $\sim$29.8 mm, respectively. For the outer double layer, a length of 34 cm at a radius of 74.9 mm and a length of 36 cm at a radius 86.3 mm are planned. On either side of the barrel, an additional double layer of pixels, called recurl station, is used to track electrons which recurl in the strong electromagnetic field. They are identical to the outer double layer described above.

The tracker will comprise a total of 2844 pixel sensors covering an active area of about 1.1 m^2.

HV-MAPS Technology

The pixel tracker of the Mu3e detector will be equipped with **High-Voltage Monolithic Active Pixel Sensors** (HV-MAPS). They are manufactured in an HV-CMOS process. A schematic of an HV-MAPS is shown in Figure 3.4. The prototype series which is currently being characterised is the MuPix family.

For MAPS, deep n-wells are implemented in a p-substrate to form diodes. A charged particle passing through or photons getting absorbed lead to the creation of electron-hole pairs. The charges are then separated and collected via diffusion.

In HV-MAPS, an additional high voltage of $\mathcal{O}(100\,\mathrm{V})$ is applied to form a depletion zone with a strong electric field across the diode such that the charge is collected via drift rather than diffusion making charge collection much faster compared to MAPS.

In a monolithic chip, the entire readout electronics is integrated in the chip. Therefore, there is no need for a separate readout chip as it is common for hybrid sensors. As a consequence, a significant amount of material can be saved making it suitable for use in the Mu3e experiment.

In addition, a large fraction of the passive bulk material can be removed such that the chips can be thinned down to 50 µm. This corresponds to merely 0.054 % of a radiation length.

The Final MuPix

The final MuPix chip is foreseen to have a size of $20 \times 23\,\mathrm{mm}^2$ with an active area of $20 \times 20\,\mathrm{mm}^2$. This corresponds to 250×250 pixels with a size of $80 \times 80\,\mu\mathrm{m}^2$. The

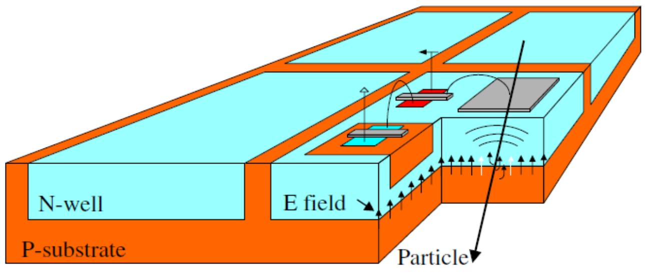


Figure 3.4: Schematic view of an HV-MAPS [21].

final MuPix will be read out via up to three serial links at a data transmission rate of 1.25 Gb/s each. Three differential links will be employed for the inner double-layer where a high occupancy requires a high bandwidth. For the outer layers, one differential link suffices due to lower particle rates.

3.3.5 The Timing Detector

The timing information from the pixel sensors does not suffice for an effective background suppression at high muon decay rates. Therefore, the Mu3e detector will be equipped with scintillating tiles and fibres for additional precise timing measurements.

Scintillating fibres with a timing resolution of ≤ 0.5 ns will be placed just underneath the outer double layer of pixel sensors. Like the pixel sensors, they are required to be as thin as possible to keep the material budget low such that traversing electrons are deflected as little as possible. Fibre ribbons with a length of 28 cm will be used.

Scintillating tiles with a timing resolution better than 100 ps complement the fibres. They will be placed underneath the pixel double layer of the recurl stations. In contrast to the fibres, the material budget does not play a role here because no further measurements are performed after the particles have passed through the tiles, as seen in Figure 3.3.

The fibres as well as the tiles will be connected to silicon photomultipliers (SiPMs) and readout by a full-custom readout chip, the so-called **MuTRiG** [22].

3.3.6 The Mu3e Readout Scheme

The Mu3e detector is read out trigger-less. It is continuous because the exact time of a muon decay is random and cannot be determined beforehand.

An overview of the readout concept of the Mu3e detector is shown in Figure 3.5. All the front-end electronics is placed inside the magnetic field of the solenoid as indicated by the *orange* box in Figure 3.5.

The data from the three subdetector systems – the pixels, tiles and fibres – is transmitted electrically to the so-called **front-end boards** via differential links at a transmission rate of 1.25 Gbit/s. On the front-end boards, FGPAs are used to time-sort the data from the pixel sensors and form packages in which hits with the same timestamp are grouped to reduce the overhead.

From the front-end boards, the data gets sent out via optical links at a transmission rate of 6 Gbit/s. The optical transmission ensures an electrical decoupling from all components inside the magnetic field and allows for the use of a high bandwidth. The optical signals are transformed back into electrical signals on the so-called **switching boards** where the data from the front-end boards is merged such that 50 ns long time slices of the three sub-detector can be sent to the **GPU filter farm**.

The PCs in the filter farm are equipped with powerful graphical processing units (GPUs) to perform an online event selection in order to reduce the amount of data and store only events of interest [23]. These events are transmitted to a data collection server and saved in a mass storage system for later analysis.

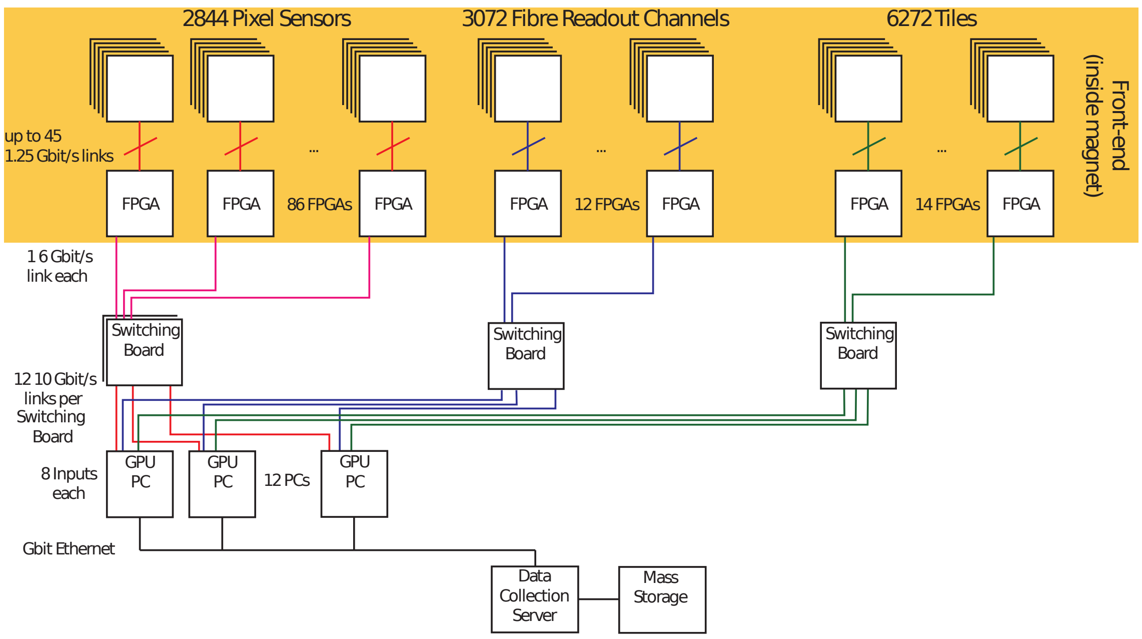


Figure 3.5: Schematic showing the readout scheme of the Mu3e detector.

3.4 Experimental Situation

A variety of experiments has been dedicated to search for lepton flavour violation in the charged sector. Until today, no signal was found. Figure 3.6 summarizes the upper limits of multiple LFV processes that have been set by experiments performed during the last 70 years. The most precise measurement of the decay $\mu^+ \rightarrow e^+e^-e^+$ was performed by SINDRUM, whereas the MEG collaboration set the lowest limit on the decay $\mu^+ \rightarrow e^+\gamma$. The SINDRUM II experiment was dedicated to the search for a muon to electron conversion in nuclei. The Mu2e experiment and the COMET experiment are expected to improve this limit in the next years.

SINDRUM (1983-86)

The SINDRUM experiment was performed at the Paul Scherrer Institute (PSI) in Switzerland, from 1983 to 1986. It was searching for the decay $\mu^+ \rightarrow e^+e^-e^+$.

To this end, a low-momentum muon beam was stopped at a target where the muons decayed at rest. The decay electrons were then tracked by a multi-wire proportional chamber and a trigger hodoscope.

The upper limit of the branching ratio was set to $\mathrm{BR}(\mu^+ \rightarrow e^+e^-e^+) \leq 10^{-12}$ at a confidence level (CL) of 90 %. The result is limited by the number of muons that have been stopped during the runtime [25].

SINDRUM II

The successor of SINDRUM, the SINDRUM II experiment, was dedicated to search for the coherent neutrinoless muon conversion into an electron in the presence of a nucleus: $\mu^- N \to e^- N$. An upper limit of 7×10^{-13} was set at 90 % CL [26].

MEG (2008-today)

The MEG experiment is also located at PSI in Switzerland. It is searching for the decay $\mu^+ \to e^- \gamma$ and is taking data since 2008. Like for SINDRUM, low-momentum muons are stopped at a target before decaying. The positrons are tracked in a drift chamber whereas the photons are detected in a liquid xenon calorimeter.

In 2016, an analysis of the full dataset has been published, giving the most stringent upper limit on $\mathrm{BR}(\mu^+ \to e^+\gamma) \leq 4.2 \times 10^{-13}$ at 90 % CL [27].

An upgrade of the experiment, MEG II, is currently being assembled to increase the sensitivity to $\sim 5 \times 10^{-14}$ [28].

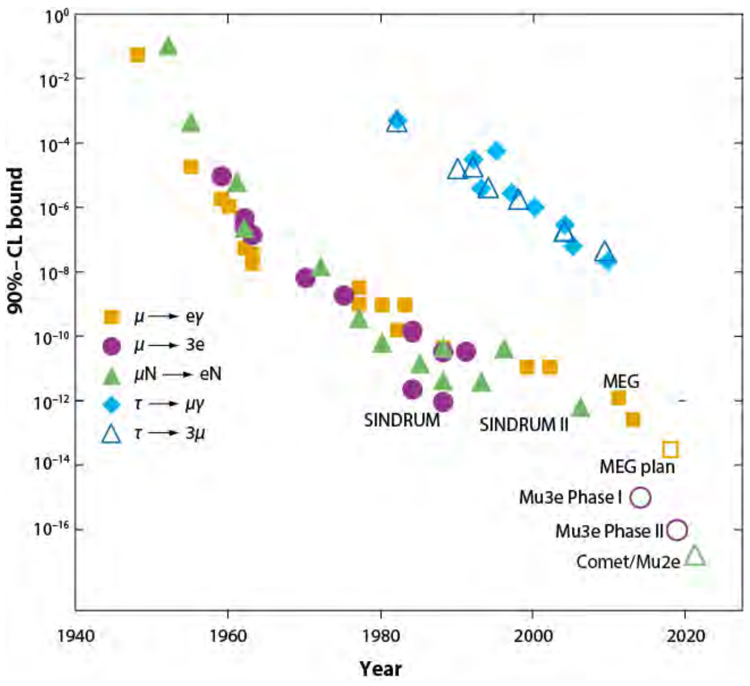


Figure 3.6: Overview of the history and future of experiments investigating LVF processes. From [24].

COMET (2018-2019 planned)

The COMET experiment [29] is located at J-PARC in Japan and will start taking data in 2018.

Low-momentum muons will be stopped at an aluminium target where they can be captured by the nuclei. The electrons will be tracked in a cylindrical drift chamber which is complemented by a set of trigger hodoscope counters.

The expected single event sensitivity in the first phase of the experiment is 3×10^{-15}. In a second phase, it is anticipated to be improved to 2.6×10^{-17}.

Mu2e (under construction)

Like COMET, the Mu2e experiment [30] is dedicated to search for a muon-to-electron conversion in the presence of a nucleus. If no signal is found, a new limit will be set at $\mathrm{BR} < 8 \times 10^{-17}$ at $90\,\%$ CL.

Again, low-momentum muons will be stopped at an aluminium target. The decay electrons will be detected by a straw tracker in combination with an electromagnetic calorimeter.

The experiment is located at Fermilab in the United States. It is currently under construction and will start taking data in 2022 [31].

4 Hardware for the MuPix8

In the following, the **MuPix8** will be described in detail. The analogue pixel electronics will be presented as well as the digital periphery and the readout state machine.

This chapter also comprises a presentation of the various **hardware components** that are required to operate the MuPix8 as well as the MuPix8 emulator. In addition, the **readout firmware** and **software** are described.

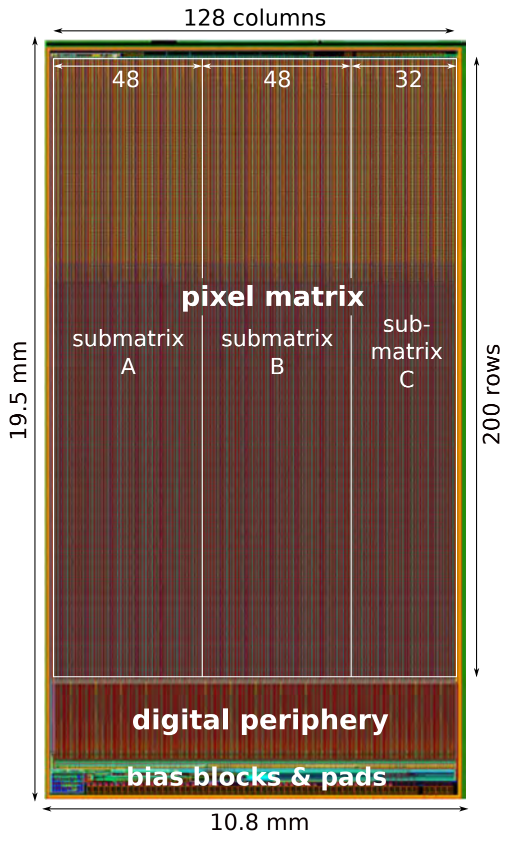
4.1 The MuPix8

The MuPix8 sensor is based on the HV-MAPS technology as described in section 3.3. With a size of 10.8×19.5 mm^2, it is the first large-scale prototype in the MuPix family. A schematic drawing and a photograph are shown in Figure 4.1.

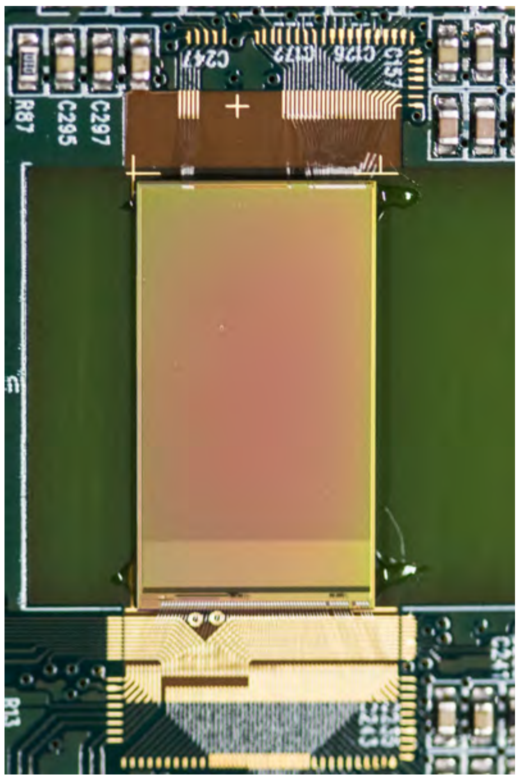
The sensor has 128 columns and 200 rows corresponding to a total of 25600 pixels which are divided into **three submatrices A, B and C** with 48, 48 and 32 columns, respectively. The pixel size is 81×80 µm^2 such that the active area of the sensor is ∼166 mm^2.

In addition to the analogue pixel matrix, the chip comprises the **digital periphery** in which the complete readout electronics is placed. It also features **bias blocks** in which reference voltages can be generated by on-chip *digital-to-analogue converters* (DACs). The **pads** along the bottom edge can be contacted with bond wires. These pads are essential to operate the chip. Further pads along the top edge (not indicated in Figure 4.1a) serve as debugging contacts for probing or can be connected to additional supply voltages if needed.

The chip features **four serial LVDS data output links**, three of which correspond to the submatrices A, B and C. On the fourth link D, one can either send out a copy of the data from A, B or C, or multiplex the data from all three submatrices in a round-robin scheme if running the readout state machine at a reduced speed. Each link runs at a data transmission rate of up to 1.6 Gb/s and is nominally op-



(a) Schematic drawing of the MuPix8 chip showing the geometric dimensions and the subdivision into three submatrices as well as the location of the digital periphery and the bias blocks and pads. From [34], modified.



(b) Photograph showing the MuPix8 chip glued and bonded to an insert (see section 4.3.2).

Figure 4.1: Schematic drawing and photograph of the MuPix8 sensor chip.

erated at 1.25 Gb/s. The data is 8b/10b encoded according to the standard IBM scheme [32, 33].

In Figure 4.2, a simplified schematic of the chip electronics of the MuPix8 including the analogue pixel electronics as well as the digital periphery and the readout state machine is depicted. In the following, its elements and their interplay are discussed in detail.

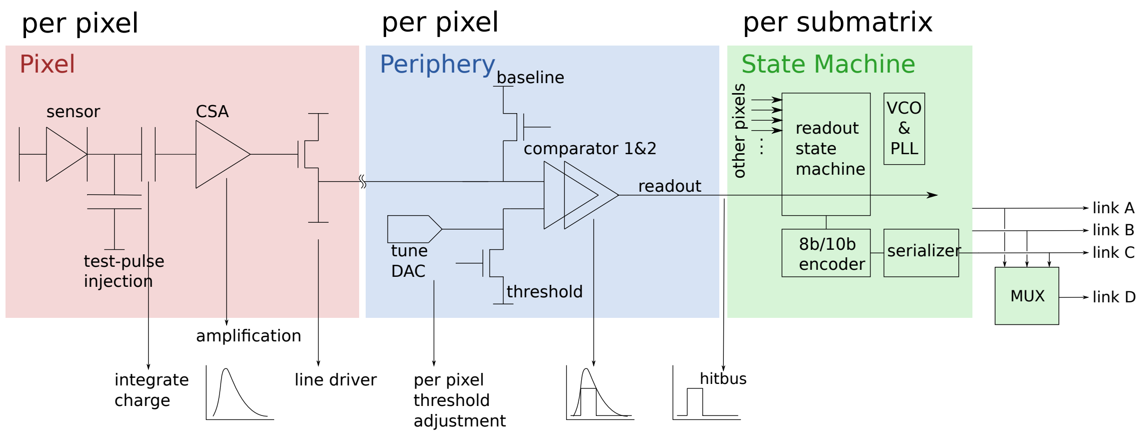


Figure 4.2: Schematic of the chip electronics of the MuPix8 including the analogue pixel electronics, the digital periphery and the readout state machine.

Analogue Pixel Electronics

A pixel in the MuPix8 (see *red* box in Figure 4.2) is based on a diode operated in *reverse-bias* mode. When an ionizing particle traverses the diode, the generated electron-hole pairs are separated by the strong electric field in the depletion zone and collected ('integrated') on a capacity. The induced voltage pulse gets enhanced by a *charge sensitive amplifier* (CSA).

The **analogue output** pulse from the CSA can be fed out of the chip to be probed externally. However, this feature is only available for pixels of the leftmost column of submatrix A. The pixel selection is done when configuring the chip (see below).

From the CSA, the signal is driven to the *digital periphery* located along the bottom edge of the chip (see Figure 4.1a). In this aspect, the three submatrices differ: the signal from submatrix A is driven by a *source follower* whereas the other two make use of a *current driven* scheme.

An artificial signal can be induced with the help of a **test-pulse injection** circuit. To this end, a capacitor which is located between the sensor diode and the CSA gets charged by applying an external voltage. The external voltage is then suddenly pulled to ground such that the capacitor discharges.

The discharge resembles a ’real’ signal and is a powerful tool in the process of commissioning the chip. The injection can be enabled for individual pixels when configuring the chip (see below).

Digital Periphery

In the periphery (see *blue* box in Figure 4.2), every pixel has its digital readout cell comprising two comparators into which the analogue pulse is fed. The two thresholds are set globally but can be fine-adjusted per pixel using a 2-bit and a 3-bit DAC, respectively. The digitized signal is then forwarded to the *readout state machine* which is described below.

As shown in Figure 4.3, two pulses with different amplitudes exceed the threshold of a comparator at different times, even if they start in exactly the same moment. This time difference is referred to as **timewalk**.

The two comparators in the digital pixel cell can be operated in three modes described below. They are designated to investigate and compare different timewalk correction strategies in order to increase the timing resolution of the MuPix8:

1. In the **time-over-threshold (ToT) mode**, a single threshold is used for the discrimination of an incoming pulse, i.e. only one of the comparators is used. When the pulse exceeds the threshold, a 10-bit timestamp is saved. A second 6-bit timestamp is saved when the pulse falls below the threshold again.

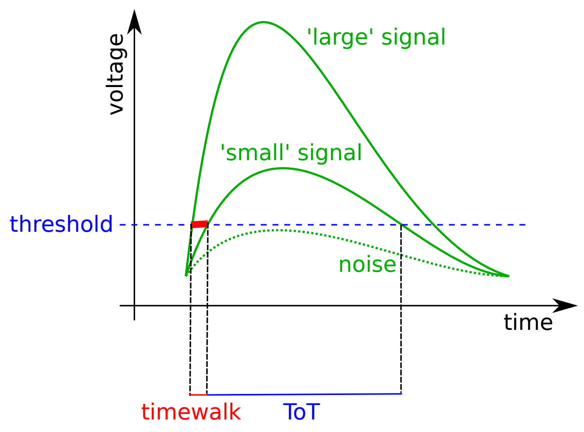


Figure 4.3: Illustration of the **time-over-threshold** (ToT) (blue) and the effect of **timewalk** (red) for pulses of different heights. From [35], modified.

Therefore, the difference between the two timestamps corresponds to the **ToT** of the pulse (see Figure 4.3).

2. In the **two-threshold mode**, an incoming pulse is compared to a low threshold to get a precise timing. Because noise is likely to surpass this low threshold, a second higher threshold is used which needs to be crossed to mark an actual hit (see Figure 4.4). Like in mode 1, the 10-bit timestamp is saved when a pulse exceeds the low threshold and the 6-bit timestamp when the pulse drops below the threshold again.

3. If a pulse exceeds the high threshold in the **ramp mode** (see Figure 4.5), the 10-bit timestamp is stored and a ramp voltage with an adjustable slope starting from the low threshold is triggered to rise. When the rising threshold crosses the pulse, the 6-bit timestamp of the hit is fixed.

The digital signal from the second comparator (which generally has the higher threshold) can also be fed out of the chip to be probed externally. It is referred to as the **hitbus** signal and corresponds to the ToT for this comparator. When configuring the chip (see below), the readout of the hitbus signal can be enabled for a column of choice such that the output signal corresponds to a logic '`or`' of the hitbus signals from all pixels in this column.

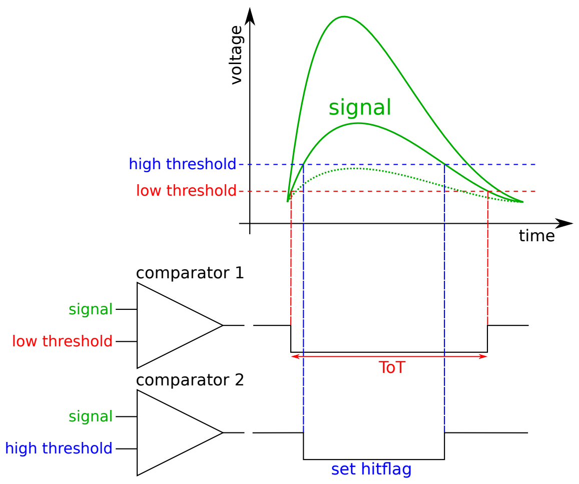


Figure 4.4: Illustration of the two-threshold mode. From [35], modified.

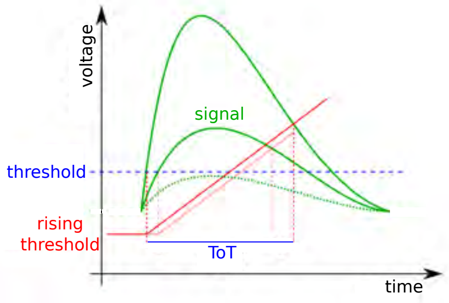


Figure 4.5: Illustration of the ramp mode. From [35], modified.

Readout State Machine and Serializer

The MuPix8 features one readout state machine for each submatrix A, B and C (see *green* box in Figure 4.2). They receive signals from the digital periphery. A description of the exact readout procedure is given in section 5.1 where the concept of the MuPix8 emulator is discussed.

The data leaving the readout state machine is 8b/10b encoded and fed into a serializer to be sent out of the chip as a *low-voltage differential signal* (LVDS). As mentioned above, the chip has **four data output links**. Three of them are used for the data from the submatrices A-C. The fourth link D can either copy the data from one of the other links or send out the multiplexed data from links A-C. In the latter case, the readout state machine needs to be run at reduced speed of 1/3 or slower (i.e. 1/6, 1/9, etc.).

Clocking

On the MuPix8, a *voltage controlled oscillator* (VCO) and a *phase-locked loop* (PLL) receive a reference clock signal, which is provided to the chip externally, and derive all internal clocks from it. The nominal frequency of the reference clock is 125 MHz but it can be varied up to 160 MHz. A minimum frequency needs to be found experimentally.

Timestamp Generation

The MuPix8 generates a binary counter and two timestamps on-chip. The **24-bit binary counter** serves as a 'chip timestamp' which is sent out during the readout procedure to allow for a sorting of the data if operating multiple chips at a time. As illustrated in Figure 4.6, the 8 least significant bits of this binary counter are Gray encoded [36] and also sent out. This is a means to test the integrity of the on-chip Gray encoding scheme.

The **10-bit timestamp** and the **6-bit timestamp** are both assigned to a particular hit. They are generated in a Gray encoded scheme with a maximal frequency of 160 MHz (nominal 125 MHz). However, the incrementation frequencies can be decreased by dedicated configuration values. More details can be found in Appendix A.2 where the configuration values and their functionality are presented in detail.

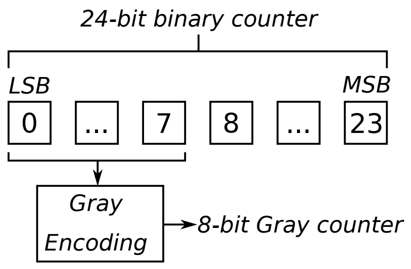


Figure 4.6: Illustration of the 24-bit binary counter and the 8 less significant bits being Gray encoded. LSB stands for *Least Significant Bit*, MSB for *Most Significant Bit*.

Configuration Registers

The MuPix8 features configuration registers realized as *linear shift registers* (LSR). The configuration registers are chained up, giving a total length of 2998 bits. They comprise:

1. One register for the configuration of the **digital part**. It allows to adjust bias voltages for the various components in the digital periphery.
2. One **digital row** register to specify which row is configured with the pixel-related values written to register 3 (see below).
3. One **column** register for a column-wise injection and the hitbus enable. It is also used for the pixel-wise fine tuning of the comparator thresholds. The register contains the tuning values for all 128 columns. To tune pixel-wise, the correct row needs to be selected in register 2.
4. Another **row** register to activate the analogue buffer output (only available in the leftmost column) and for a row-wise injection enable. The injection is enabled in the pixel which is marked in both registers 3 and 4.
5. One register to configure the **readout state machine**.
6. Finally, one register block is used for the **on-chip voltage DACs** which generate thresholds and baselines.

To configure the chip, the configuration bits are pushed into the LSR serially, as indicated in Figure 4.7. At the end of chain, the bits 'drop out' of the LSR and are fed back to the DAQ PC which allows to read back the values from the configuration register.

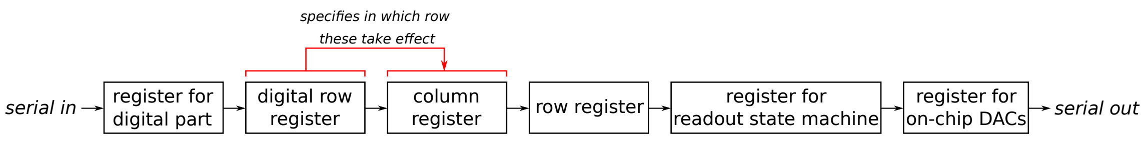


Figure 4.7: Illustration of the chain of **configurations registers**.

