Department of Physics and Astronomy

University of Heidelberg

Master thesis

in Physics

submitted by

Fabian Alexander Förster

born in Schwetzingen

2014

HV-MAPS Readout and Direct Memory Access

for the Mu3e Experiment

This Master thesis has been carried out by

Fabian Alexander Förster

at the

Physikalisches Institut

under the supervision of

Dr. Niklaus Berger

HV-MAPS Auslese und Direct Memory Access für das Mu3e Experiment:

Das geplante Mu3e Experiment sucht nach dem Lepton-Flavor-verletztenden Zerfall $\mu^+ \to e^+e^-e^+$ mit einer Sensitivität, die besser als 1 in 10^{16} μ^+-Zerfällen ist. Dieser Zerfall hat im Standardmodell der Teilchenphysik ein Verzweigungsverhältnis von weniger als 10^{-54}, wodurch jedes beobachtete Ereignis ein klares Zeichen für neue Physik wäre.

Um den Zerfall $\mu^+ \to e^+e^-e^+$ von Untergrundereignissen zu trennen, wird der Impuls, Zerfallsort- und Zeitpunkt der Zerfallsprodukte möglichst genau gemessen. Da die Energie der Zerfallsprodukte maximal 53 MeV beträgt, wird die Impulsauflösung durch Vielfachstreuung dominiert. Um diese zu reduzieren, wurden dünnbare Monolithische Aktive Pixel-Sensoren in Hochspannungs-Technologie (HV-MAPS) entwickelt. In dieser Arbeit wurde im Rahmen der Entwicklung und Charakterisierung der HV-MAPS eine automatische Bestimmung des Spannungspulses der Pixelelektronik entwickelt und implementiert. Weiterhin wurde der Einfluss unterschiedlich starker Bias-Ströme in der Elektronik auf die Pulsform untersucht. Die Effizienz des Prototypen MuPix4 wurde in einem 5 GeV Elektronenstrahl auf nahezu 100% bestimmt.

Die geplante Sensitivität von 10^{-16} kann innerhalb von wenigen Jahren nur erreicht werden, wenn ein intensiver Myonenstrahl mit $2 \cdot 10^9 \mu^+$-Zerfällen pro Sekunde verwendet wird. Die dabei entstehenden Detektorereignisse mit einer Datenrate von 1 TBit/s werden mithilfe von rechenstarken Grafikkarten (GPUs) online rekonstruiert. Die Daten werden aus dem Detektor über optische Kabel zu FPGAs gesandt, die die Daten über PCIe in den Speicher der Grafikkarten schreiben. Um eine schnelle Datenrate zwischen FPGA und GPU zu ermöglichen wird die Methode des direkten Speicherzugriffs (DMA) verwendet, bei welchem ein Gerät ohne CPU-Interaktion in einen Speicherbereich schreiben kann. DMA wurde für ein Stratix IV FPGA über PCIe 2.0 implementiert, wobei eine maximale Datenrate von 3.5 GByte/s erreicht wurde. Als Test wurde diese Übertragung vier Tage am Stück durchgeführt, wobei insgesamt 1200 TByte Daten ohne Übertragungsfehler versendet und empfangen wurden.

HV-MAPS Readout and Direct Memory Access for the Mu3e Experiment:

The planned Mu3e experiment looks for the lepton flavor violating decay $\mu^+ \to e^+e^-e^+$ with a sensitivity of better than 1 in 10^{16} μ^+-decays. This decay has a branching ratio of less than 10^{-54} in the Standard Model – any observation of a signal would be a clear sign for new physics.

The decay $\mu^+ \to e^+e^-e^+$ can be separated from background by measuring momentum, vertex and time point of the decay particles. The maximum energy of the decay particles is 53 MeV, thus the momentum resolution is dominated by multiple scattering. In order to reduce this, thinnable High Voltage Monolithic Active Pixel Sensors (HV-MAPS) were developed. For the characterization of the HV-MAPS an automatic determination of the voltage signal in the pixel electronics was developed and implemented in this thesis. Furthermore, the influence of different bias-currents on the pulse shape was studied. The efficiency of the MuPix4 prototype was measured in an 5 GeV electron beam to be very close to 100%.

The high aimed sensitivity of 10^{-16} can only be achieved in a few years if an intense muon beam with up to $2 \cdot 10^9 \mu^+/\mathrm{s}$ is used. This creates a data rate of event information of 1 TBit/s, which has to be reduced online using powerful Graphical Processing Units (GPUs). The data stream, coming from the detector, is sent via optical links to FPGAs that write into the memory of the GPUs through PCIe. A fast transmission between FPGA and GPU is possible by using Direct Memory Access. This allows sending data without the interaction of the CPU. This was implemented for a Stratix IV FPGA using PCIe 2.0. The transfer was tested at a speed of 3.5 GByte/s for four days, transferring 1200 TByte of data without any transmission error.

„Long live the King"

— Scar
in: *The Lion King*

Contents

Part I

Theoretical Background and Introduction

Chapter 1

Theory

1.1 The Standard Model

The Standard Model of particle physics consists of 6 quarks and 6 leptons as the constituents of matter and their respective antiparticles (see Figure 1.1). Both quarks and leptons are arranged in three generations. The interactions between these particles are mediated by the gauge bosons.

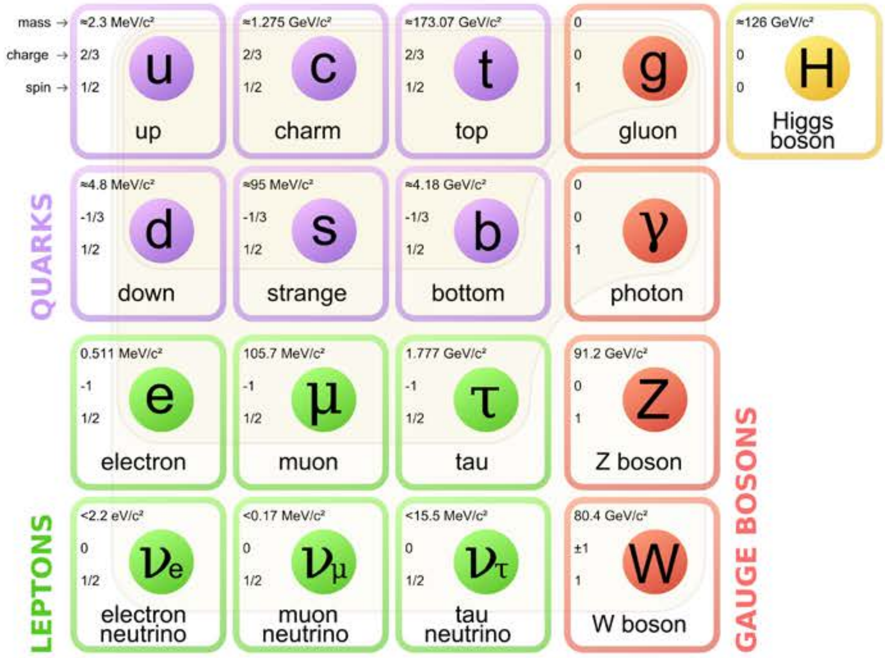


FIGURE 1.1: Elementary particles of the Standard Model [1].

The first generation of leptons consists of the negatively charged electron e^- and the massless neutral electron-neutrino ν_e. Each lepton generation has an associated lepton

flavor number, which is conserved. In this case, this is the electron lepton flavor number L_e. The second and third generation are arranged in an equivalent way. They consist of the muon μ^- and muon-neutrino ν_μ with lepton flavor number L_μ and tau τ^- and tau-neutrino ν_τ with lepton flavor number L_τ.
Quarks are also arranged in three generations. The first generation consists of the up quark u and the down quark d, the second generation of the strange quark s and the charm quark c and the third by the bottom quark b and the top quark t.
The eight gluons mediate the strong force, the photon is responsible for the electromagnetic interaction and W^+, W^- and Z^0 for the weak force. The gravitational interaction is not included in the Standard Model.
The discovery of a Higgs boson in 2012 by the ATLAS and CMS experiment at the LHC [2, 3] established the last missing particle in the Standard Model. The Higgs field causes the large masses of the weak gauge bosons W^+, W^- and Z^0 and gives mass to the other elementary particles.
The Standard Model however does not explain the change from a neutrino of one generation into a neutrino of another generation, which has been observed by Super-Kamiokande [4], SNO [5], KamLAND [6] and many others. This neutrino oscillation implies that neutrinos have a non vanishing mass. The Standard Model can be extended to contain this lepton flavor violating process by introducing a massive right-handed neutrino that gives a small mass to the left-handed neutrinos (seesaw mechanism [7]). This allows neutrino-oscillations, but still leaves open questions. The origin of Dark Matter, the matter-antimatter asymmetry of the universe, the lack of gravitation and why there are 3 generations are questions that are not answered by the Standard Model. This gives rise to new theories beyond the Standard Model.

1.2 Lepton Flavor Violating Muon Decays

The Mu3e experiment searches for the charged lepton flavor violating decay $\mu^+ \rightarrow e^+e^-e^+$. The extended Standard Model with neutrino oscillation allows for this decay (see Figure 1.2) at 1-loop level. The transition amplitude is suppressed by a factor of $\sim \left(\frac{\Delta m_\nu^2}{m_{W^+}^2}\right)^2$. The low mass difference between the neutrinos ($\mathcal{O}(0.01\,\mathrm{eV}/c^2)$) is much smaller than the mass of the W^+ ($80.4\,\mathrm{GeV}/c^2$); this results in an unobservably low branching ratio of BR $\ll 10^{-50}$ [8]. Any signal of the decay $\mu^+ \rightarrow e^+e^-e^+$thus would be a clear sign for new physics beyond the Standard Model (BSM).
Many BSM theories predict a higher branching ratio by introducing new particles, leading to new possible Feynman diagrams. The decay with a heavy super-symmetric (SUSY) particle like a slepton (Figure 1.3a) could lead to an observable branching ratio. At tree-level, possible lepton flavor violation can be mediated by a single particle coupling to e and μ (Figure 1.3b).

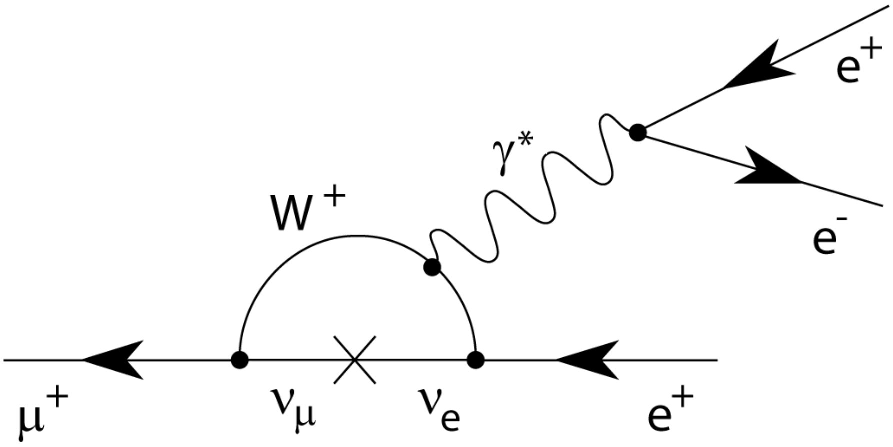
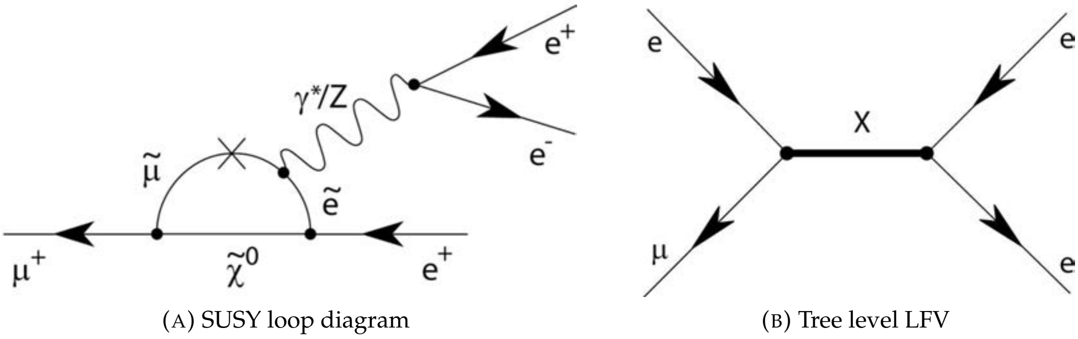


FIGURE 1.2: Feynman diagram for the decay $\mu^+ \rightarrow e^+e^-e^+$via neutrino mixing at 1-loop level [9].



(A) SUSY loop diagram

(B) Tree level LFV

FIGURE 1.3: Possible lepton flavor violating diagrams [9].

Chapter 2

The Mu3e Experiment

The Mu3e experiment searches for the charged lepton flavor violating decay $\mu^+ \rightarrow e^+e^-e^+$with a sensitivity up to one in 10^{16} μ^+-decays. This high sensitivity can only be achieved in a reasonable time if a high muon rate is used. The background can only be effectively suppressed by using a detector with high spatial, momentum and timing resolution.

2.1 Muon Decays

The muon is the charged lepton of the second generation with a mass of 105.659 MeV/c^2 and a mean lifetime of 2.197 µs [10]. Its small mass combined with charge conservation allows only for the decay into electrons, neutrinos and photons.

Lepton Flavor Conserving Decays:

The most likely decay (BR nearly 100%) is the so-called Michel decay $\mu^+ \rightarrow e^+\nu_e\overline{\nu}_\mu$. Other lepton flavor conserving decays are $\mu^+ \rightarrow e^+\gamma\overline{\nu}_\mu\nu_e$ (BR 1.4 %) and $\mu^+ \rightarrow e^+e^-e^+\overline{\nu}_\mu\nu_e$ (BR $3.4 \cdot 10^{-5}$) [10].

The Decay $\mu^+ \rightarrow e^+e^-e^+$:

If the muon decays at rest, the total energy E_{tot} of the final state has to be the rest mass of the muon:

$$E_{\text{tot}} = \sum_{i=1}^{3} E_i = m_\mu c^2 \tag{2.1}$$

An muon decaying at forces the decay particles to have a vanishing total momentum $\vec{p}_{\text{tot}}$:

$$\vec{p}_{\text{tot}} = \sum_{i=1}^{3} \vec{p}_i = \vec{0} \tag{2.2}$$

This restricts the energy ranges of the decaying electrons to be within $m_e c^2$ and 53 MeV, corresponding to half of the muon mass. A coincident signal from a common vertex of three particles with vanishing total momentum and total energy of $m_\mu c^2$ is the rare signal that the Mu3e experiment is searching for.

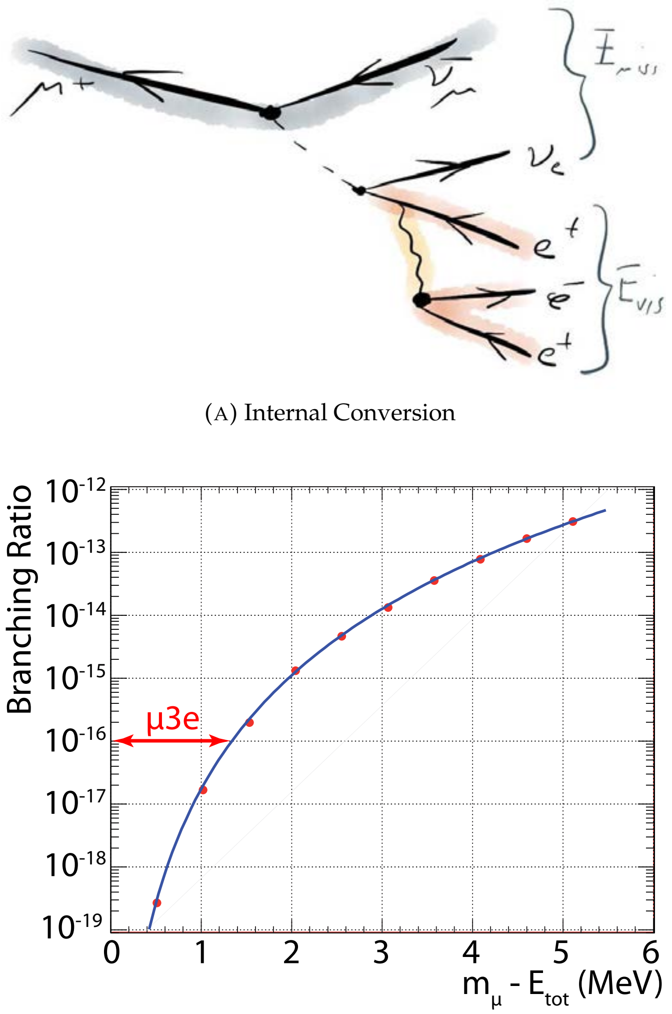
2.2 Background

Internal Conversion Background

The most severe background is a muon decay with an additional internal photon that creates an electron-positron pair (Figure 2.1a). Here, all decay particles are coincident in time and share one vertex, but the neutrinos carry away energy and momentum that is not measured in the detector. This background can be suppressed by having an excellent momentum resolution. The branching fraction as a function of the missing energy is shown in Figure 2.1b. At the aimed sensitivity of 10^{-16} the background is $\sim$ 1.4 MeV distant from the signal. Thus the invariant mass resolution has to be better than 1 MeV.

Accidental Background

Operating at high muon decay rates leads to another source of background: Two positrons from Michel decays and an electron from different sources can mimic the signal (see Figure 2.2). The electron could come from Bhabha scattering, a muon decay with internal conversion or a photon conversion process. Since all particles come from different decays, they will most likely not have the required momentum and energy conservation. Neither do they share a common vertex or are coincident in time. This background can be reduced by having a good vertex, time and momentum resolution.



(A) Internal Conversion

(B) Branching ratio of missing energy

FIGURE 2.1: (a) Michel-decay with an internal conversion to an e^+e^- pair. The missing momentum through neutrinos allows a suppression of this background by having a good momentum resolution. The branching ratio as a function of the missing energy due to neutrinos is shown in (b) from [11].

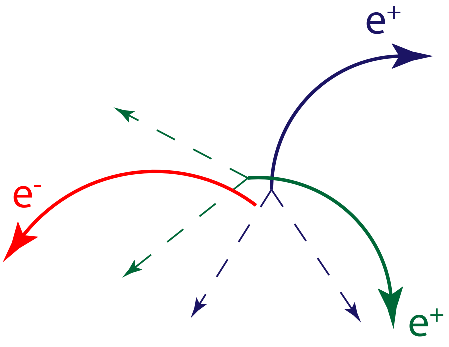


FIGURE 2.2: Accidental background: two positrons from Michel decays are combined with another electron and mimic the signal. The dashed lines indicate the neutrinos from the Michel decays.

2.3 Experimental Situation

The current upper limit for the decay $\mu^+ \to e^+e^-e^+$was set in 1988 by the SINDRUM experiment at PSI to BR($\mu \to eee < 10^{-12}$) (90% C.L.)[12].
Other LFV sensitive experiments are MEG searching for the decay $\mu \to e\gamma$ with BR($\mu^+ \to e^+\gamma$) $< 5.7 \cdot 10^{-13}$ (90% C.L.) [13] and SINDRUM II searching for the conversion $\mu \to e$ in presence of a nucleus with a BR($\mu Au \to eAu < 7 \cdot 10^{-13}$) (90% C.L.) [14].
A summary of the upper limits for different LFV decay modes versus time is shown in Figure 2.3.