For a **global configuration** of the MuPix8, the entire chain of 2998 bits needs to be pushed into the LSR such that the sections corresponding to the different configuration registers described above are in the right position. They are then 'loaded' to take effect as DAC values. The exact working principle is described in section 6.1 where it is tested.

For a **pixel-wise configuration** such as a pixel tuning, the 2998-bit chain needs to be written 200 times. In this case, the pixel-tuning values from register 3 are loaded into the row which is marked by the digital row register 2. By design, the entire 2998 bits need to be written 200 times even though only the sections corresponding to the registers 2 and 3 are changed in each iteration.

Temperature Diode

The MuPix8 features an on-chip temperature diode. A defined current can be injected into the diode and the resulting voltage drop across the diode can be measured. For a constant current, the measured voltage drop is temperature dependent and a calibration can be performed (see section 6.6).

4.2 MuPix8 Setups

Two setups have been developed to debug, examine, and characterise the MuPix chips. In addition, a closely related setup has been derived from these to be able to run the 'external' MuPix8 emulator which has been developed within the scope of this thesis (see Chapter 5).

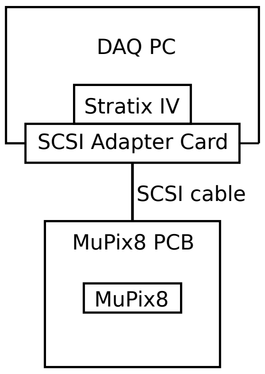
4.2.1 The MuPix8 Single Setup

The **single setup** is used to operate one single MuPix8 chip. Schematic drawings of the two variations in use are shown in Figure 4.8. The respective hardware components are described in detail in the section 4.3.

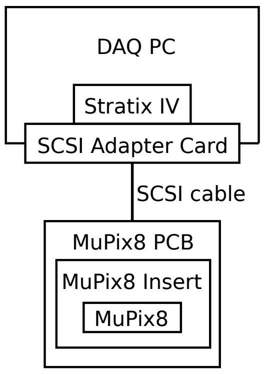
The setup as shown in Figure 4.8a resembles older single setups for the predecessors of the MuPix8. A MuPix8 chip is directly glued and bonded to the **MuPix8 PCB** (see section 4.3.1) which is linked to the **MuPix8 SCSI Adapter Card** (see section 4.3.5) via a SCSI cable. The adapter card connects to a **Stratix IV dev board** (see section 4.3.6) mounted into a PC which is running the data acquisition software (see section 4.4).

The novelty in Figure 4.8b is the additional **MuPix8 Insert** (see section 4.3.2) on which the MuPix8 is glued and bonded. The insert slides into an edge connector on the MuPix8 PCB and can thus be exchanged quickly without having to unplug SCSI and power cables. Photographs of this setup are shown in Figure 4.9.

The single setup is used in the lab for debugging and measurements with radioactive sources as well as for an integration into the EUTelescope at DESY test beam campaigns [37].

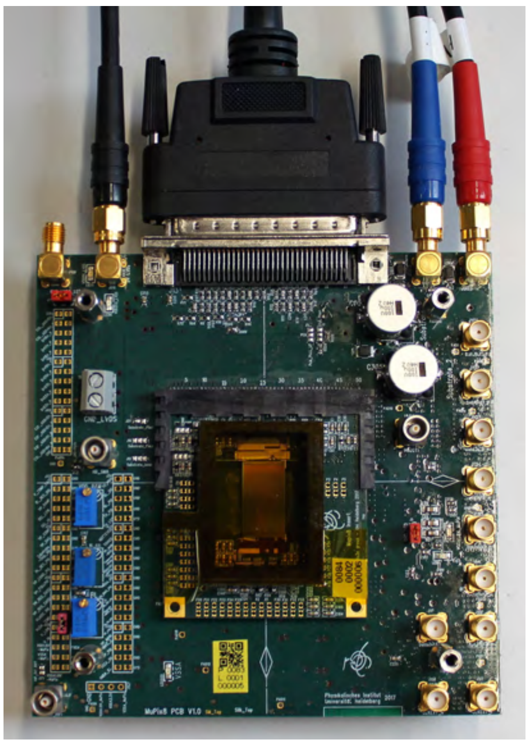


(a) Single setup with the MuPix8 glued and bonded **directly** onto the MuPix8 PCB.

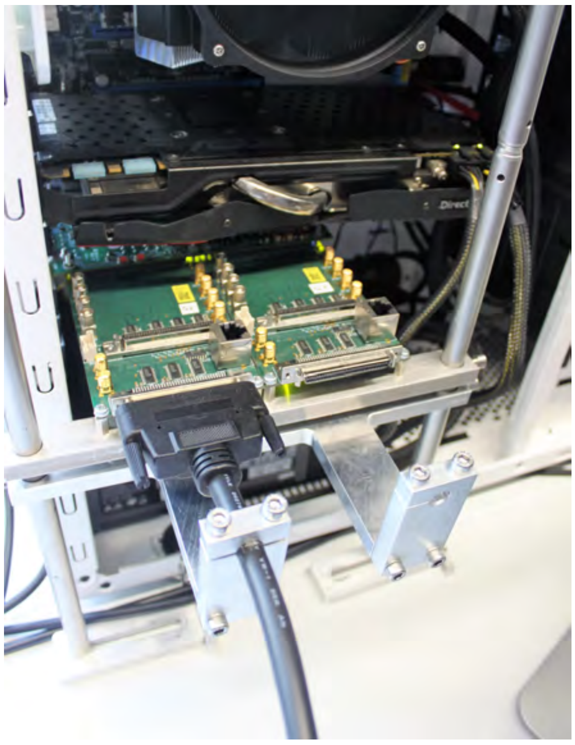


(b) Single setup with the MuPix8 glued and bonded onto the **MuPix8 Insert**.

Figure 4.8: Schematic drawings of the two versions of the MuPix8 single setup.



(a) The MuPix8 is glued and bonded to an insert which is mounted on the MuPix8 PCB.



(b) The SCSI cable links the MuPix8 PCB to an HSMC Adapter Card which is connected to the Stratix IV in the PC.

Figure 4.9: Photographs showing **MuPix8 single setup** with an insert.

4.2.2 The MuPix8 Telescope

The **telescope** is made to operate multiple (typically four) MuPix8 chips at a time as illustrated in Figure 4.10. It utilizes the same hardware components as the single setup. In order to operate four MuPix8 chips, again one PC and one Stratix IV are needed. In contrast to the single setup, there are now two MuPix8 SCSI Adapter Cards connected to the FPGA. Each adapter card has two SCSI connectors such that two MuPix8 PCBs can be linked in parallel. Therefore, four MuPix8 chips can be controlled and read out by one PC.

This setup is used at DESY and PSI test beam campaigns. It allows to perform position resolved efficiency measurements as well as timing resolution measurements. In principle, it can also be employed as a tracking telescope for the characterisation of other devices-under-test (DUTs). The four chips are placed behind each other in a beam line as shown in Figure 4.11. Three planes are typically used as reference planes for one DUT for which measurements can be performed.

There is also the option to assemble a so-called **octoscope** integrating eight MuPix8 sensors using two FPGAs in a single PC. This requires an additional synchronisation between the two FPGAs using an external reference clock.

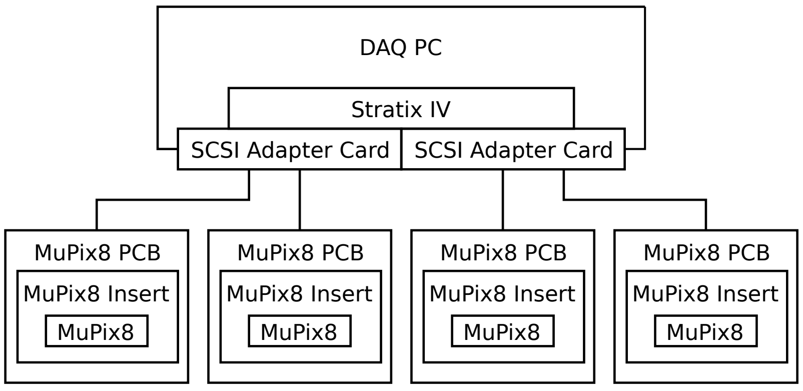


Figure 4.10: Schematic drawing of the **MuPix8 telescope.**

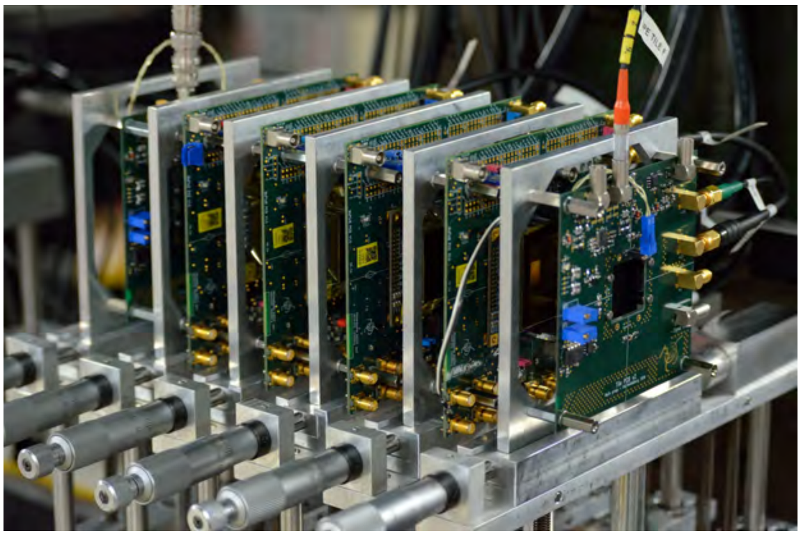


Figure 4.11: Photograph of the **MuPix8 telescope** as used in a DESY test beam campaign. The PCBs are held in place by aluminium frames, micrometer screws are used for mechanical alignment. In addition to four MuPix8 layers, scintillating tiles are mounted in the front and back of the setup.

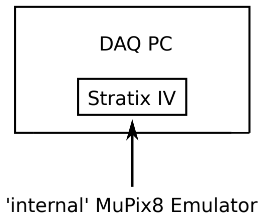
4.2.3 The MuPix8 Emulator Setup

As part of this thesis, an 'internal' as well as an 'external' emulator have been developed. The exact concept as well as measurements performed with the emulator are discussed in chapter 5.

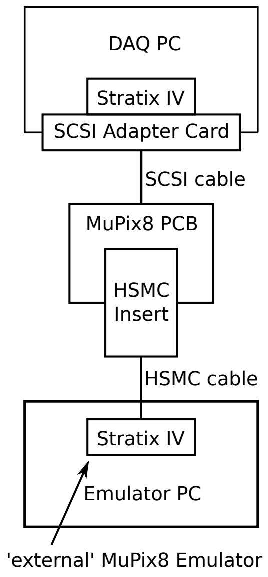
Concerning the hardware, the setup of the 'internal' emulator is very simple (see Figure 4.12a). The 'internal' emulator is embedded in the DAQ firmware running on the Stratix IV in the PC which runs the data acquisition software. Consequently, no hardware is needed in addition to the Stratix IV and the DAQ PC.

On the contrary, the setup for the 'external' emulator (see Figure 4.12b) comprises the full hardware required for the MuPix8 single setup except for the MuPix8 Insert which needs to be replaced by a dedicated adapter insert – the **MuPix8 HSMC Insert** (see section 4.3.4). This connects to a separate FPGA via an HSMC-to-HSMC flat ribbon cable. The FPGA is mounted in a PC which is in turn used for the control of the 'external' emulator via a GUI (see Figure 4.13).

One FPGA can be connected to up to two MuPix8 HSMC Inserts via the HSMC ports A and B. It is also possible to mount two FPGAs – which both run the emulator firmware – in one PC. This way, one can emulate four chips in parallel in order to test the *telescope* setup.



(a) The 'internal' emulator is embedded in the readout firmware.



(b) The 'external' emulator runs on a separate FPGA.

Figure 4.12: Schematic drawings of the setups for the 'internal' and the 'external' MuPix8 emulator.

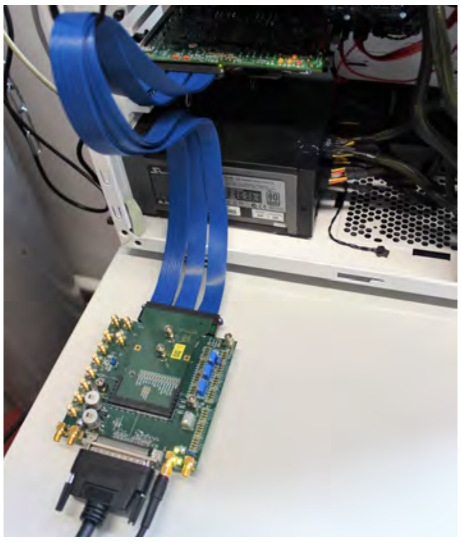


Figure 4.13: The MuPix8 PCB is connected to a Stratix IV in a PC via the MuPix8 HSMC Insert. The other half of the setup is identical to the single setup as shown in Figure 4.9b.

4.3 Hardware

In this section, the various **hardware components** required for the operation of the MuPix8 setups presented above are described. In addition, the **readout firmware** is presented.

4.3.1 The MuPix8 PCB

The **MuPix8 PCB** (see Figure 4.14) provides the interface for the powering, the slowcontrol, and the readout of one MuPix8 chip. The area in the centre of the PCB is left blank such that the MuPix8 chip can be glued here as illustrated in Figure 4.8a. It is not only free from components but also free from copper on all layers to make it usable in test beams where multiple layers of chips are placed behind each other. Copper in the beam would strongly increase the amount of scattering of the beam particles. Small metal marks help to position the chip when gluing and bonding it to the fine golden fanout along the bottom and the top edge of the yellow area.

There is also the option to mount a SAMTEC^{TM}edge connector [38] onto the PCB to be able to use different **MuPix8 Inserts** (see sections 4.3.2 - 4.3.4) as indicated in Figure 4.8b.

Differential Signals

Communication with the Stratix IV in the readout PC (see section 4.3.6) is realized via *low-voltage differential signals* (LVDS) sent to and from the **MuPix8 SCSI Adapter Card** (see section 4.3.5) through a *Small Computer System Interface* (SCSI) cable, which is a ribbon cable with 68 shielded cores. The differential signals going towards the MuPix8 sensor are terminated with $100\,\Omega$ resistors close to the fanout where the sensor can be glued and bonded.

The four differential data links from the MuPix8 are fed into an *LVDS repeater* to ensure that the signal strength is high enough for a transmission through the SCSI cable over a length of $\mathcal{O}(2\,\mathrm{m})$.

Single-Ended Signals

All other signals are single-ended on the MuPix8 side to reduce the number of required pins. However, they have to be transmitted differentially through the SCSI

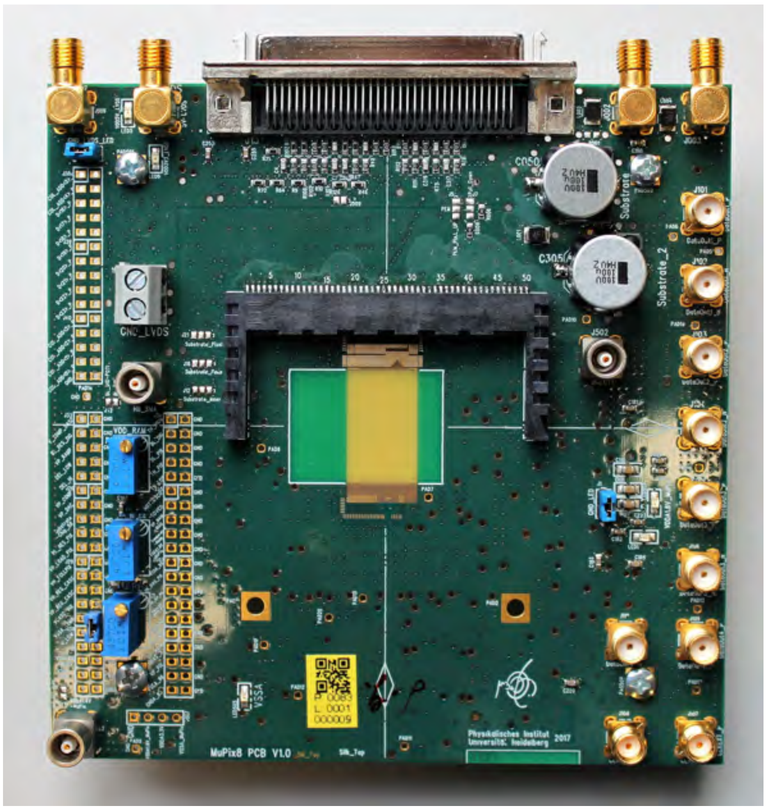


Figure 4.14: Photograph of the **MuPix8 PCB** equipped with an edge connector in the centre.

cable because single-ended signals cannot be driven strongly enough through the cable. Consequently, there is a number of *LVDS converters* on the MuPix8 PCB which transform single-ended signals from the MuPix8 into differential signals going to the SCSI Adapter Card and differential signals from the SCSI Adapter Card into single-ended going to the MuPix8.

Powering

The MuPix8 PCB is supplied with 5 V for the *LVDS repeater* and *converters*, a separate 5 V from which all supply voltages for the chip are derived using commercial *voltage regulators*, and a high voltage (HV) which is directly wired through to the MuPix8. The sensor is supplied with very clean ('low-ripple') voltages which are VDD of 1.8 V for the digital periphery, VSSA of 1 V for the CSAs in the pixels, and VDDA of 1.8 V to power the analogue pixel electronics. All supply and high voltages that are fed into the chip are separately filtered via 10 nF and 100 nF capacitors which are placed in close proximity to the chip.

Injection Circuitry

A dedicated circuitry exists to be able to generate *injection pulses* of variable amplitudes with the help of a 14-bit DAC. The pulses are fed into the chip to mimic a hit signal in a particular pixel.

External Threshold Generation

The MuPix8 features on-chip DACs which allow to adjust thresholds when configuring the chip. However, these thresholds can also be applied to the chip externally with the help of *DACs* and *potentiometers* on the MuPix8 PCB, when solder jumpers are set accordingly.

Temperature Circuitry

A dedicated circuitry is implemented which allows to address the on-chip temperature diode of the MuPix8. The circuitry involves a *DAC*, *amplifiers* and *precision resistors* and is used to generate an adjustable current which is sent into the chip. The voltage drop across the temperature diode on the chip is measured and digitized by an *analogue-to-digital converter* (ADC) on the MuPix8 PCB.

4.3.2 The MuPix8 Insert

The **MuPix8 Insert** is a small and relatively simple PCB. Photographs are shown in Figures 4.15. It is designed to carry one MuPix8 chip which can be glued and bonded to it. The insert can then be pushed into the edge connector of the MuPix8 PCB (see section 4.3.1). It features only passive components, namely resistors and capacitors. There are no active components which are expensive and increase the risk of soldering imperfections between fine-pitched contacting pins.

The insert has been developed and tested within the scope of this thesis and has proven to work in laboratory setups as well as on three DESY test beam campaigns. Results from the commissioning of the MuPix8 are presented in Chapter 6.

Compared to a design in which the chip is glued and bonded directly onto the MuPix8 PCB, this concept has various advantages:

- Smaller numbers of the large and expensive MuPix8 PCB need to be purchased and tested. Also, if one of the active components on the PCB fails, no MuPix8 chip is lost as would be the case if it was glued onto the MuPix8 PCB. Vice versa, a MuPix8 PCB can be used further on, even if a particular MuPix8 chip breaks.
- A functional MuPix8 PCB can be used for the operation of multiple chips such that the chips can be compared better as systematic errors arising from the use of different MuPix8 PCBs are excluded.
- It is much quicker and easier to exchange chips in an assembled setup such as a *telescope* because there is no need to unplug cables or loosen screws in order to exchange the entire MuPix8 PCB.
- Lastly, the insert can be replaced by the MuPix8 HSMC Insert which allows to assemble an 'external' emulator setup. Without an edge connector on the MuPix8 PCB, this would be much more complicated as the fine bonding fanout on the MuPix8 PCB would somehow need to be connected to the 'external' emulator.

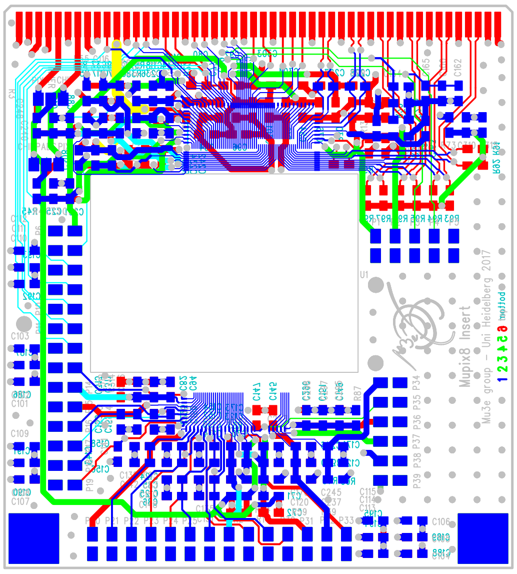
Features

Figure 4.15a shows a schematic view of the MuPix8 Insert. Like on the Mupix8 PCB, the area in the centre of the PCB is free from components and copper on all six layers. Small metal marks help to position the chip when gluing and bonding it to the fine golden fanout along the bottom and the top edge of the blank area. The pads in red along the top edge of the board make the connection to the *edge connector* which is mounted on the MuPix8 PCB.

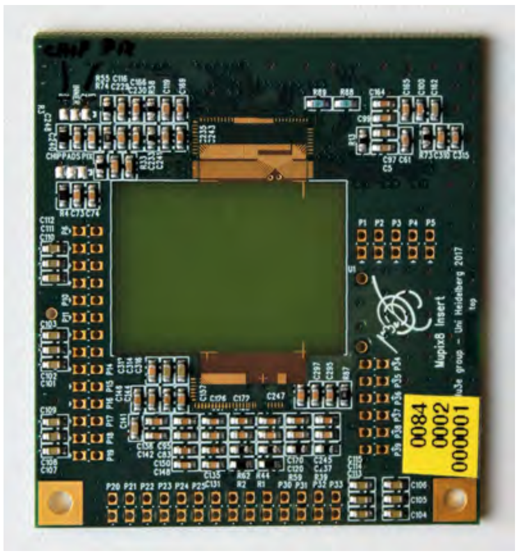
Like on the MuPix8 PCB, all supply and high voltages that are fed into the chip are separately filtered with the help of 10 nF and 100 nF capacitors which are placed as close to the chip as possible. They are connected in series with 22 mΩ resistors which can be removed to disconnect individual chip pads from the supply voltage for debugging purposes.

In addition, there are test points in the form of pin header holes which can be contacted with a voltage probe. The differential signals that are fed to the chip are terminated with 100 Ω very close to the fanout. The two large blue pads in the bottom left and right corner are mounting holes which allow to screw the insert to the MuPix8 PCB for a higher mechanical stability. Solder jumpers allow to connect the four existing high voltage nets in different combinations.

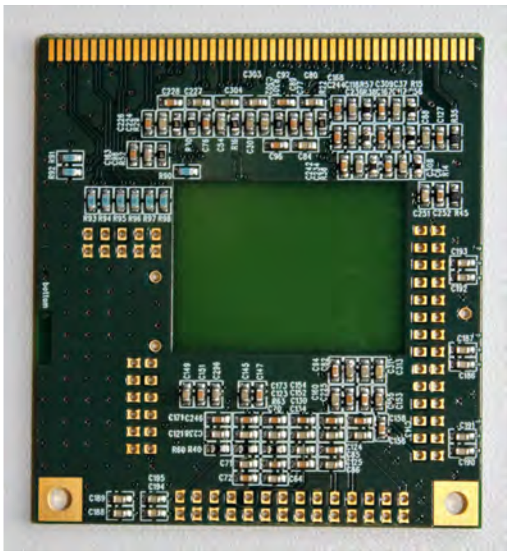
Three holes allow to put a custom-designed 3D-printed cap onto the insert after a chip has been mounted. It protects the chip and the bonds from dust and mechanical damage.



(a) Schematic. The top layer is coloured in blue, the bottom layer in red. Other colours indicate inner layers.



(b) Photograph – top view.



(c) Photograph – bottom view.

Figure 4.15: Schematic and photographs of the fully equipped **MuPix8 Insert**.

4.3.3 The MuPix8 Insert light

The **MuPix8 Insert *light*** (see Figure 4.16) is derived from the MuPix8 Insert and can be considered a *simplified* version. It has also been developed within the scope of this work.

The idea behind this insert is to have a PCB which is even cheaper and also faster and easier to populate and test compared to the MuPix8 Insert presented above. Therefore, only the most crucial components are placed on this insert *light*. At the time of writing, this PCB is in production and has not been delivered yet. As the PCB will be very easy and fast to assemble and test, it is suitable for production in larger quantities.

Features

The design of the fully equipped MuPix8 Insert followed a conservative approach with many decoupling capacitors. In contrast, the number of components on the MuPix8 Insert *light* is reduced to 10. Two critical bias voltages connected to the on-chip voltage controlled oscillator are grounded via a 100 pF capacitor and a 1 kΩ resistor. In addition, the ground and power of the digital logic are filtered through a 1 Ω resistor in series with parallel 10 nF and 100 nF capacitors. All other supply voltages as well as the high voltage are applied directly to the chip and are not filtered again.

As described in section 4.1, only the pads along the bottom edge are essential to operate the sensor. The pads along the top edge are auxilary and can be used to probe voltage drops across the chip. They are therefore wired to pin header holes as test points which can be contacted with a voltage probe. However, if it turns out that one or several of them are needed to operate the sensor they can be contacted to their corresponding supply voltage net with a jumper.

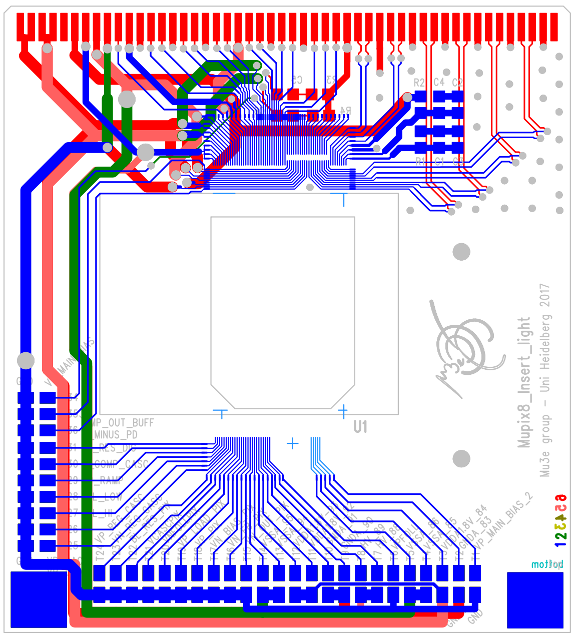


Figure 4.16: Schematic of the **MuPix8 Insert *light***. The top layer is coloured in blue, the bottom layer in red. Other colours indicate inner layers.

4.3.4 The MuPix8 HSMC Insert

In order to test the entire data acquisition chain from the MuPix8 PCB all the way through to the software without having the MuPix8 chip at hand, the 'external' MuPix8 emulator setup has been developed (see section 4.2.3). To realize this setup, a dedicated insert is required which can be connected to a separate FPGA running the 'external' MuPix8 emulator. To this end, the **MuPix8 HSMC Insert** (see Figure 4.17) has been designed and tested as a part of this work.

Features

The PCB is designed to suit the pinout of a Stratix IV running firmware to emulate the MuPix8. As a mere 'adapter card' its design is kept as simple as possible.

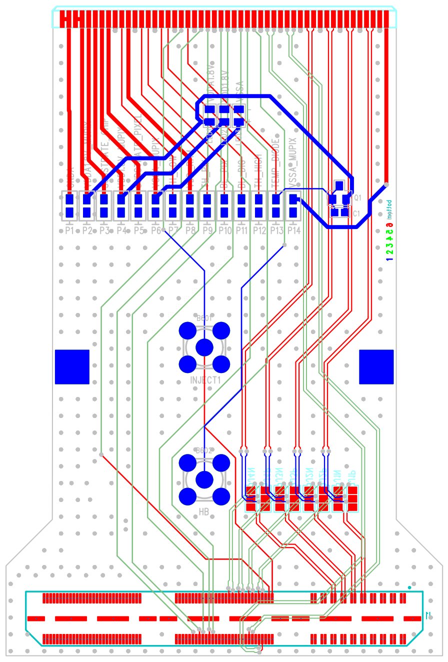
The MuPix8 HSMC Insert can be linked to a Stratix IV via HSMC-to-HSMC cable plugged onto the *HSMC connector*. The four fast data lines corresponding to the three submatrices A, B and C plus the multiplexed link D can be wired to two different types of transmitter blocks on the FPGA – the *Fast GX Transceivers* or the *LVDS Transmitters* – by setting solder jumpers. The Fast GX Transceivers have been used for all measurements (see section 5.3). The optional connection to the LVDS Transmitters is implemented as a backup option. Furthermore, there are differential lines for the external reference clock provided to the MuPix8 (emulator) from the readout FPGA and synchronous reset signal.

All slow signal lines are wired through as single-ended transmission lines. Two *LEMOs* are placed on the PCB as well, one for the hitbus signal generated by the 'external' MuPix8 emulator and one to probe the injection pulse generated on the MuPix8 PCB. They allow to investigate these signals with an oscilloscope.

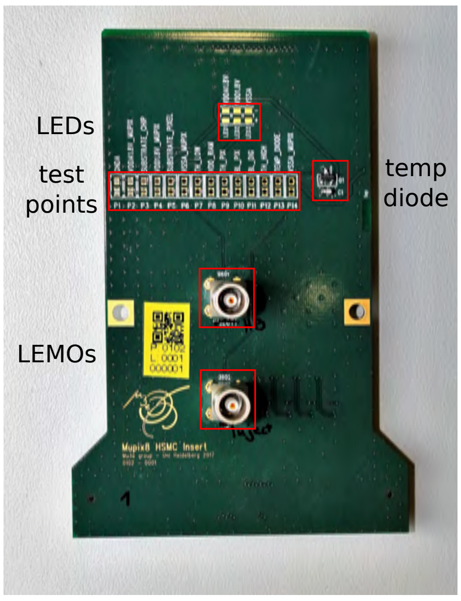
The following features are **not related** to the firmware of the 'external' emulator but allow further measurements with the MuPix8 emulator setup:

There is a *temperature diode* circuitry which is realized using a transistor in diode mode and a 470 pF decoupling capacitor. It is addressed by the temperature diode circuitry on the MuPix8 PCB to test the functionality of the of the latter.

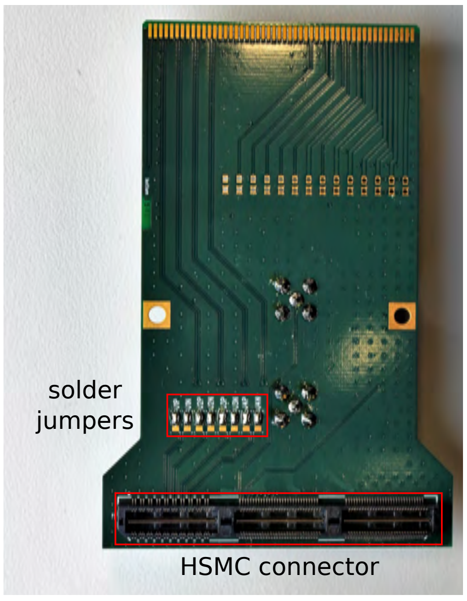
Furthermore, there are three LEDs which indicate whether the three supply voltages VDDA, VSSA and VDD are applied correctly. 14 pin header hole pairs serve as testpoints at which the baseline and threshold voltages, generated on the MuPix8 PCB and applied to the MuPix8 externally, can be probed.



(a) Schematic. The top layer is coloured in blue, the bottom layer in red. Other colours indicate inner layers.



(b) Photograph – top view.



(c) Photograph – bottom view.

Figure 4.17: Schematic and photographs of the **MuPix8 HSMC Insert**.

4.3.5 The MuPix8 SCSI Adapter Card

As depicted in Figure 4.8, the **MuPix8 SCSI Adapter Card** is required to make a connection between the Stratix IV in the data acquisition PC and the MuPix8 PCB. A schematic view and photographs are shown in Figure 4.18.

This PCB has been designed and tested in the context of this thesis and has proven to work in setups involving the 'external' MuPix8 emulator as well as the real MuPix8 chip. There is an *HSMC connector* for the link to the Stratix IV. Two *SCSI-68 connectors* allow for the connection to two MuPix8 PCBs.

Differential Signals

The fast signals, which comprise the fast data links, two clock signals and a synchronous reset, are sent from the Stratix IV to the MuPix8 or vice versa as differential signals. Solder jumpers can be set to connect the fast differential data from the chip either to the *Fast GX Transceivers* or the *LVDS Receivers* of the Stratix IV (see also section 4.3.6). One of the clock signals is fed into the on-chip PLL from which all on-chip clocks, like those for the readout state machine or the timestamp generation, are derived. The second clock is a backup for the LVDS transmission block of the MuPix8 in case the on-chip PLL causes problems.

Single-Ended Signals

All other signals are single-ended on the FPGA side to reduce the number of required pins. However, they have to be transmitted differentially through the SCSI cable because single-ended signals cannot be driven strongly enough over a length of $\mathcal{O}(2\,\mathrm{m})$. Consequently, there is a number of LVDS converters on the SCSI Adapter Card which transform single-ended signals from the Stratix IV into differential signals going to the MuPix8 PCB and differential signals from the MuPix8 PCB into single-ended going to the Stratix IV. The converters are powered by 3.3 V from the HSMC connector which is filtered by an inductor and a number of decoupling capacitors.

LEMOs and NIM-to-TTL

In addition, there are four *LEMO inputs* for external **'trigger'** signals which can be used for a coincidence setup with scintillators. A connector is needed for the $\pm 5\,\mathrm{V}$ supply voltage for the *NIM-to-TTL level shifting* of the external trigger signals.

In this case, **’trigger’** does **not** imply a signal that triggers the readout of the chip. It is an external signal used to generate ’trigger’ timestamps on the FPGA which can be merged into the data stream for later offline analysis.

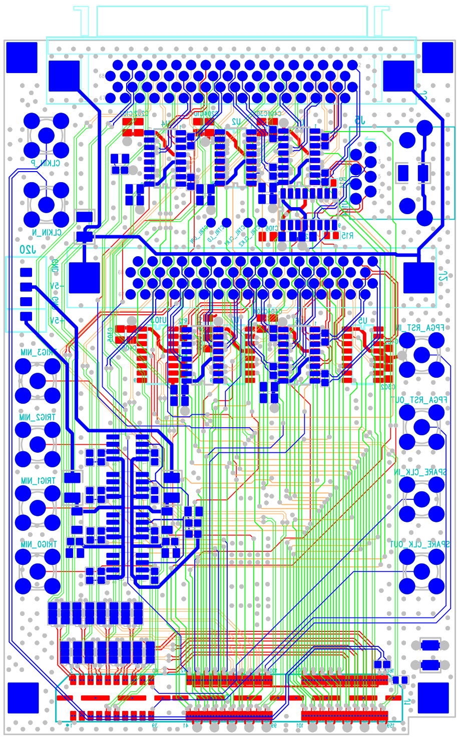
RJ45 Connector

An *RJ45 connector* allows for an integration of a **Trigger Logic Unit (TLU)** [39] in the EUDAQ system at DESY test beams in Hamburg. It comprises four differential signals. The TLU sends out a *reset* – currently not used – and a *trigger* signal, and receives a *clock* and a *busy* signal from the FPGA. The latter is used to indicate that the FPGA cannot process any incoming trigger signals temporarily.

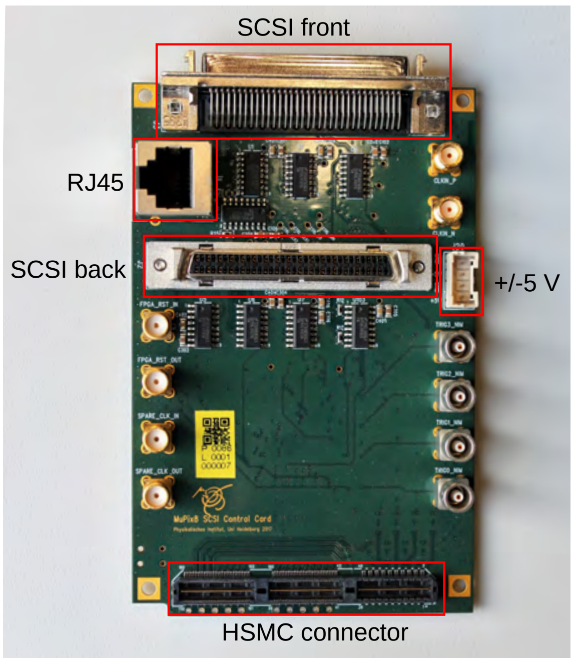
SMA Connectors

An external differential clock signal, e.g. from a clock board, can also be provided to the FPGA through a pair of *SubMiniature version A (SMA) connectors*. This feature is required when two FPGAs need to be operated in sync if running the *Octoscope*. Further *SMA connectors* comprise spare in- and outgoing clocks and a spare outgoing reset signal, as well as two spare general purpose inputs to the FPGA.

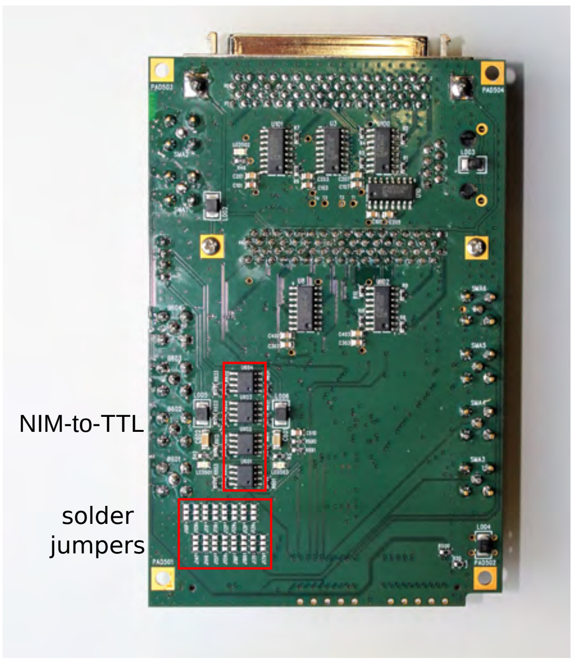
The metal housings of the SCSI connectors and the RJ45 connector are grounded and decoupled from the general ground through inductors. LEDs indicate that the 3.3 V for the LVDS converters as well as the ±5 V for the level shifters are applied.



(a) Schematic. The top layer is displayed in blue, the bottom layer in red. Inner layers are indicated by other colours.



(b) Photograph – top view.



(c) Photograph – bottom view.

Figure 4.18: Schematic and photographs of the **MuPix8 SCSI Adapter Card**. The large SMD components which are not highlighted correspond to the *LVDS converters*.

4.3.6 The Stratix IV Dev Board & DAQ Firmware

The **Stratix IV GX** is a *field-programmable gate array* (FPGA) from Altera. It is mounted on a commercial *development board* (dev board) [40] which can be used as the interface between the custom hardware and the PC running the DAQ software. A photograph is shown in Figure 4.19.

The dev board can be mounted in the *Peripheral Component Interconnect Express* (PCIe) slot on the mainboard of a PC. On the other side, it connects to up to two MuPix8 HSMC Adapter Cards (see section 4.3.5) through the *HSMC ports A* and *B*.
In the **single setup**, only port A is used. For the **telescope setup**, a second (identical) adapter card is connected to port B. The FPGA is running firmware which comprises a range of functionalities described below.

The configuration of the firmware is realised via *write*- and *read*-registers which are addressed by the DAQ software. The data which the FPGA acquires from the MuPix8 can be written to the **Random-Access Memory** (RAM) of the DAQ PC in two different modes:

- In the case of **Polling**, the FPGA tells the PC that new data is available. This data is written to the RAM of the PC when the PC accepts the write request.

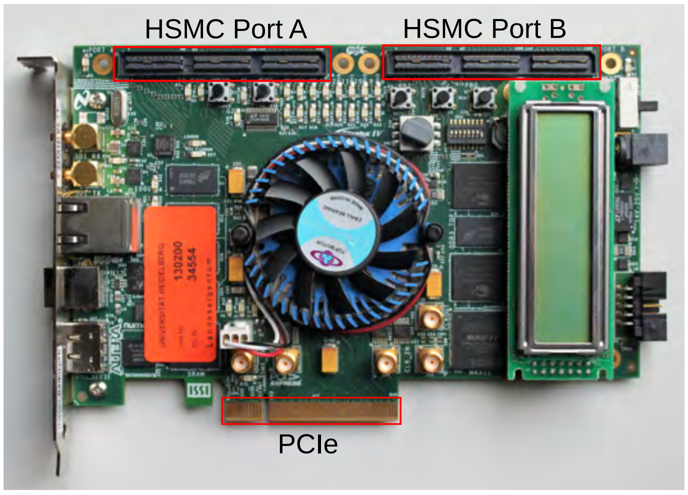


Figure 4.19: Photograph of the **Stratix IV dev board**. The FPGA itself is hidden behind the fan in the centre.

- The other option is **Direct Memory Access** (DMA) in which case the FPGA writes directly to the RAM of the PC without CPU involvement. This mode allows for a higher bandwidth compared to polling as the FPGA does not need to wait for feedback from the PC. Regular 'interrupts' are sent to the CPU in order to acknowledge new data.

The FPGA also runs a reconfigurable **phase-locked loop** (PLL) which generates the reference clock signal that is sent out to the sensor.

The Data Path

The **data path** is the entity comprising the various data processing steps in the DAQ firmware. It is illustrated in Figure 4.20 and described step by step in the following.

One FPGA is able to process the data from up to four MuPix8 chips – two chip per HSMC port. For simplicity, the description of the data path below is given for the data from one chip only.

The FPGA receives data from the MuPix8 via four differential links corresponding to the three submatrices A, B and C plus the multiplexed link D. On the FPGA, two different types of receiver blocks with are implemented in parallel: the **Fast GX Transceiver** block and the **LVDS Receiver** block. However, on the MuPix8 SCSI Adapter Card, solder jumpers need to be set such that **only one** of the receiver blocks receives physical signals. The respective other one is unused.

The **Fast GX Transceivers** have the advantage of an integrated 8b/10b decoding including an error detection feature. Therefore, they are ideal for the phase of commissioning the MuPix8 as the error detection is a powerful tool in the process of debugging. However, on the HSMC Port B (see Figure 4.19), only three links (for the submatrices A, B and C) can be connected to the fast transceivers. This is a limitation given by the availability of receiver blocks on the Stratix IV. Consequently, the multiplexed link D cannot be used in this case. HSMC Port B is only used in the telescope setup. Therefore, the telescope cannot be run using the multiplexed link D when using the Fast GX Transceivers.

On the contrary, the **LVDS Receivers** are foreseen to be used on the front-end boards in the final readout chain of the Mu3e detector. For them, the 8b/10b decoding is implemented separately by hand. An integrated error detection feature

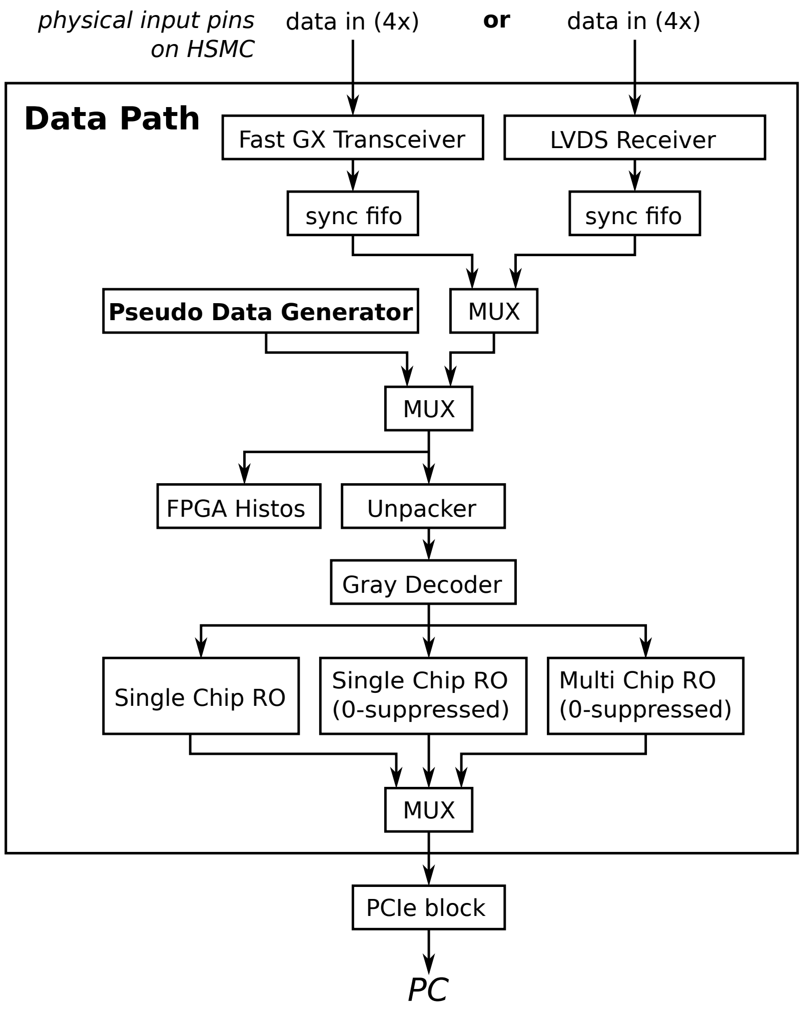


Figure 4.20: Schematic drawing of the **data path** in the MuPix8 readout firmware.

as for the Fast GX Transceivers is not available making them less suitable for the phase of commissioning the readout system. So far, only the Fast GX Transceivers have been used.

In a first step, the incoming data is linked to the selected receiver block, where it is deserialized and 8b/10b decoded. Afterwards, the data needs to be fed into a *FIFO* (first in, first out register) to ensure the synchronisation of all incoming links from the three submatrices and the multiplexed link. From here, the data is fed into a *multiplexer* (MUX) to select the output data from the recevier block which actually receives the signals from the chip. Then, there is an additional MUX to select between 'real' data from the MuPix8 and 'generated' data from the *Pseudo Data Generator*, which has been implemented as part of this thesis (see section 5.2).

The data is afterwards sent into the *Unpacker* where sections of the data stream are interpreted as link identifiers, hit addresses and timestamps such that hit packages are formed. Optionally, a number of histograms can be generated on the FPGA. They are implemented as an additional debugging tool.

In the next two steps, the hit addresses and the timestamps are optionally *Gray decoded* and forwarded into three different readout entities which further process the data according to the chosen readout mode. There are five different readout modes:

1. The first corresponds to a simple dumping of the raw data from all four differential inputs into the memory such that the output from the deserializer can be investigated.
2. For the second readout mode, one of the input channels is selected. The raw data from this link is then dumped into the memory. All other links are ignored.
3. The third mode resembles the second, but zero suppression is implemented in addition. This means that no data is written to memory when the data stream does not contain any hit information but consists of control words only.
4. Then, there is a readout mode which is dedicated to the readout of a telescope. It includes data sorting and package building for multiple chips.
5. Finally, the fifth readout mode is using a simple round robin arbitration scheme. Also in this mode, events are built for all connected links and written to memory.

In the last two modes, hitbus information are included as well as trigger timestamps if external trigger signals are fed into the MuPix8 HSMC Adapter Card via the dedicated *LEMO* connectors.

The data from the different readout blocks are fed into another multiplexer. Then, the output of the MUX is forwarded to the *PCIe block* which handles the communication with the DAQ PC such that the data can be written to the memory of the PC.

4.4 DAQ PC & Software

The DAQ software is implemented in C++ and based on **Qt** [41], **Eigen** [42] and **Boost** [43]. It comprises the readout software, an online monitoring feature and a Graphical User Interface (GUI) for a user-friendly control.

The software is designed in a modular approach. The **PCIe driver** handles the communication with the FPGA in that it maps the registers and memory of the Stratix IV to the local RAM of the PC. The **readout block** processes the data which is written to the FPGA interface and writes files to a local disk for later analysis. It requires a high computational power and is thus designed utilizing concurrent programming to make use of multiple cores. Several threads run in parallel and asynchronously and are controlled by a **MainThread** which also handles the GUI. Communication between the threads is done via queues.

From the **MainWindow** of the GUI, several dialog windows can be opened. A more detailed description of the GUI and the available features can be found in Appendix A.1.

5 Emulating the MuPix8 in Firmware

For the MuPix8, an FPGA-based chip emulator (*Pseudo Data Generator*) has been developed which is embedded in the readout firmware. It was of great help in the process of debugging the readout firmware and software because it excludes any hardware components other than the Stratix IV and the readout PC from the system. Therefore, one does not have to rely on the functionality of any custom-designed hardware of the readout chain or even the MuPix8 chip itself while debugging the readout firmware and software (see section 5.2).

In a second step, the emulator has been migrated onto a separate FPGA being connected to the MuPix8 PCB via a custom adapter card. This allowed to validate the functionality of the entire data acquisition chain from the MuPix8 PCB to the readout PC (see section 5.3).

In particular, emulating the MuPix8 was of great value because the hardware components required for the MuPix8 setups had arrived and were tested individually long before the actual MuPix8 chips were delivered. The use of the emulator allowed to test the integrity of these hardware components and to debug the *single setup* such that the commissioning of the MuPix8 could be prepared optimally.

In this chapter, the concept of the MuPix8 emulator is discussed. Furthermore, a detailed description of the firmware implementation and the measurements performed with both the 'internal' and the 'external' MuPix8 emulator is given.

5.1 Concept

The digital part as implemented in the MuPix8 has been synthesized from **Verilog HDL** code. This *hardware description language* (HDL) is also used for FPGA programming. It contains the readout state machine which sends and receives signals to and from the digital periphery and includes the 8b/10b encoding as well as the serializer. This piece of code has been taken as the foundation for the MuPix8 emulator.

The data acquisition firmware as described in section 4.3.6 is implemented in **VHDL** (*Very High Speed Integrated Circuit Hardware Description Language*). It is a powerful feature of these hardware description languages that firmware projects can be composed of a combination of entities implemented in both Verilog – such as the MuPix8 digital part – and VHDL – such as the readout firmware.

Consequently, the code description of the digital part of the MuPix8 did not have to be translated into a different HDL in order to be embedded in the readout firmware.

A major part of developing the emulator was to implement a wrapper containing the readout state machine as well as a firmware-based version of the digital pixel electronics to mimic its behaviour with respect to the communication with the readout state machine and to generate hits. The analogue matrix in its full complexity comprising charge-sensitive amplifier, line-drivers etc. as implemented in the MuPix8 is not required for a firmware-based emulator. Instead of the generation of analogue pulses which need to be discriminated to get digital signals, hits can be generated 'digitally' through a number of logic conditions.

Another important step was to extend the exisiting MuPix8 GUI by a dedicated control window for the emulator (see section 5.2.2). The same control window is used for both the 'internal' and the 'external' emulator. However, some features are only available in the 'internal' **or** the 'external' emulator as explained below.

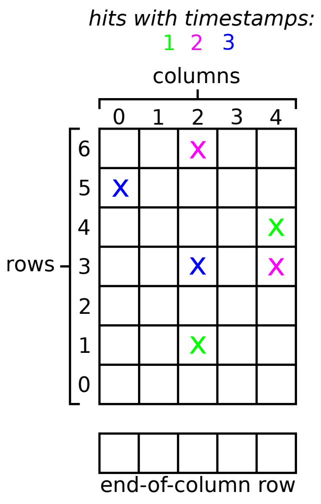
Non-Chronological Readout Scheme

The firmware implementation of the digital periphery of the MuPix8 must interact correctly with the MuPix8 readout state machine. An exact description of the state sequence and the signals sent and received by the state machine is given in Appendix A.2. At this point, it suffices to understand the readout concept: In the MuPix8, hits are read out in a non-chronological order.

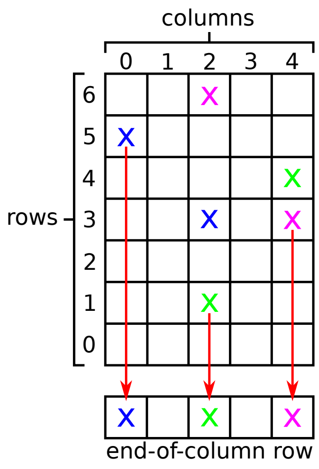
As described in section 4.1, the pulses from the analogue pixel matrix are driven to the periphery where every pixel has its digital cell. In this cell, the pulse is digitized by two comparators and the corresponding timestamps are saved until read out. In Figure 5.1, a matrix of these digital pixel cells is shown. For simplicity, its size is reduced to 5×7. In this example, the matrix is filled with six hits which have occured at different times (timestamps 1, 2 and 3, see Figure 5.1a).

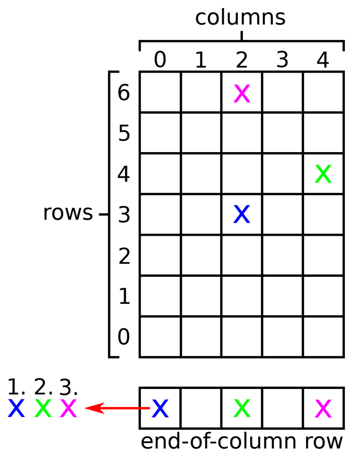
The readout procedure works as follows: A ’snapshot’ of the matrix is taken, i.e. the hits which have occurred are ’confirmed’ and ready to be read out. New hits which occur after the ’snapshot’ are not read out during this readout cycle but will be processed later when the next ’snapshot’ is taken.

From every column which contains at least one hit, the hit with the lowest row address is pulled down into the end-of-column row (see Figure 5.1b). Then, the hits are sent out consecutively (see Figure 5.1c). As can be seen, this scheme is non-chronological as it mixes up the time order of the hits. Once the end-of-column row is empty, the next load of hits is pulled down. This way, a maximum of 63 hits can be sent out before the next ’snapshot’ is taken. If less than 63 hits are contained in the ’snapshot’, the next ’snapshot’ is taken after these have been sent out. The maximum of 63 can also be reduced by a dedicated configuration value for the purpose of debugging.



(a) The hits are stored in the digital pixel cells. The three colours indicate different timestamps.





(b) Per column, the hit with the lowest row address is loaded into the end-of-column row.

(c) The hits in the end-of-column row are sent out consecutively by the readout state machine.

Figure 5.1: Illustration of the **non-chronological readout scheme**. For simplicity, 5×7 pixels are shown. They correspond to the digital pixel cells in the periphery which contain the comparators and **not** the pixels in the analogue matrix.

5.2 The Internal Emulator

The 'internal' emulator is a mean to test and validate the functionality of the data acquisition software and (most of) the readout firmware as described below. It is embedded in the firmware running on the Stratix IV in the data acquisition PC as illustrated in Figure 5.2.

Consequently, no hardware other than a Stratix IV and a DAQ PC are required to run the 'internal' emulator as described in section 4.2.3.

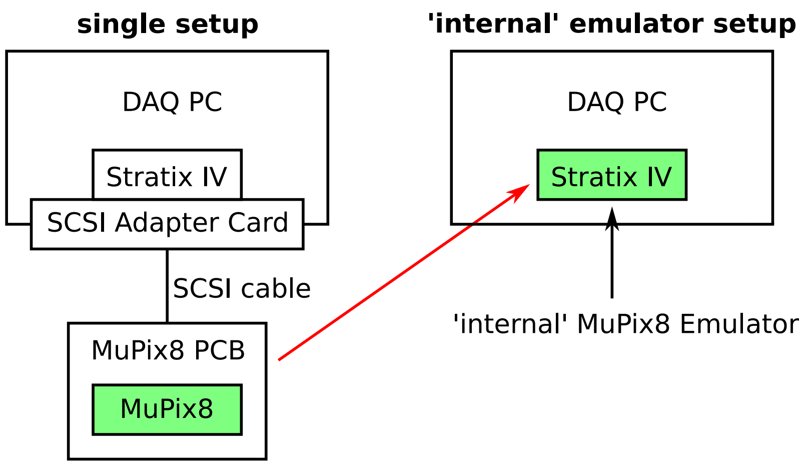


Figure 5.2: Schematic illustrating the concept of the MuPix8 emulator embedded in the DAQ firmware.

5.2.1 Firmare

The 'internal' emulator is embedded in the readout firmware as an entity called **Pseudo Data Generator**. It is located right in front of the *Unpacker* (see section 4.3.6, Figure 4.20) where the data from the emulator is multiplexed with data from the real MuPix8 sensor. Thus, all entities in the data path before the *Unpacker* cannot be tested using this emulator. These are the *Fast GX Transceiver* and the *LVDS Receiver* including the *deserialization* and the *8b/10b decoding*. For them, electrical input signals to the FPGA are required as provided by the 'external' emulator (see section 5.3).

The **Pseudo Data Generator** contains the following subentities, as illustrated in Figure 5.3:

- The *MuPix8 Digital Part* which itself comprises
  - the timestamp generation,
  - the three readout state machines A, B and C,
  - the serializing block which also includes the multiplexing of link D.
- The three *submatrices A*, *B* and *C* in which the hit information is generated.