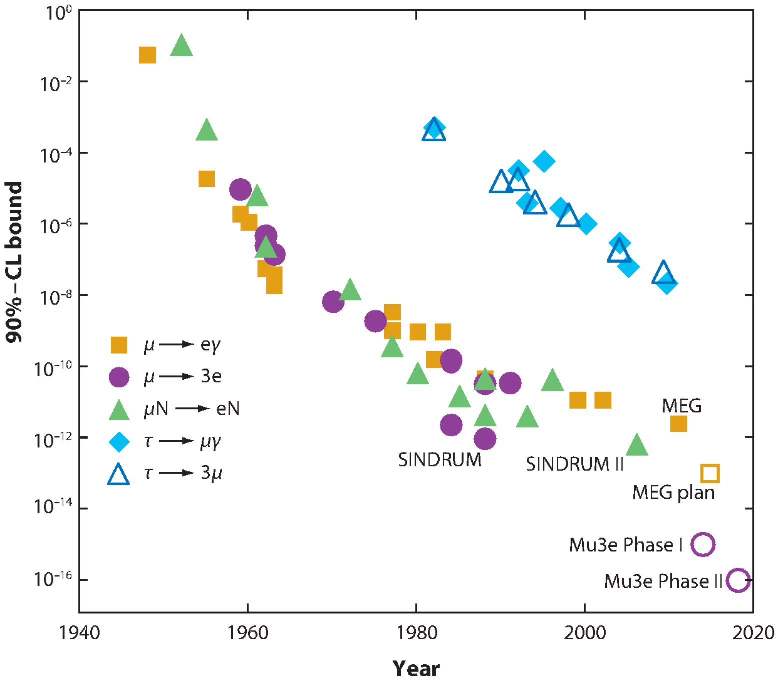


FIGURE 2.3: Summary of the experimental results from different experiments searching for LFV. Adapted from [8].

2.4 Muon Beam

The Mu3e experiment is planned to be carried out in two phases. The high targeted sensitivity requires a high muon-beam rate. This is available at the Paul Scherrer Institute (PSI) in Switzerland.
Phase I of the experiment aims for a sensitivity of 10^{-15} in a runtime of about 3 years. To reach this sensitivity, a muon rate of 10^8 Hz is required. The PSI offers such a beam in the experimental hall (Figure 2.4). A cyclotron produces a proton beam of 2.4 mA with an energy of 590 MeV. The beam hits the rotating carbon target E, producing pions. Resting pions decay and produce muons for the muon beam used in the πE5 beamline. This beamline provides muons with a momentum of 28 MeV/c at a rate of 10^8 Hz, which is used for phase I of the Mu3e experiment.

Phase II aims for a sensitivity up to 10^{-16}, thus a higher beam intensity is required. This is fulfilled by the planned high intensity muon beamline (HiMB) at PSI. Here, muons are produced at a spallation neutron target (SINQ, Figure 2.4). This beam is planned to deliver an intensity of up to $3 \cdot 10^{10}\ \mu/\mathrm{s}$ [9], where $2 \cdot 10^{9}\ \mu/\mathrm{s}$ are required for phase II.

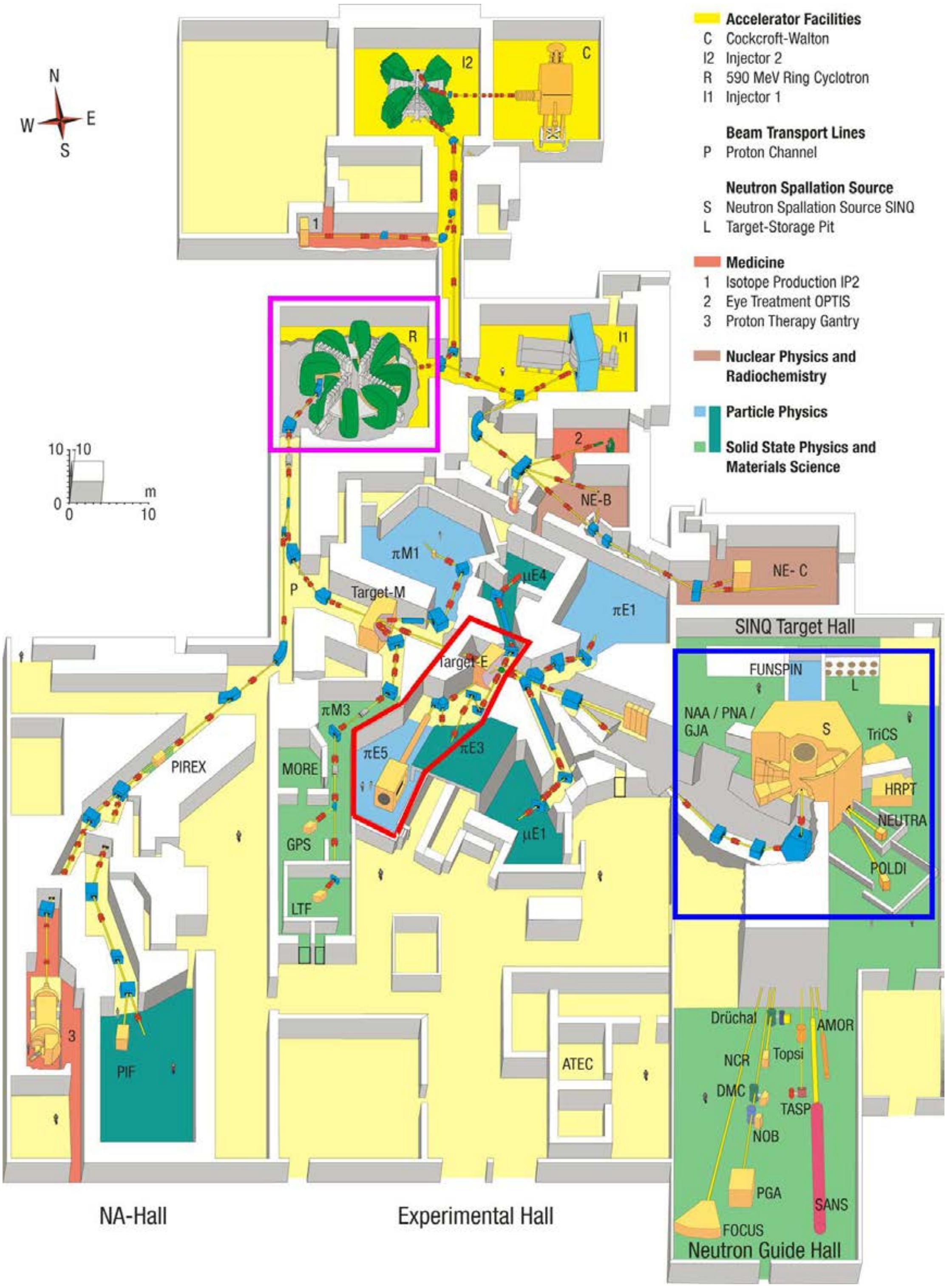


FIGURE 2.4: Experimental Hall of the Paul Scherrer Institute. The cyclotron (magenta) produces a 2.4 mA proton beam with a momentum of 590 MeV/c. In phase I of the Mu3e experiment, the muon beam generated from decaying pions in target E is used at the πE5 beamline (red). In phase II a high intensity muon beamline from the spallation neutron target SINQ (blue) is used.

2.5 Detector Design

The Mu3e detector [9] consists of a thin, hollow double cone aluminum target with a length of 100 mm and a diameter of 20 mm. Here, muons are stopped and decay on a large surface, allowing for better separation between vertices. The pixel detector consists of two double layers, where the inner layers determine the vertex and the outer layers measure the momentum through the bending of recurling tracks in a 1 T magnetic field. Scintillating fibres inside the central layers and scintillating tiles in the recurl layer yield a good timing resolution to suppress combinatoric background.

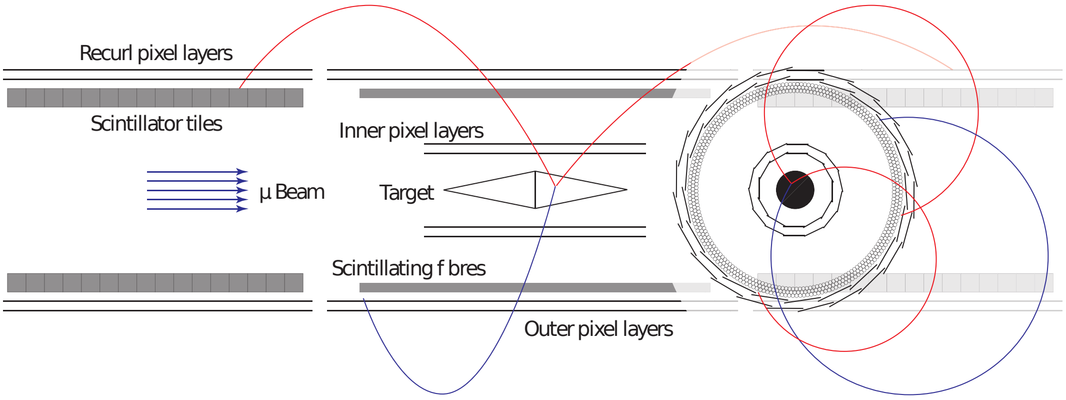


FIGURE 2.5: Longitudinal view of the Mu3e detector: Muon beam hitting the target. For signal events, stopped muons decay into three charged particles that are bent in a solenoidal magnetic field. On the right side, a transverse view in beam direction of the detector is shown.

2.5.1 Pixel Detector

The Mu3e pixel detector uses silicon High-Voltage Monolithic Active Pixel Sensors (HV-MAPS) [15, 16] thinned to 50 µm. They are placed upon two double cylinders of 50 µm thick Kapton foil. 15 µm thick aluminum traces on the self supporting Kapton structure allow powering and reading out the pixel sensors without adding much additional material. The central double layers allows the vertex reconstruction for decaying particles.

The momentum of electrons can be determined by their curvature in a magnetic field. The Mu3e experiment uses a solenoidal magnetic field of 1 T. The curvature of the electrons is not only measured by the inner and outer pixel layers at first passage, but also the recurling tracks are used. This is shown in the transverse projection of the detector in Figure 2.5. The recurl stations allow a first order cancellation of the error induced by multiple scattering as seen in Figure 2.6.

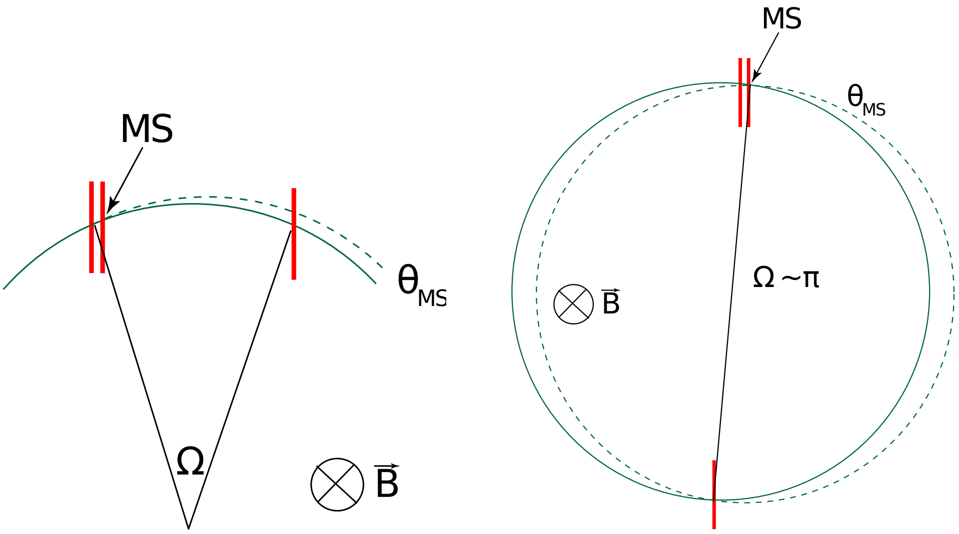


FIGURE 2.6: Left: the effect of multiple scattering induces an error in the track. Right: Both possible tracks due to multiple scattering yield approximately the same hit position after a half turn.

The pixel sensors will have a pixel size of 80 × 80 µm^2 – smaller pixel sizes are not necessary, since momentum resolution is dominated by multiple scattering in the detector material. Current prototypes of these pixel sensors are desribed in section 3.1.

2.5.2 Timing Detector

Fiber detector Multiple layers of scintillating fibres are placed inside the central outer double pixel layers. With a small material budget, they allow for a timing resolution better than $\sim$ 1 ns [17].

Tile detector Better time resolution can be achieved by using scintillating tiles, inside the recurling pixel layers. Since the recurl particles are stopped here, higher material budget can be chosen. They offer a time resolution of $\sim$ 0.1 ns [18, 19].

Part II

Mu3e Pixel Sensor

Chapter 3

High Voltage Monolithic Active Pixel Sensors

This chapter gives a detailed introduction to the High Voltage Monolithic Active Pixel Sensors (HV-MAPS) used in the Mu3e experiment.

3.1 High Voltage Monolithic Active Pixel Sensors

The Mu3e experiment requires a fast sensor with low latency jitter ($\sim$10–20 ns) and low material budget ($\sim 1‰$ of a radiation length per tracking layer). These requirements are met by the High Voltage Monolithic Active Pixel Sensors (HV-MAPS).
HV-MAPS are produced in a commercial 180 nm HV-CMOS process. N-wells are implanted on the p-doped wafer, each representing a pixel (see Figure 3.1). Applying a high voltage of $\sim$ 60 V between n-well and substrate increases the volume of the depletion zone to a thickness of $\sim$ 9 µm. Furthermore deposited charge from passing ionizing particles gets collected via drift – much faster than diffusion. This leads to a low latency variation, which is required for the Mu3e experiment.
Transistors can be implemented in the n-well of each pixel which can be used to integrate readout electronics, e.g. a charge sensitive amplifier (CSA). The integrated electronics of the HV-MAPS prototype will be discussed in detail in section 4.1.

Since the depletion zone of the HV-MAPS chip is $\sim$ 9 µm thick, the p-substrate can be thinned, resulting in a chip thickness of $\sim$ 50 µm. This thickness is low enough to reduce multiple scattering to an acceptable level for the Mu3e experiment.

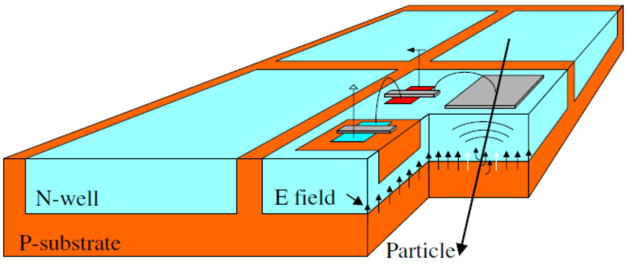


FIGURE 3.1: Pattern of a HV-MAPS showing four pixels with integrated readout electronics [15].

3.2 Interaction of Electrons with Silicon

The detection of a particle passing a layer of a detector relies on the interaction with the detector material. In the energy region relevant for Mu3e (10–53 MeV) electrons lose energy mostly through two electromagnetic interactions: inelastic Coulomb scattering and Bremsstrahlung.

Inelastic Scattering

The energy loss of heavy particles through interaction with shell electrons of the passing material is well described by the Bethe-Bloch formula. The low mass of electrons and positrons require a correction to this formula. A good description of the energy loss by inelastic scattering is given by the Berger-Seltzer formula [20]:

$$-\frac{\mathrm{d}E}{\mathrm{d}x} = \rho\frac{0.153536\,E}{\beta^2}\frac{Z}{A}B(T). \tag{3.1}$$

Here, ρ is the density of the medium in $\mathrm{g/cm^3}$, β is the speed of the particle in units of the speed of light, Z and A are the charge and nucleon number of the medium, and $B(T)$ is the stopping power in the medium for a certain kinetic energy T.

Bremsstrahlung

When a charged particle passes through matter it gets deflected by the fields of the nuclei. This deflection is stronger for lighter particles. The deflection leads to the emission of photons and therefore loss of energy. For relativistic particles this energy loss is given by:

$$-\frac{\mathrm{d}E}{\mathrm{d}x} = \frac{E}{X_0}. \tag{3.2}$$

Here, X_0 denotes the radiation length of the passed material, where passing particles drop to $1/e$ of their initial energy. HV-MAPS consist of silicon with a radiation length of $\sim 9.5\,\mathrm{cm}$.

Adding both energy losses yields an energy loss as shown in Figure 3.2. The energy loss differs slightly for electrons and positrons, since their stopping power in the Berger-Seltzer formula is different. In the Mu3e relevant electron energy region of 10–50 MeV an

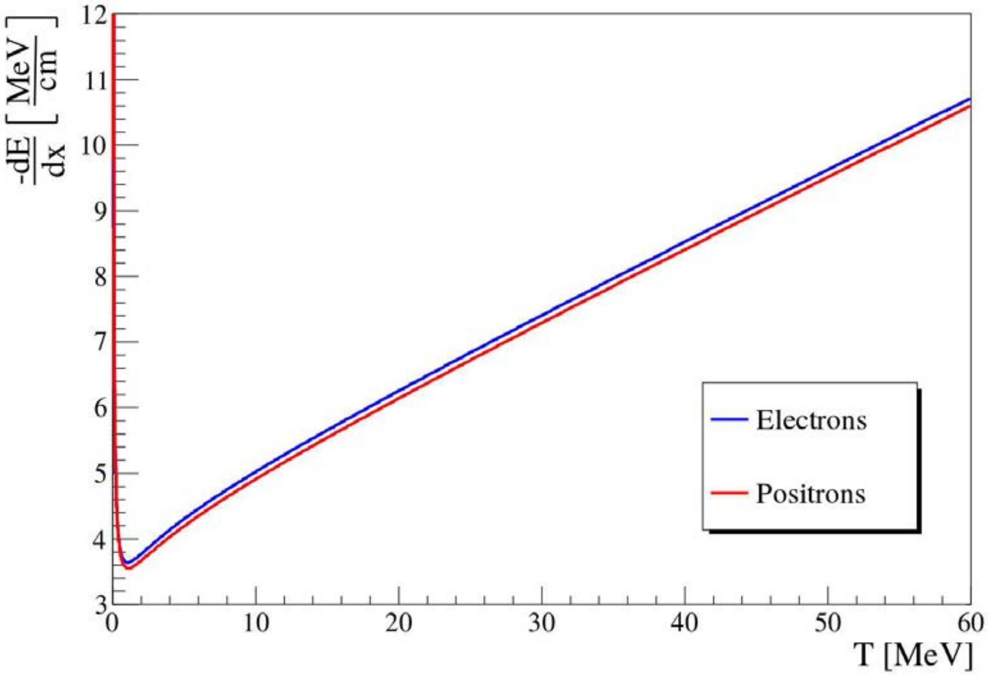


FIGURE 3.2: Energy loss of electrons and positrons in silicon due to Bremsstrahlung and inelastic scattering [21].

energy loss of 0.5–1 keV per µm of path length is expected. The mean required energy to create an electron-positron hole pair in silicon is 3.6 eV [22]. For the expected depletion zone of ∼9 µm this leads to the creation of 1250 – 2500 electron-positron pairs.

3.3 MuPix Pulse Shape

Electrons passing through the depletion zone of the MuPix chip create electron-hole pairs. These electrons are collected within $\sim 10^{-10}$ s [23] due to drift movement in the HV E-field. A charge sensitive amplifier (CSA) is then used to generate a signal of reasonable amplitude.