The serializer in its original form is not needed for the 'internal' emulator. As the data from the *Pseudo Data Generator* is fed into the *Unpacker*, no 8b/10b encoding of the generated data is required. In addition, the transmitters could be removed as no physical output signal from the FPGA is needed. As a consequence, the clocking scheme could be slightly simplified as the fast internal clock related to the actual serializer is not used anymore. However, the serializer block could not be removed completely because for link D the functionality of either sending a copy of the data link A, B or C, or multiplexing the three links should remain.

Hit Patterns

The hit generation happens in the *Submatrix* entities. The generated hit pattern is repetitive and deterministic. Three modes, which differ slightly, can be chosen:

- In the **first** mode, the pattern corresponds to a 'subdiagonal' across each submatrix, i.e. from pixel $(0, 0)$ to $(47, 47)$, from $(48, 0)$ to $(95, 47)$, and from $(96, 0)$ to $(127, 31)$.

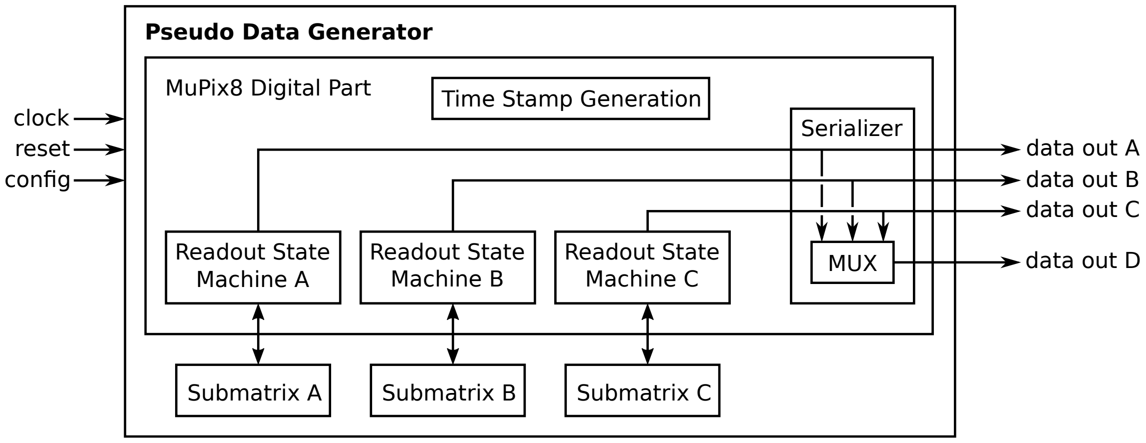


Figure 5.3: Illustration of the **Pseudo Data Generator** and its subentities.

- In the **second** case, a full 'diagonal' across all three submatrices from pixel $(0, 0)$ to $(127, 127)$ is generated.
- On the MuPix8 chip, the digital addresses, which the chip sends out, do not correspond to the physical pixel addresses on the chip. Therefore, the **third** mode – based on the second mode – includes the transformation from physical to digital address which is summarized in Table 5.1. The corresponding reverse transformation from digital back to physical addresses is implemented in the readout software.

Physical Column	Physical Row	Digital Address
$0 - 127$		physical column + 128
	$0 - 83$	physical row + 56
	$84 - 99$	physical row + 156
	$100 - 199$	physical row + 40

Table 5.1: Transformation from the physical position of the pixel to the digital pixel address as sent out by the MuPix8.

For all three cases described above, one can reduce the hit pattern to a fraction of the subdiagonals with a variable number of hits (`numhits`) which is equal for all three submatrices.

In the third mode, for instance, one then gets 3 subdiagonals from pixel (0,0) to (`numhits`,`numhits`) for submatrix A, from (48,48) to (48+`numhits`,48+`numhits`) for submatrix B, and from (96,96) to (96+`numhits`,96+`numhits`) for submatrix C including the address transformation. Examples are shown in section 5.2.3 (see Figures 5.5b and 5.5c).

There are good reasons to keep the hit pattern as simple as possible here: As the emulator is embedded in the DAQ firmware, it is supposed to be small in terms of resource usage of the FPGA. Moreover, it is advantageous to have a simple data pattern which can easily be identified and checked by eye when investigating the data written to memory. In addition, in modes two and three, it can be derived from the hit address which submatrix the hit stems from. This allows to investigate whether the multiplexing of the data from the submatrices A, B and C on link D works properly.

Random or pseudo-random data would make it much harder to cross-check whether all hits of a block are read out correctly or whether a loss of part of the data occurs.

Hit Generation & Readout Procedure

The hit generation works as follows: The number of hits in the end-of-column row (see Figure 5.1) is implemented as a simple counter (`hitcount`) which is decremented when a hit is sent out by the readout state machine. An actual (digital) pixel matrix and an end-of-column row are not required as the hit addresses are deterministic – the hit addresses do not need to be stored in the end-of-column row. They can be derived from `hitcount` in the moment they are sent out.

When the next 'snapshot' is taken, `hitcount` is reset to `numhits` or to 48, 48 and 32 for submatrices A, B and C, respectively, if `numhits` is chosen larger than the number of columns in the corresponding submatrix. This means that hits are always available and no 'idle' time occurs in which the state machine waits for new hits. It also means that submatrix C sees a higher per-pixel rate if `numhits` is chosen larger than 32. The reason is that in this case the procedure repeats after 32 hits for submatrix C compared to a higher number (up to 48) for submatrices A and B.
In order to be able to generate an 'idle' time, is also possible to reduce the hit rate using the configuration value `slowdown`. If `slowdown` is larger than 1, `hitsize` is kept at 0 as long as a counter arrives at `slowdown` $- 1$.

The timestamps corresponding to the hits are assigned in the moment they are sent out. This means, the hits get ordered timestamps unlike in the case of 'real' hits which are read out non-chronologically. Again, this simplification is sensible because it reduces the resource usage on the FPGA as the timestamps are not stored.

Four identical copies of the *Pseudo Data Generator* are implemented such that four MuPix8 chips can be emulated at a time. This allows to test the readout firmware and software with respect to its usability for the telescope setup.

5.2.2 Software & GUI

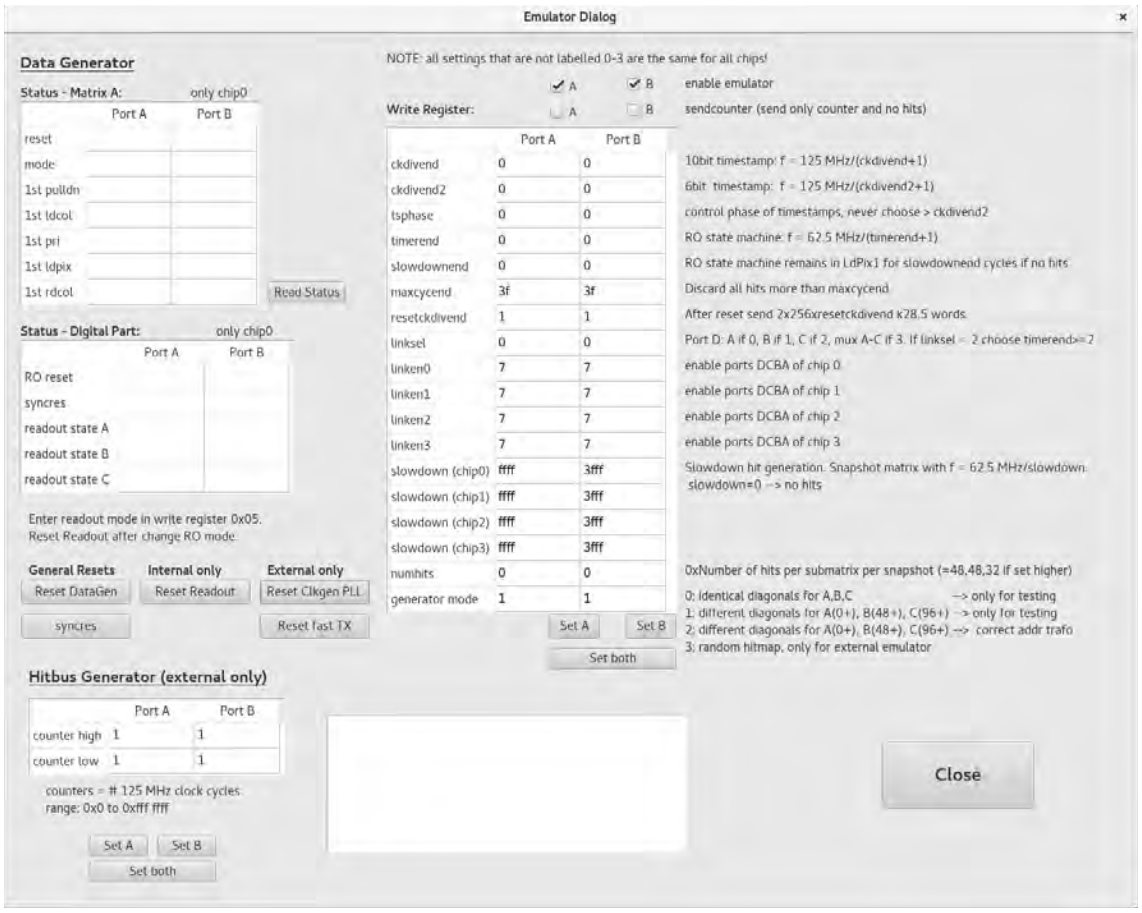
The graphical user interface used for the control of the MuPix8 emulator is integrated in the data acquisition GUI. The **Emulator Dialog** can be opened from the

Single MainWindow or the Telescope MainWindow to run the 'internal' emulator. It can also be opened from the Emulator MainWindow if running the 'external' emulator. A more detailed description of the MainWindows is given in Appendix A.1. The Emulator Dialog is used for the control of both the 'internal' and the 'external' emulator.

The GUI (see Figure 5.4) comprises the control of the *Pseudo Data Generator* as well as a *Hitbus Generator* which is described in 5.3.1. For the 'internal' emulator, only the *Pseudo Data Generator* is available.

Pseudo Data Generator

As the MuPix8 emulator runs on an FPGA, its configuration and control is realized via the same PCIe interface which is used for the communication with the readout



Data Generator

Status - Matrix A: only chip0

	Port A	Port B
reset		
mode		
1st pulldn		
1st ldcol		
1st pri		
1st ldpix		
1st rdcol		

Status - Digital Part: only chip0

	Port A	Port B
RO reset		
syncres		
readout state A		
readout state B		
readout state C		

Enter readout mode in write register 0x05.
Reset Readout after change RO mode.

General Resets **Internal only** **External only**

Hitbus Generator (external only)

	Port A	Port B
counter high	1	1
counter low	1	1

counters = # 125 MHz clock cycles
range: 0x0 to 0xfff ffff

NOTE: all settings that are not labelled 0-3 are the same for all chips!

[✓ A] [✓ B] enable emulator

Write Register: A B sendcounter (send only counter and no hits)

	Port A	Port B	
ckdivend	0	0	10bit timestamp: f = 125 MHz/(ckdivend+1)
ckdivend2	0	0	6bit timestamp: f = 125 MHz/(ckdivend2+1)
tsphase	0	0	control phase of timestamps, never choose > ckdivend2
timerend	0	0	RO state machine: f = 62.5 MHz/(timerend+1)
slowdownend	0	0	RO state machine remains in LdPix1 for slowdownend cycles if no hits
maxcycend	3f	3f	Discard all hits more than maxcycend
resetckdivend	1	1	After reset send 2x256xresetckdivend k28.5 words
linksel	0	0	Port D: A if 0, B if 1, C if 2, mux A-C if 3. If linksel = 2 choose timerend>=2
linken0	7	7	enable ports DCBA of chip 0
linken1	7	7	enable ports DCBA of chip 1
linken2	7	7	enable ports DCBA of chip 2
linken3	7	7	enable ports DCBA of chip 3
slowdown (chip0)	ffff	3fff	Slowdown hit generation. Snapshot matrix with f = 62.5 MHz/slowdown. slowdown=0 --> no hits
slowdown (chip1)	ffff	3fff	
slowdown (chip2)	ffff	3fff	
slowdown (chip3)	ffff	3fff	
numhits	0	0	0xNumber of hits per submatrix per snapshot (=48,48,32 if set higher)
generator mode	1	1	0: identical diagonals for A,B,C --> only for testing 1: different diagonals for A(0+), B(48+), C(96+) --> only for testing 2: different diagonals for A(0+), B(48+), C(96+) --> correct addr trafo 3: random hitmap, only for external emulator

Figure 5.4: The GUI for the control of the emulator.

firmware. To this end, dedicated *read*- and *write*-registers can be accessed. The configuration values described in the following are written to such registers.

As the 'external' emulator can be used on the HSMC ports A and B of a Stratix IV, all configuration values exist twice. For the 'internal' emulator, only the left column (Port A) is of relevance.

The configuration values for the readout state machine correspond to those of the 'real' MuPix8. They are summarized in Appendix A.2. Further emulator specific control values are presented below. They do **not** exist for the 'real' MuPix8 and are merely for the control of the emulator.

- The emulator can be enabled/disabled with the check box `enable`.
- For debugging purposes it can be useful to enable only individual links from individual chips. This can be done using `linken0 - linken3` for the four chips which can simultaneously be emulated with the 'internal' emulator. For the 'external' emulator which can only emulate one MuPix8 per port, only `linken0` is relevant.
- Like the `linken` register values, the `slowdown` exists for each of the four emulated chips. When set to 0, no hits are generated. Otherwise, 'snapshots' of the matrix are performed at a rate of $f = 62.5\,\text{MHz}/\texttt{slowdown}$.
- `numhits` corresponds to the number of hits per submatrix when generating a 'diagonal' hit pattern.
- Again, for debugging purposes, it is useful to be able to switch between the different data patterns explained above using `generator mode`.

In the top left corner of the GUI (see Figure 5.4), the table '*Status - Matrix A*' displays status information of the entity corresponding to submatrix A on the first chip. It allows to check whether the reset signal is enabled and which mode is selected for the hit generation. In addition, a number of flags indicate whether the state machine has arrived at a certain state for the first time. In the process of debugging, this feature was particularly helpful to detect in which state the state machine got stuck.

The table '*Status - Digital Part*' (see Figure 5.4) displays the status of the digital part corresponding to the first chip. It shows the readout reset signal as well as the synchronous reset signal. It also displays in which state the readout state machine is at the moment of pushing the button `Read Status`.

Further buttons allow to quickly reset various entities of the emulator and the DAQ, respectively. They comprise:

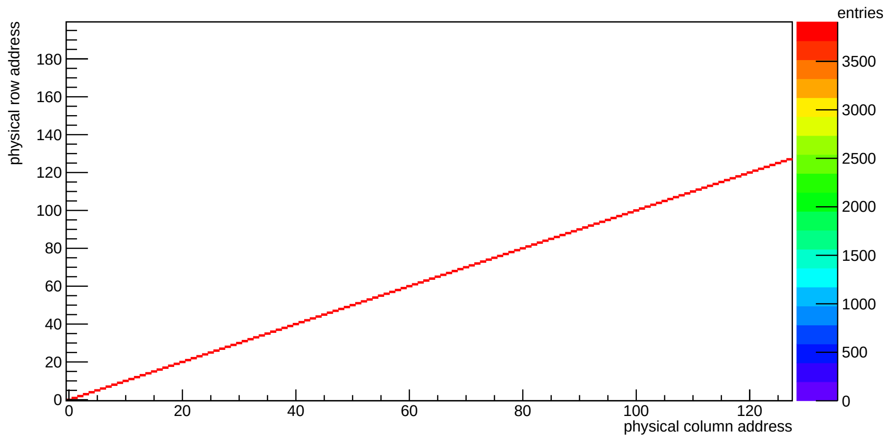
- `Reset DataGen` to reset the *Pseudo Data Generator*
- `syncres` to synchronously reset the timestamp generating entity in the digital parts of the four emulated MuPix8 chips
- `Reset Readout` to reset the readout entities in the data path after changing the readout mode (only 'internal' emulator)
- `Reset Clkgen PLL` to reset the PLL to which the input clock is fed in case of the 'external' emulator
- `Reset fast TX` to reset the *Fast GX Transceivers* (only 'external' emulator)

5.2.3 Measurements

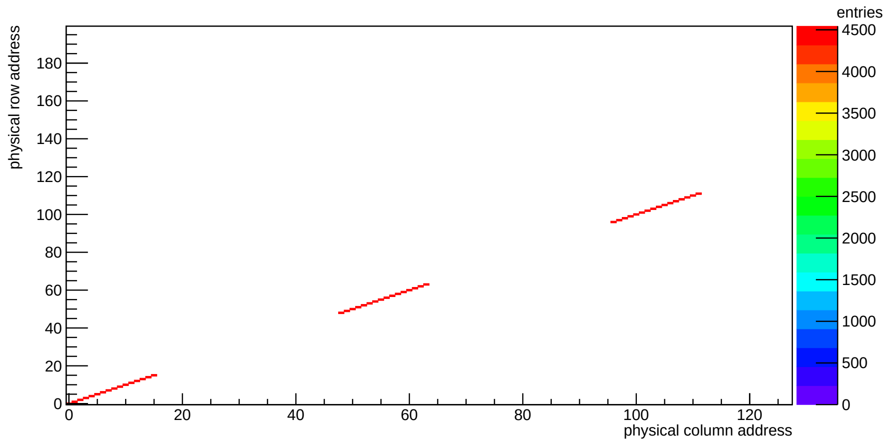
Hitmaps

In Figure 5.5, hitmaps are shown which have been generated with the 'internal' emulator using `generatormode` = 3. Therefore, the transformation between digital and physical addresses is included. The software used for the processing and storage of the data is exactly the same used on DESY testbeams.

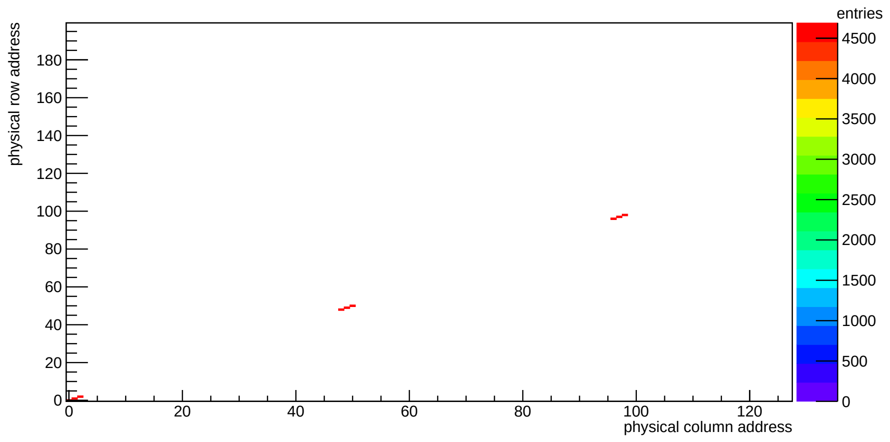
The shown hitmaps prove the functionality of the *data path* the readout firmware. No data is lost, it is written to the memory of the DAQ PC and processed correctly. In addition, it could be shown that the online monitoring is functional, i.e. during the run time, a 'real-time' hitmap is displayed in the GUI.



(a) Full diagonal across all three sub-matrices.



(b) 16 hits per submatrix.



(c) 3 hits per submatrix.

Figure 5.5: Hitmaps with different numbers of hits per 'subdiagonal', generated with the 'internal' emulator. The difference in the number of entries stems from different run times.

5.3 The External Emulator

The 'external' emulator is suitable to test the entire data acquisition chain from the MuPix8 PCB over the SCSI cable to the SCSI Adapter Card, the readout firmware, and the DAQ software as illustrated in Figure 5.6. Being derived from the 'internal' emulator, the 'external' emulator is also based on the original Verilog HDL code which has been synthesized for the digital part of the MuPix8.

A major part of the implementation of the 'external' emulator was a modification of the *submatrix* entity to allow for a generation of pseudo-random hits. In addition, the 8b/10b encoding as well as the transceiver blocks had to be re-implemented.

Like the readout firmware, it is running on a Stratix IV which, in contrast, is mounted in a separate PC. For maximal synergy, the control GUI for the 'external' emulator is implemented in the same software framework running on the DAQ PC. A dedicated Emulator MainWindow (see Appendix A.1) allows to open the *Emulator Dialog* as described in section 5.2.2 for the 'internal' emulator.

As described in section 4.2.3, the setup comprises the full hardware required for the MuPix8 single setup except for the MuPix8 Insert which needs to be replaced

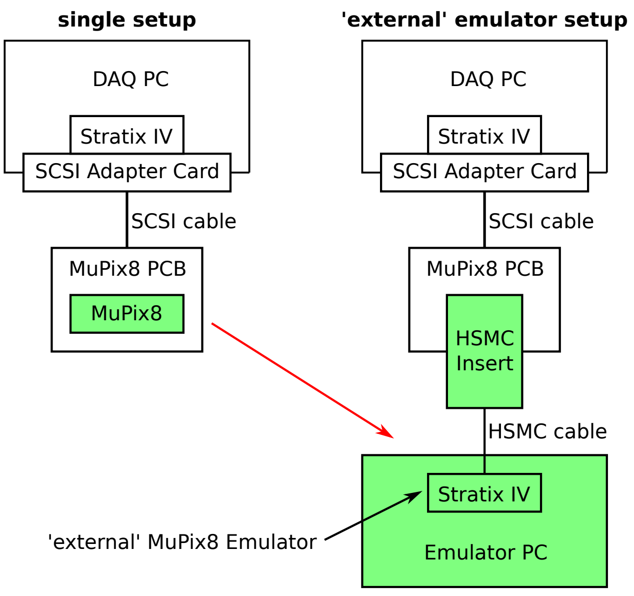


Figure 5.6: A schematic illustrating the concept of the MuPix8 emulator running on a separate FPGA.

by a dedicated adapter insert – the **MuPix8 HSMC Insert**. This connects to a separate FPGA. The FPGA is mounted in a PC which is in turn used for the control of the 'external' emulator via a GUI.

5.3.1 The Firmware

The firmware has been derived from the 'internal' emulator as explained above (see section 5.2). The *MuPix8 Digital Part* and the *Submatrix* entities are in principle identical to the 'internal' emulator. However, there is now the possibility to generate pseudo-random hits in addition to simplistic diagonal hit patterns. This pseudo-random hit generation is based on a *pseudo-random binary sequence* (PRBS) as explained below.

As a basis, the MuPix8 readout firmware has been used for the 'external' emulator. In particular, the *PCIe block*, responsible for the communication with the PC via *read*- and *write*-registers has been kept. Only the use of the registers needed to be adapted to the needs of the 'external' emulator.

On the other hand, all readout-related entities were removed and the *data path* was restructured as illustrated in Figure 5.3. Because the FPGA now has to transmit signals via physical output pins, 8b/10b encoding as well as serialization are required.

For the 8b/10b encoding, a freely available VHDL implementation [44] has been integrated behind the *Pseudo Data Generator*. This solution has been chosen instead of using the original MuPix8 Verilog code such that the exact *MuPix8 Digital Part* from the 'internal' emulator could be used.

Firmware needs to be designed in consideration of which building blocks are available on the type of FPGA at hand. Because the type of serializer implemented in the MuPix8 is not available on the Stratix IV, two alternative solutions have been realized in parallel. Like in the readout firmware, a *Fast GX Transceiver Block* and an *LVDS Transmitter Block* are implemented. Both include the serialization and generate electrical output signals which are available on the HSMC ports A and B to be transmitted to the MuPix8 HSMC Insert via a HSMC-to-HSMC cable. On the insert, solder jumpers can be set to select the data from one of the transmitters. The *Fast GX Transceivers* have been used for all measurements presented below. The *LVDS Transmitters* have been implemented as a backup option.

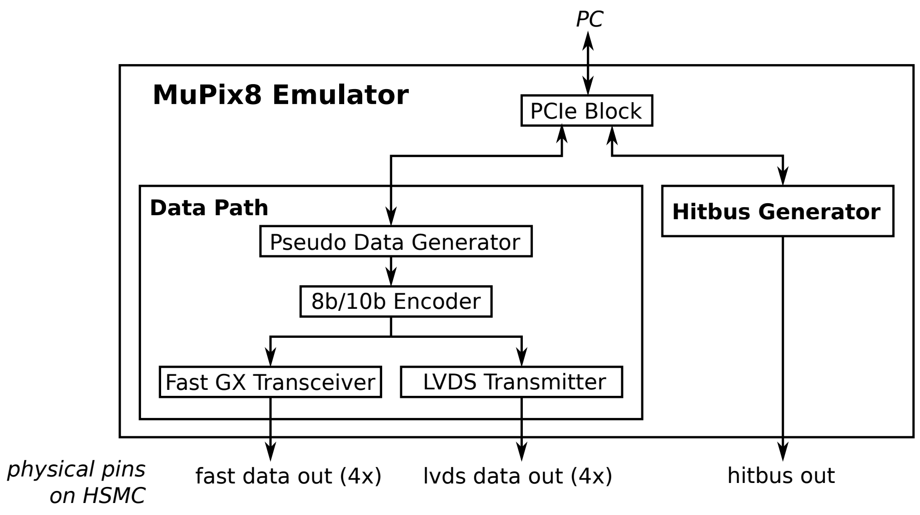


Figure 5.7: Illustration of the **MuPix8 Emulator** and its subentities.

In addition, a *Hitbus Generator* as explained below has been added.

Pseudo-Random Hit Generation & Readout Procedure

A *pseudo-random binary sequence* (PRBS) is a bit sequence which is generated with a deterministic algorithm but is very similar to a truly random sequence on a short time scale. In hardware, it is typically implemented using a *linear feedback shift register* (LFSR) [45].

For the *Pseudo Data Generator*, a so-called **PRBS31** has been employed. A schematic drawing is shown in Figure 5.8. The PRBS31 is realized as a LFSR with a length of 31 bits. Every clock cycle, the result of a logic '`xnor`' operation between bit 30 and bit 27 is fed back into the zeroth bit of the chain. As the starting condition, which can be chosen arbitrarily, all bits are set to '0'. The bit pattern repeats after $2^{31} \approx 2.15 \times 10^9$ steps which corresponds to $\sim$34 s at a clock frequency of 62.5 MHz.

The generation of pseudo-random hits works as follows: Every clock cycle, a 3-bit section of the PRBS is compared to a fixed pattern. This condition ensures that a hit is not generated every clock cycle and the time between two consecutive hits varies. The length of 3 bits as well as the pattern are chosen arbitrarily. A longer pattern would reduce the hit generation rate as matches happen less frequent. The

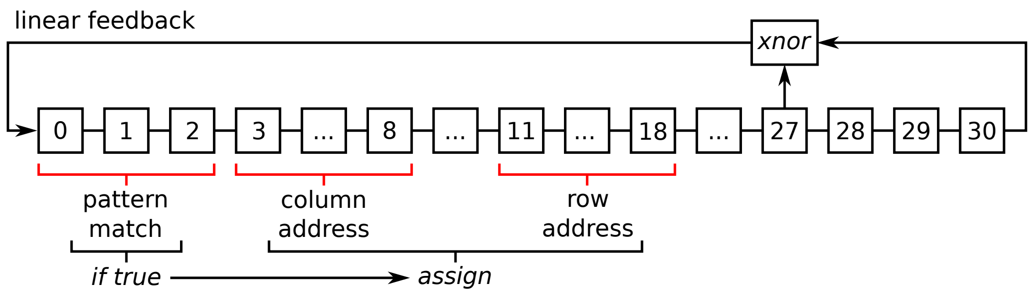


Figure 5.8: Schematic drawing the pseudo-random hit generation using a **PRBS31** implementation in the form of a LFSR.

pattern does not matter as all combinations are equally likely.

If the chosen 3-bit section of the PRBS matches a fixed pattern, another 6-bit section is used as the column address and yet another 8-bit section as the row address of the hit as illustrated in Figure 5.8. In addition, the timestamp received from the *Digital Part* in the clock cycle of the pattern match is assigned to the hit.

6 bits correspond to numbers between 0 and 63 which suffices for the column address space of one submatrix. The correct 'offset' of 48 and 96 for submatrices B and C is added later. The 8 bits for the row allow values between 0 and 255. Consequently, a 'valid' hit is only generated when the column and row addresses lie within the range of the respective submatrix. If the column address is larger than 47 (for submatrices A and B) or 31 (for submatrix C), or the row address is larger than 199, the hit is discarded.