A CR-RC filter is displayed in Figure 3.3. If we assume that the charge deposition is a step-function-like input signal, the resulting output signal can be calculated. The expected output signal is described by Equation 3.3 [21]. Here U_0 is the amplitude of the input signal, $\tau_1 = C_1 R_1$ the time constant of the high pass and τ_2 of the low pass. Plugging in typical values for the MuPix4 chip results in a pulse shape as shown in Figure 3.4. Pulse shapes for different time constants are shown in Figure 3.5.

$$U(t) = U_0 \frac{\tau_1}{\tau_1 - \tau_2} \left(e^{-\frac{t}{\tau_1}} - e^{-\frac{t}{\tau_2}} \right) \tag{3.3}$$

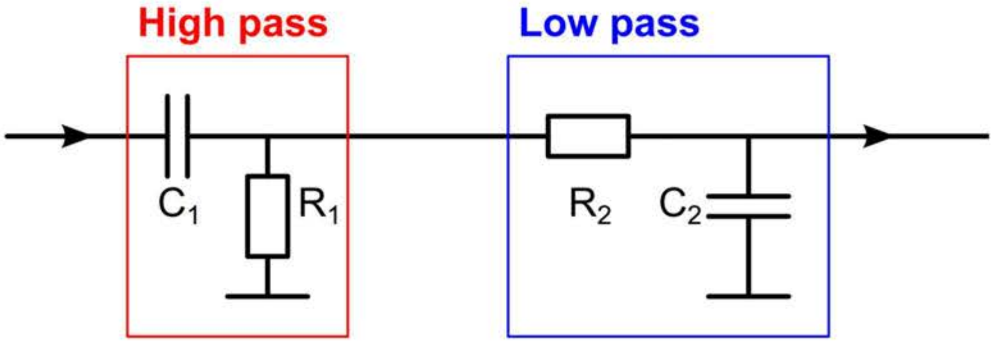


FIGURE 3.3: Circuit of the CR-RC filter that yields the pulse shape from [21].

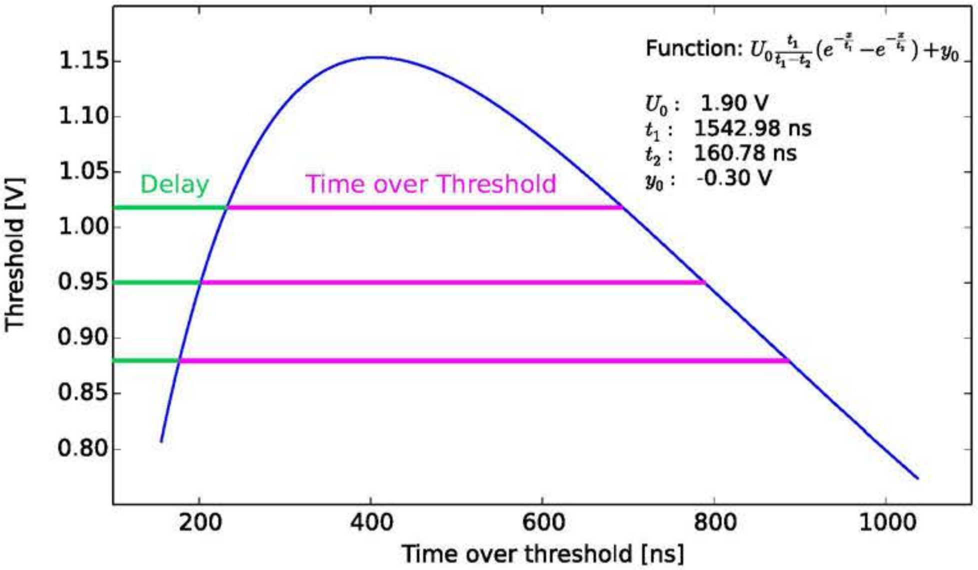


FIGURE 3.4: The resulting shape of a pulse after the CR-RC-filter. The signal can be described by defining the delay (green) and the Time over Threshold (magenta) for a certain threshold. Doing this for multiple thresholds yields the pulse shape. The parameters used are the result of a measurement with the MuPix4.

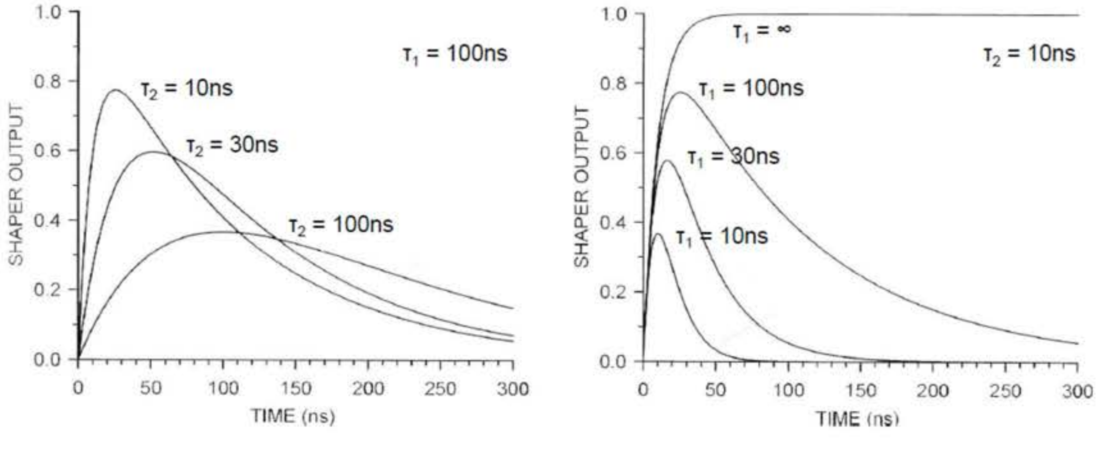


FIGURE 3.5: Change in pulse shape for different values of low pass time constant τ_2 (left) and high pass τ_1 (right) [24].

Chapter 4

Measurements

In this chapter the setup used to analyze the pulse shape of two MuPix versions is described. The test setup and software was successfully tested with the MuPix3. The MuPix3 chip did not allow a more precise measurement of the pulse shape due to a bug in the configuration electronics. A more advanced technique was used to determine the pulse shape of the MuPix4 chip, including the influence of bias voltages on the pulse shape.

4.1 Experimental Setup

For the determination of the pulse shape we used a MuPix3/4 chip glued and bonded to a ceramic carrier which is placed on a custom PCB[1]. The PCB is connected via two 40 pin flat ribbon cables to a Stratix IV PCIe development kit [25]. The FPGA works both as readout and slow control device.

Hits in the pixel cells are generated by a laser diode with a wavelength of 850 nm [26] centered above the MuPix chip. A single detached wire of the readout cable triggers a signal generator that powers the laser diode. The digital output of the comparator from a single pixel can be multiplexed to a pin – this output is called hitbus. Both the trigger of the LED and the hitbus output are linked to an oscilloscope. An overview of this setup is given in Figure 4.1. A closer look at the PCB with connections is given in Figure 4.2.

[1] Designed by Dr. Dirk Wiedner, Physikalisches Institut Heidelberg.

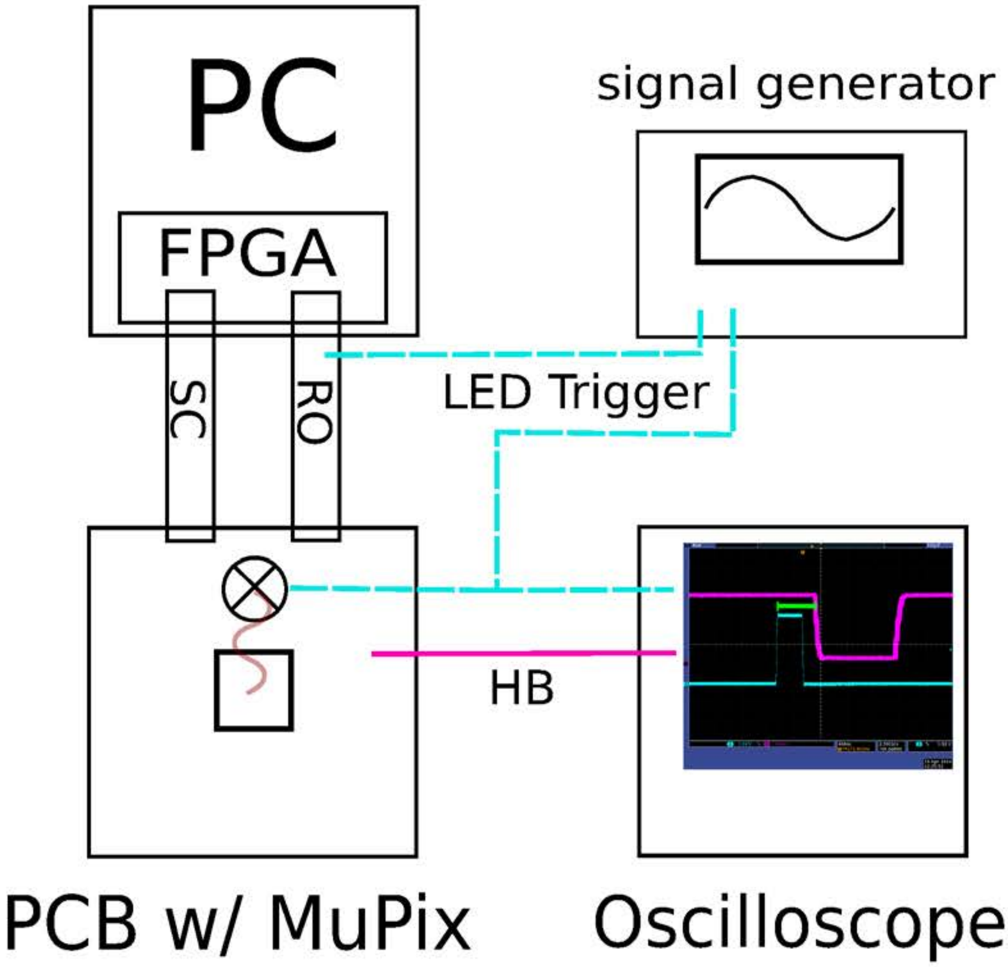


FIGURE 4.1: Setup for the determination of the pulse shape of the MuPix chip, with SC/RO: Slow control and Readout ribbon cable.

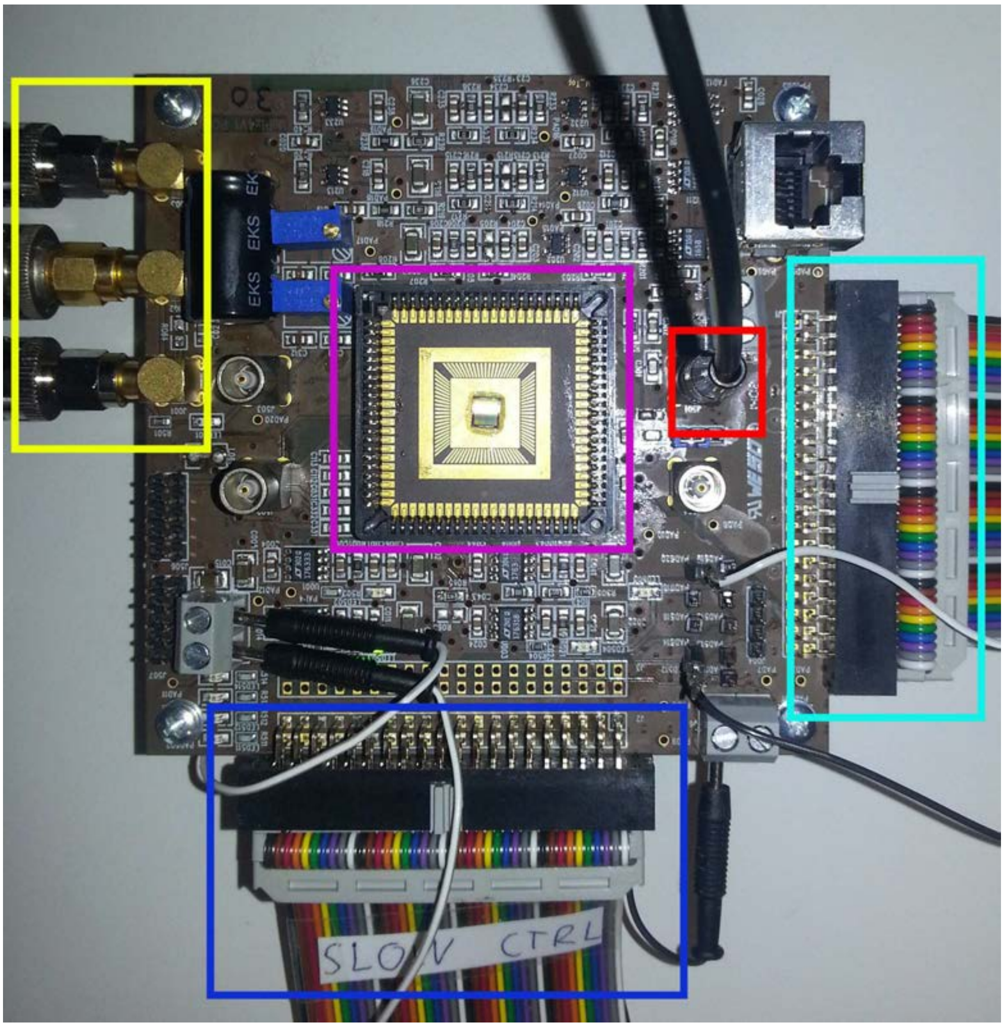


FIGURE 4.2: PCB containing a MuPix chip. Purple: MuPix chip glued and bonded on a carrier. Slow control (blue) and readout cable (teal) connected to the FPGA. The black and white wires are required for grounding. Yellow: power connection to the HAMEG power supply. Red: Hitbus output connected to the oscilloscope.

4.2 Pulse Shape Measurements

The CR-RC filter after each pixel (section 3.3) generates a pulse shape as shown in Figure 3.4. The delay is taken as the time between an arbitrary, fixed reference signal and the time at which the signal is above a certain threshold. We use the rising edge of the pulse generator signal activating the LED as the starting time, which roughly corresponds to the time of depositing charges into the selected pixel. The time over threshold (ToT) is the time difference between the signal rising and falling below a specific threshold.
In order to determine the pulse shape of a signal, one has to measure those two values at each threshold:

Time over threshold The ToT can be determined in several ways. The width of the signal is just the ToT. Since the injected charge is not always the same, the ToT fluctuates, and one needs to average these values. This can be done by taking the signal and looking at it on the scope or by sending the digital signal to the FPGA and counting the number of clock cycles ($f_{clock} = 50\,\mathrm{MHz}$) that the signal is high.

Delay The delay measurement requires more attention, since a reference point is needed. We split the signal that we used as a trigger for the LED and fed it into the oscilloscope. The time difference between the rising edge of this signal and rising edge of the ToT signal gives the delay.

4.2.1 MuPix3

As a first test, the MuPix3 was illuminated with a Laser LED at a constant threshold while sampling the Time over Threshold with the FPGA (see Figure 4.1). It has a binning of 20 ns resulting from the 50 MHz clock used. Looking at the histogram in Figure 4.4a a broad width is visible, which is due to a feature of the MuPix3 chip. The MuPix3 hitbus output is a logical OR of all activated pixel comparator outputs. Arbitrary pixels should be selectable by sending in a selection pattern. A bug in the configuration logic did not allow this, so only the full configuration with the logical OR of all pixels was tested.
Sampling the delay with the oscilloscope (Figure 4.3) yields also large time fluctuations – putting the values into a pulse shape results in Figure 4.4b. Higher thresholds than 0.98 V could not be measured, since the amplitude of the signal jitters and therefore activates and deactivates the hitbus signal at a high rate. The measurement seemed to work, but the feature of the MuPix3 chip lead to a bad resolution. Therefore no further investigations of the pulse shape of the MuPix3 chip has been done.

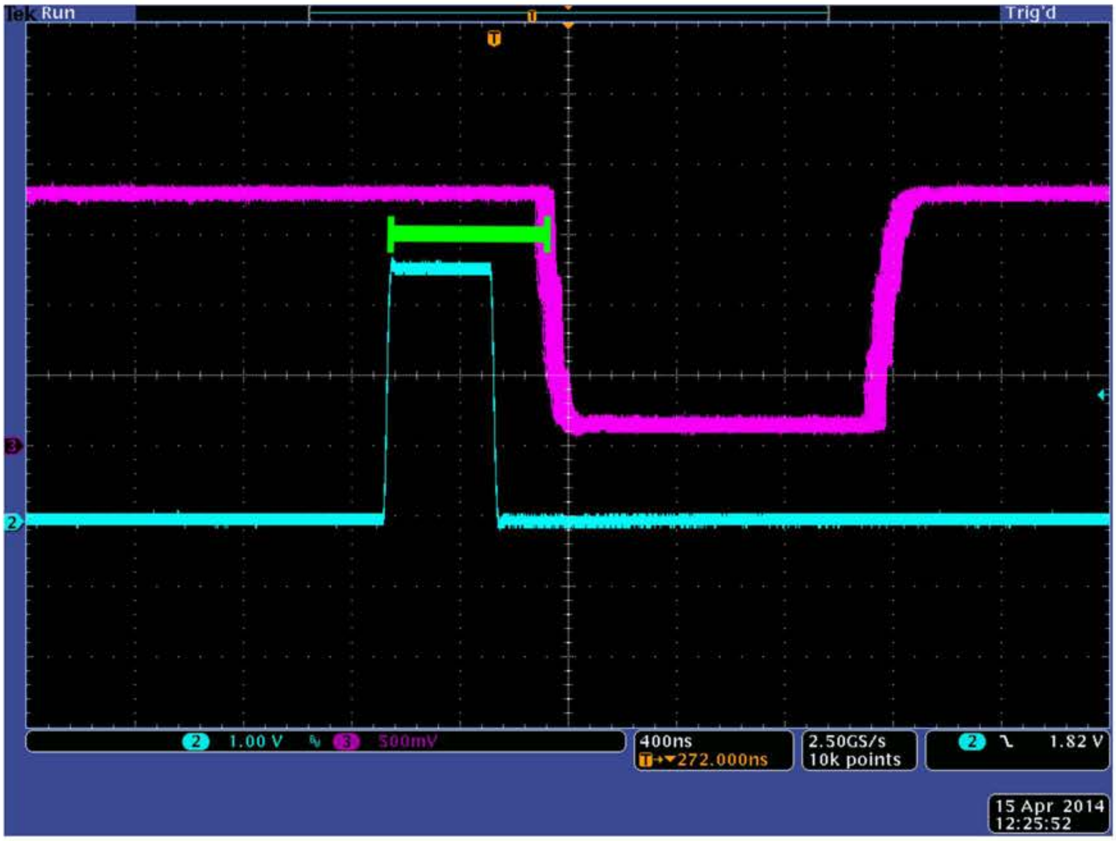
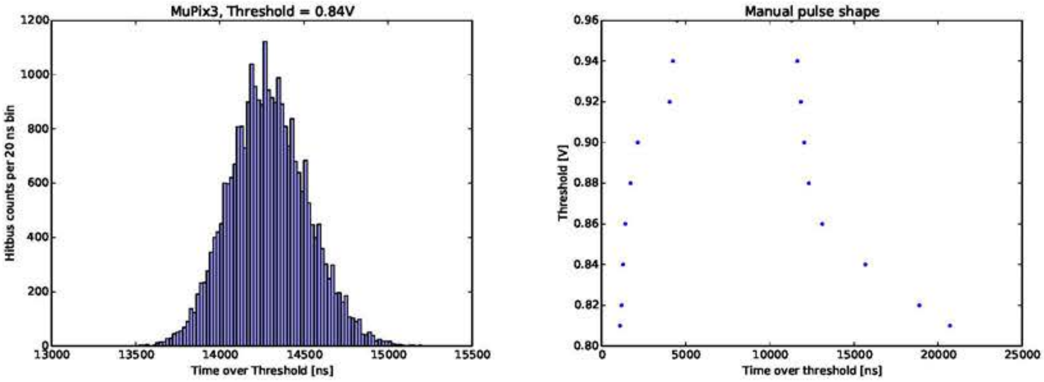


FIGURE 4.3: Teal: trigger for the LASER LED sent by the FPGA. Purple: hitbus analog output of MuPix chip - width represents ToT. Green: Time difference between trigger and ToT determines the delay.



(A) FPGA ToT histogram

(B) Manual pulse shape

FIGURE 4.4: Results of the manual pulse shape measurement: (a) The histogram of ToT measured by the FPGA. (b) The voltage threshold against the time over threshold shows the silhouette of the pulse shape. Due to the bad resolution of the delay, the shape is not smooth.

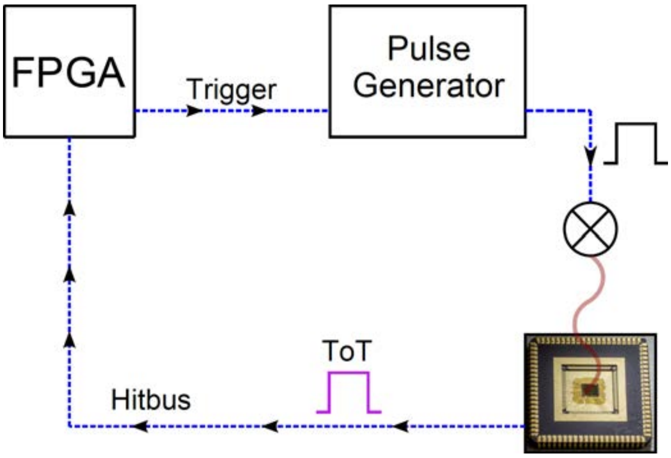


FIGURE 4.5: Measurement scheme for the pulse shape. The FPGA is sending the signal via the pulse generator to a LED which triggers the MuPix chip. The hitbus is then fed back into the FPGA.

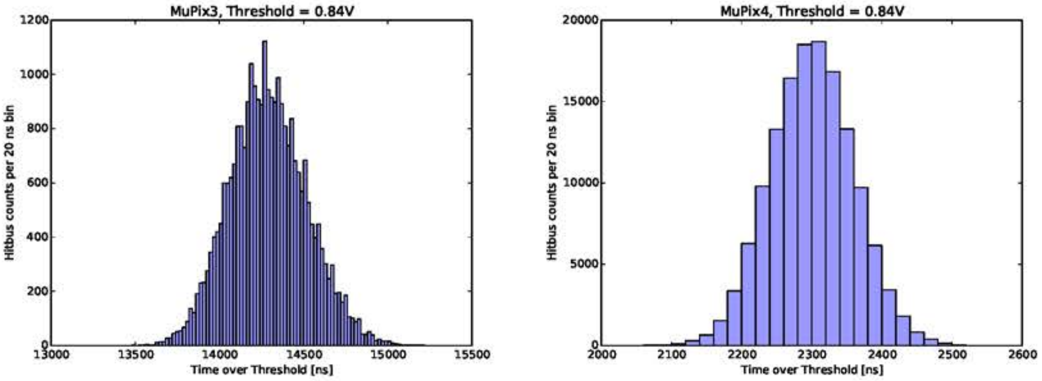
4.2.2 MuPix4

With the newer MuPix4 chip the hitbus output was selectable for a single pixel. This allowed more precise measurements of the pulse shape – a comparison of the ToT between MuPix3 and 4 is shown in Figure 4.6. The manual determination of the delay with the sampling function of the oscilloscope proved to be tedious when measuring a pulse shape with the help of multiple threshold values. A more elegant way was chosen: The FPGA triggers the pulse generator as seen in Figure 4.5, so that the laser diode can emit a light pulse onto the MuPix4 Chip. The hitbus signal is then read out by the FPGA. Since the FPGA sent out the starting signal, it has a reference point and can use the time difference between sending a signal and receiving the hitbus as delay.

There is another advantage of doing the measurement with the FPGA alone: It is also responsible for the slow control settings. Thus it can change the threshold after the determination of the ToT and delay has been done. This way, the pulse shape can be scanned in a few minutes, instead of taking several hours.

The resulting pulse shape can be seen in Figure 4.7. The FPGA itself uses a 50 MHz sampling clock, which results in a binning of 20 ns with a intrinsic timing resolution[2] of $\frac{20}{\sqrt{12}}$ns $\approx 6\,$ns.

[2] The variance of a uniform distribution with width a is given by $\frac{a^2}{12}$ [27].



(A) MuPix3 ToT histogram

(B) MuPix4 ToT histogram

FIGURE 4.6: Histograms of the Time over Threshold with a Laser LED for MuPix3 (a) and MuPix4 (b). The ToT is the logical or of all comparators in MuPix3, resulting in a much higher ToT than the MuPix4 ToT.

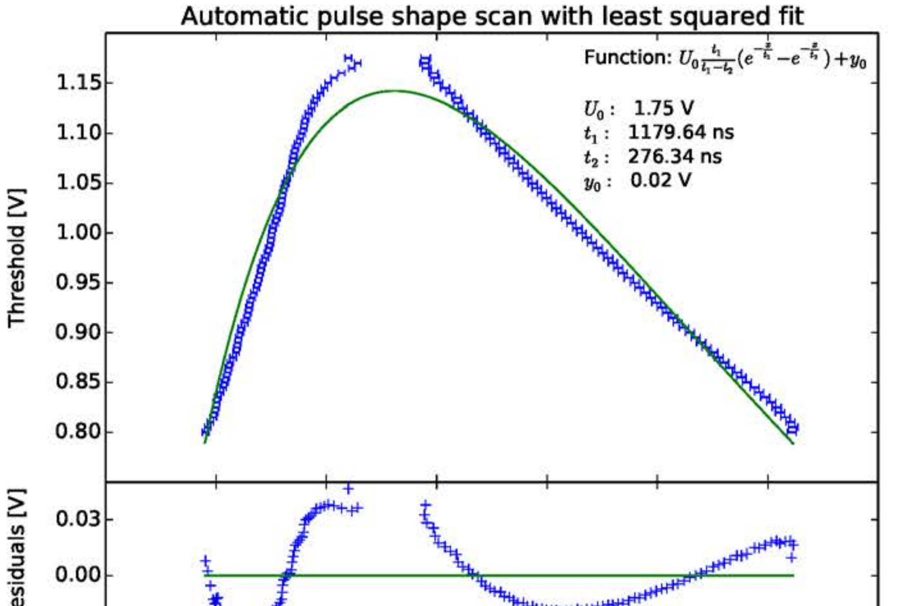


FIGURE 4.7: Results of the automatic pulse shape measurement. Since the FPGA has a sampling clock of 50 MHz, we have a timing bin of 20 ns which can be seen in the step behavior at the rising edge. The residuals show that the least squared fit shows systematic deviations from the fit function. The used fit function is the result of a step-function like charge deposition that only gets shaped by an ideal CR-RC filter, which is only a simplified model.

4.3 Chip DAC Influence on Pulse Shape

Now that the pulse shape can be precisely determined, the influence of internal currents on the shape can be studied. Thus we take a further look at the electronics in the pixel. The schematic of the electronics from [28] is displayed in Figure 4.8 – a further description is given in this section. Using DACs (digital-to-analog converters) implemented on the chip, we can control the current in certain elements of the chip. The chip contains 9 DACs, each represented by 6 bits. In the following the influence of each Chip DAC on the pulse shape is discussed and scanned, while it is compared with a simulation from Shruti Shreshta [29]. While varying one value, we kept the others at default values (Table 4.1) chosen by the chip designer.
Before the discussion of each Chip DAC is given, another element of the chip structure is introduced. The signal after the CR-RC filter of each pixel is compared to the same threshold voltage. Due to small differences between the pixels also the amplitude varies for different pixels, even though the signal was generated from equivalent deposited energy. This can be handled by introducing a small change in the threshold for each pixel, with the help of individual tune DACs (TDACs). While the MuPix4 allows this kind of tuning, the TDACs were not studied for the present work. Hence all chip DAC values controlling the influence of TDACs are not considered for the pulse shape studies.

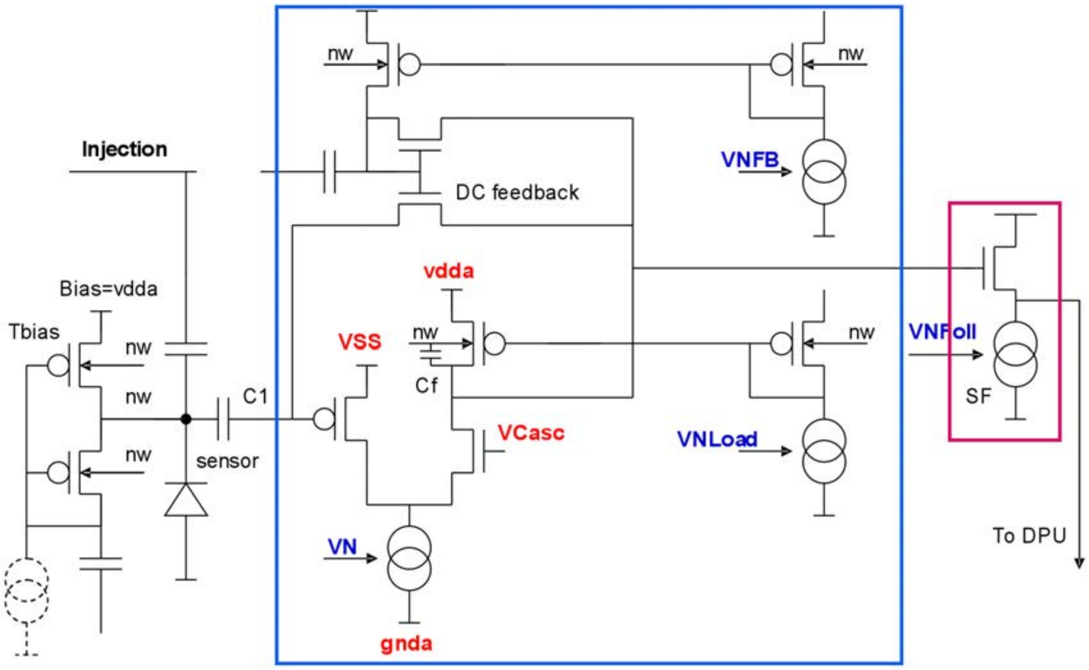
VNLoad, VNFoll, VN All of these control voltages increase the amplification in the CSA and therefore result in larger pulses. Comparing the measurements with the simulation, one can see the expected behavior: see Figure 4.9, Figure 4.10 and Figure 4.11.

VNFB Regulates the current in the feedback system, which stabilizes the amplifier. This is visible in the falling edge of the signal, which is described by $t_2 = RC_2$. Higher currents lead therefor to a faster falling time. Figure 4.12 shows this effect both in simulation and measurement.

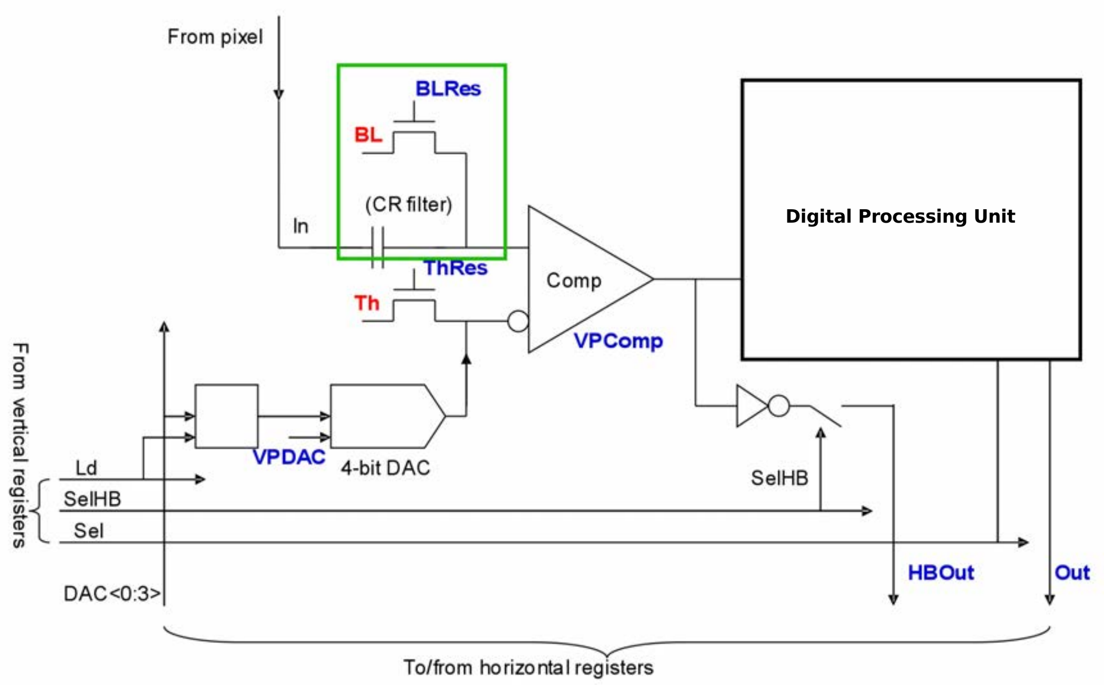
VCasc The main purpose of this DAC is to keep the potential of the amplifier at a nearly constant level. This DAC can not be adjusted by slow control. The potential is generated on the MuPix-PCB instead with a rotary potentiometer.

BLRes The **B**ase**L**ine **Res**torer controls the current that flows off the second capacitor in the CR-RC filter. Higher values in BLRes lead to a higher flow and therefore shorter pulse shapes. This changes the ToT from 1–2 µs. A comparison between measurement and simulation is shown in Figure 4.13.

VPDAC Acts as multiplier for the influence of the TDACs of all pixels. Higher values in VPDAC increase the influence of each tuning. Since our setup was not using TDACs, no influence on the pulse shape could be measured.

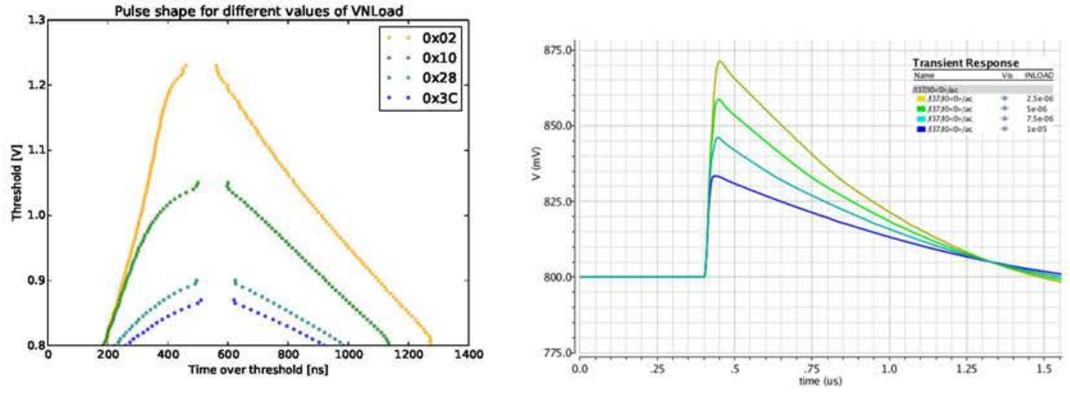


(A) Pixel electronics



(B) Digital part

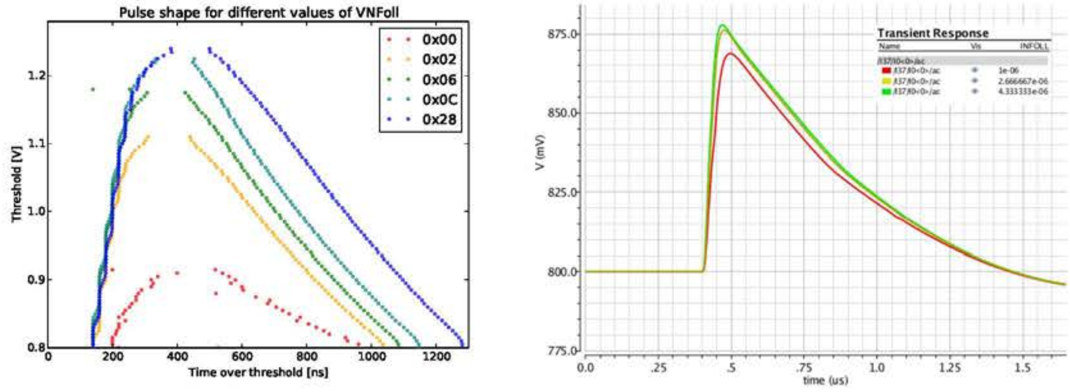
FIGURE 4.8: (a) Pixel electronics with CSA (blue) and source follower (red). (b) digital part with CR-RC filter (green), comparator and Digital Processing Unit (DPU varies from MuPix3 to MuPix4)[28].



(A) Measurement

(B) Simulation

FIGURE 4.9: Variation of pulse shape with different values of VNLoad. Low to high values are: yellow, green, teal and blue. (a) shows the measured variance of the pulse shape. The simulation (b) yields likewise results. VNLoad acts as an amplification of the initial signal and therefore higher pulse shapes are achieved. The simulated pulse shape has even for all values the same rising speed.



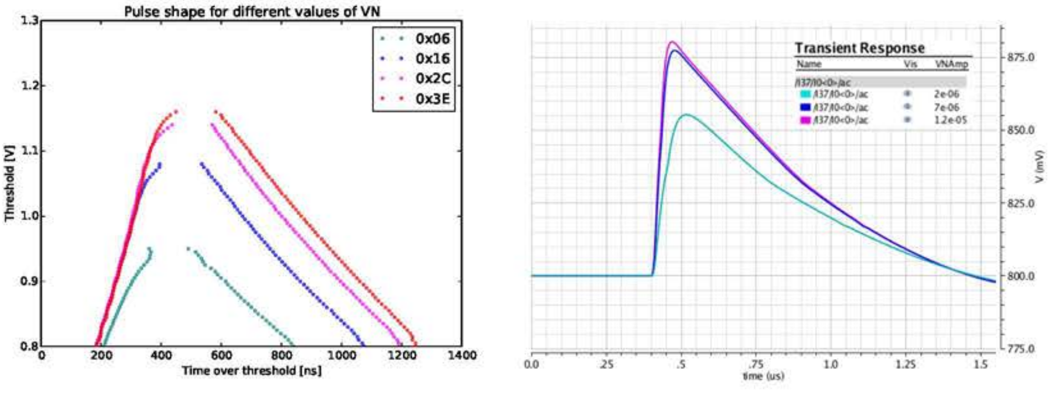
(A) Measurement

(B) Simulation

FIGURE 4.10: Variation of pulse shape with different values of VNFoll. In the measurement (a) we have low to high values for VNFoll: red, yellow, green, teal and blue. In the simulation (b) it is red, yellow and green. The DACs in the simulation have a weaker influence on the shape. This can be explained by the small simulated current range. The rising edge in the measurement has step like behavior. This is the result of an unintentional added cable on the PCB that caused reflections.