With this strategy, hits are generated which look random on a short time scale but are repetitive after ∼34 s. Like for the diagonal hit pattern, the hit generation can be slowed down using the configuration value `slowdown`. If `slowdown` = 0, no hits are generated. Otherwise, the pattern matching is performed at a rate of 62.5 MHz/`slowdown`.

When a pattern match occurs, the corresponding hit information needs to be stored until the hit is loaded into the end-of-column row to be sent out. This problem is solved by the use of *Random Access Memory* (RAM) blocks. As depicted in Figure 5.9, one RAM block with a depth of 16 bits and 200 memory addresses is used per column. The 10-bit and the 6-bit timestamp of a hit is be stored at the memory address corresponding to the row address of the hit. In addition, a hit flag is stored in a 48×200 (submatrices A, B) or a 32×200 matrix (submatrix C), to keep an information about which RAM address contains valid hit information. The

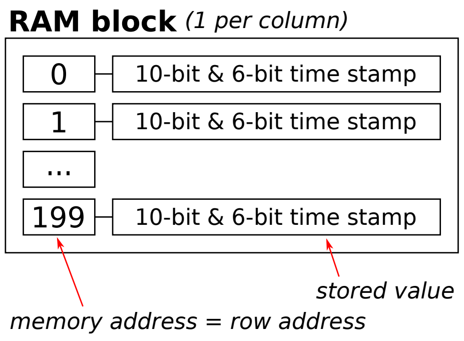

RAM block (1 per column)
0
10-bit & 6-bit time stamp
1
10-bit & 6-bit time stamp
...
199
10-bit & 6-bit time stamp
stored value
memory address = row address

Figure 5.9: Schematic drawing of one RAM block used to store the 10-bit and 6-bit timestamps of a hit at the memory address corresponding to the row address of the hit.

hit 'pixel' is now occupied until read out. If a hit with the same address occurs before the hit has been read out, it is **not** overwritten.

When a matrix 'snapshot' is requested from the readout state machine, a readout priority logic chain is triggered to check each column for a hit. If at least one hit is found, the one with the lowest row address is pulled into the end-of-column row and the corresponding flag in the hit flag matrix mentioned above is reset, i.e. the 'pixel' is active again and the RAM cell can be written again. These hits are sent out consecutively as described in section 5.1 (see Figure 5.1). When the end-of-column row is empty (or 63 hits have been sent out), the readout procedure starts again.

This implementation of the pseudo-random hit generation is very resource intensive. For this reason, it is not possible to fit more than three submatrices into one Stratix IV. Consequently, two firmware versions have been developed. One contains the pseudo-random hit generation but only one MuPix8 can be emulated at a time, i.e. only HSMC port A is used. In the second version, only the three 'diagonal' hit patterns as in the 'internal' emulator are available. In this case, two MuPix8 chips can be emulated on one FPGA making use of HSMC ports A and B.

The Hitbus Generator

The **Hitbus Generator** can be used to generate a hitbus signal. It is considerably simple and therefore a good tool for a first test of the electrical connectivity between the Stratix IV and the MuPix8 PCB. The output signal stays high until a counter with an incrementation frequency of 125 MHz reaches the value of `counter high`.

Then the signal is low for `counter low` clock cycles. Therefore, a repetitive hitbus signal with an adjustable repetition rate and duration is generated. The signal can be investigated with an oscilloscope when probed at the *LEMO* connector on the MuPix8 HSMC Insert.

5.3.2 Measurements with the MuPix8 Emulator Setup

In the following, the various measurements, which have been performed using the 'external' MuPix8 emulator and the MuPix8 HSMC Insert, are presented.

Hitbus

On the MuPix8 chip, the hitbus signal corresponds to a simple time-over-threshold for a chosen pixel column. It is wired through to the readout firmware but can also be probed at the MuPix8 PCB.

As a first test, a hitbus signal has been generated with the 'external' emulator and visualized with an oscilloscope. The signal (see Figure 5.10) has a high time of 2 µs and a low time of 8 µs and is repetitive. The measurement proves that the electrical connectivity between the Stratix IV running 'external' emulator and the MuPix8 PCB is established. It also shows that the 'external' emulator can be configured via the GUI.

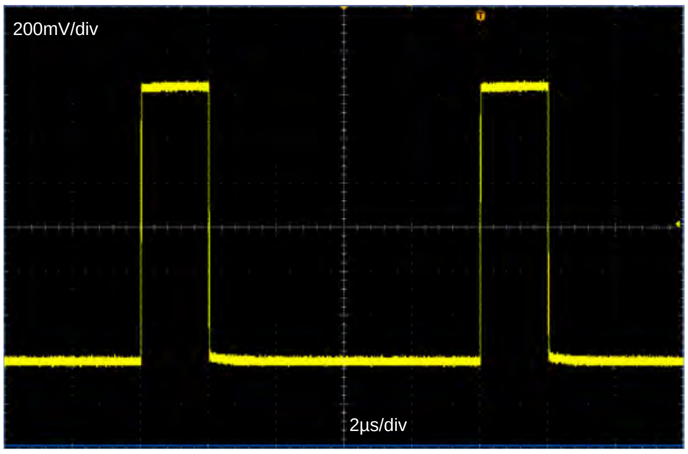


Figure 5.10: Screenshot of an oscilloscope attached to the hitbus *LEMO* on the MuPix8 PCB. A high time of 2 µs and a low time of 8 µs have been chosen as an example.

The signal could also have been used to further test the readout chain, but efforts have been focussed on the hitmap instead (see below).

Hitmaps

Figure 5.11 shows three exemplary hitmaps which have been created from the data generated by the 'external' MuPix8 emulator, sent through the entire readout chain, processed by the readout software and then saved to file as would be done for 'real' data.

The first hitmap (Figure 5.11a) shows a simple diagonal as presented in the previous section for the 'internal' emulator. It proves that the electrical connection between the MuPix8 PCB and the data acquisition PC is functional. In addition, it shows that the firmware entities which could not be tested with the 'internal' emulator – the *receiver blocks* including the deserialisation and 8b/10b decoding and the synchronisation *FIFOs* – work properly.

The hitmaps depicted in Figures 5.11b and 5.11c show pseudo-random hit patterns at different hit rates. The generated hit rate can be calculated as shown in the following. The fixed bit pattern is compared to a section of the PRBS31 with a frequency of 62.5 MHz/`slowdown`, a value which is set in the emulator GUI. As explained above, the pattern to be matches comprises 3 bits, such that there is a chance of $(\frac{1}{2})^2 = \frac{1}{8}$ for a match. In addition, the generated 6-bit column and 8-bit row address needs to lie within the address space of the corresponding submatrix, i.e. the row is required to be < 200 and the column in an interval of 48 values for submatrices A and B, and an interval of 32 values for submatrix C. Consequently, the expected hit rate is given by

$$f_{\text{hits,AB}} = \frac{1}{8} \times \frac{62.5\,\text{MHz}}{\texttt{slowdown}} \times \frac{200}{256} \times \frac{48}{64} = \frac{4.6\,\text{MHz}}{\texttt{slowdown}} \tag{5.1}$$

and

$$f_{\text{hits,C}} = \frac{1}{8} \times \frac{62.5\,\text{MHz}}{\texttt{slowdown}} \times \frac{200}{256} \times \frac{32}{64} = \frac{3.1\,\text{MHz}}{\texttt{slowdown}}. \tag{5.2}$$

For the whole chip this adds up to

$$f_{\text{hits, chip}} = 2 \times f_{\text{hits,AB}} + f_{\text{hits,C}} = \frac{12.3\,\text{MHz}}{\texttt{slowdown}}. \tag{5.3}$$

Note that the per-pixel rate is identical for all submatrices as the different number of columns is cancelled out in equations 5.1 and 5.2. Consequently, a homogeneous distribution across the entire pixel matrix is expected.

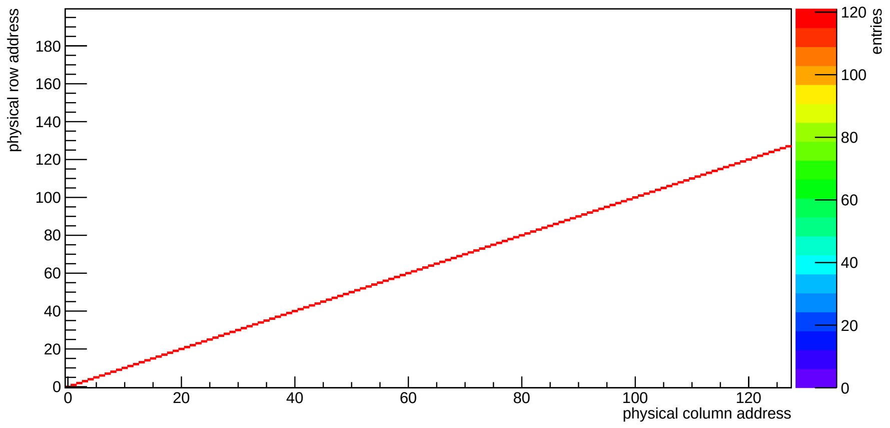
Indeed, Figure 5.11b shows a homogeneous distribution across the chip. The occupancy lies well within the limits of the capability of the MuPix8 readout state machine.

On the contrary, in Figure 5.11c a clear inhomogeneity is visible. Towards the upper edge of the chip, i.e. far away from the digital periphery, the hit number is strongly reduced. This effect can be explained by the non-chronological readout scheme which was described in section 5.1. As described above, in every column the hit with the lowest row address is pulled into the end-of-column row, even if it has a much later timestamp compared to hits with higher row addresses. Consequently, for a very high hit rate, the hits in the upper rows are less likely (in an extreme case never) read out because the maximal 63 hits, which can be transmitted consecutively after one 'snapshot', are always taken from the lower rows which are read out first.

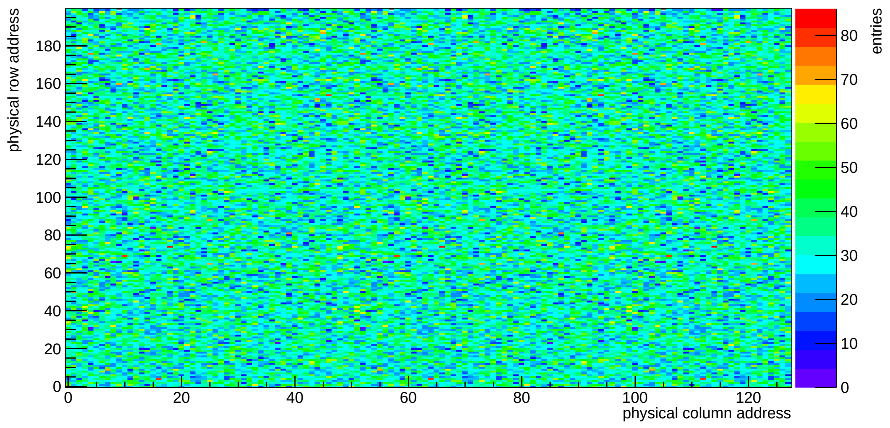
In the figure, it is also apparent, that the submatrix C sees a higher hit rate than submatricess A and B. The reason is that the end-of-column rows for submatrices A and B have a length of 48 compared to 32 for submatrix C. Consequently, up to 48 and 32 hits, respectively, can be pulled into the end-of-column row at a time. The readout state machines then send out hit by hit until the end-of-column row is empty and a load of hits is pulled down. It is evident that 48 hits for the submatrices A and B take longer to be sent out than the 32 for submatrix C, i.e. the procedure can be repeated more frequently for submatrix C until the maximum of 63 hits is reached. Consequently, submatrix C has a higher per-pixel rate capability. The rate capability of all submatrices decreases if the readout state machine is run at a reduced speed.

In separate tests, the links A-D have been enabled one by one. The corresponding hitmaps look as expected. If using only A, B or C, only one of the submatrices is active as shown in Figure 5.12. Also port D has shown to be functional. For `linksel` $= 0, 1, 2$, copies of the links A, B or C, respectively, are transmitted such that the hitmaps are only filled on one submatrix. The multiplexing of data from submatrices A-C (`linksel` $= 3$) has not been investigated yet.

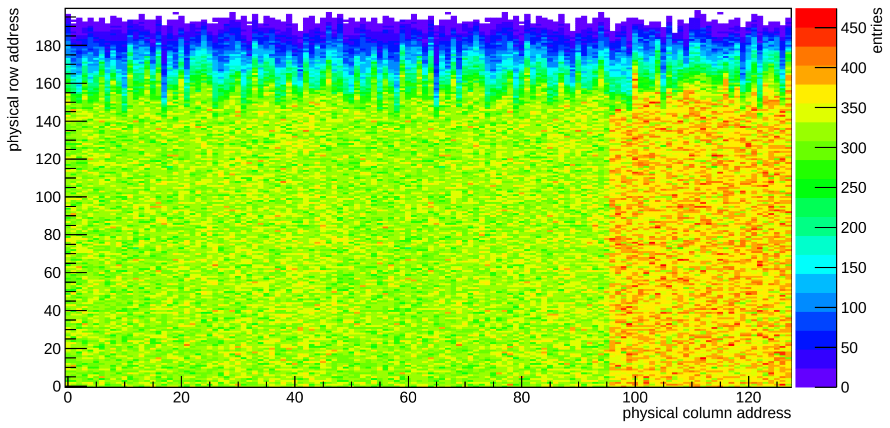
It can be concluded that the electrical connectivity of the entire readout chain from the MuPix8 PCB all the way through to the data acquisition PC is functional for all four links. It has also been shown that the readout system can handle a non-chronological hit order. A hit rate of 12.3 MHz exceeds the capability of the MuPix8 emulator and the same can be expected for the MuPix8 if proving an external clock with a frequency of 125 MHz and running the state machine at the full readout speed of 62.5 MHz.



(a) Hitmap with a full diagonal across all three sub-matrices.

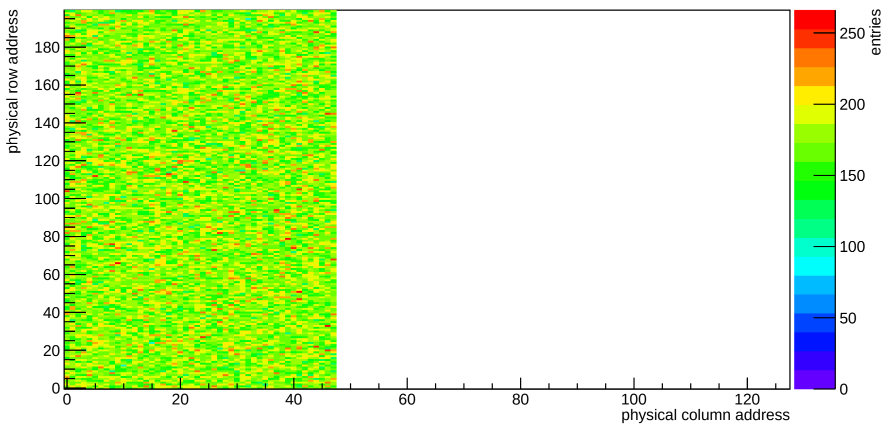


(b) Hitmap with pseudo-random hits at a rate of ∼820 kHz.

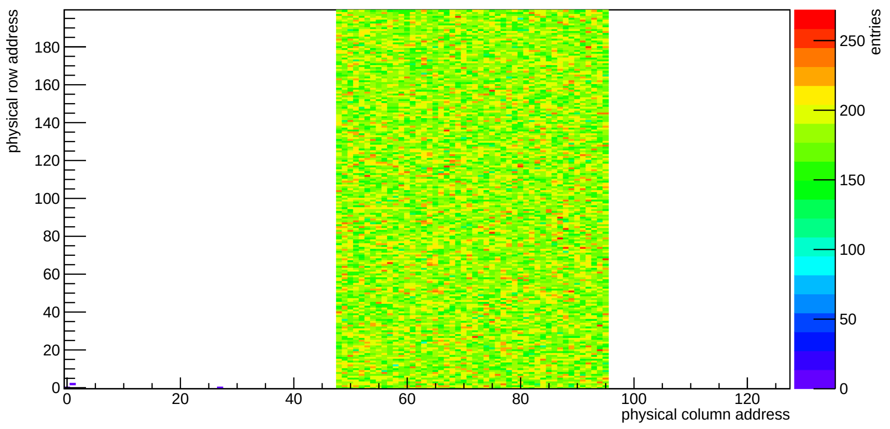


(c) Hitmap with pseudo-random hits at a rate of ∼12.3 MHz. The hit rate exceeds the rate capability of the MuPix8 (emulator).

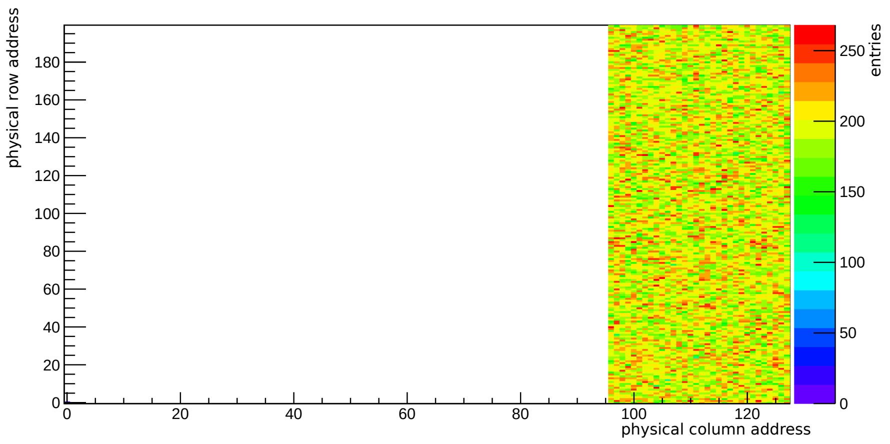
Figure 5.11: Hitmaps with a diagonal hit structure and a pseudo-random hit distribution.



(a) Port A.



(b) Port B.



(c) Port C.

Figure 5.12: Hitmaps for individual links with pseudo-random hits at a hit rate of ∼400 kHz.

Temperature Diode

To prove the basic functionality of the temperature measurement circuitry, automated scans over the whole range of possible temperature DAC values have been performed, i.e. different currents have been injected into the temperature diode on the MuPix8 HSMC Insert and the corresponding voltage drop across the diode has been measured. The result is shown in Figure 5.13.

This measurement **does not** involve any firmware related to the MuPix8 emulator. However, it makes use of the 'external' MuPix8 emulator setup as the temperature diode on the MuPix8 HSMC Insert is required for the measurement.

The first scan (*red* data points) has been performed at room temperature, i.e. T ≈ 20 °C. For the second scan (*blue* data points), the temperature diode on the MuPix8 HSMC Insert has been heated with a soldering iron at the lowest possible temperature of T ≈ 100 °C for about 1 min. As the heat flow from the diode onto the PCB cannot be quantified, the temperature of the diode is difficult to estimate. However, it can be assumed to be warmer than during the first measurement such that the result can be evaluated qualitatively.

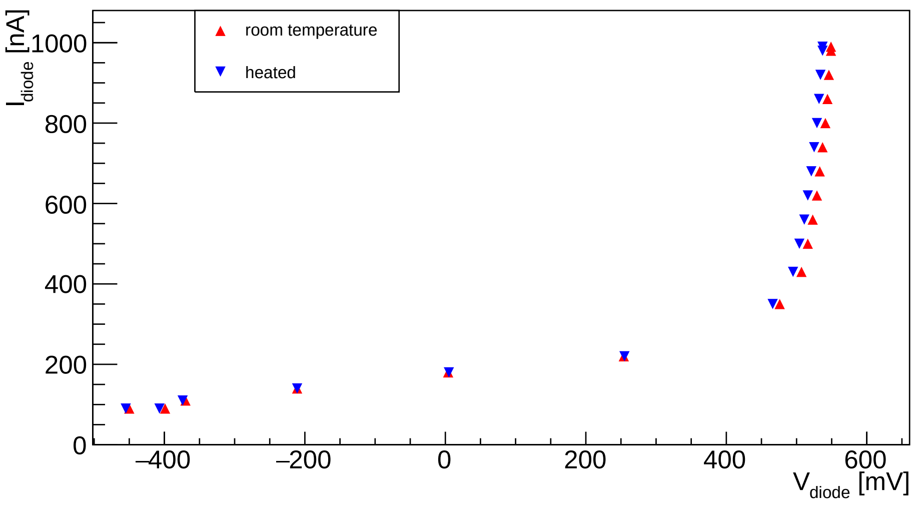


Figure 5.13: IV scans with the MuPix8 HSMC Insert for two temperatures. The current has been generated by a dedicated circuitry on the MuPix8 PCB and pushed through the temperature diode on the MuPix8. The corresponding voltage drop across the diode is measured by an ADC.

As can be seen, the curves show the typical exponential behaviour of a diode as described by the Shockley equation [46]:

$$I = I_\mathrm{S}\left(e^{eV_\mathrm{D}/k_\mathrm{B}T} - 1\right) \tag{5.4}$$

where I is the current through the diode, I_S the reverse bias saturation current, V_D the voltage dropping across the diode, k_B the Boltzmann constant, T the temperature, and e the elementary charge.

In contradiction to the Shockley equation, it can be observed that negative voltages are measured for positive injected currents. This 'offset' presumably stems from a voltage which drops across the diode even if no current is generated by the temperature diode circuitry on the MuPix8 PCB. However, its exact origin has not been investigated because it is irrelevant for all further steps.

The important result is the following: The IV curve shows a temperature dependence in that the current rises earlier for higher temperatures (*blue* data points). This measurement does not suffice for an absolute temperature determination of the diode. It is a qualitative demonstration that the temperature circuitry on the MuPix8 PCB is functional. A calibration yielding an absolute temperature was performed with the actual MuPix8 chip and is presented in section 6.6.

6 Commissioning the MuPix8

In this chapter, measurements which have been performed while taking the MuPix8 chip into operation are presented.

6.1 Configuring the MuPix8

To test the functionality of the configuration registers of the MuPix8, a number of tests has been performed. Before the results from these tests are presented, a description of the working principle of the linear shift register as implemented in the MuPix8 is provided.

The configuration register of the MuPix8 has a length of 2998 bits. A schematic drawing of one elementary building block of the LSR is shown in Figure 6.1. It is designed following a two-clock approach. From the DAQ PC, the MuPix8 receives a serial bit stream and the two clock signals `clock1` and `clock2` as well as the `readback` and the `load` signal which are explained in the following.

In the first step, the bit value which is sent from the DAQ PC (or from the previous cell in the chain) gets fed into the 'first' latch if `clock1` = '1' (and `readback` = '0').
Once `clock2` = '1', it gets forwarded to the 'second' latch. If another `clock1` signal arrives now, the data bit leaves the 'second' latch and gets forwarded to the next block which looks identical to the one shown in the sketch. After the last latch in the chain, the signal 'drops out' of the LSR and is fed back to the DAQ PC.
If a `load` signal arrives instead, the bit is transferred into the 'upper' latch where it takes effect as a DAC configuration value. The value from the 'upper' latch can be read back. To this end, the `readback` needs to be '1' while `clock1` is high so that the value from the 'upper' latch is fed back into the 'first' latch instead of the value from the previous cell.

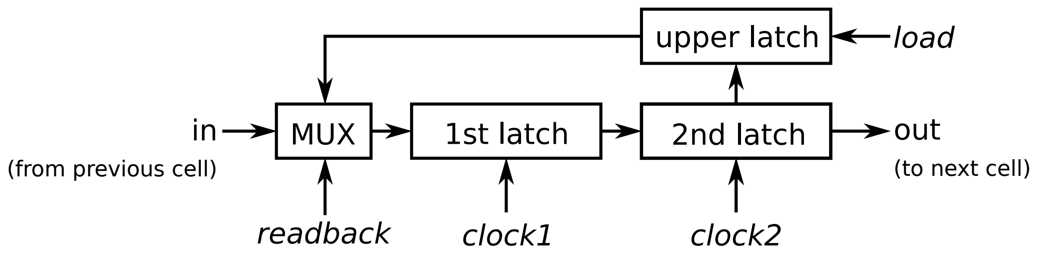


Figure 6.1: Schematic drawing of one elementary building block of the **linear shift register**.

Length Test

As a first test, the length of 2998 bits had to be validated to make sure that the configuration files built in the software fit the size of the MuPix8 register.

To this end, a large number ($\gg 2998$) of 'zeros' has been pushed into the LSR to ensure that it is filled with 'zeroes' only. Afterwards 2998 'ones' have been pushed into the LSR. Then the bits 'dropping out' of the LSR as decribed in section 4.1 have been written to file while pushing 'zeros' into the LSR. If the number of bits is correct, 2998 'ones' should 'drop out' of the LSR before the first 'zero'. The length of 2998 bits could be confirmed.

Readback Test

The length test described above only allows to test the chain of 'first' and 'second' latches (see Figure 6.1). In another test, the functionality of the 'upper' latch was examined.

For this purpose, a certain bit pattern with the correct length of 2998 bits has been pushed into the LSR. Afterwards these bits have been loaded into the 'upper' latches by applying a `load` signal. To test whether the correct bits have been written to the 'upper' latches, a `readback` signal was sent such that the values from all 'upper' latches are fed back into the 'first' latch of the LSR. In the next step, all these values have been pushed out of the LSR by pushing in 2998 'zeros' and written to file. Thus, the bit pattern could be compared to the pattern that has been pushed into the LSR in the first step.

Indeed, the two bit patterns matched. However, the pattern read back was inverted compared to the first pattern. The reason is that the 'upper' latches have two outputs. The non-inverting output is connected to those parts of the chip where the DAC values actually take effect. The inverting output is connected to the MUX for the readback functionality.

It can concluded that the 'upper' latches can be written and the configuration bits can be read back correctly.

Linearity Test of On-Chip DACs

As described in section 4.1, threshold and baseline voltages can be generated by on-chip DACs using the configuration register 6. The voltages can be measured with a voltage probe on the MuPix8 Insert.

The measurement of the voltage **ThHigh** is shown below. It corresponds to the global threshold of the comparator which sets the hit flag in the digital pixel cell. The on-chip DAC has a range from 0 to 1.8 V. For the measurement, the corresponding 10-bit DAC value has been increased bit-wise, i.e. values of 0, 1, 2, 4, 8 up to 1023 have been set and ThHigh has been probed on a testpoint on the MuPix8 Insert. The result is shown in Figure 6.2. In addition, a linear fit has been performed. The errors on the voltage measurement have been calculated according to the handbook [47]:

$$\Delta V = \pm(0.5\,\% + 8\,\text{mV}) \text{ for } |V| \leq 2\,\text{V}. \tag{6.1}$$

As can be seen, the fit yields

$$\text{ThHigh} = p0 + p1 \times \text{DAC} = 45.7\,\text{mV} + 1.7\,\text{mV} \times \text{DAC} \tag{6.2}$$

with a high precision. The very low value of $\chi^2/ndf \approx 0.05$ indicates that the errors have been overestimated. This can be explained by the fact that all measurement points have been recorded using the ± 2 V range of the voltmeter.

A minimal voltage of 45 mV and a maximal voltage of 1778 mV have been reached. However, thresholds in the operating range of roughly 500−900 mV will be used when operating the chip as the comparator baseline is set at ∼500 mV and pulses are superimposed onto this baseline with an amplitude of up to ∼400 mV (see section 6.2). In this range, the DAC has a linear behaviour with a high precision.

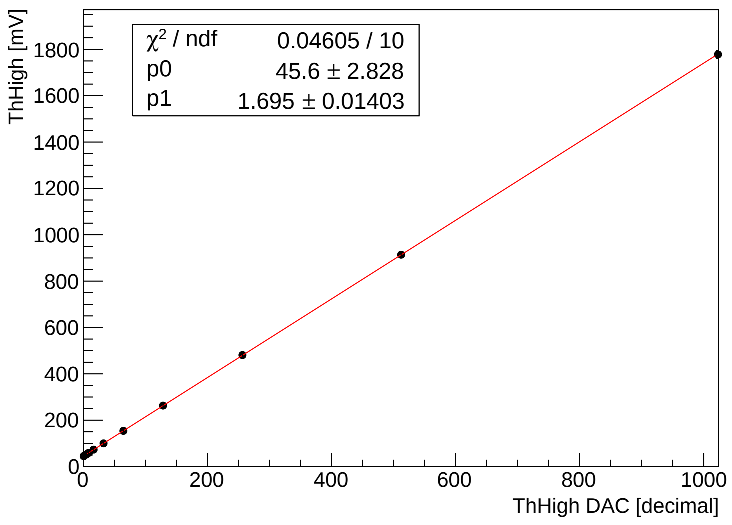


Figure 6.2: Linear fit of the voltage **ThHigh** vs. its corresponding DAC value as generated by the on-chip DAC.

Further Configuration Tests

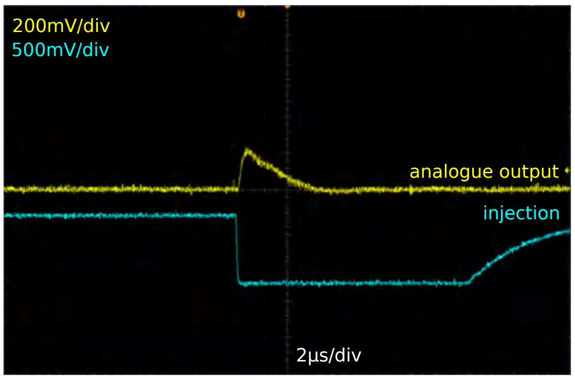
Further configuration tests include the enabling of the **analogue amplifier output** signal from the CSA in the pixel for a selected row (see section 6.2) and the **hitbus** signal for a column of choice (see section 6.3). As these have been performed in combination with dedicated measurements of the corresponding signals, they are addressed in separate sections below.

6.2 Amplifier Output

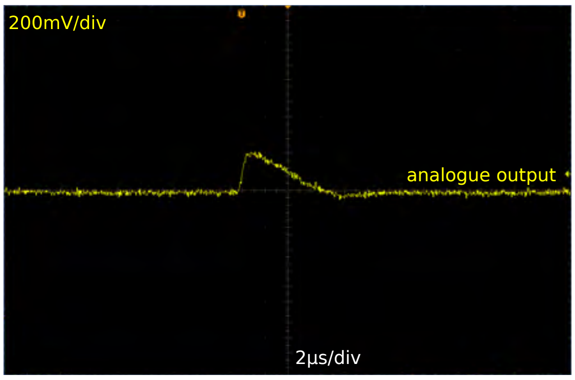
The MuPix8 chip allows to probe the analogue output of the amplifier for a pixel of choice in the leftmost column. It can be selected by setting an enable bit in the row register. At a testpoint on the MuPix8 Insert, the signal can be contacted with a voltage probe attached to an oscilloscope.

In Figure 6.3a, the analogue signal is shown for a testpulse injection. Figures 6.3b-6.3d show the corresponding signals for a strontium-90 (Sr-90) source. As can be seen, the signals reach different amplitudes and lengths.

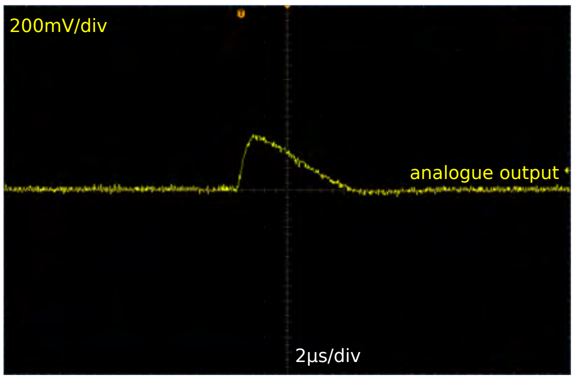
A look at the β-spectrum of Sr-90 (see Figure 6.4) helps to understand the different



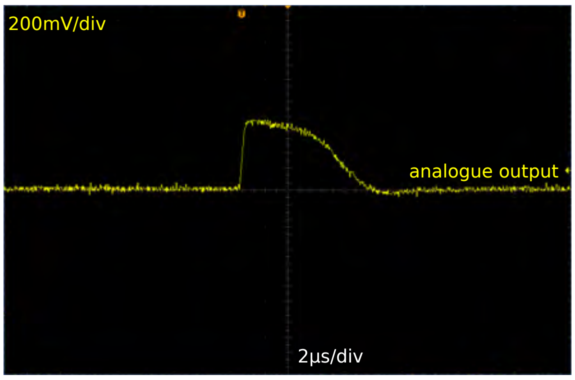
(a) Amplifier output for testpulse injection.



(b) Example 1 for the amplifier output for Sr-90 source.



(c) Example 2 for the amplifier output for Sr-90 source.



(d) Example 3 for the amplifier output for Sr-90 source.

Figure 6.3: Oscilloscope screenshots showing the analogue amplifier outputs for a testpulse injection and three different signal of a Sr-90 source.

pulse heights. Sr-90 has a half-life of 28.5 years. Being a β-source, it emits electrons with an energy up to 546 keV. The daughter product of the decay – yttrium-90 (Y-90) – it also a β-source and has a comparatively short half-life of 64.1 hours and an end-point energy of 2274 keV. Consequently, its spectrum is superimposed with that of the Sr-90 [48].

As the electrons from the β-decays have a broad energy spectrum, they deposit a varying amount of energy in the sensor such that the analogue amplifier output reaches different amplitudes. Figure 6.3d also shows that the signal starts to saturate for pulses above a certain height.

In another measurement, the amplitude of the analogue amplifier output signal has been histogrammed using the same Sr-90 source (see Figure 6.5). The number of entries per bin is plotted against the amplitude of the pulse.

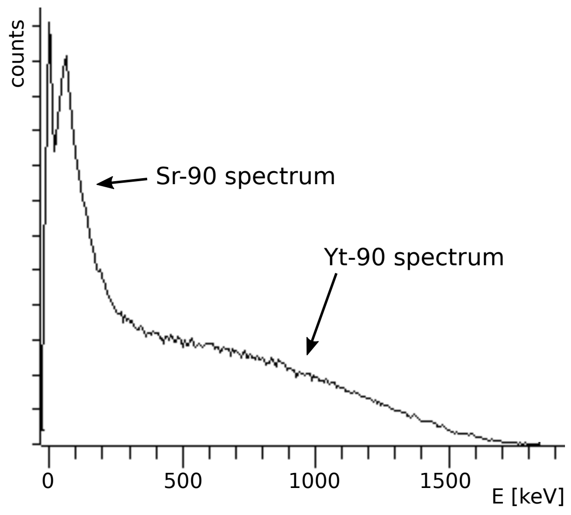


Figure 6.4: β-spectrum of Sr-90, superimposed by the β-decay of the daughter product Y-90, measured with a scintillation counter. From [49], modified.

The sharp cut-off at ∼0.14 V appears due to the chosen trigger threshold of the oscilloscope. The high peak on the left and its falling edge to the right corresponds to the 'low-energy' peak of the β-spectrum of the Sr-90 source as shown in Figure 6.4.

As observed in Figure 6.3d, the analogue amplifier output begins to saturate for pulses above a certain height. Thus, the peak to the right in Figure 6.5 can be understood as an integral over the 'high-energy' tail of the spectrum as the amplifier saturates for signals in this range.

The measurements presented above demonstrate that the MuPix8 chip generates proper analogue pulses. The CSA amplifies these pulses and saturates at amplitudes of ∼400 mV. They also show that the signals can be fed out of the chip to be investigated with an oscilloscope. This feature can be used in the future to study and optimise the configuration of the analogue pixel electronics.

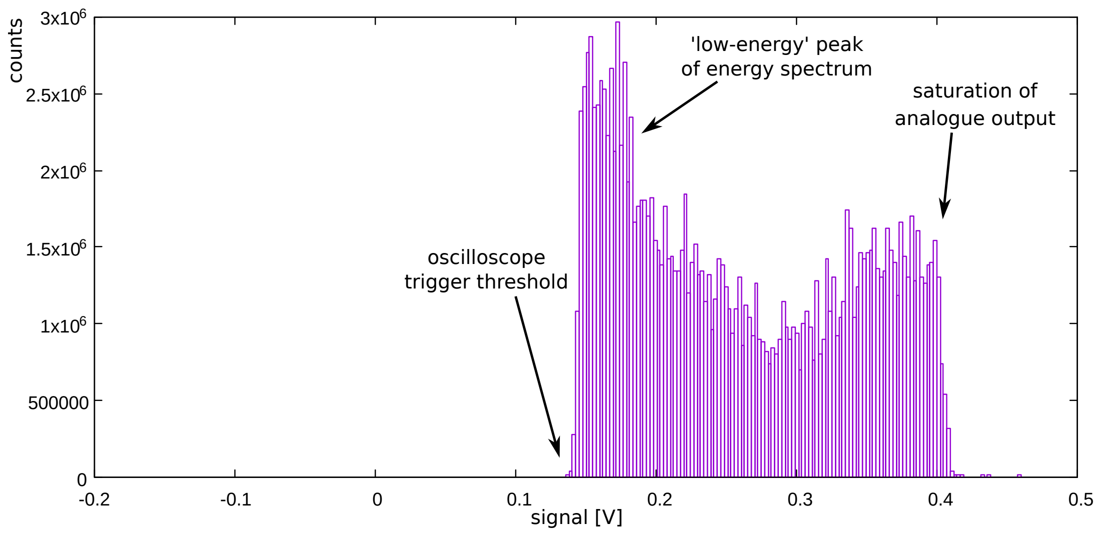


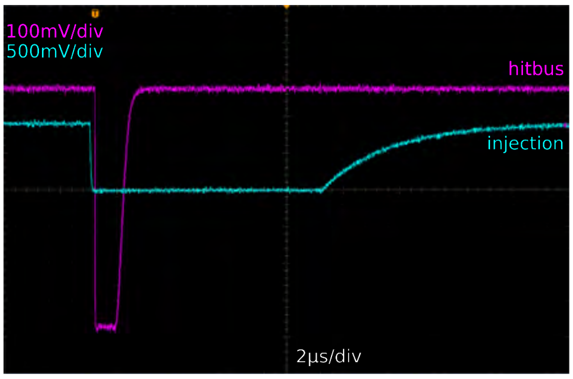
Figure 6.5: Histogram of the analogue amplifier output recorded with an oscilloscope for a Sr-90 source.

6.3 Hitbus

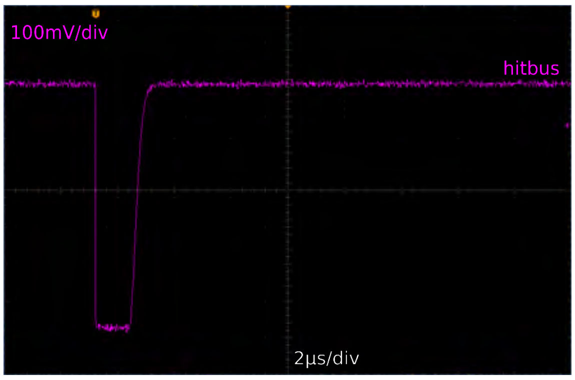
The hitbus signal from a column of choice can be probed on the MuPix8 PCB. For the measurements shown in Figure 6.6, an oscilloscope has been connected with a *LEMO* cable and the input was terminated with 1 MΩ.

In the case of Figure 6.6a, an injection signal has been generated. The length of the hitbus signal is constant as the injected signal is reproduced with a high precision. For Figure 6.6b, a strontium-90 source has been used. Here, the length of the hitbus signal varies depending on how large the input pulses to the comparator have been.

The hitbus signal, i.e. the output of the comparator (with the higher threshold), has an amplitude of ∼600 mV. The measurements prove the functionality of the comparator in the digital periphery.



(a) Hitbus signal for testpulse injection.



(b) Hitbus signal for strontium-90 source.

Figure 6.6: Oscilloscope screenshots showing exemplary hitbus signals for a test-pulse injection and the signal of a strontium-90 source.

6.4 Data Quality

Figure 6.7 shows a measurement of the data stream from link D (copying the data from link A) at a data transmission rate of 1.25 Gb/s. For this measurement, solder jumpers on the MuPix8 PCB have been set such that the differential signal lines from the MuPix8 chip are wired to SMA connectors on the PCB which allow to attach a Digital Serial Analyzer [50]. An eye diagram with a wide vertical and horizontal opening can be seen. The low and the high level are clearly distinguishable.

A mean *eye height* of 198.8±1.1 mV is recorded taking into account a damping factor of −6.6 dB introduced by the measurement method. The mean *eye width* is 528±1 ps and the *jitter* has been determined to be 45.2±0.3 ps.

When the signal is sent through to the readout PC, the `Nios` interface of the GUI (see Appendix A.1) can be used to check whether the Fast GX Transceivers detect 8b/10b errors. This is not the case for the tested chip, the transmission works flawless.

It can be concluded that the serializer and the LVDS driver on the MuPix8 chip work properly and produce a clean signal.

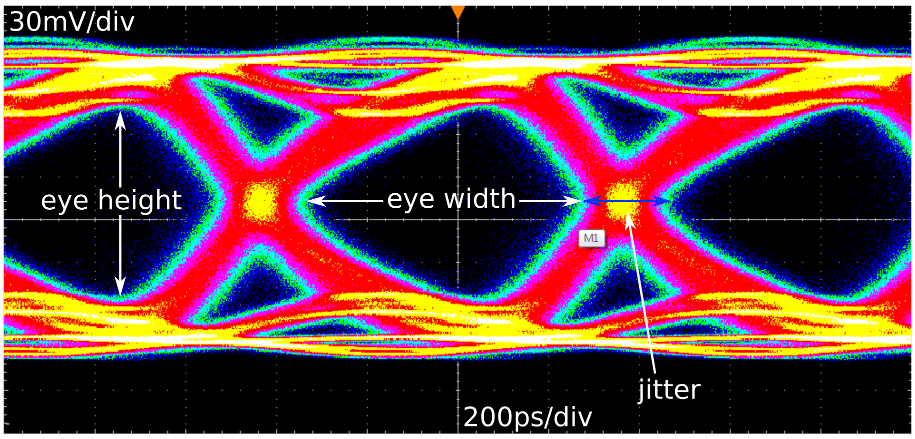


Figure 6.7: LVDS output from the MuPix8 chip on link D (copying the data from link A) at a transmission rate of 1.25 Gb/s, probed on the MuPix8 PCB and measured with a Digital Serial Analyzer.

6.5 Hitmaps

In Figure 6.8, a hitmap of a collimated strontium-90 source is shown. It has been recorded in the laboratory and was used to verify the transformation scheme from digital to physical pixel addresses. If the transformation is done incorrectly, blank rows without hits show up or rows containing pixels with many hits are displayed in a region of fewer hits such that the spot of the source is distorted.

By moving the spot into different corners of the pixel matrix it was also possible to verify the correct sense of counting, namely that the pixel rows and columns are indeed addressed in a rising order from bottom to top and from left to right, respectively.

On the hitmap, a clear distinction between submatrix A and the submatrices B and C is observed. It stems from a different intrinsic or 'untuned' efficiency of the current driven part (submatrices B and C) compared to the source follower (submatrix A).

Dependence on the Readout Speed

As explained in section 4.1, the speed of the readout state machine can be reduced. The corresponding configuration value is `timerend`. It sets the frequency with which the state machine changes states to $f = 62.5\,\mathrm{MHz}/(\texttt{timerend} + 1)$.

The hitmap shown in Figure 6.8 has been recorded with a readout state machine

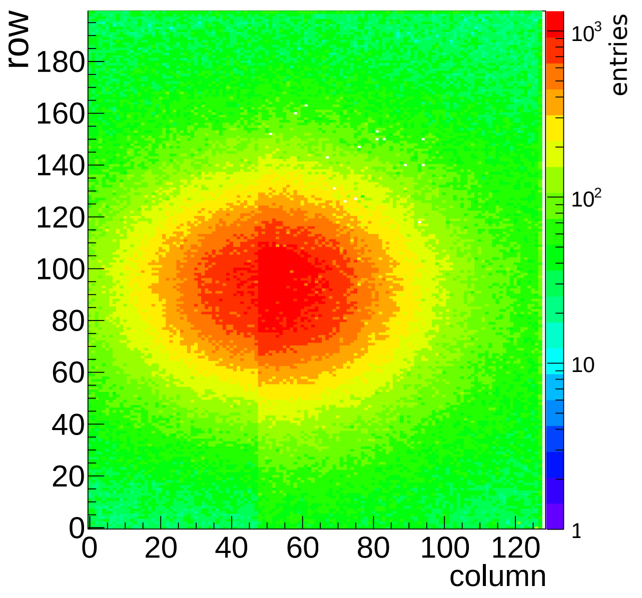


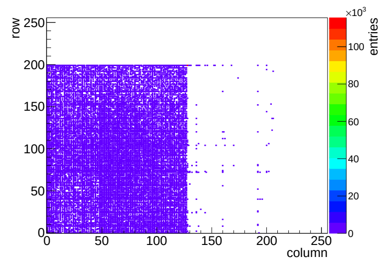
Figure 6.8: Hitmap of a collimated strontium-90 source pointing onto the centre of the MuPix8 chip. There is a clear distinction between submatrix A (source follower) and submatrices B and C (current driven).

running at 1/8 of the maximal speed, i.e. `timerend` = 7. It has been found that the hit addresses are not sent out correctly if running at the maximal speed.

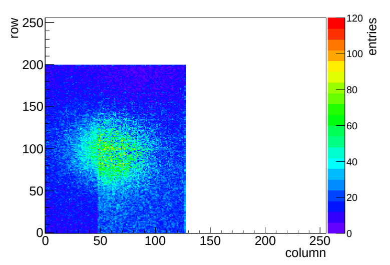
A direct comparison between `timerend` values of 0, 1 and 2 is shown in Figure 6.9. The entire 8-bit address space from 0 to 255 is displayed for both the column and the row. In the case of `timerend` = 0, invalid hit addresses are sent out by the readout state machine. The spot of the collimated Sr-90 source is not visible. Instead, a rectangular pattern shows up. In addition, most hits are assigned the address (128,199). This bin has $\mathcal{O}(10^5)$ entries whereas all other bins have $\mathcal{O}(10)$ entries only. The reason might be that the transistor logic of the state machine does not switch fast enough.

In contrast, for `timerend` = 1, 2 (and larger), the hits lie within the expected address space and the collimated spot from the Sr90 is clearly visible.

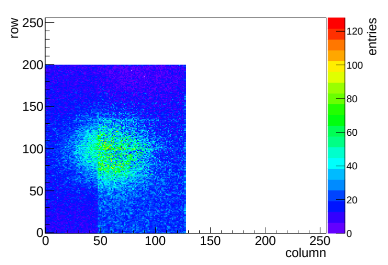
In separate measurements, the links A-C have been enabled one by one. If using only A, B or C, only one of the submatrices is active. Also link D has shown to be functional. For `linksel` $= 0, 1, 2$, copies of the links A, B or C, respectively, are transmitted (see Figure 6.10). The multiplexing of data from submatrices A-C on link D (`linksel` = 3) has not been investigated yet.



(a) `timerend` = 0.

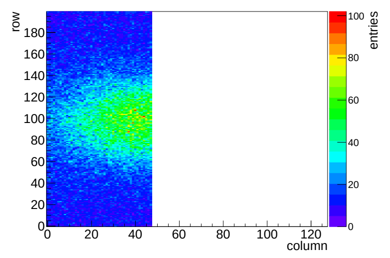


(b) `timerend` = 1.

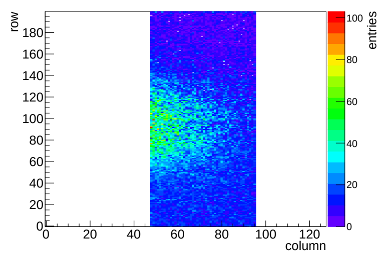


(c) `timerend` = 2.

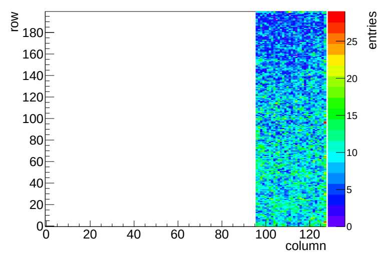
Figure 6.9: Hitmaps recorded with a Sr90 source for different values of `timerend`.



(a) `linksel` = 0.



(b) `linksel` = 1.



(c) `linksel` = 2.

Figure 6.10: Hitmaps recorded with a Sr90 source for link D copying the data from submatrix A, B or C, respectively.

6.6 Temperature Measurements

The functionality of the temperature diode circuitry on the MuPix8 PCB has already been demonstrated using the MuPix8 HSMC Insert (see section 5.3.2). For the MuPix8, such measurements needed to be repeated to validate the functionality of the on-chip temperature diode.

To this end, current-voltage (IV) scans as described in section 5.3.2 have been carried out for various chip settings drawing different currents such that the chip is expected to have different temperatures. No attention has been paid to the functionality of the chip. For the minimal power consumption, for instance, all configuration values have been set to 'zero'. IV curves for three different settings are shown in Figure 6.11. Analogue to the measurement with the temperature diode on the MuPix8 HSMC Insert shown in section 5.3.2, the current starts to rise at lower voltages for higher temperatures.

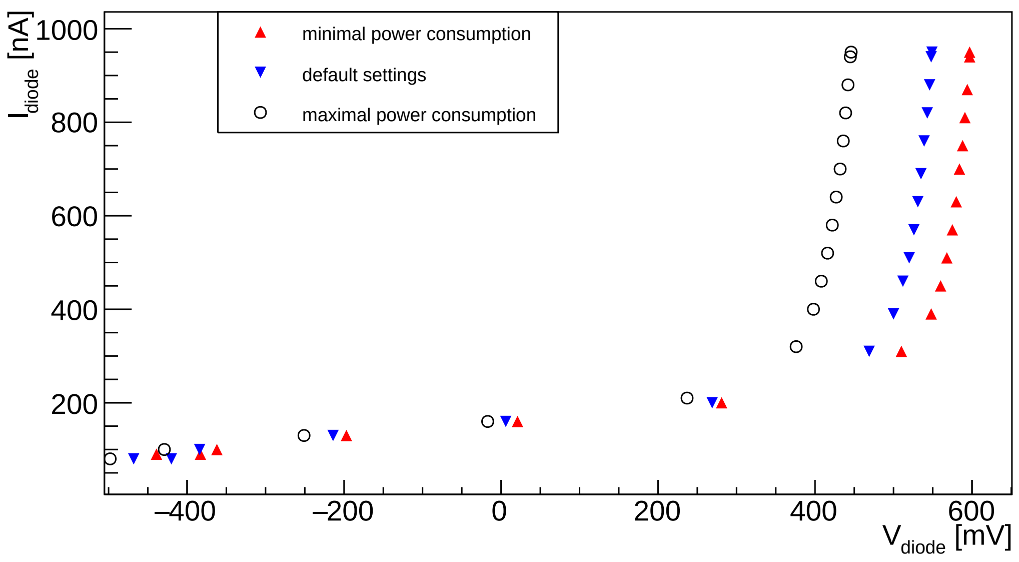


Figure 6.11: IV scans for three exemplary power settings of the MuPix8. A similar plot including 8 different power settings can be found in Appendix A.3.

These measurements show that a temperature measurement is possible in principle. To relate voltages to absolute temperatures, a calibration is needed. This has been done using a *TROTEC IC 080 LV* infrared (IR) camera [51], as explained below.

As a first step, a calibration of the IR camera itself is required because the temperature measurement with the camera is material dependent due to different surface emissivities.

Therefore, a broken MuPix7 chip has been brought into thermal contact with a 3.3 Ω resistor in an aluminium housing which was heated to different temperatures by varying the applied voltage. After a while, the system reaches thermal equilibrium. Then, the temperature of MuPix7 could be read off from the IR camera. Simultaneously, the temperature of the resistor was measured using a PT1000 element in contact with the resistor.

The measurement yields a calibration curve with a linear behaviour as shown in Figure 6.12. The uncertainty ΔT_{IR} on the temperature T_{IR} measured with the IR camera has been approximated to be $\pm 0.5\,^{\circ}\mathrm{C}$ which corresponds to the read-off variations during the measurement. In comparison, the uncertainties introduced by the PT1000 can be neglected. Consequently, an accurate temperature estimation of

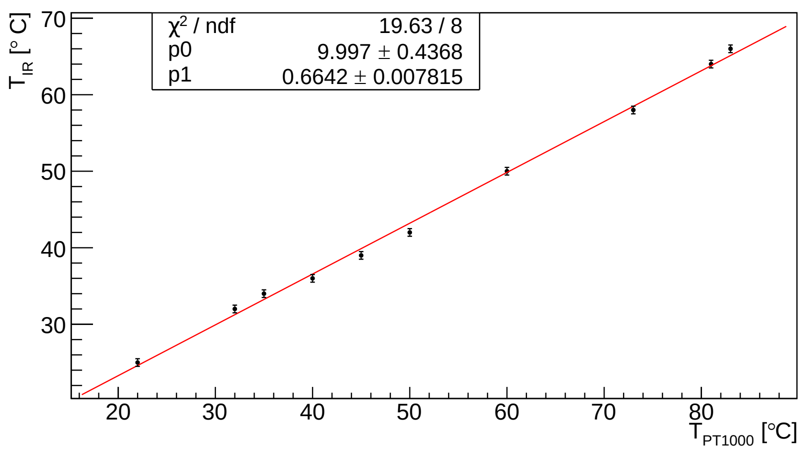


Figure 6.12: Calibration curve between an infrared camera and a PT1000 element for a MuPix chip.

the MuPix8 can be done with the infrared camera using the inverted fit function

$$T_{\text{MuPix8}} = \frac{T_{\text{IR}} - p_0}{p_1}. \tag{6.3}$$

A Gaussian error propagation yields the uncertainty on this conversion:

$$\Delta T_{\text{MuPix8}} = \sqrt{\left(\frac{T_{\text{IR}} - p_0}{p_1^2}\Delta p_1\right)^2 + \left(\frac{\Delta p_0}{p_1}\right)^2} \tag{6.4}$$

ΔT_{IR} does not appear in equation 6.4 as it is correlated with the uncertainties of p_0 and p_1.

In the following, it is assumed that the MuPix7 and the MuPix8 have a comparable emissivity. This approach is valid because both chips are made out of silicon. A MuPix7 has been used instead of a MuPix8 because a broken MuPix8 chip which was not glued to an insert was not available at the time.

The calibration curve can be utilized to correct the temperatures measured with the IR camera for the different power settings mentioned above and plot these corrected temperatures against the voltage measured across the diode for a constant temperature DAC setting, i.e. for a constant injected current.

The working point has been chosen at a current of 950 nA where the spread of the IV curves for different temperatures is well pronounced. The result is shown

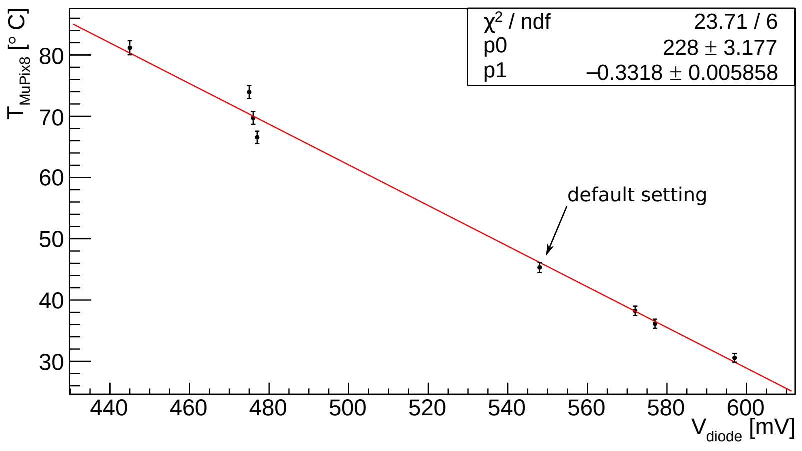


Figure 6.13: The temperature of the MuPix8 in relation to the voltage drop across the temperature diode on the chip, measured with a dedicated ADC on the MuPix8 PCB.

in Figure 6.13. In this plot, the uncertainties on the temperature measurement correspond to the error introduced by the conversion using equation 6.3. The ADC on the MuPix8 PCB offers a precision of ±1 mV which is negligible in comparison.

A linear fit seems appropriate to describe the data. It is highly non-trivial to derive this dependence from equation 5.4 because the saturation current I_S is strongly temperature dependent. The temperature on the MuPix8 can be determined using the fit parameters from Figure 6.13:

$$T_{\text{MuPix8}} = 228.0\,^\circ\text{C} - 0.332\,^\circ\text{C/mV} \times V_{\text{diode}} \tag{6.5}$$

when measuring the voltage drop across the temperature diode on the chip. The temperature uncertainty follows by error propagation (neglecting the error on the voltage measurement):

$$\Delta T_{\text{MuPix8}} = \sqrt{(3.2\,^\circ\text{C})^2 + (0.006\,^\circ\text{C/mV} \times V_{\text{diode}})^2} \tag{6.6}$$

For the default settings which are currently used in the lab and on testbeam campaigns, a diode voltage of ∼550 mV has been recorded. This corresponds to a temperature of (45.4±4.6)°C.

The measurements presented above shows that a temperature measurement with the on-chip temperature diode can be performed. In the future, this can be utilized for an automated temperature monitoring integrated in the readout software. Temperature measurements could be performed on a regular basis and compared to a given critical temperature. If the chip exceeds this temperature, its power consumption could be drastically reduced automatically by setting all configuration values to ’0’ to prevent an overheating. In addition, the user could receive a warning via the GUI.