THRes Amplifies the influence of each Tune DAC, which increases the threshold for each pixel. Since the control of Tune DACs was not implemented, no influence could be measured.

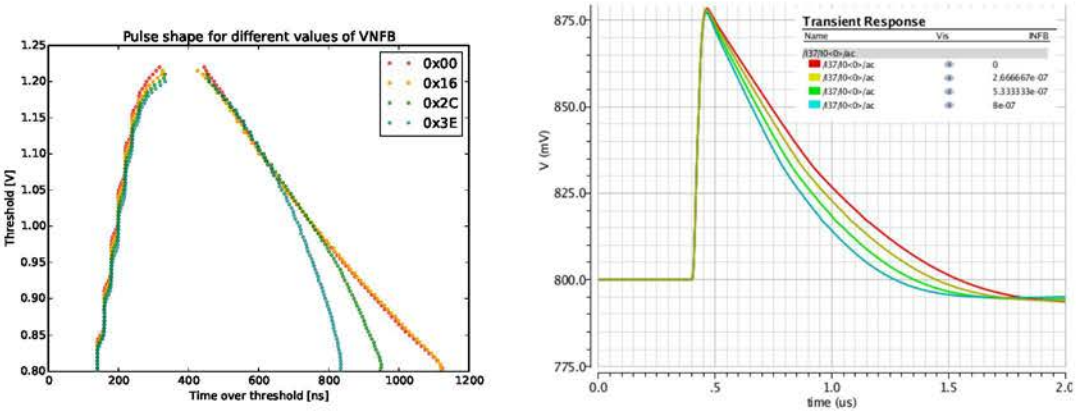
VPComp Controls the current of the comparator. Higher values increase the gain. In order to produce an output signal, it needs a current that differs from 0 – the resulting pulse shape is seen in Figure 4.14. Too small values increase the delay of the pulse shape but as soon as it is above 0x0E, the pulse shapes match. For power savings a moderate value is therefore recommended. The comparator doesn't have the property of delaying in the simulation.



(A) Measurement

(B) Simulation

FIGURE 4.11: Variation of pulse shape with different values of VN. Low to high values of VN (teal, blue, magenta and orange (measurement only)) result in a shift of the baseline to higher values in (a) . The simulation (b) shows the same behavior.

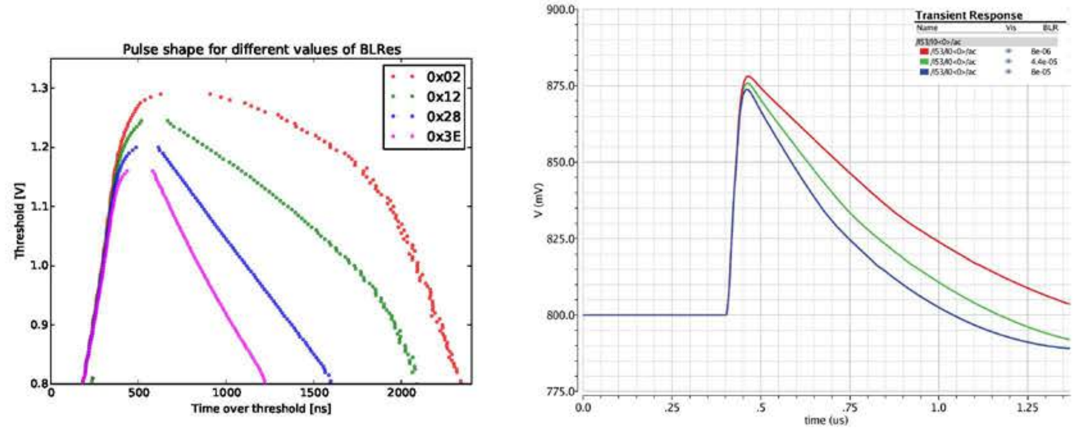


(A) Measurement

(B) Simulation

FIGURE 4.12: Variation of pulse shape with different values of VNFB. Low to high values are: red, yellow, green and teal. In both measurement (a) and simulation (b) the falling tail is shorter for higher VNFB values, even though the slope of the simulated shape gets lower near the baseline. This measurement also had the additional cable plugged, causing reflections and the step-like rising edge.

VNDel Controls the edge detector after the comparator. This current has no influence on the pulse shape.



(A) Measurement

(B) Simulation

FIGURE 4.13: Variation of the pulse shape with different values of BLRes. Low to high values are: red, green, blue and magenta (measurement only). (a) shows the measured variance of the pulse shape. The simulation (b) shows similar influence on the amplitude – it is not possible to select a certain 6 bit value in the simulation, but a specific current has to be chosen. This can not be done in the same range as the chip DACs. Thus the dependency appears to be weaker.

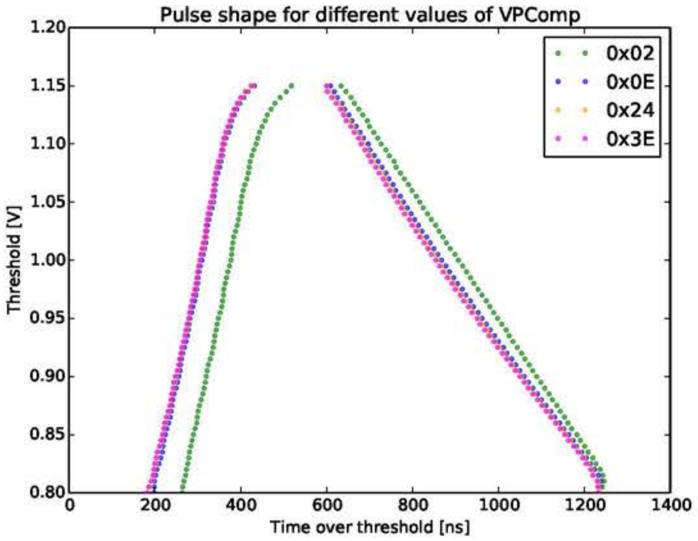


FIGURE 4.14: Variation of pulse shape with different values of VPComp. Low to high values are: green, blue, yellow and magenta. Even relatively low values of the current (blue) allow fast timing.

Chip DAC	Value[hex]
VPDAC	`0x0`
VPComp	`0x3C`
VNDel	`0xA`
VNLoad	`0x5`
VNFoll	`0xA`
VNFB	`0xA`
VN	`0x3C`
THRes	`0x3C`
BLRes	`0x3C`

TABLE 4.1: Default Chip DAC values as chosen by Ivan Perić.

Chip DAC	observation
VNLoad	increases amplification
VNFoll	increases amplification
VN	increases amplification
BLRes	quicker falling edge
VPDAC	TDAC not implemented
THRes	TDAC not implemented
VNFB	quicker falling edge
VPCOMP	non-zero value needed, else no influence
VNDel	no change

TABLE 4.2: Overview of Chip DACs influence on pulse shape.

4.4 MuPix Efficiency Measurement at DESY

4.4.1 Setup

The DESY testbeam line T22 offers an electron beam with an energy of 1–6 GeV [30]. The maximum beam rate is 1.8 kHz at a beam energy of 3 GeV. One goal of this testbeam was the efficiency determination of the MuPix4 chip – thus a low rate and low multiple scattering at an electron energy of 5 GeV was chosen. The efficiency can be measured by using a telescope that reconstructs particle tracks as a reference. In T22 this is the EUDET telescope, which consists of 6 planes of MIMOSA26 chips. These are MAPS chips with a spatial resolution of 2–3 μm [31]. Combining scattering effects in the EUDET planes with individual MIMOSA26 resolution, a track resolution of 15 μm at the device under test is available [32]. This is better than the individual pixel size of the MuPix4 chip with $92 \times 80\,\mu\text{m}^2$. The used setup with the EUDET telescope and the MuPix4 is shown in Figure 4.15. The chip is here rotated in a 45° angle, giving a bigger effective depletion zone in the pixels and thus resulting in a higher efficiency. The hit information from all 7 chips is read out and stored in two time frames of the MIMOSA26 chips of $2 \cdot 115\,\mu\text{s}$. The small size of the MuPix4 results in a hit rate in the order of a few Hz in the MuPix4, thus allowing a matching of EUDET tracks with MuPix4 hits.

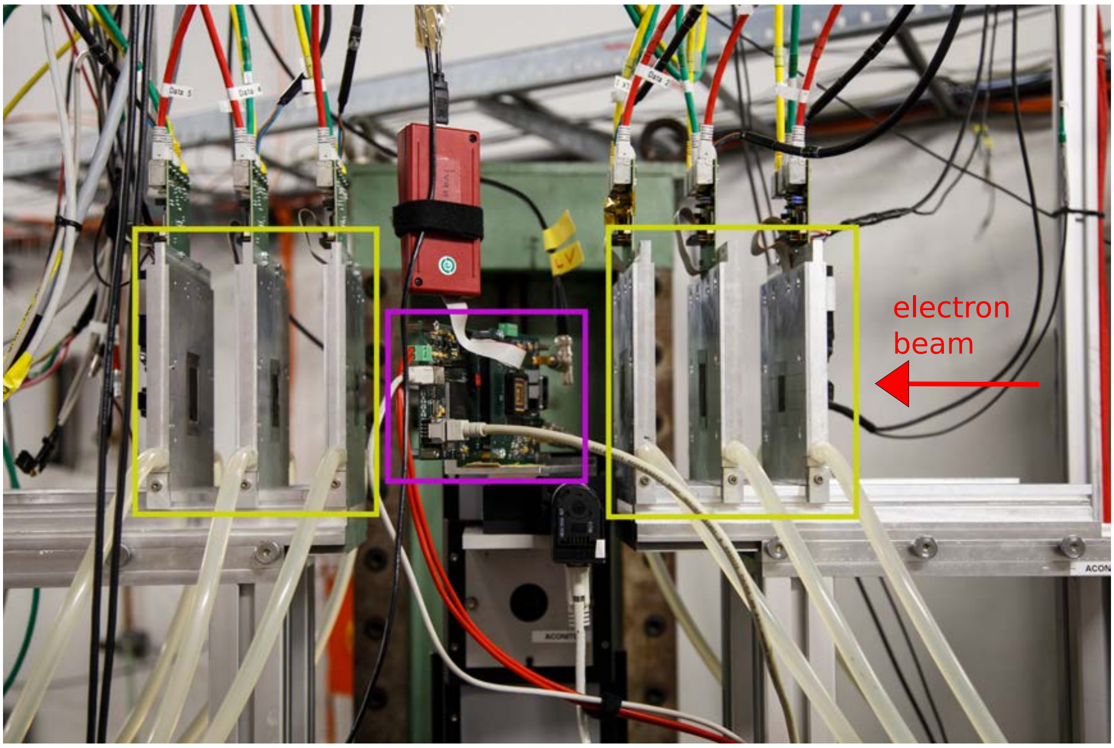


FIGURE 4.15: DESY testbeam setup. The MuPix chip on the PCB (purple) is put as a device under test in a 45° angle between the six planes of the EUDET telescope (yellow). The electron beam of 5 GeV is coming from the right (red).

4.4.2 Results

The electron tracks are reconstructed using the six planes of the EUDET telescope. This allows a determination of the MuPix4 pixel that should be hit by this electron. An electron track is successfully matched if this pixel has a hit information stored. The efficiency ϵ for each pixel is then calculated by:

$$\epsilon = \frac{N_{\text{matched}}}{N_{\text{tot}}} \tag{4.1}$$

where N_{matched} is the number of successful matching and N_{tot} the total amount of reconstructed tracks through this pixel. The resulting efficiency for the MuPix4 chip is shown in Figure 4.16. The mean efficiency of each column and row is also included. If the efficiency for a certain column or row has two dashes instead of three, the mean efficiency for it is $\sim$ 100%. The overall efficiency of the Chip is very close to 100%.

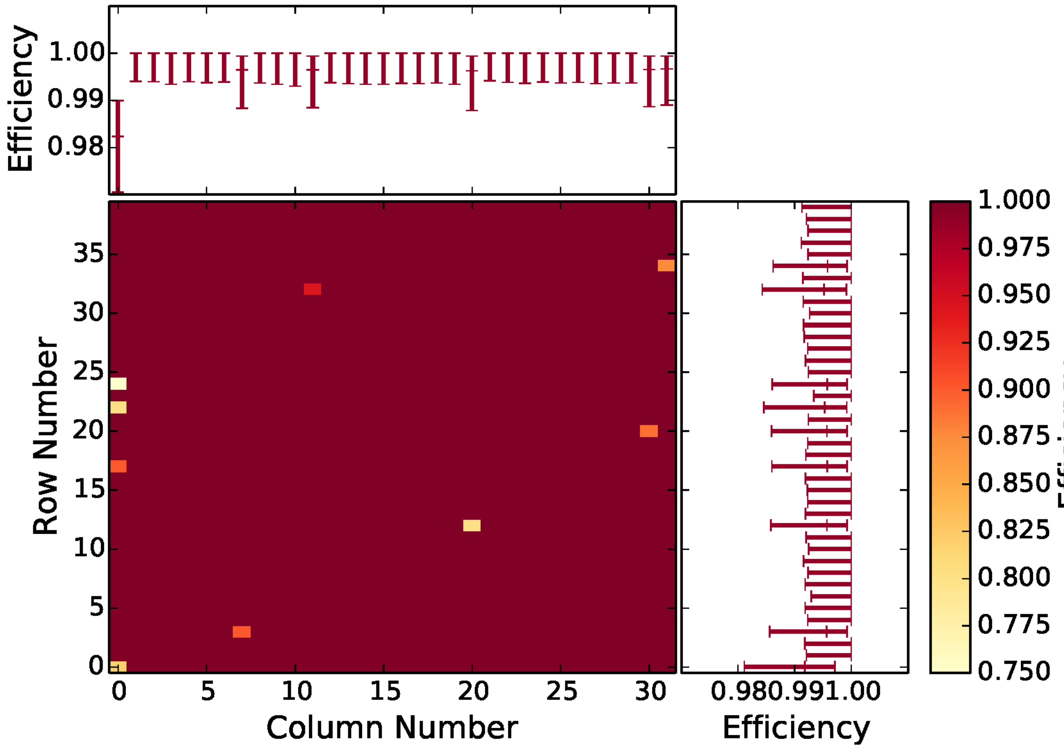


FIGURE 4.16: Efficiency of each MuPix4 pixel. The mean efficiency for each column and row is displayed [32].

Chapter 5

MuPix Interface

5.1 Main Window

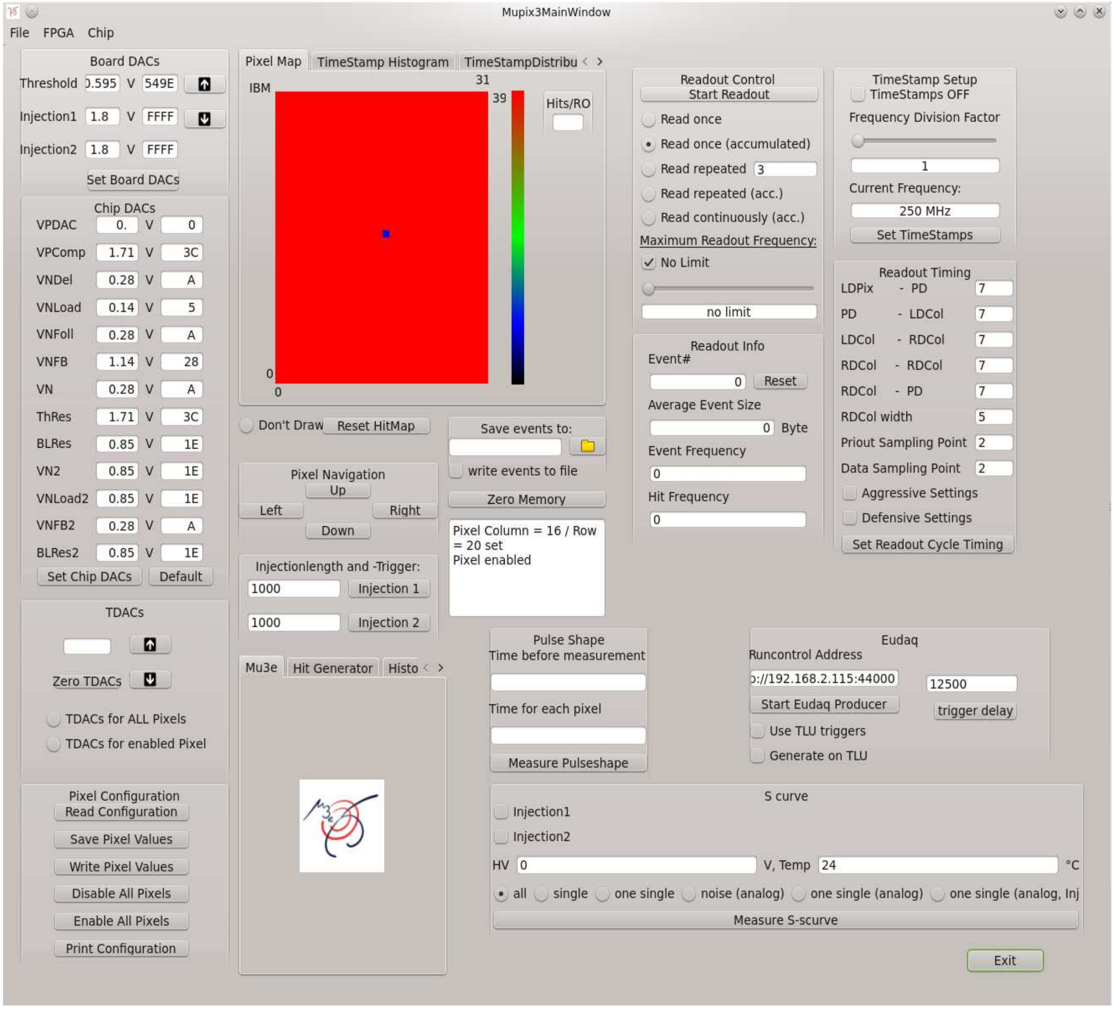
A graphical user interface (GUI) has been built for the readout and control of the MuPix chip (Figure 5.1). The features of this interface are described in this chapter.

Board DACs *Set Board DACs* sends DACs located on the PCB. This includes the threshold, which compares the output signal of the CR-RC filter to the selected threshold. The baseline is usually around 0.8 V. For Laser LEDs maximum thresholds of 1.25V can be chosen, at which the peak of the signal is reached. Injection1 deposits charge in the odd double-rows (rows $4n + 2$ and $4n + 3$). Injection2 does the same for the even double rows (rows $4n$ and $4n + 1$). Since these rows have an address readout problem in the MuPix4 version, all of them are mapped to row number 0 and 1. The amplitude voltage of the charge deposition can be set here. Values can be changed on the fly, but are only set when pressing *Set Board DACs* or when using the up and down buttons.

Chip DACs Every 6 bit value controls the maximum current for specific function. The function of each DAC is described in section 4.3. *Set Chip DACs* sends them to the specific MuPix chip, while *Default* loads the standard values see Table 4.1. Additional DACs (VNLoad2, VNFB2, BLRes2) are required for MuPix6, since it uses a two-stage amplifier. This is not used for MuPix4.

TDACs Tune DACs are used to increase the threshold for a single pixel (in contrast to the Board DAC threshold, which is global for all pixels). This way the efficiency of each pixel can be increased and 'hot pixels' may be muted.

Pixel Configuration Allows the saving, loading TDAC configurations as well as sending these configurations to the chip.



Board DACs

Threshold 0.595 V 549E

Injection1 1.8 V FFFF

Injection2 1.8 V FFFF

Chip DACs

VPDAC	0.	V	0
VPComp	1.71	V	3C
VNDel	0.28	V	A
VNLoad	0.14	V	5
VNFoll	0.28	V	A
VNFB	1.14	V	28
VN	0.28	V	A
ThRes	1.71	V	3C
BLRes	0.85	V	1E
VN2	0.85	V	1E
VNLoad2	0.85	V	1E
VNFB2	0.28	V	A
BLRes2	0.85	V	1E

TDACs

○ TDACs for ALL Pixels
○ TDACs for enabled Pixel

○ Don't Draw

Save events to:
□ write events to file

Pixel Column = 16 / Row = 20 set
Pixel enabled

Injectionlength and -Trigger:
1000 Injection 1
1000 Injection 2

Readout Control

○ Read once
● Read once (accumulated)
○ Read repeated 3
○ Read repeated (acc.)
○ Read continuously (acc.)

Maximum Readout Frequency:
☑ No Limit
no limit

Readout Info
Event# 0
Average Event Size 0 Byte
Event Frequency 0
Hit Frequency 0

TimeStamp Setup
□ TimeStamps OFF
Frequency Division Factor 1
Current Frequency: 250 MHz

Readout Timing

LDPix - PD	7
PD - LDCol	7
LDCol - RDCol	7
RDCol - RDCol	7
RDCol - PD	7
RDCol width	5
Priout Sampling Point	2
Data Sampling Point	2

□ Aggressive Settings
□ Defensive Settings

Pulse Shape
Time before measurement
Time for each pixel

Eudaq
Runcontrol Address ›://192.168.2.115:44000
12500 trigger delay
□ Use TLU triggers
□ Generate on TLU

S curve
□ Injection1
□ Injection2
HV 0 V, Temp 24 °C
● all ○ single ○ one single ○ noise (analog) ○ one single (analog) ○ one single (analog, Inj

FIGURE 5.1: Mainwindow for MuPix readout and control.

Pixel map During readout this area shows the amount of hits in each pixel. Red is the color of the pixel with the most hits, dark blue/black corresponds to 0 hits. Once *Don't Draw* is selected, the pixel map will not update, resulting in a much lower CPU load. Another available display mode is TimeStamp Histogram that puts all the timestamps of all measured hits in a histogram. TimeStampDistribution averages the time stamp for each pixel and displays it in the pixel map.

Pixel Navigation Allows an alternative navigation through the pixel map.

Save events to: Writes the hit information to the selected file.

Zero Memory Wipes both read and write FPGA memories.

Pulse shape Starts the pulse shape scan as described in section 4.3. The scanned Chip DACs have to be selected in the source code.)

Readout Control Starts the readout of the pixel hits with the following options:

- *Read once*: Saves the hit information only for this readout period.

- *Read once (accumulated)*: Accumulates the hit information.
- *Read repeated*: Repeats N times and accumulates the hits of all readouts. Afterwards, this information is deleted.
- *Read repeated (acc.)*: Accumulates the data for N readouts, the entries are kept.
- *Read continuously (acc.)*: Reads hit information with a adjustable frequency and accumulates their hit information.
- *Maximum Readout Frequency*: Allows controlling the frequency that the FPGA uses to send readout signals. Lower frequencies can be used to test the influence for crosstalk.

Readout Info Counts the number of events, the average event size and the event and hit frequency in Hz.

TimeStamp Setup Timestamps are generated on the FPGA using a gray counter with the selected frequency and sent to the chip.

Readout Timing The MuPix chip is read out sending multiple signals. The time difference between tehse signals in 50 MHz clock cycles between these can be adjusted here.

Eudaq Used for testbeams at DESY to interact with control software.

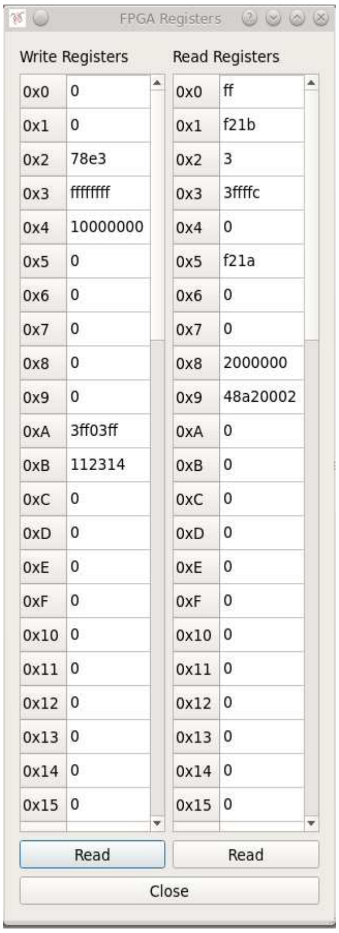
S curve Used as a test environment for determination of the S-curve. This counts the ratio of pixel hits per injection for different thresholds. A complete description including measurements is given in [33].

5.2 FPGA Registers

The control registers are accessible through the main window and represent two memory areas of the FPGA, where each memory consists of $64 \cdot 32$ bit, see Figure 5.2:

Write Registers Controls the behavior of the FPGA – each cell is changeable by the user. The task of each 32 bit register can be changed by each version of the FPGA. This includes reading out the chip, setting injections to the chip or resetting the FPGA memory. Internal notes contain the complete description of each register.

Read Registers Memory that is used by the FPGA to give specific information to the user. This can be version number, or the address of last written event. An up-to-date description can be found in the Mu3e online encyclopedia.



Write Registers		Read Registers	
0x0	0	0x0	ff
0x1	0	0x1	f21b
0x2	78e3	0x2	3
0x3	ffffffff	0x3	3ffffc
0x4	10000000	0x4	0
0x5	0	0x5	f21a
0x6	0	0x6	0
0x7	0	0x7	0
0x8	0	0x8	2000000
0x9	0	0x9	48a20002
0xA	3ff03ff	0xA	0
0xB	112314	0xB	0
0xC	0	0xC	0
0xD	0	0xD	0
0xE	0	0xE	0
0xF	0	0xF	0
0x10	0	0x10	0
0x11	0	0x11	0
0x12	0	0x12	0
0x13	0	0x13	0
0x14	0	0x14	0
0x15	0	0x15	0

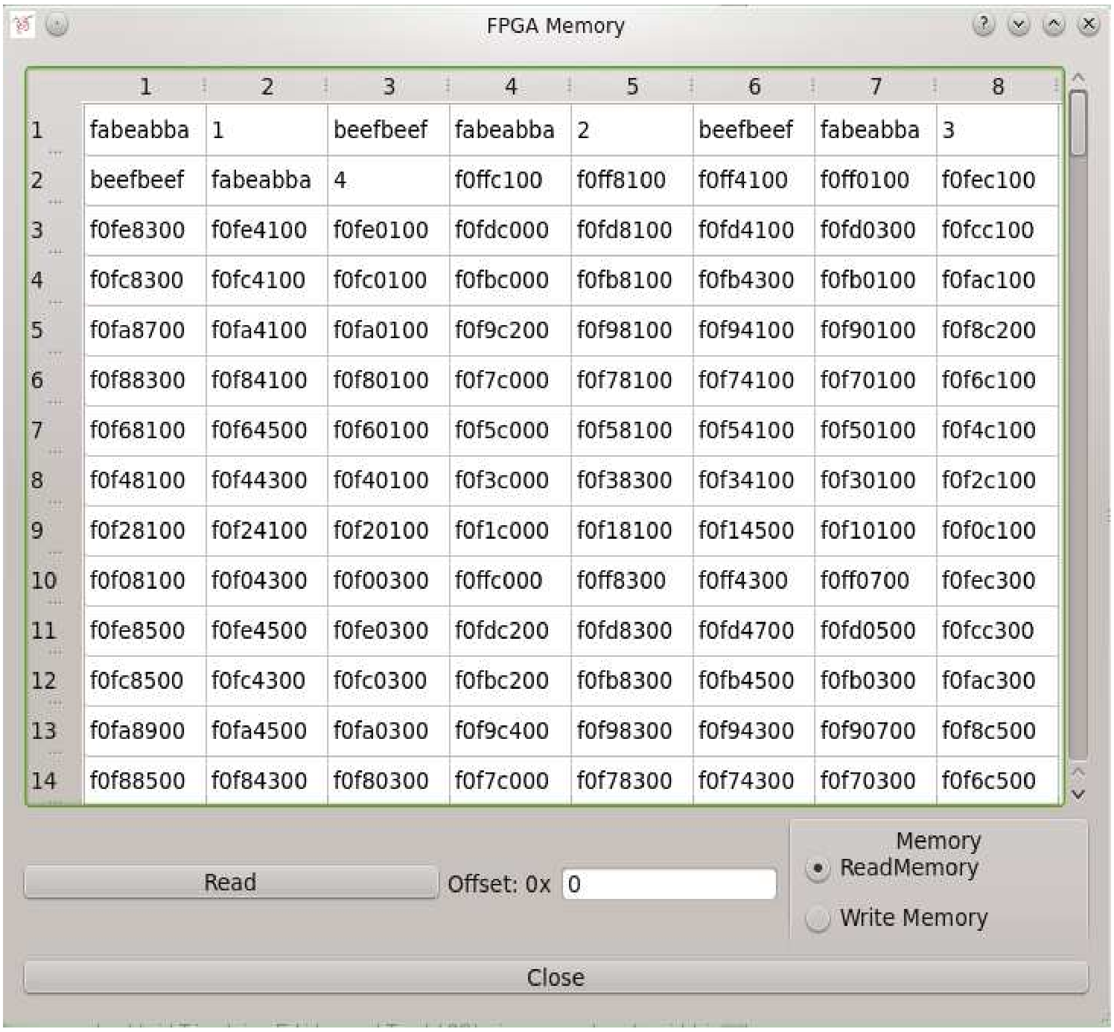
FIGURE 5.2: FPGA register window. Controls properties in behavior of the FPGA.

5.3 FPGA Memory

There are two memory blocks located on the FPGA each with a memory size of $2^{16} \cdot 32$ bit $= 256$ kByte. The writeable memory area is currently not used, the readable area contains events that are read out by the MuPix chip. Each event has the following format:

- Start of event pattern: `0xFABEABBA`
- 32 bit event number
- Hits with `0xF0F`, Column (6 bit), Row (6 bit), Timestamp (8 bit)
- End of event pattern `0xBEEFBEEF`

Note that this event structure varies for newer FPGA firmware versions, since they have to take a second MuPix chip into account (for the MuPix telescope, c.f. [34]).
The memory acts as a ringbuffer memory – the memory position of last written event can be read off from a FPGA read register value.



	1	2	3	4	5	6	7	8
1	fabeabba	1	beefbeef	fabeabba	2	beefbeef	fabeabba	3
2	beefbeef	fabeabba	4	f0ffc100	f0ff8100	f0ff4100	f0ff0100	f0fec100
3	f0fe8300	f0fe4100	f0fe0100	f0fdc000	f0fd8100	f0fd4100	f0fd0300	f0fcc100
4	f0fc8300	f0fc4100	f0fc0100	f0fbc000	f0fb8100	f0fb4300	f0fb0100	f0fac100
5	f0fa8700	f0fa4100	f0fa0100	f0f9c200	f0f98100	f0f94100	f0f90100	f0f8c200
6	f0f88300	f0f84100	f0f80100	f0f7c000	f0f78100	f0f74100	f0f70100	f0f6c100
7	f0f68100	f0f64500	f0f60100	f0f5c000	f0f58100	f0f54100	f0f50100	f0f4c100
8	f0f48100	f0f44300	f0f40100	f0f3c000	f0f38300	f0f34100	f0f30100	f0f2c100
9	f0f28100	f0f24100	f0f20100	f0f1c000	f0f18100	f0f14500	f0f10100	f0f0c100
10	f0f08100	f0f04300	f0f00300	f0ffc000	f0ff8300	f0ff4300	f0ff0700	f0fec300
11	f0fe8500	f0fe4500	f0fe0300	f0fdc200	f0fd8300	f0fd4700	f0fd0500	f0fcc300
12	f0fc8500	f0fc4300	f0fc0300	f0fbc200	f0fb8300	f0fb4500	f0fb0300	f0fac300
13	f0fa8900	f0fa4500	f0fa0300	f0f9c400	f0f98300	f0f94300	f0f90700	f0f8c500
14	f0f88500	f0f84300	f0f80300	f0f7c000	f0f78300	f0f74300	f0f70300	f0f6c500

Offset: 0x 0

Memory
● ReadMemory
○ Write Memory

FIGURE 5.3: FPGA memory window. Shows the saved data that are stored in the FPGA memory.

Part III

Direct Memory Access

Chapter 6

Background

The aimed sensitivity in branching ratio below 10^{-16} requires the Mu3e experiment to run at high rates of $2 \cdot 10^9$ muon decays per second. The resulting data rate of approximately 1 TBit/s for the whole detector needs to be transferred out of the detector, while the readout system has to fulfill strict space and material budget requirements. The generated data will be filtered in strong Graphical Processing Units (GPUs), which perform track fits using massively parallel computing. This way 10^9 track reconstructions/s can be achieved, leading to a data reduction of a factor ~ 1000.
The hit information from the detector reaches the PC containing the GPU via optical links connected to a FPGA. This FPGA has to send the data through the Peripheral Component Interconnect Express (PCIe) to the memory of the GPU. Full usage of the bandwidth of the PCIe bus can only be achieved when performing this data transfer without the usage of a CPU. This is called Direct Memory Access (DMA).
As a first test, this data transfer is performed into CPU memory instead of GPU-RAM, since it is easier to allocate and possible transfer errors can be safely detected and the expected transfer speed is the same.
In this chapter, the readout scheme of the Mu3e experiment is introduced. Furthermore the two different data transfers from FPGA to GPU (polling and DMA) are described.

6.1 Mu3e Readout Scheme

The readout scheme of the Mu3e experiment is described for phase I of the experiment (see Figure 6.1). MuPix sensors are linked via Low Voltage Differential Signaling (LVDS) aluminum links on Kapton flexprints to the front-end FPGAs. Each front-end FPGA performs the slow control for several MuPix sensors. The MuPix sensors send zero-suppressed hit information to the front-end FPGA. This data is merged, time-sorted and sent via optical links to two readout-boards. These are located outside of the detector, and send the hit information from the whole detector of a small time slice to the FPGA inside a PC with a powerful GPU. In the GPU, online track- and event-reconstruction is done, reducing the data rate by a factor of ∼ 1000. In order to perform the data transfer from the FPGA to the GPU via PCIe this interface has to be studied in detail.

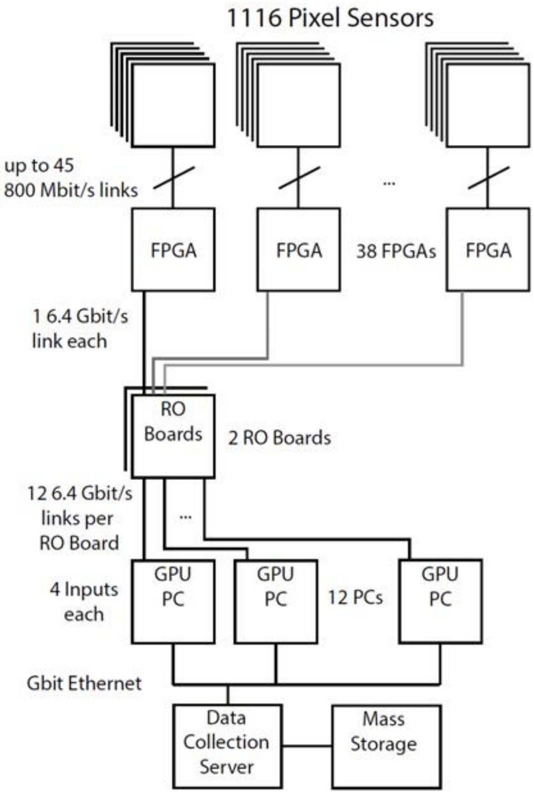


FIGURE 6.1: Phase I readout scheme. MuPix sensors sending zero-suppressed data via LVDS links to the front-end FPGAs. Time-ordered data is merged and sent via optical links to the readout-boards located outside the detector. The readout-baords send the hit information of the whole detector for a small time slice to one GPU PC, where data reduction is performed.

6.2 Memory Management

Processes in a modern computer do not have access to the physical address in the memory that they operate. This would quickly lead to fragmentation of the memory and the allocation of large continuous memory blocks would be impossible. Instead a virtual memory is used that is much bigger than the physical memory. The memory management unit (MMU) maps the addresses of the virtual memory to physical memory. A continuous block of virtual memory can be mapped to different parts of memory. This mapping is shown for one process in Figure 6.2.

Modern operating systems segregate the available memory in a computer in kernel space and user space:

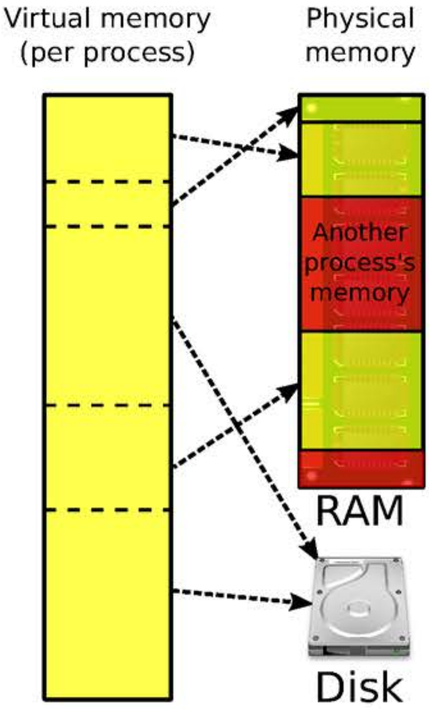


FIGURE 6.2: The used memory of an application appears to be continuous, however the memory management unit maps this memory to physical memory which is not continuous [35].

Kernel Space The kernel, kernel extensions and device drivers use kernel space. They can access both virtual and physical memory addresses. This makes it very powerful but also dangerous. Changes in memory that is currently used by another program is memory corruption.

User Space Instances of applications run in user space. They can only access their own designated virtual memory.

6.3 Data Transfer

PCIe is not a bus, where the physical wires of several devices are merged and passing through a shared slot. PCIe is a Point-to-Point network where every device is connected to a switch. Connecting these switches creates a network where data can be sent. Data has to be in packages of a predefined structure. The packages for reading and writing are described in this section. PCIe uses a differential pair of wires for both the reading and writing direction per lane. This allows PCIe to both read and write at the same time. Up to 16 lanes can be bundled into one connection. The speed per lane is shown in Table 6.1.

PCIe version	Transfer Speed per lane
v1.x	250 MB/s
v2.x	500 MB/s
v3.0	985 MB/s
v4.0	1969 MB/s

TABLE 6.1: PCIe transfer speed per lane per direction[36].

Write request:
The header of each write packet consists of 3 to 4 32-bit words (depending whether a 32- or 64-bit addressing scheme is used). This includes an indication that the package is a write packet, the write address for the first 32-bit word, how many bits of the first 32-bit word should be written and the amount of sent 32-bit words. A full description of the packet head is given in [36]. The remainder of the package are the data words that should get transferred.

Read request:
A read request always consists of sending two packages: one read package that demands data from a device and the completion package as an answer. The read package contains a marker that it is a read package, a unique package number and the ID of the device that sent the request. This allows the target device to answer this request and allows an identification of the data once they arrive.
A successful answer consists of IDs for the requester, completer and the sent package. It also includes the amount of sent data (as a crosscheck), the origin address of the data and the data itself.