6.7 Power Consumption

As described in section 4.3.1, the MuPix8 PCB is provided with 5 V from which power regulators generate the three supply voltages for the chip. These are VDD and VDDA with a value of 1.8 V and VSSA with a value of 1.0 V.

As the power regulators consume power themselves, the power consumption of the MuPix8 cannot be inferred from the current drawn on the 5 V channel of the external power supply. Consequently, a MuPix8 PCB needed to be modified such that the three supply voltages could be applied directly to the chip. A photograph is shown in Figure 6.14. The currents which are drawn on the three channels corresponding to VDD, VDDA and VSSA could then be used to calculate the power consumption of the MuPix8.

For the measurement, the MuPix8 has been operated in the settings which are used as a default for all lab measurements in Heidelberg and on DESY testbeam campaigns.

The currents have been read off from the power supply (a HAMEG HMP4040 [52]). Multiple measurements have been performed with intermediate power-downs. In 1 of 5 measurements, it has been observed that the currents of VDD and VDDA deviate by more 23 % from the other measurements with identical settings. This case has been interpreted as a misconfiguration of the chip and is excluded in the following.

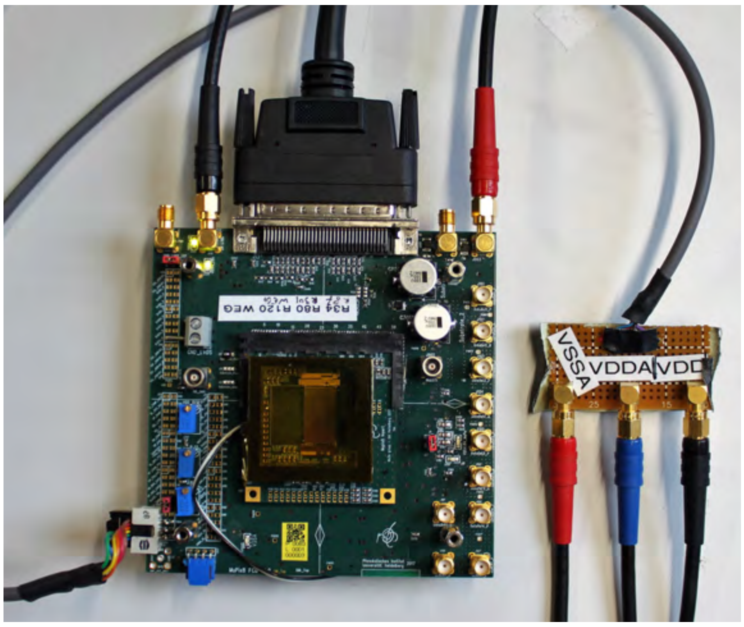


Figure 6.14: Photograph showing the modified MuPix8 PCB to which VDD, VDDA and VSSA are applied externally.

The power consumption is calculated as the product of the voltages and respective currents. The errors stem from the averaging over multiple measurements. The results are summarised in table 6.1.

VDD (1.8 V)	VDDA (1.8 V)	VSSA (1.0 V)	power consumption
(103.4 ± 4.9) mA	(74.0 ± 3.6) mA	(101.9 ± 0.4) mA	(421.1 ± 5.0) mW

Table 6.1: Summary of the power consumption measurement.

The measurement shows that the power consumption of the MuPix8 is ~421 mW for the settings currently used. As a result, it can be concluded that the MuPix8 – having an active area of ~2 cm^2 – can be operated with a power density of ~210 mW/cm^2 which lies well within the cooling capability of the final detector design which is specified up to 400 mW/cm^2 [19].

However, it is not clear how much the power consumption might change when the settings of the sensor are changed further. Therefore, the measurement needs to be repeated when the exact operating settings are found.

7 Conclusion & Outlook

In the first part of this work, a number of hardware components – the MuPix8 HSMC Adapter Card, the MuPix8 Insert, and the MuPix8 Insert *light* – have been designed to complete the data acquisition chain of the MuPix8 chip. Their functionality has been proven successfully. The MuPix8 SCSI Adapter Card is needed as the interface between the Stratix IV running the readout firmware and the MuPix8 PCB carrying the MuPix8 Insert. The MuPix8 Insert in turn carries one MuPix8 chip and allows to exchange chips easily in an existing setup. A simplified version – the MuPix8 Insert *light* – has been designed as well. It is in production at the time of writing.

Secondly, an FPGA-based chip emulator has been developed and integrated in the MuPix8 readout firmware. Producing a simple deterministic 'diagonal' hit pattern, it was a valuable tool to debug the readout firmware as well as the software. In a further step, the emulator has been migrated onto a separate FPGA which is connected to the MuPix8 readout chain via a dedicated custom-designed adapter card – the MuPix8 HSMC Insert – which was also designed within the scope of this thesis. This 'external' emulator can produce pseudo-random hits in addition to the simple 'diagonal' hit pattern of the 'internal' emulator. It was successfully used to validate the functionality data acquisition chain of the MuPix8 single setup.

Furthermore, contributions to the commissioning of the MuPix8 chip have been made using the newly developed hardware. The general functionality of the sensor has been proven. The configuration register can be addressed and the values can be read back correctly. The charge-sensitive amplifier integrated in the pixel cell generates large pulses which are processed in the digital periphery. A data stream containing valid hit addresses is produced by the readout state machine, correctly 8b/10b encoded, serialized and transmitted as LVDS signals. However, the state machine must be run at a clock speed reduced by a factor of 1/2 or more in order to send out valid addresses.

In addition, a calibration for the on-chip temperature diode has been performed. For the current default settings, the chip reaches a temperature of ∼45 °C. Finally, a measurement of the power consumption of the chip has been carried out yielding a power density of ∼210 mW/cm^2 for the default settings currently in use. This value lies well within the specifications of the cooling system of the final detector.

In the following, open questions related to the characterisation of the MuPix8 are discussed. Furthermore, an outlook is provided to what the hardware and firmware, which was developed and tested within the scope of this work, can be used for in the future.

Further MuPix8 Characterisation

The investigation of further features of the MuPix8 chip is ongoing [53, 54, 55]. Measurements of particular importance comprise the determination of the **sensor efficiency** and its **timing resolution**. To this end, data from testbeam campaigns at DESY in Hamburg and at PSI in Villigen, Switzerland, will be analysed.

Another important step is an **optimisation of the chip settings** and a **pixel-wise tuning** (i.e. a pixel-wise adaption of the comparator thresholds) to increase the performance of the sensor with respect to noise, efficiency, time resolution, and power consumption.

Furthermore, the different **timewalk correction** strategies related to the available operating modes of the comparators in the digital periphery (time-over-threshold, 2-threshold and ramp) need to be tested and compared.

MuPix8 Hardware

The **ATLASPix** sensor is a monolithic chip prototype similar to the MuPix8 which is designed to investigate the usability of the HV-MAPS technology in a future ATLAS upgrade [56]. It has been produced in an engineering run together with the MuPix8.

The readout chain for the operation of a single ATLASPix chip is currently being commissioned [57]. For maximal synergy, it employs many hardware components from the MuPix8 readout chain: The MuPix8 PCB is used in combination with a dedicated ATLASPix Insert.

In addition, the MuPix8 SCSI Adapter Card, which was also designed as part of this thesis, is used in the ATLASPix readout chain as well. The firmware and the software require changes in consideration of a different data structure and configuration options. For instance, the ATLASPix features only one fast data output such that a merging of the data from multiple submatrices is superfluous. However, the readout software is structured in a modular way such that it can easily be modified to operate the ATLASPix.

The concept of a small and simple PCB (MuPix8 Insert) carrying a chip and being mountable to a large and complex PCB (MuPix8 PCB) has been proven to work. Consequently, the same strategy is applied for the **MuPix9** and **future MuPix generations**. The readout hardware for the MuPix9 is currently under development.

MuPix8 Emulator

After the **MuPix8 single setup** has been taken into operation successfully, the 'internal' as well as the 'external' MuPix8 emulator are actively used as a debugging and testing tool further on. The 'external' emulator is employed for the quality assurance following the production of further MuPix8 PCBs.

The **MuPix8 telescope setup** is currently being commissioned during a DESY testbeam campaign. In this process, the emulator is utilized to develop and debug the setup with respect to the firmware as well as the software. The 'external' emulator is a particularly valuable tool due to the small number of fully functional MuPix8 chips currently available.

Another project which is well underway is the build-up of a **Vertical Slice** of the Mu3e detector readout chain [58]. In this project, the 'internal' emulator could be embedded into the firmware running on the FPGA on the front-end board. The 'external' emulator would be of particular use to verify the correct functionality of the data transmission through the physical transmission lines.

In the current status, the front-end board can be connected to up to four custom-designed adapter cards which allow to connect up to two MuPix8 PCBs each via SCSI cables. In this case, a total of eight MuPix8 HSMC Inserts can be used in combination with MuPix8 PCBs just as for the single setup. In future steps, a new custom adapter card might be necessary when the readout chain comprises more components like flexprints to which the MuPix8 will eventually be glued and bonded.

As the firmware-based MuPix8 emulator has shown to be a valuable tool in the process of debugging and commissioning a newly developed setup, this concept will be employed for future MuPix generations as well.

A Appendix

A.1 Graphical User Interface

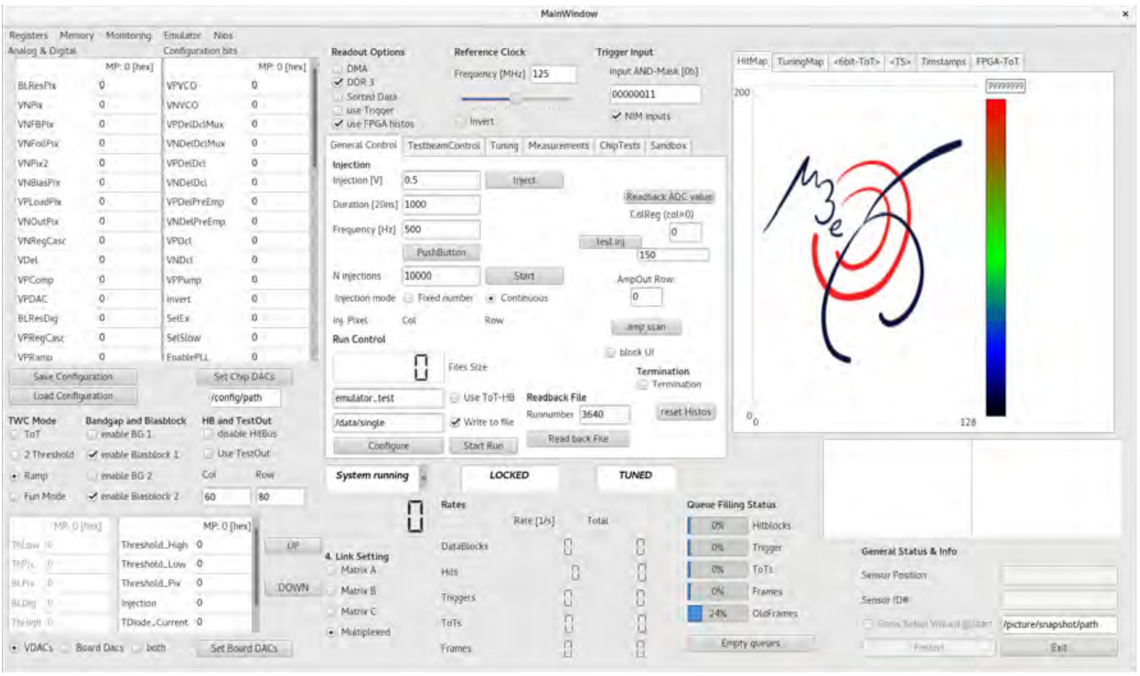
The Graphical User Interface is built in a way such that multiple dialog windows can be accessed from a **MainWindow**. There are three MainWindows for different applications.

The **Single MainWindow** (see Figure A.1) is dedicated to the single setup such that a large spectrum of functions is provided to operate the single setup and do quick and easy adjustments to chips settings. From the MainWindow, several dialogs can be opened:

- The **Registers** tab opens a window in which the values in the read- and write registers of the FPGA can be displayed (see Figure A.2).
- The **Memory** tab opens a window in which the FPGA memory can be read out and displayed (see Figure A.3).
- The **Monitoring** tab is currently not used but will later provide access to online monitoring functions.
- The **Emulator** tab opens a control window with which the **MuPix8 emulator** can be enabled and controlled. It has been presented in section 5.4.
- Finally, the **Nios** tab provides access to information about the synchronisation status of the receivers, 8b/10b errors, and unpacker errors (see Figure A.4).

The **Telecope MainWindow** (see Figure A.5) can be used to operate the telescope. It provides a different range of functionalities which is optimized to the specific requirements of the telescope like automated tuning. Like for the Single MainWindow, various windows can be opened from tabs. They comprise the registers, the memory, a tuning window, a window to set values for the on-board DACs, a window for the on-chip DACs, a window in which the hit maps of all chips can be displayed, a slow control panel for MIDAS integration [59], the EmulatorDialog and the NiosDialog.

The **Emulator MainWindow** (see Figure A.6) is comparatively simple. It allows to initialize a Stratix IV if it is not automatically mapped correctly when starting the software. In addition, a second FPGA can be initialized if one PC is used for the operation of two Stratix IV with the emulator to be able to test the telescope. In addition, it allows to open the EmulatorDialog and the RegistersDialog. The MemoryDialog can also be opened but is not used because the emulator does not write to the FPGA memory.



Analog & Digital

	MP: 0 [hex]
BLResPix	0
VNPix	0
VNFBPix	0
VNFollPix	0
VNPix2	0
VNBiasPix	0
VPLoadPix	0
VNOutPix	0
VNRegCasc	0
VDel	0
VPComp	0
VPDAC	0
BLResDig	0
VPRegCasc	0
VPRamp	0

Configuration bits

	MP: 0 [hex]
VPVCO	0
VNVCO	0
VPDelDclMux	0
VNDelDclMux	0
VPDelDcl	0
VNDelDcl	0
VPDelPreEmp	0
VNDelPreEmp	0
VPDcl	0
VNDcl	0
VPPump	0
invert	0
SelEx	0
SelSlow	0
EnablePLL	0

/config/path

TWC Mode: ToT [✓ Ramp] 2 Threshold, Fun Mode

Bandgap and Biasblock: enable BG 1, [✓ enable Biasblock 1], enable BG 2, [✓ enable Biasblock 2]

HB and TestOut: disable HitBus, Use TestOut

Col 60 Row 80

	MP: 0 [hex]
ThLow	0
ThPix	0
BLPix	0
BLDig	0
ThHigh	0

	MP: 0 [hex]
Threshold_High	0
Threshold_Low	0
Threshold_Pix	0
Injection	0
TDiode_Current	0

[✓ VDACs] Board Dacs, both

Readout Options: DMA, [✓ DDR 3], Sorted Data, use Trigger, [✓ use FPGA histos]

Reference Clock: Frequency [MHz] 125; Invert

Trigger Input: Input AND-Mask [0b] 00000011; [✓ NIM inputs]

General Control | TestbeamControl | Tuning | Measurements | ChipTests | Sandbox

Injection
Injection [V] 0.5
Duration [20ns] 1000
Frequency [Hz] 500
N injections 10000
Injection mode: Fixed number [✓ Continuous]
Inj. Pixel Col Row

ColReg (col=0) 0
150
AmpOut Row: 0
block UI
Termination: Termination

Run Control
0 Files Size
emulator_test
/data/single
Use ToT-HB
[✓ Write to file]
Readback File: Runnumber 3640

System running | LOCKED | TUNED

0

4. Link Setting: Matrix A, Matrix B, Matrix C, [✓ Multiplexed]

Rates

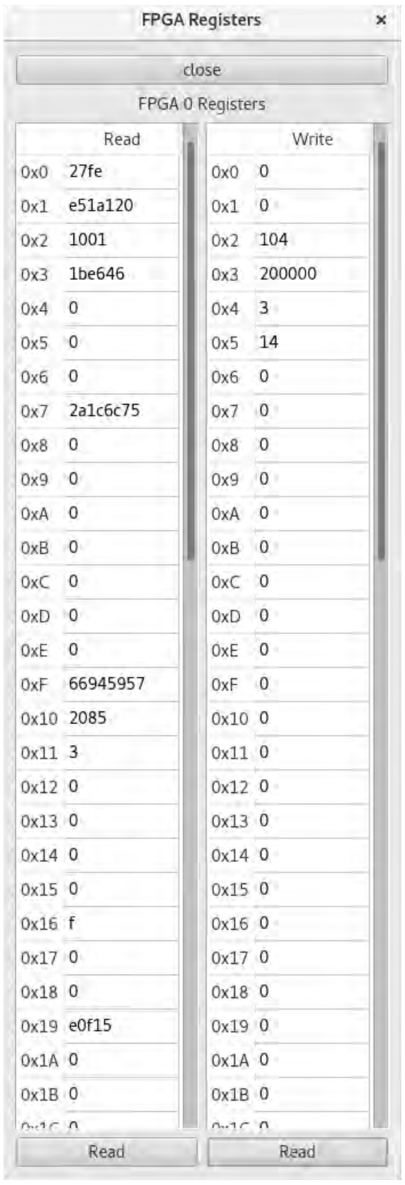
	Rate [1/s]	Total
DataBlocks	0	0
Hits	0	0
Triggers	0	0
ToTs	0	0
Frames	0	0

Queue Filling Status

0%	Hitblocks
0%	Trigger
0%	ToTs
0%	Frames
24%	OldFrames

General Status & Info
Sensor Position
Sensor ID#
Show Setup Wizard @Start
/picture/snapshot/path

Figure A.1: The Single MainWindow.



FPGA Registers ×

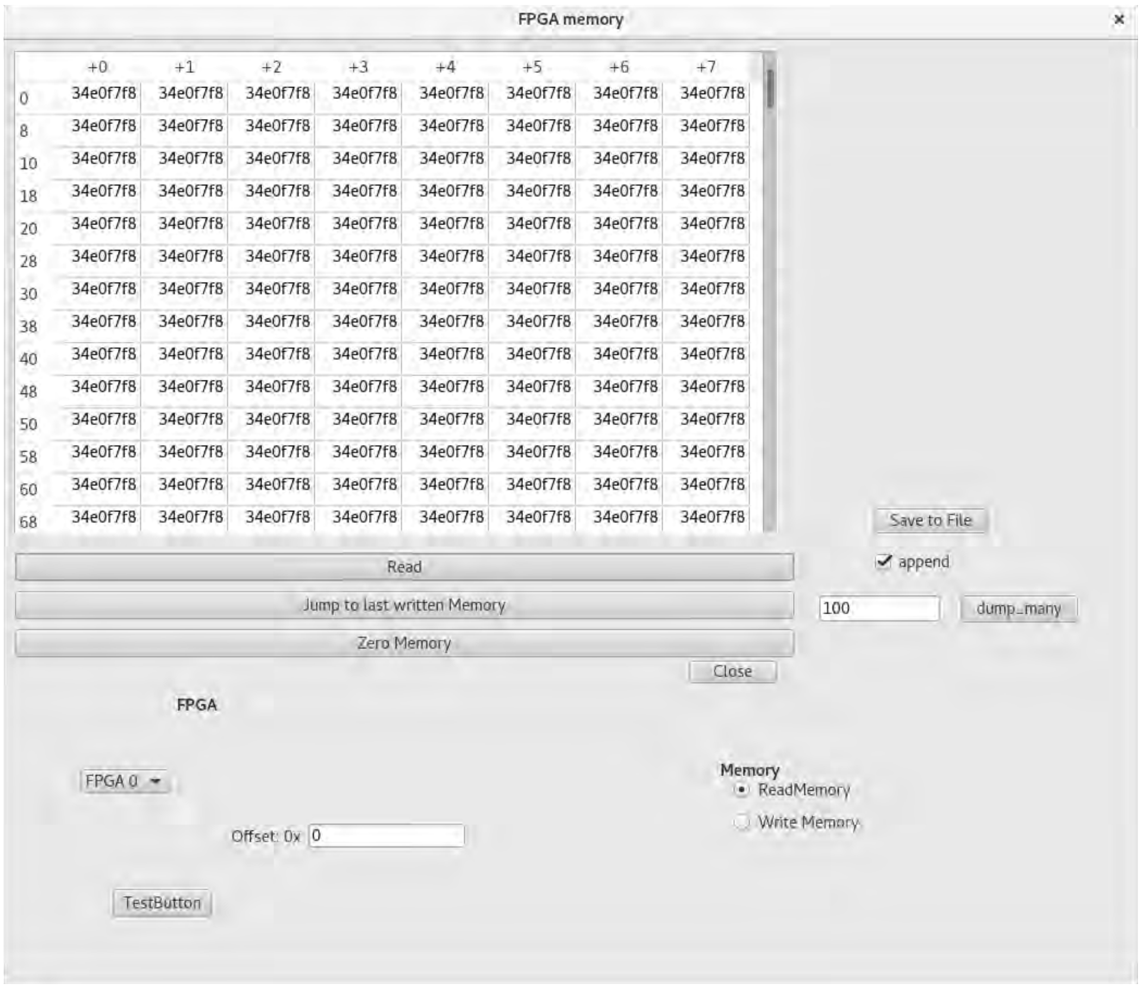
close

FPGA 0 Registers

Read		Write	
0x0	27fe	0x0	0
0x1	e51a120	0x1	0
0x2	1001	0x2	104
0x3	1be646	0x3	200000
0x4	0	0x4	3
0x5	0	0x5	14
0x6	0	0x6	0
0x7	2a1c6c75	0x7	0
0x8	0	0x8	0
0x9	0	0x9	0
0xA	0	0xA	0
0xB	0	0xB	0
0xC	0	0xC	0
0xD	0	0xD	0
0xE	0	0xE	0
0xF	66945957	0xF	0
0x10	2085	0x10	0
0x11	3	0x11	0
0x12	0	0x12	0
0x13	0	0x13	0
0x14	0	0x14	0
0x15	0	0x15	0
0x16	f	0x16	0
0x17	0	0x17	0
0x18	0	0x18	0
0x19	e0f15	0x19	0
0x1A	0	0x1A	0
0x1B	0	0x1B	0
0x1C	0	0x1C	0

Read Read

Figure A.2: The RegistersDialog.


FPGA memory

	+0	+1	+2	+3	+4	+5	+6	+7
0	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
10	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
18	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
20	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
28	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
30	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
38	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
40	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
48	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
50	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
58	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
60	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8
68	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8	34e0f7f8

[✓ append]

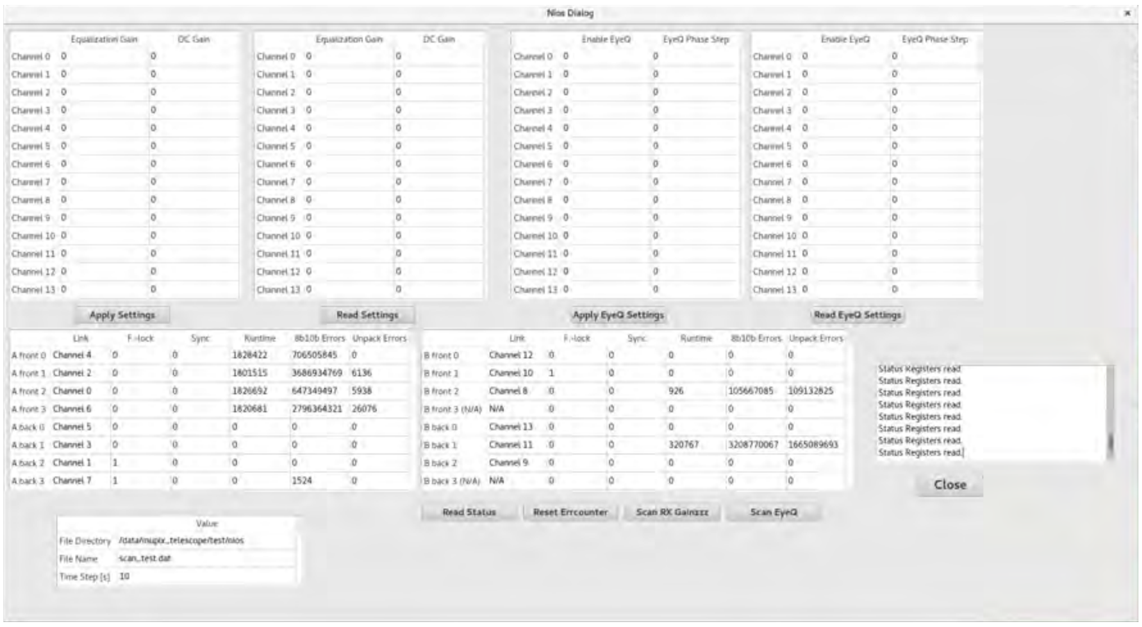
100

FPGA: FPGA 0

Offset: 0x 0

Memory
[✓ ReadMemory]
Write Memory

Figure A.3: The MemoryDialog.



	Equalization Gain	DC Gain
Channel 0	0	0
Channel 1	0	0
Channel 2	0	0
Channel 3	0	0
Channel 4	0	0
Channel 5	0	0
Channel 6	0	0
Channel 7	0	0
Channel 8	0	0
Channel 9	0	0
Channel 10	0	0
Channel 11	0	0
Channel 12	0	0
Channel 13	0	0

	Equalization Gain	DC Gain
Channel 0	0	0
Channel 1	0	0
Channel 2	0	0
Channel 3	0	0
Channel 4	0	0
Channel 5	0	0
Channel 6	0	0
Channel 7	0	0
Channel 8	0	0
Channel 9	0	0
Channel 10	0	0
Channel 11	0	0
Channel 12	0	0
Channel 13	0	0

	Enable EyeQ	EyeQ Phase Step
Channel 0	0	0
Channel 1	0	0
Channel 2	0	0
Channel 3	0	0
Channel 4	0	0
Channel 5	0	0
Channel 6	0	0
Channel 7	0	0
Channel 8	0	0
Channel 9	0	0
Channel 10	0	0
Channel 11	0	0
Channel 12	0	0
Channel 13	0	0

	Enable EyeQ	EyeQ Phase Step
Channel 0	0	0
Channel 1	0	0
Channel 2	0	0
Channel 3	0	0
Channel 4	0	0
Channel 5	0	0
Channel 6	0	0
Channel 7	0	0
Channel 8	0	0
Channel 9	0	0
Channel 10	0	0
Channel 11	0	0
Channel 12	0	0
Channel 13	0	0

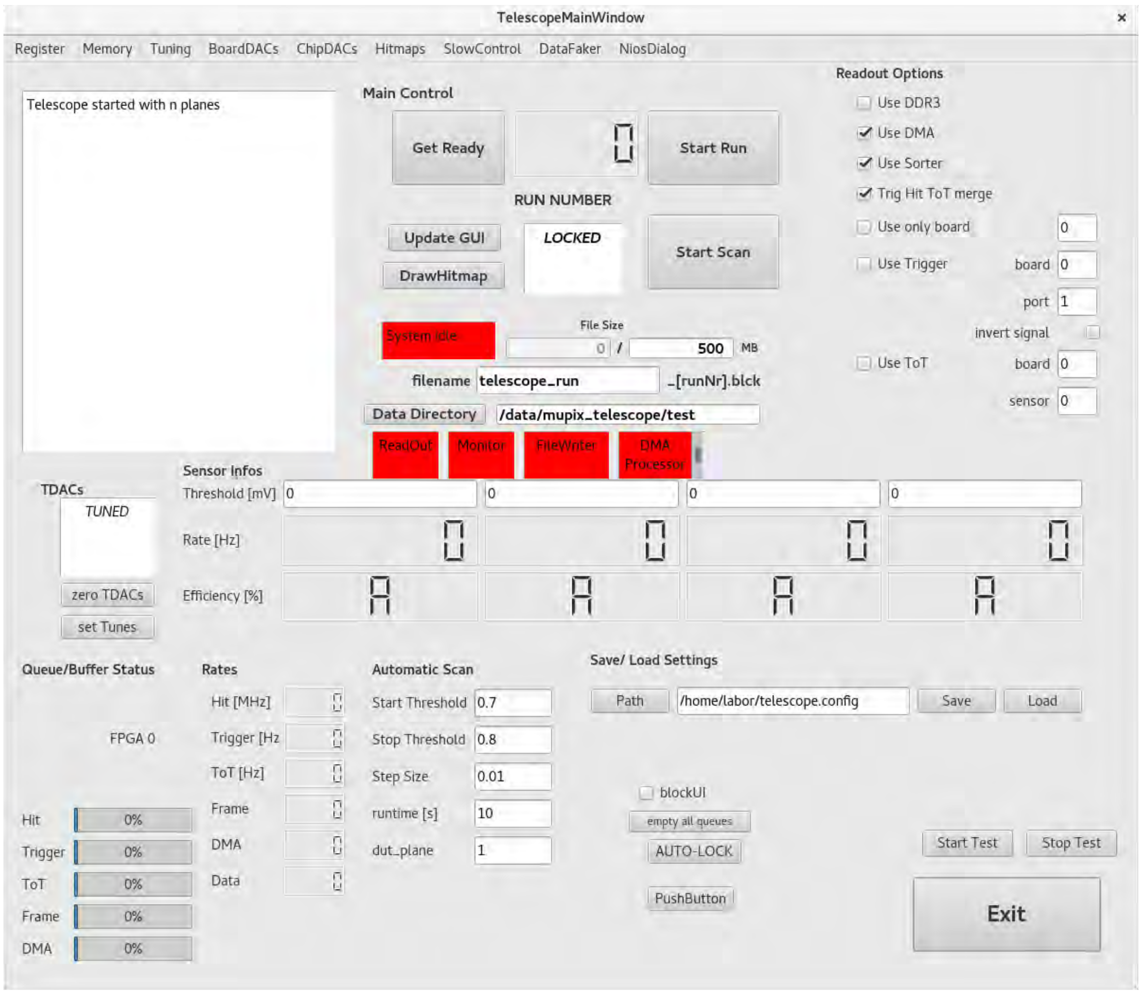
	Link	F.-lock	Sync	Runtime	8b10b Errors	Unpack Errors
A front 0	Channel 4	0	0	1828422	706505845	0
A front 1	Channel 2	0	0	1801515	3686934769	6136
A front 2	Channel 0	0	0	1826692	647349497	5938
A front 3	Channel 6	0	0	1820681	2796364321	26076
A back 0	Channel 5	0	0	0	0	0
A back 1	Channel 3	0	0	0	0	0
A back 2	Channel 1	1	0	0	0	0
A back 3	Channel 7	1	0	0	1524	0

	Link	F.-lock	Sync	Runtime	8b10b Errors	Unpack Errors
B front 0	Channel 12	0	0	0	0	0
B front 1	Channel 10	1	0	0	0	0
B front 2	Channel 8	0	0	926	105667085	109132825
B front 3 (N/A)	N/A	0	0	0	0	0
B back 0	Channel 13	0	0	0	0	0
B back 1	Channel 11	0	0	320767	3208770067	1665089693
B back 2	Channel 9	0	0	0	0	0
B back 3 (N/A)	N/A	0	0	0	0	0

```
Status Registers read.
Status Registers read.
Status Registers read.
Status Registers read.
Status Registers read.
Status Registers read.
Status Registers read.
Status Registers read.
```

	Value
File Directory	/data/mupix_telescope/test/nios
File Name	scan_test.dat
Time Step [s]	10

Figure A.4: The NiosDialog.