6.3.1 Polling

This data transfer is performed by the CPU and fulfills the rules of a **read request**. Here, a user space program wants to receive data from a device and copy them to an accessible location (i.e. in the RAM, see Figure 6.3). This is done by the CPU sending a read request to the device. After sending this single request, the CPU has to wait for a completion packet from the device. If data is available, the device sends the completion package to the CPU containing the data. The CPU then writes the contained data into the RAM. This data transfer method is comfortable, since it is easier to implement and the access to the data is initiated by a user space program. However the waiting times lead to a dead time of the used PCIe link.

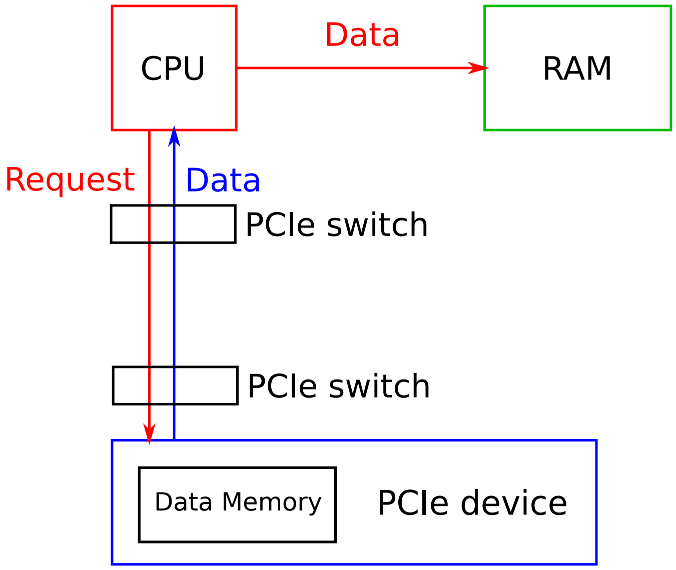


FIGURE 6.3: Read request performed by the CPU to a PCIe device. The answer (called write) of the PCIe contains the demanded data. The process of sending two packages with the included wait time reduces the effective speed of this data transfer. This is called polling.

6.3.2 Direct Memory Access

Direct Memory Access allows quicker data transfers, while losing comfort for the user space program. The data transfer is similar to a **write request** and is done by the PCIe device itself without any interference by the CPU. For this to work, the device needs to know the memory address it can write to.

Before a transfer can start, the CPU has to allocate a certain memory area in the target memory. The target memory is allocated as a fixed size memory block. After giving the target memory information to the PCIe device it can perform the writing process without CPU usage. This way the full bandwidth of PCIe (Table 6.1) can be used[1]. A typical data transfer is shown in Figure 6.4.

Devices in the Point-to-Point network have an internal currency (credits) that regulates how much data a device is allowed to send. After initialization, each device has a certain amount of credits, while sending a package reduces the amount of credits. If these packages travel successfully through the PCIe switches, credits are transferred back to the transmitter allowing it to send more data. This flow control prevents overflowing caches in the target memory. This can also show the underlying peripheral: When a device sends a write packet with 256 bytes of data, while the PCIe switches only allow 128 bytes of data the second half of data will be chopped. Since the header still indicates that the packet contains 256 bytes of data, the credit for this transfer will not be given

[1] Each write package contains $4 \cdot 32$ bit header and up to 128, 256 or 512 bytes data (depending on the peripheral). This way one still has a overhead that reduces the effective speed.

back to the sender. Sending off many incorrect packets results in a device having 0 credits – it is no longer accessible through PCIe[2].

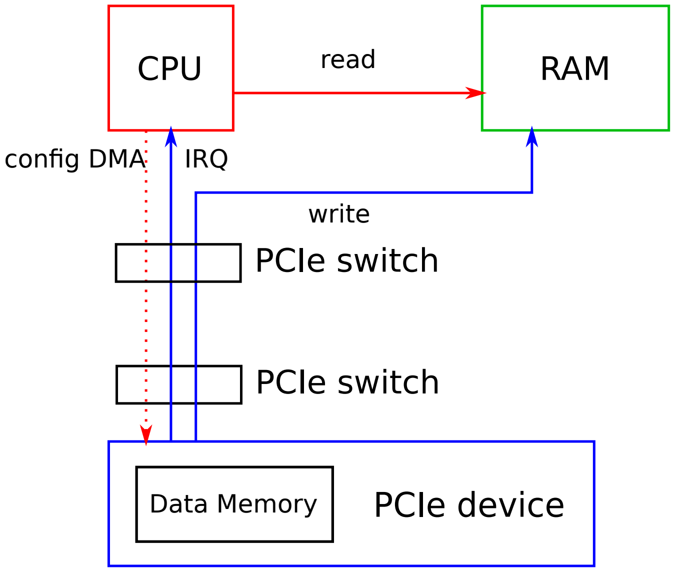


FIGURE 6.4: Schematic of a DMA data transfer between a PCIe device and a RAM. The CPU sets up the DMA by giving the target memory address. The data transfer can be started independent of the CPU. The CPU is notified by sending an interrupt request (IRQ).

[2] A new intialization of the device by replugging or repowering gives access again.

Chapter 7

Measurements

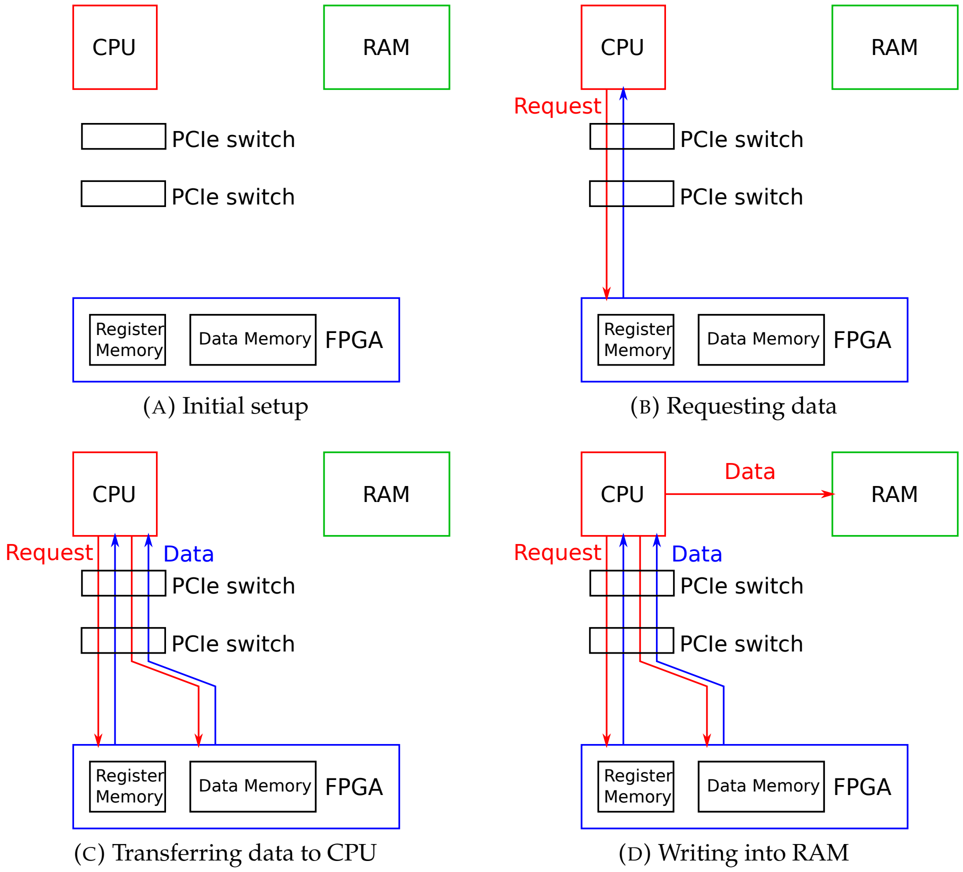
7.1 Setup

The PCIe device used for the polling transfer is a Stratix IV development kit [25] that is plugged to a PCIe 3.0 slot. Since the device only supports 8 lanes of PCIe 2.0, the maximum transfer speed is 4 GByte/s via PCIe (for PCIe transfer speeds see Table 6.1). Testing the PCIe speed is done with the CPU-RAM as a target instead of GPU-RAM. This allows easier error handling and easier implementation, while using likewise procedures. The target RAM is a Corsair DDR3-1600 SDRAM with a maximum data rate of 12.8 GByte/s per module [37], which is much faster than the PCIe transfer rate. The resulting bottle neck is the PCIe link. The transaction speeds for polling and DMA are measured in this chapter.

7.2 Polling

When polling, the CPU acts as the data transmitter requesting data from the FPGA and writing them into the memory. The data transfer from the FPGA to the CPU-RAM is displayed in Figure 7.1. The CPU does not know the position of the last written data inside the FPGA memory. This is written in a small register memory sitting on the FPGA. The CPU sends out a polling read request for this specific register. After receiving the address of the actual data, it can perform another polling request for the data memory and write them into the RAM.
The speed of this transaction is determined by using an indirect method:
The RAM of the FPGA is organized as a ring buffer (Figure 7.2). This memory is filled by the FPGA with an adjustable speed. In parallel, data is read by the CPU. If this is successful over a long period (rewriting the 4 kByte memory of the FPGA $> 10^5$ times) the generation speed is increased until the reading fails. This coarse determination of

the transaction speed is a sufficient estimate.
This method works up to transfer speeds of 800 MByte/s. This is insufficient for the Mu3e experiment, since $\sim$ 1 TBit/s front-end data would require $>$ 160 FPGAs to transfer the data into the GPUs. This would be theoretically possible but the space requirements and financial restrictions of the Mu3e experiment do not allow for so many machines.



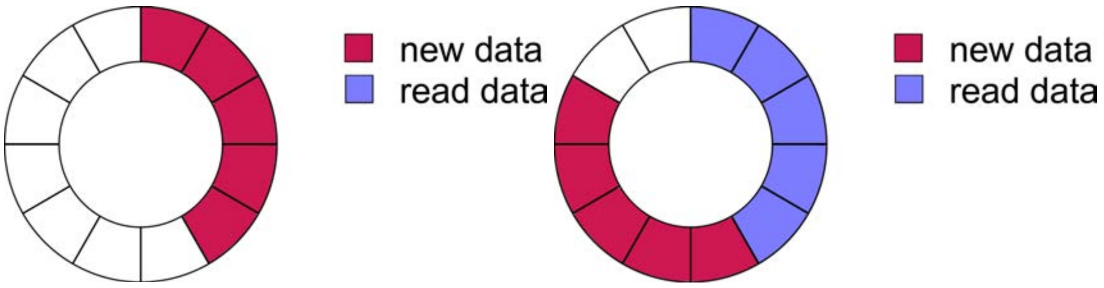
(A) Initial setup

(B) Requesting data

(C) Transferring data to CPU

(D) Writing into RAM

FIGURE 7.1: Schematics of a polling data transfer between a PCIe device (FPGA: blue) and a RAM (green). The CPU (red) engages the data transfer. The CPU asks the FPGA for new data, and gets the memory address if data are available (b). After requesting the data at this address (c), data are sent to the RAM (d).



(A) new generated Data

(B) old data read, new data available

FIGURE 7.2: The write speed of polling is dominated by the read speed of FPGA memory by the CPU. This is determined by generating data on the memory of the FPGA (new data), while reading them by the CPU (read data). The used memory domain is used as a ring buffer.

7.3 Direct Memory Access

DMA works like a write request on PCIe level, where a device (Stratix IV FPGA) sends a packet to a target (RAM) without any CPU interaction or waiting for an answer packet. This allows a higher saturation of the used PCIe 2.0 slot.

The advantage of DMA also has trade-offs: Without any CPU interaction, a user space program does not know if a data transfer has happened, where and what was written. This can be done by using interrupt requests (IRQ). These signals are special PCIe messages sent to the CPU. Once the CPU receives an interrupt, an interrupt handler in the driver is called, which performs a specific task for each IRQ. In this application it notifies a user space program that reads the data and compares them with the original data.

The CPU only knows the position and size of the DMA memory, but not at which position data was written. This can be solved by using a small part of the DMA memory as a register. This register contains the last written memory address and the number of IRQs sent to allow comparing the received and FPGA-sent IRQ. Once the CPU receives an IRQ, it checks the last written memory address. Since the CPU knows the last address it read and the latest that has been written, the new data will be in between, since this memory is contiguous.

The transaction speed of DMA can be measured in different ways:

Counting Offset: The target of the DMA is a 4 MB memory in the RAM. When performing DMA operations this is used as a ring buffer. Once a data transfer was finished, the last written position is written in the DMA control register and an IRQ is sent. The CPU now reads the last written position (offset in the memory) and saves it with the corresponding time. The writing speed v_t can be calculated by using the change of offset per second:

$$v_t = \frac{\Delta\text{offset}}{\Delta\text{t}}. \tag{7.1}$$

Another possibility is using the periodicity of the ring buffer. Since the memory size s_{mem} is known the writing speed can be calculated by:

$$v_t = \frac{\Delta s_{\text{mem}}}{\text{T}}. \tag{7.2}$$

where T is time required to fill the memory i.e. the period. An example is shown in Figure 7.3. Both methods result in comparable writing speeds. This is a blind determination, since we do not check if the bits sent are the bits written.

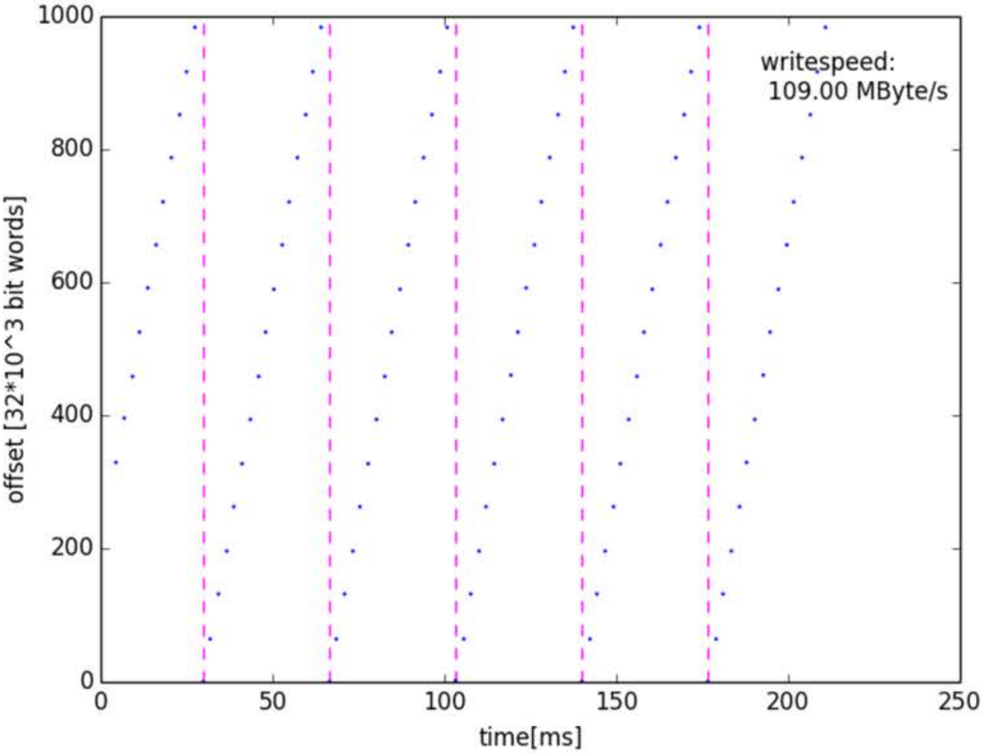


FIGURE 7.3: Memory offset read out after each IRQ. The write speed can be calculated by the slope of the offset or using the periodicity of the memory.

Counting Words: Comparing the sent data with the original data is non-trivial, since the comparison requires to read off data from the FPGA. When performing DMA, this can not be done, since requesting data from the FPGA is a read request that works for data rates of ∼ 800 MByte/s. One possibility is starting the DMA transfer and stopping it again once the sent data correspond to the internal FPGA memory. With a FPGA memory of 4 kB and a write speed of ∼ 4 GByte/s, this happens after ∼ 1 µs. For a reasonable measurement, longer transaction rates are required and another method has to be chosen.

This is done by generating data in a deterministic way: a 64-bit number is written in two consecutive 32-bit memory blocks. Every time an IRQ is sent, the CPU checks the new data in the RAM and compares if the difference between two sequenced 64-bit words is 1. As soon as an error happens, an internal error counter is increased. This method works since reading from the CPU-RAM is much faster than DMA and the comparisons between the two 64-bit numbers occur in the CPU-cache.

As long as this transfer occurs without any bit errors the writing speed can be increased. The writing speed has been increased up to 3.5 GByte/s, which is in the same order to the expected upper limit of PCIe 2.0 with 4 GByte/s. The overhead of the write packages does not allow a full exploitation of the PCIe transfer speed.

The rate of 3.5 GByte/s was tested for 4 days straight without any transmission error. This corresponds to a total transmitted data sample of 1200 TByte.

PCIe uses hardware controllers that help reducing bit errors. An excluded bit error rate[1] can be calculated. Assuming that all errors are Poisson distributed and no error occurred, the excluded bit error rate for N transmitted bits at 95% confidence level yields [27]:

$$BER \lesssim \frac{2.996}{N}(95\%\ C.L.) \approx 3 \cdot 10^{-16}(95\%\ C.L.) \tag{7.3}$$

In phase II of the experiment, a front-end data stream of $\sim$ 1 TBit/s is expected. Using the 48 proposed GPU PCs a total bandwidth of $\sim$ 2.6 GByte/s has to be handled per PCIe bus. The used PCIe 2.0 fulfills this requirement. However an upgrade to PCIe 3.0 with a Stratix V can still be done if a higher bandwidth is required.

[1] Bit error rate: Number of wrong transmitted bits / total transmitted bits

Part IV

Summary and Outlook

HV-MAPS

The readout for the MuPix3 and MuPix4 was successfully implemented. Furthermore the pulse shape resulting from the CR-RC filter was determined for the MuPix3 and MuPix4 chip. The bug in the configuration logic of the MuPix3 did not allow a precise determination of this shape, since the comparator output was always the logical OR of many pixels, resulting in a large smearing of the pulse shape.

The MuPix4 allowed the selection of a single pixel, thus allowing a more precise determination of the pulse shape. This measurement was improved by doing it completely automatically using a FPGA. Since this measurement was so precise, even the influence of internal bias currents on the pulse shape could be determined. The used default values of the chip designer produce a reasonable pulse shape. Furthermore, the efficiency of the MuPix4 was tested at DESY, using a 5 GeV electron beam. The resulting efficiency was close to 100%.

The time over threshold varies for different pixels, which can be adjusted by using the TDACs. This allows a fine threshold adjustment for each pixel, leading to a more homogeneous chip response and allowing muting possible hot pixels.

In the latest version, MuPix6[2], a two-stage amplifier was added. This was implemented by adding an amplification stage close to the comparator, besides the one in each pixel, possibly allowing a better signal to noise ratio, which has to be investigated. Up to now, MuPix chips have been thinned one by one – however they of them can also be thinned by the manufacturer on the wafer. The quality of this thinning and the effect for the MuPix chip needs to be studied in more detail.

Further improvements would be using low voltage differential signaling (LVDS). This results in less crosstalk, that is currently a problem in the flat ribbon cables.