Telescope started with n planes

RUN NUMBER: 0

LOCKED

File Size: 0 / 500 MB

filename: telescope_run _[runNr].blck

Data Directory: /data/mupix_telescope/test

Readout Options
- [ ] Use DDR3
- [x] Use DMA
- [x] Use Sorter
- [x] Trig Hit ToT merge
- [ ] Use only board: 0
- [ ] Use Trigger: board 0, port 1, invert signal [ ]
- [ ] Use ToT: board 0, sensor 0

TDACs: TUNED

Sensor Infos				
Threshold [mV]	0	0	0	0
Rate [Hz]	0	0	0	0
Efficiency [%]	A	A	A	A

Queue/Buffer Status

FPGA 0

Hit	0%
Trigger	0%
ToT	0%
Frame	0%
DMA	0%

Rates

Hit [MHz]	0
Trigger [Hz	0
ToT [Hz]	0
Frame	0
DMA	0
Data	0

Automatic Scan

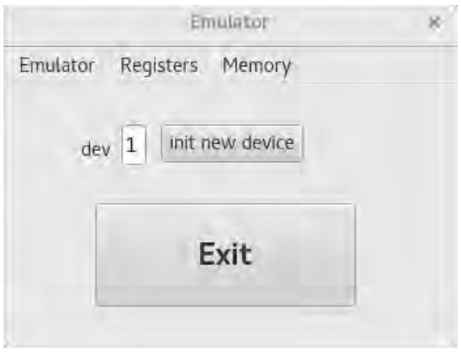
Start Threshold	0.7
Stop Threshold	0.8
Step Size	0.01
runtime [s]	10
dut_plane	1

Save/ Load Settings

Path: /home/labor/telescope.config

- [ ] blockUI

Figure A.5: The Telescope MainWindow.



dev 1

Figure A.6: The Emulator MainWindow.

A.2 The MuPix8 Readout State Machine

In this section, a summary of the configuration values available for the control of the on-chip MuPix8 readout state machine is presented. In addition, a detailed description of the exact working principle of the generation of a hit signal and the readout procedure of the state machine are given. The firmware-based MuPix8 emulator which has been developed as part of this thesis follow this concept.

A.2.1 Configuration Values for Readout State Machine

The following list of configuration values is used to control the readout state machine. They can be set in the GUI (see Appendix A.1) to configure the MuPix8. For the firmware-based emulator (see chapter 5), the same values are used. In this case, they are set in the dedicated EmulatorDialog (see section 5.4).

- With `ckdivend`, the frequency with which the 10-bit timestamp is incremented, can be decreased: $f = 125\,\text{MHz}/(\texttt{ckdivend} + 1)$.
- `ckdivend2` can likewise be used to slow down the incrementation frequency of the 6-bit timestamp: $f = 125\,\text{MHz}/(\texttt{ckdivend2} + 1)$.
- If `ckdivend` or `ckdivend2` is chosen > 0, `tsphase` can be used to adjust the phase of the two timestamps relative to their generating clock. It cannot be chosen to be larger than `ckdivend` or `ckdivend2`. If, for instance, `ckdivend` $=$ 5 and `tsphase` $=$ 3, the 10-bit timestamp is increased whenever a counter (from 0 up to `ckdivend` $- 1$) reaches 3.
- With `timerend`, the frequency of the readout state machine can be controlled: $f = 62.5\,\text{MHz}/(\texttt{timerend}+1)$. It is required to be ≥ 2 if the link D is supposed to send out a multiplexed data stream from all three submatrices.
- The readout state machine remains in the state `LdPix1` (see below) for `slow-downend` clock cycles if no hits are received.
- `maxcycend` corresponds to the maximal number of hits which are read out subsequently before a new 'snapshot' is taken. The hits which have not been sent out, are not lost but are contained in the new 'snapshot' as well. In general, it is assigned its maximal value of 63.

- The number of `K28.5` control words which is sent after a reset is $2 \times 256 \times$ `resetckdivend`.
- `linksel` defines the mode of link D. For values of 0 to 2, copies of the data from submatrices A, B or C, respectively, are sent out. For a value of 3, link D multiplexes the data from A-C. The multiplexing requires `timerend` to be ≥ 2.

A.2.2 Hit Generation and Readout Procedure in the MuPix8

In the following, the generation of hit information in the analogue pixel and its transfer to the digital periphery is described. For the sake of simplicity, all signals are assumed to be active-high here, even though this is not the case in the actual MuPix8 chip in order to reduce the number of on-chip transistors.

If an amplified analogue pulse from the pixel diode exceeds the threshold of a comparator, a hit flag is stored in a latch. At the same time, the timestamp value gets stored. The pixel is now occupied and cannot receive a new hit until read out. On a `LdPix` signal from the state machine, the hit signal is transferred to a second latch. Throughout this thesis, this step is referred to as taking a 'snapshot' of the matrix.

A readout priority chain exists to check which column contains a hit. The second latch mentioned above serves as a means of synchronisation to prevent the situation that a pixel is hit just in the moment when the priority chain checks the column for a hit. If at least one hit has been detected, the readout state machine receives a `Priout` signal.

On a `PullDown` signal from the state machine, the address bus lines, which connect the digital pixel cells and the end-of-column cells, are pulled to ground to ensure a well-defined state.

If a `LdCol` signal is received from the state machine, the hit with the lowest row address gets loaded into the end-of-column cell from each column with at least one hit. On a `RdCol`, another priority chain becomes active and sends a `PriFromDet` signal to the readout state machine if at least one hit is stored in the end-of-column row. The hits in the end-of-column row are then sent out in the readout procedure described below.

A schematic illustrating the state sequence of the readout state machine is shown in Figure A.7. One clock cycle corresponds to two 8-bit words. The state machine runs at a maximal speed of up to 80 MHz (nominal 62.5 MHz) which can be reduced using the register value `timerend`. It stays in every state for `timerend` + 1 clock cycles, i.e. the readout frequency is given by $f_{\text{readout}} = 62.5\,\text{MHz}/\texttt{timerend}{+}1$.

After resetting, the state machine starts off in `StateSync`. It sends an adjustable number of $2 \times 256 \times$ `resetckdivend` control words (`K28.5`) to allow the receiver in the readout firmware to synchronise on the data stream and find the correct word boundaries. If `sendcounter` is enabled, the chip then enters a debug mode in which it constantly sends out a counter instead of hit information. In this case, the state flips between `StateSendCounter1` and `StateSendCounter2` continuously until `sendcounter` is disabled.

Otherwise, the actual readout procedure begins:

- In `StatePD1`, the `Pulldown` signal is enabled.
- In `StatePD2`, the `Pulldown` goes low again.

Until now, the chip sends out further `K28.5` control words.

- In `StateLdCol1`, one hit per column – the one with the lowest row address – is transferred to the end-of-column cell in the digital periphery on `LdCol`.
- In `StateLdCol2`, `LdCol` goes low again and it is waited for three clock cycles in order to allow `PriFromDet` to settle.

While waiting, the chip sends out further `K28.5` control words. Then, a `K28.0` control word and a link identifier which is `0xAA`, `0xBB`, `0xCC` for the three submatrices, respectively.

- In `StateLdPix1`, `LdPix` goes high.
- In `StateLdPix2`, `LdPix` goes low again. If one or more hits are registered in the end-of-column row (`PriFromDet` is on), the next state `StateRdCol1` is entered. Otherwise one returns to `StatePD1`.

In the above two state, the binary counter as well as the Gray counter are sent out.

- In `StateRdCol1`, the `RdCol` is high.
- In `StateRdCol2`, `RdCol` goes low again. If there are remaining hits in the end-of-column row and less than `maxcycend` hits have been read out, the state jumps back to `StateRdCol1`, otherwise it goes to `StatePD1` and the readout procedure is repeated.

In these two states, the 10-bit and the 6-bit timestamps as well as the column and row addresses are sent out. Table A.1 provides a summary of which data word is sent out in which state of the readout state machine.

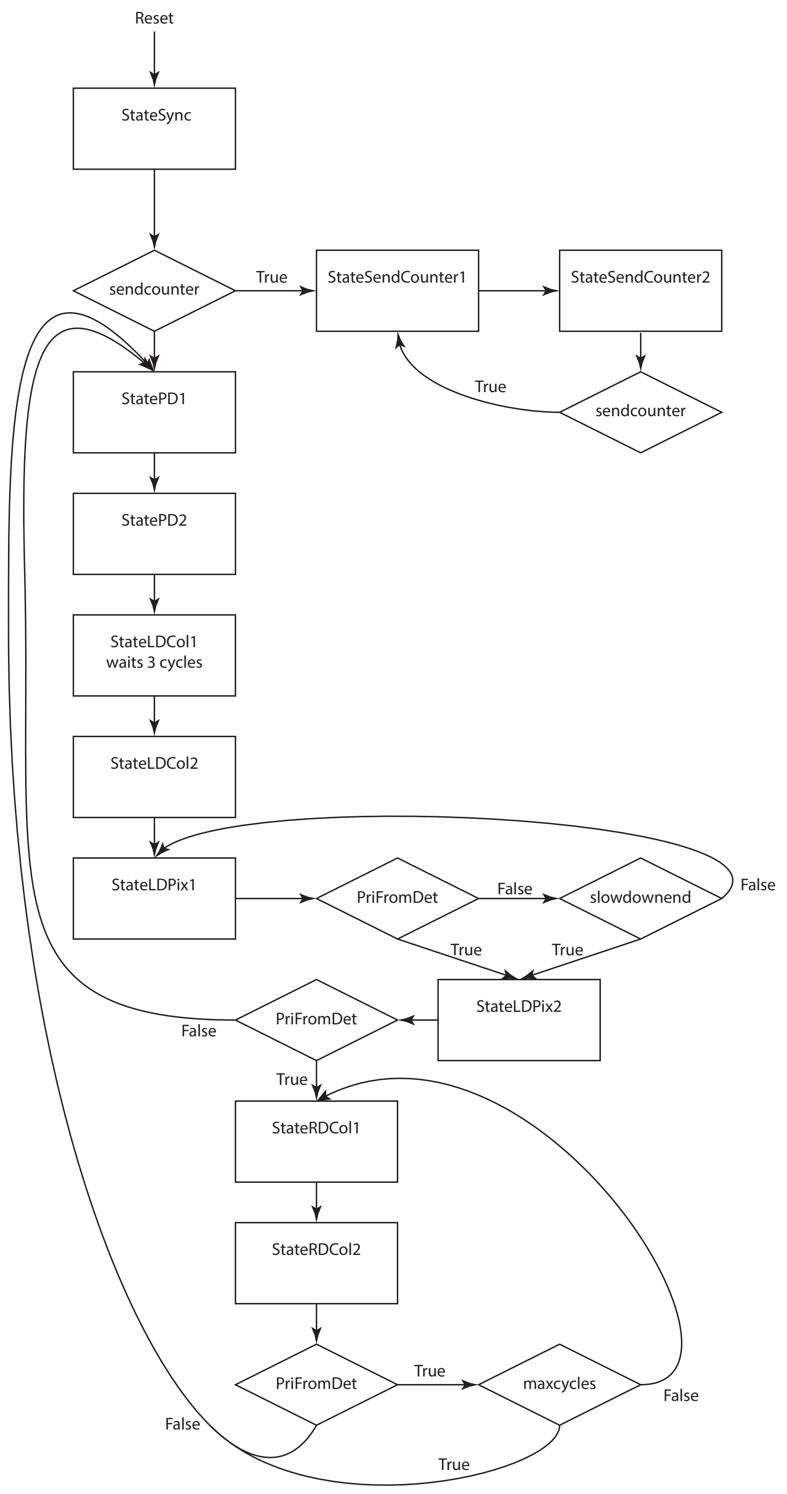


Figure A.7: Schematic drawing of the MuPix8 readout state machine. From [60].

Cycle	State	Data
0	`SyncState`	`K28.5`
1	`SyncState`	`K28.5`
...	`SyncState`	`K28.5`
$N_{res} - 1$	`SyncState`	`K28.5`
N_{res}	`StatePD1`	`K28.5`
$N_{res} + 1$	`StatePD1`	`K28.5`
$N_{res} + 2$	`StatePD2`	`K28.5`
$N_{res} + 3$	`StatePD2`	`K28.5`
$N_{res} + 4$	`LdColWait`	`K28.5`
$N_{res} + 5$	`LdColWait`	`K28.5`
$N_{res} + 6$	`LdColWait`	`K28.5`
$N_{res} + 7$	`LdColWait`	`K28.5`
$N_{res} + 8$	`LdColWait`	`K28.5`
$N_{res} + 9$	`LdColWait`	`K28.5`
$N_{res} + 10$	`LdColWait`	`K28.5`
$N_{res} + 11$	`LdColWait`	`K28.5`
$N_{res} + 12$	`StateLdCol1`	`K28.0`
$N_{res} + 13$	`StateLdCol1`	`Link identifier`
$N_{res} + 14$	`StateLdCol2`	`K28.0`
$N_{res} + 15$	`StateLdCol2`	`Link identifier`
$N_{res} + 16$	`StateLdPix1`	`BinaryCounter[23 down to 16]`
$N_{res} + 17$	`StateLdPix1`	`BinaryCounter[15 down to 8]`
$N_{res} + 18$	`StateLdPix2`	`BinaryCounter[7 down to 0]`
$N_{res} + 19$	`StateLdPix2`	`GrayCounter[7 down to 0]`
$N_{res} + 20$	`StateRdCol1`	`Charge[5 down to 0] & Timestamp[9 down to 8]`
$N_{res} + 21$	`StateRdCol1`	`Timestamp[7 down to 0]`
$N_{res} + 22$	`StateRdCol2`	`Column`
$N_{res} + 23$	`StateRdCol2`	`Row`
...	`StateRdCol1`	`Charge[5 down to 0] & Timestamp[9 down to 8]`
...	`StateRdCol1`	`Timestamp[7 down to 0]`
...	`StateRdCol2`	`Column`
...	`StateRdCol2`	`Row`
	Repeat `maxcycend` times, then back to cycle N_{res}	

Table A.1: Data format. The link identifiers are `0xAA`, `0xBB` and `0xCC` for the three links. From [60].

A.3 Temperature Measurements

Figure A.8 shows the IV curves for all eight settings which have been used for the temperature calibration in 6.6. The settings have been chosen arbitrarily such that the chip draws different currents and thus heats differently. No attention has been paid to the functionality of the chip.

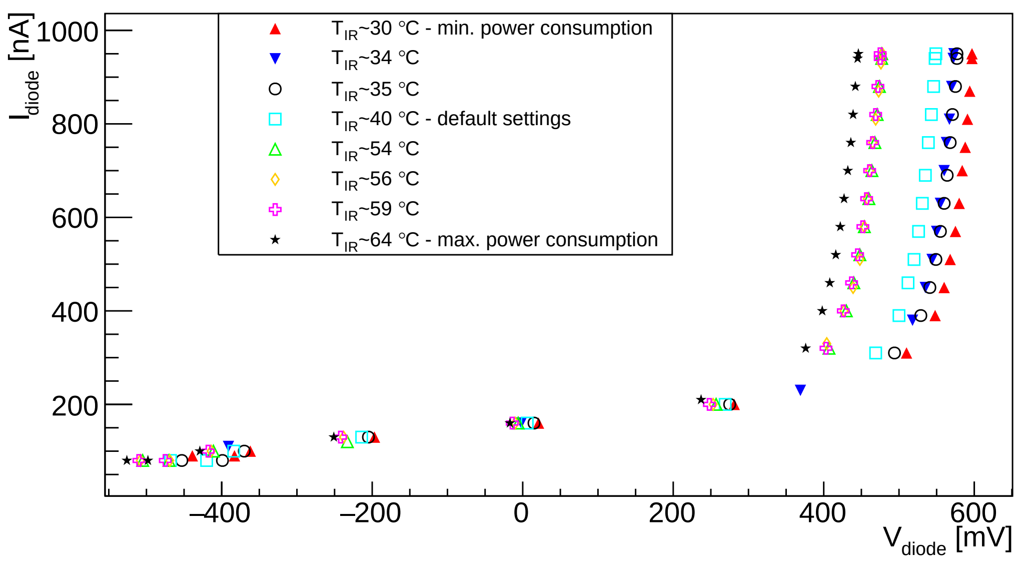


Figure A.8: IV scans for eight different settings of the MuPix8.

A.4 List of Figures

A.5 List of Tables

B Bibliography

[1] M. Thompson, *Modern Particle Physics.* Cambridge University Press, 2013.

[2] https://en.wikipedia.org/wiki/Quantum_chromodynamics#/media/File:Standard_Model_of_Elementary_Particles.svg, accessed 2017-10-02.

[3] G. Aad et al., *"Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC"*, Physics Letters B, **716**(1) 1 – 29, 2012.

[4] S. Chatrchyan et al., [CMS Collaboration], *"Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC"*, Phys.Lett.B, 2012.

[5] Y. Fukuda et al., [Super-Kamiokande Collaboration], *"Evidence for oscillation of atmospheric neutrinos"*, Phys. Rev. Lett., **81** 1562–1567, 1998.

[6] Q. R. Ahmad et al., [SNO Collaboration], *"Measurement of the charged current interactions produced by B-8 solar neutrinos at the Sudbury Neutrino Observatory"*, Phys. Rev. Lett., **87** 071301, 2001.

[7] K. Eguchi et al., [KamLAND Collaboration], *"First results from KamLAND: Evidence for reactor anti- neutrino disappearance"*, Phys. Rev. Lett., **90** 021802, 2003.

[8] F.P. An et al., [Daya Bay Collaboration], *"Observation of electron-antineutrino disappearance at Daya Bay"*, Phys.Rev.Lett., **108** 171803, 2012.

[9] F.P. An et al., [Daya Bay Collaboration], *"Improved Measurement of Electron Antineutrino Disappearance at Daya Bay"*, arXiv:1210.6327v2, 2012.

[10] L. Canetti and M. Shaposhnikov, *"The νMSM and muon to electron conversion experiments"*, Hyperfine Interact. DOI: 10.1007/s10751-013-0796-7, 2013.

[11] A.T. Bajkova and V.V. Bobylev, *"Rotation Curve and Mass Distribution in the Galaxy from the Velocities of Objects at Distances up to 200 kpc"*, Astronomy Letters, Vol. 42, No. 9, pp. 567-582, 2016.

[12] R. Massey et al., *"The dark matter of gravitational lensing"*, arXive, 2010.

[13] http://physicsanduniverse.com/wp-content/uploads/2014/02/galaxy-rotation-curve.gif, accessed 2017-11-12.

[14] C. Patrignani et al., *"Review of Particle Physics (RPP)"*, Phys.Rev., 2016.

[15] Y. Kuno and Y. Okada, *"Muon decay and physics beyond the standard model"*, Rev. Mod. Phys., **73** 151–202, 2001.

[16] R. M. Djilkibaev and R. V. Konoplich, *"Rare Muon Decay $\mu^+ \rightarrow e^+e^-e^+\nu_e\bar{\nu}_\mu$"*, Phys.Rev., **D79** 073004, 2009.

[17] http://aea.web.psi.ch/beam2lines/beam_pie5.html, accessed 2017-11-12.

[18] P.-R. Kettle, *"HiMB – Towards a new High-intensity Muon Beam"*, Future Muon Sources Workshop, 2015.

[19] A. Blondel et al., *"Technical deisgn of the Phase I Mu3e Experiment"*, 2016.

[20] DuPont, *DUPONT KAPTON HN*, 2016.

[21] I. Perić, *"A novel monolithic pixelated particle detector implemented in high-voltage CMOS technology"*, Nucl.Instrum.Meth., **A582** 876, 2007.

[22] H. Chen et al., *"MuTRiG: a mixed signal Silicon Photomultiplier readout ASIC with high timing resolution and gigabit data link"*, Journal of Instrumentation, **12**(01) C01043, January 2017.

[23] D. vom Bruch, *Pixel Sensor Evaluation and Online Event Selection for the Mu3e Experiment*, PhD thesis, Heidelberg University, 2017.

[24] W. J. Marciano, T. Mori and J. M. Roney, *"Charged Lepton Flavor Violation Experiments"*, Ann.Rev.Nucl.Part.Sci., **58** 315–341, 2008.

[25] U. Bellgardt et al., [SINDRUM Collaboration], *"Search for the Decay $\mu^+ \rightarrow e^+e^+e^-$"*, Nucl.Phys., **B299** 1, 1988.

[26] J. Kaulard et al., [SINDRUM II Collaboration], *"Improved limit on the branching ratio of $\mu^- \rightarrow e^+$ conversion on titanium"*, Phys.Lett., **B422** 334–338, 1998.

[27] A. M. Baldini et al., *“Search for the lepton flavour violating decay $\mu^+ \to e^+\gamma$ with the full dataset of the MEG experiment”*, Eur. Phys. J. C 76:434, 2016.

[28] P.W. Cattaneo, *“The MEG II detector”*, arXiv:1705.10224v3, 2017.

[29] R. Abramishvili et al., *“COMET - Technical Design Report”*, 2016.

[30] L. Morescalchi, *“The Mu2e Experiment at Fermilab”*, Proceedings of Science, 2016.

[31] T. Nam, Status of the Mu2e Experiment, In *Flavour Physics Conference, ICISE, Quy Nhon, Vietnam, August 13-19*, 2017.

[32] A. X. Widmer and P. A. Franaszek, *“A DC-Balanced, Partitioned-Block, 8B/10B Transmission Code”*, IBM Journal of Research and Development, **27** 440, 1983.

[33] P.A. Franaszek and A.X. Widmer, Byte oriented DC balanced (0,4) 8B/10B partitioned block transmission code, December 4 1984, US Patent 4,486,739.

[34] A. Weber and I. Peric, Documentation MuPix8 – preliminary version 1, 2017.

[35] J. Grimm, *Measurements of the Signal to Noise Ratio of High Voltage Monolithic Active Pixel Sensors and Simulation Studies for Timewalk Correction*, Bachelor’s thesis, Heidelberg University, 2017.

[36] F. Gray, Pulse code communication, March 17 1953, US Patent 2,632,058.

[37] `http://eutelescope.web.cern.ch/`, accessed 2017-10-18.

[38] SAMTEC, *MB1 Datasheet: Mini Edge Card Socket with Guides.*

[39] `https://telescopes.desy.de/Hardware#Trigger_Logic_Unit`, accessed 2017-10-18.

[40] Altera Corporation, *Stratix IV GX FPGA Development Kit - User Guide*, 9.1.0 edition, 2010.

[41] `https://www.qt.io/`, accessed 2017-11-02.

[42] `https://eigen.tuxfamily.org/dox/group__QuickRefPage.html`, accessed 2017-11-12.

[43] http://www.boost.org/, accessed 2017-11-12.

[44] K. Boyette, OpenCores 8bit/10bit encoder and decoder http://opencores.org/project,8b10b_encdec.

[45] D. Riccardi and P. Novellini, *"An Attribute-Programmable PRBS Generator and Checker"*, Application Note: Xilinx FPGAs, 2011.

[46] H. Spieler, *Semiconductor Detector Systems.* Oxford Science Publications, 2005.

[47] Voltcraft, *Digital-Multimeter Bedienungsanleitung VC250, VC270, VC290.*

[48] https://www.ld-didactic.de/software/524221de/, accessed 2017-11-22.

[49] https://www.ld-didactic.de/software/524221de/Content/Resources/Images/Sr90.png, accessed 2017-11-11.

[50] Tektronix, *DSA8300 Digital Serial Analyzer – Quick Start User Manual.*

[51] TROTEC GmbH & Co. KG, *LV/V-Serie Bedienungsanleitung - Infrarotkamera.*

[52] HAMGEG Instruments, *Power Supply HMP4030 HMP4040.*

[53] H. Augustin, *Sensors for the Mu3e Pixel Detector*, PhD thesis, Heidelberg University, started in 2014.

[54] L. Huth, *The Mu3e Pixel Detector*, PhD thesis, Heidelberg University, started in 2014.

[55] J. Hammerich, *Characterisation of the MuPix8 (preliminary title)*, Master's thesis, Heidelberg University, started in 2017.

[56] I. Perić et al., *"Status of HVCMOS developments for ATLAS"*, J. Instrum., 2017.

[57] A. Herkert, *work title not fixed*, PhD thesis, Heidelberg University, started in 2015.

[58] S. Dittmeier, *Readout of the Mu3e Pixel Detector*, PhD thesis, University of Heidelberg, started in 2014.

[59] https://midas.triumf.ca/MidasWiki/index.php/Main_Page, accessed 2017-11-15.

[60] N. Berger, Internal Note 24 – MuPix8 Data Format, 2017.

Acknowledgements

At the end of this work, I would like to express my gratitude towards everyone who supported me to carry out this thesis.

First of all, I would like to thank **Prof. Dr. André Schöning** for giving me the opportunity to work in the Mu3e group and to take part in testbeam campaigns at DESY and PSI, and the DPG Spring Meeting in Münster.

I would also like to thank **Prof. Dr. Norbert Hermann** for being my second examiner.

My particular appreciation goes to **Sebastian Dittmeier**, who was a great tutor and supported me during my whole time in the Mu3e group.

Moreover, I would like to express my appreciation towards the following people in no specific order:

- **Dr. Dirk Wiedner** for his commitment on long-term planning and his advice on PCB design
- **Heiko Augustin** for his support on MuPix-related questions of all kind
- **Lennart Huth** for his valuable support on software and coding, and for critically questioned my methods and strategies
- **Jan Hammerich** and **David Immig** for sharing a lot of time in the lab and being great room mates on testbeam campaigns and conferences
- **Frank Schumacher** and his team from the electronics workshop for all their service and support
- The entire **Mu3e group** for the pleasant and humorous work atmosphere and all the fun we had beyond physics
- Last but not least, everyone who sacrificed some of their valuable time for proof-reading my thesis and proposing improvements

Erklärung:

Ich versichere, dass ich diese Arbeit selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, den 01.12.2017 . . . .