In the final detector, the chip should send the hit information by itself, and not require readout control signals from a FPGA. This is called streaming mode. Therefore, a clock signal has to be sent to the chip. Newer chip versions will need to be bigger, since the aimed final chip size is $1.1 \times 2\,\mathrm{cm}^2$ (inner sensors) and $2 \times 2\,\mathrm{cm}^2$ (outer sensors). Another upscaling of the tracking test system is already under test by the development of a low momentum particle telescope, consisting of four layers of MuPix chips [34].

Direct Memory Access

The transfer speed for polling transfers was measured for a Stratix IV development kit. The resulting speed of ~800 MByte/s is not sufficient for the Mu3e experiment, which requires ~2.6 GByte/s in phase II.

The fast data transfer Direct Memory Access was successfully implemented for a transfer into CPU memory. The transfer speed was tested up to 3.5 GByte/s. A total data amount

[2] The chip version 5 was skipped.

of 1200 TByte was transferred within 4 days without a single bit error. This speed is sufficient for the Mu3e experiment. If a lower saturation of the PCIe bus is favored an upgrade to PCIe 3.0 can be done by using a Stratix V, allowing up to 7.9 GByte/s of transfer rate.

The next step is the implementation and testing of DMA into GPU memory. Once this is successful, track reconstructions with the numerous GPU cores can be done.

Part V

Appendix

Appendix A

Readout State Machine

A description of the state machine that is used to read out the MuPix chip with the Stratix IV development kit is given. The state machine is displayed in Figure A.1.
The starting state is the state ’Waiting’. There are two ways of reading out the MuPix chip: sending the signals by hand or having the FPGA sending the predefined structure with a high rate. If the first one is chosen, the ’readmanual’ bit should be set to true, leading to the state ’readman’. This allows sending the readout signals by settings bits in the FPGA register memory.
The automatic readout works the following way: it sends a ’loadpix’ signal to the MuPix chip, thus storing the pixel map with its hit information. Then the signal ’pulldown’ is sent, which initializes the bus used to transmit the hit information After initializing the bus, a maximum of 1 hit per column is put into the column bus by sending the signal ’loadcol’. If hits are stored in any column, a signal called ’priout’ (priority out) is on. As long as this signal is activated, each column by column gets read out by sending ’readcol’. Once the bus has no more hits available (’priout’ off), the bus is cleared again using ’pulldown’ and new hits are put to the bus by sending ’loadcol’.

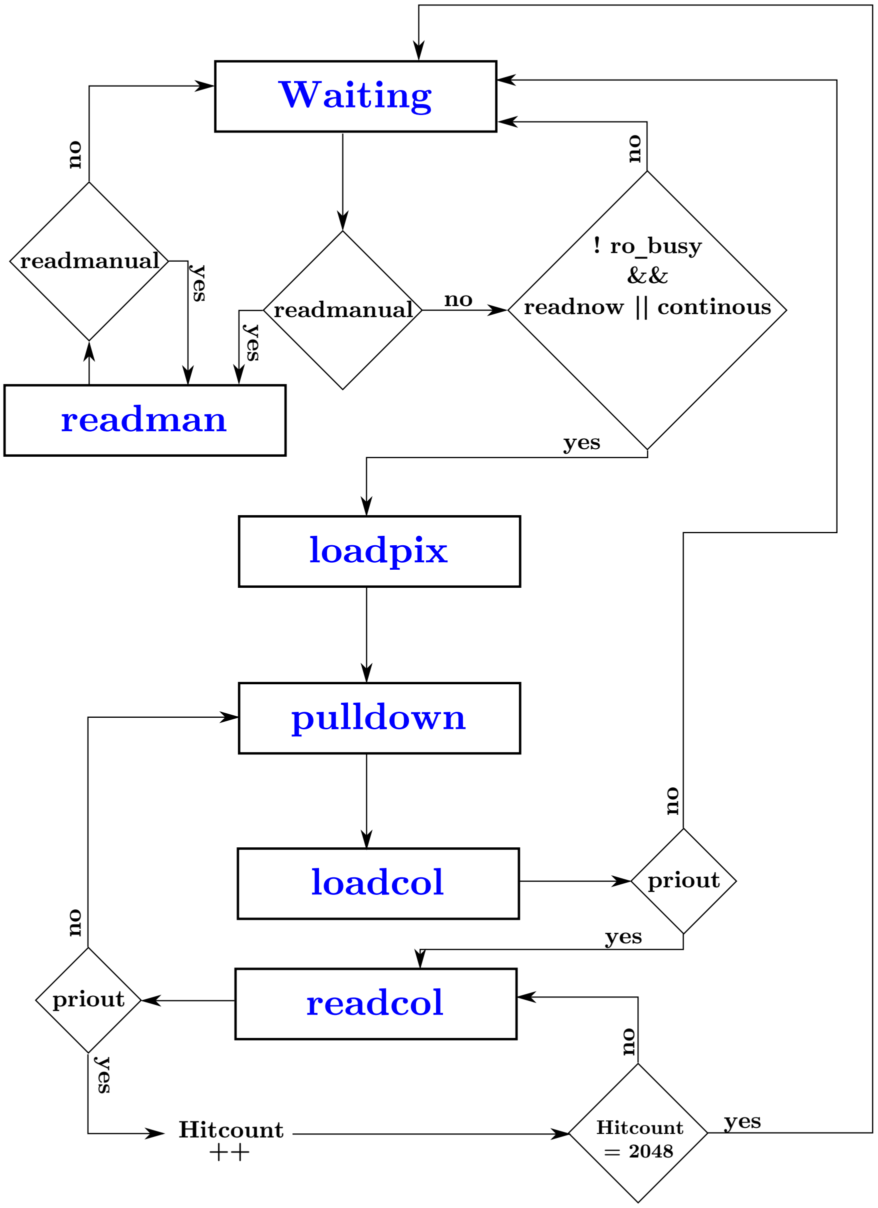


FIGURE A.1: State machine for reading out the MuPix chip with the FPGA.

Appendix B

MuPix Addressing Scheme

The design of current MuPix3/4 uses a 32×40[1] pixel sensor (see Figure B.1). Each pixel can be assigned by a certain row and column address. The design of the chip maps two physical rows to one digital row (one pixel is twice as wide as the associated digital part) – this makes the chip a 64×20 pixel sensor if you address it digitally.

Two transformations can be performed in order to change from digital to physical addresses[2] (see Figure B.2 and Figure B.3):

$$\text{phys\_row} = \text{dig\_row} \cdot 2 + (\text{dig\_col} \bmod 2) \tag{B.1}$$

$$\text{phys\_col} = \text{dig\_col}/2 \tag{B.2}$$

Reversing the transformation yields:

$$\text{dig\_row} = \text{phys\_row}/2 \tag{B.3}$$

$$\text{dig\_col} = \text{phys\_col} \cdot 2 + (\text{phys\_row} \bmod 2) \tag{B.4}$$

[1] The first argument always refers to the column, the second to the row.

[2] Note that all divisions are Integer divisions, i.e. rounded down.

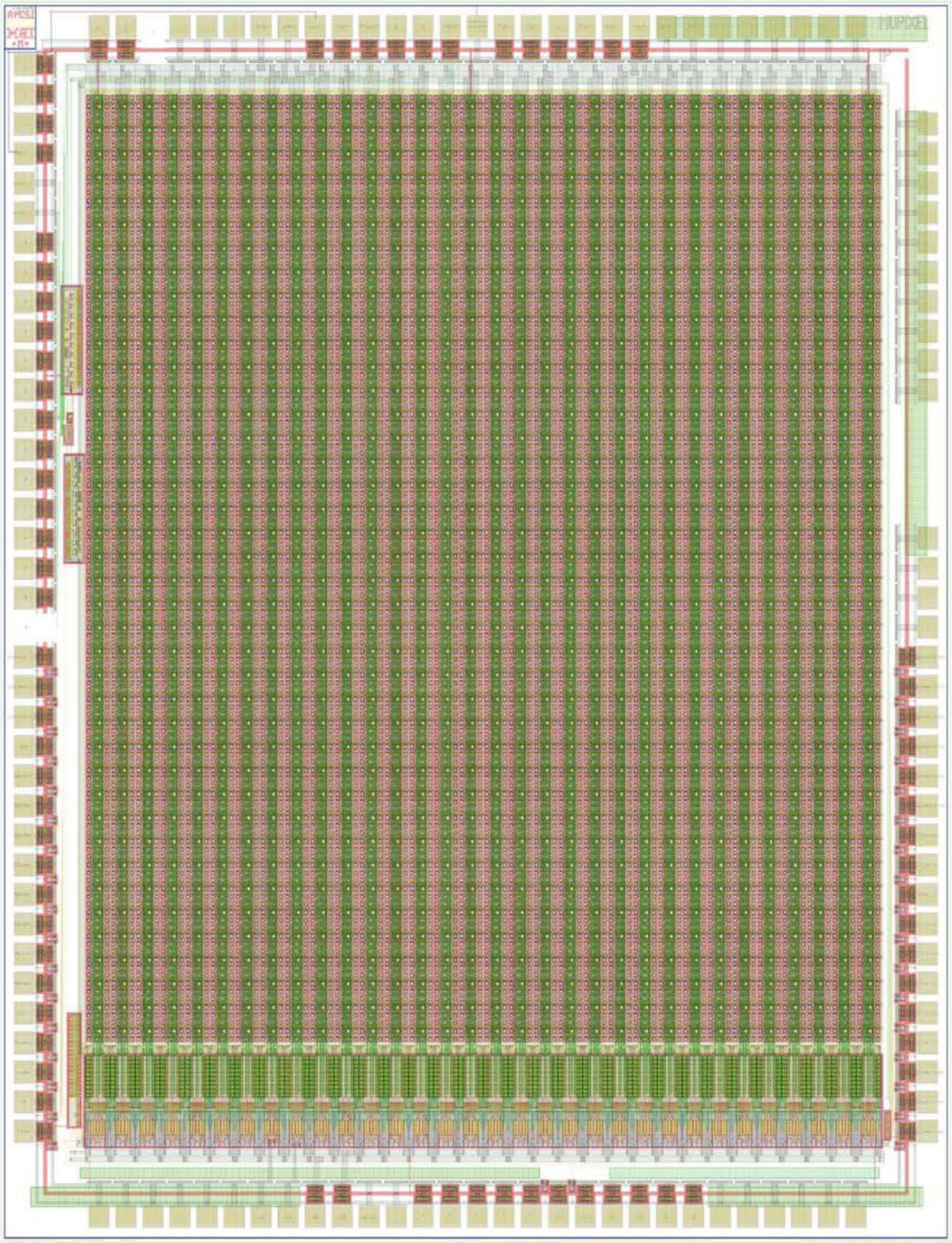


FIGURE B.1: Layout of the MuPix4 chip, showing a 32 × 40 Pixel sensor.

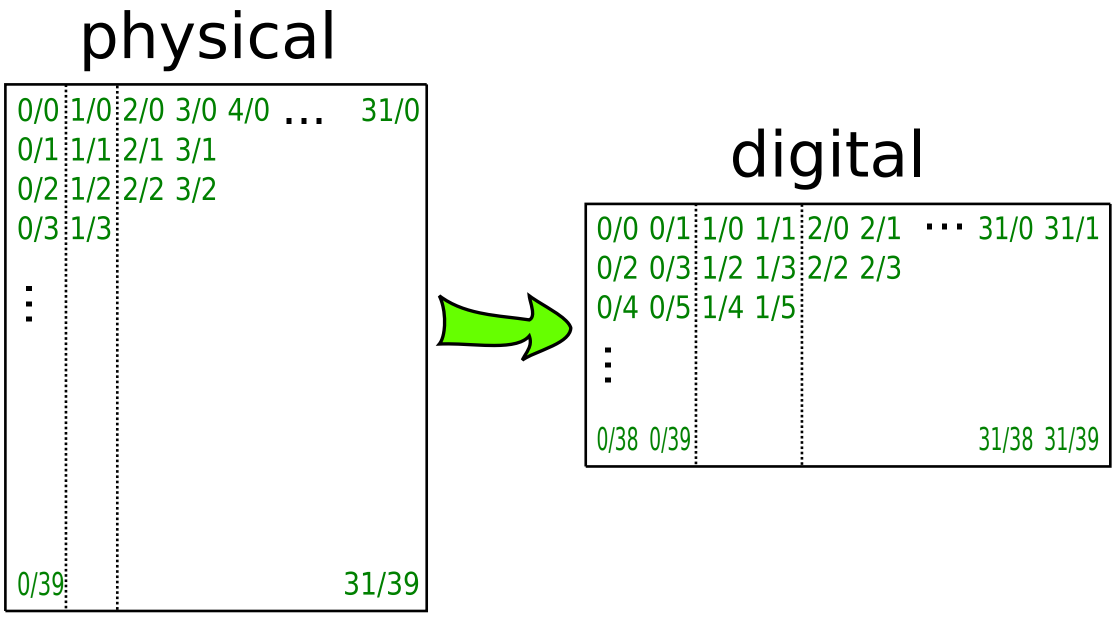


FIGURE B.2: Conversion from physical to digital addresses. Here a/b indicates column a and row b.

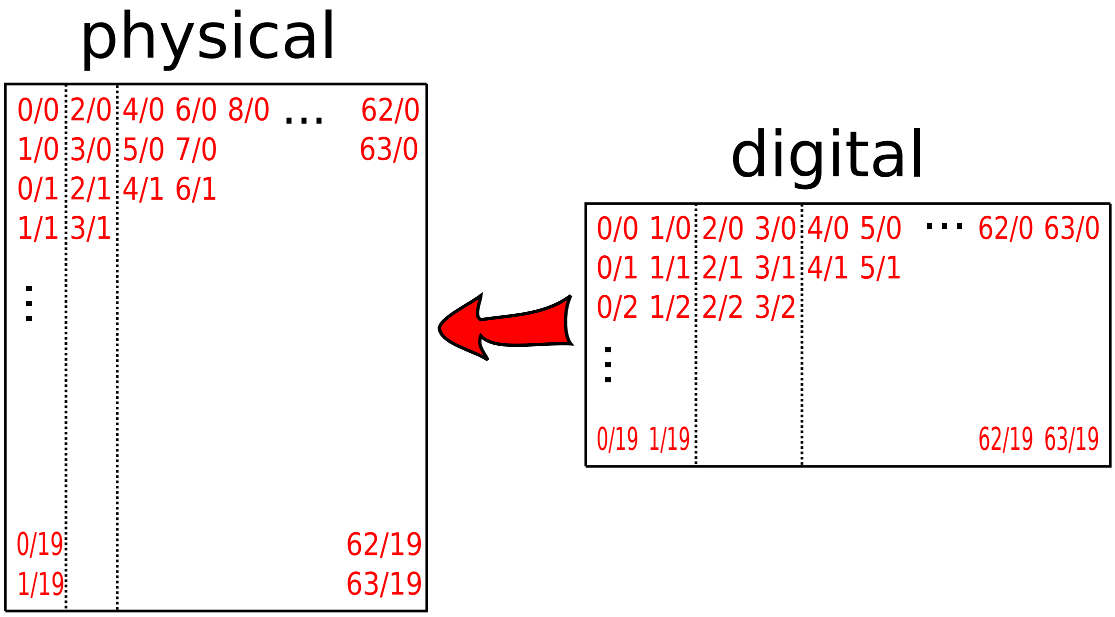


FIGURE B.3: Conversion from digital to physical addresses. Here a/b indicates column a and row b.

List of Figures

List of Tables

Bibliography

[1] Wikimedia Commons, Standard model of elementary particles, 2014, [Online; accessed 17-april-2014].

[2] G. Aad et al., [ATLAS Collaboration], *"Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC"*, 2012, (`arXiv:1207.7214 [hep-ex]`).

[3] 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, (`arXiv:1207.7235 [hep-ex]`).

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

[5] 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, (`arXiv:nucl-ex/0106015`).

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

[7] W. Rodejohann and A. Schöning, The Standard Model of Particle Physics, Lecture notes, 2012.

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

[9] A. Blondel et al., *"Research Proposal for an Experiment to Search for the Decay $\mu \to eee$"*, ArXiv e-prints, January 2013, (`arXiv:1301.6113 [physics.ins-det]`).

[10] J. Beringer et al., [Particle Data Group], *"Review of Particle Physics (RPP)"*, Phys.Rev., **D86** 010001, 2012.

[11] R. M. Djilkibaev and R. V. Konoplich, *"Rare Muon Decay $\mu^+ \to e^+e^-e^+\nu_e\bar{\nu_\mu}$"*, Phys.Rev., **D79** 073004, 2009, (`arXiv:0812.1355 [hep-ph]`).

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

[13] J. Adam et al., [MEG Collaboration], *"New Constraint on the Existence of the $\mu^+ \to e^+\gamma$ Decay"*, Phys. Rev. Lett., **110** 201801, May 2013.

[14] W. H. Bertl et al., [SINDRUM II Collaboration], *"A Search for muon to electron conversion in muonic gold"*, Eur.Phys.J., **C47** 337–346, 2006.

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

[16] I. Perić et al., *"High-voltage pixel detectors in commercial CMOS technologies for ATLAS, CLIC and Mu3e experiments"*, Nucl.Instrum.Meth., **A731** 131–136, 2013.

[17] A. Damyanova, *Development of a Scintillating Fibre Tracker/Time-of-Flight Detector with SiPM Readout for the Mu3e Experiment at PSI*, Master's thesis, Geneva University, 2013.

[18] P. Eckert, PhD thesis, Kirchhoff Institut für Physik, 2014, personal contact.

[19] C. Licciulli, *Präzise Zeitmessung für das Mu3e-Experiment*, Master's thesis, Heidelberg University, 2013.

[20] S. M. Seltzer and M. J. Berger, *"Improved Procedure for Calculating the Collision Stopping Power of Elements and Compounds for Electrons and Positrons"*, The International Journal of Applied Radiation and Isotopes, **35**, 1984.

[21] A.-K. Perrevoort, *Characterisation of High Voltage Monolithic Active Pixel Sensors for the Mu3e Experiment*, Master thesis, Heidelberg University, 2012.

[22] H.-C. Schultz-Coulon and J. Stachel, The Physics of Particle Detectors, Lecture notes, 2011.

[23] H. Augustin, *Charakterisierung von HV-MAPS*, Bachelor thesis, Heidelberg University, 2012.

[24] H. Spieler, *Semiconductor Detector Systems*. Oxford University Press, Oxford, 2008.

[25] Altera Corporation, *Stratix IV Device Handbook, Volume 1*, September 2012.

[26] OPTEK Technology Inc., *Vertical Cavity Surface Emitting Laser*, 2009.

[27] G. Cowan, *Statistical Data Analysis*. Oxford University Press, Oxford, 1998.

[28] I. Peric, Mupixel small pixel detector description, Technical report, Heidelberg University, ZITI-Mannheim, 2012.

[29] S. Shreshta, simulated pulse shapes, private communication.

[30] T. Behnke et al., Test Beams at DESY, Technical report, DESY Hamburg, 2007.

[31] J. Behr, Test Beam Measurements with the EUDET Pixel Telescope, Technical report, University of Hamburg, DESY, 2010.

[32] M. Kiehn, PhD thesis, Physikalisches Institut, Heidelberg, 2014, personal contact.

[33] R. Philipp, *Characterisation of High Voltage Monolithic Active Pixel Sensors for the Mu3e Experiment*, Master thesis, Heidelberg University, 2014.

[34] L. Huth, Master's thesis, Heidelberg University, 2014, personal contact.

[35] Wikimedia Commons, Virtual memory, 2014, [Online; accessed 25-april-2014].

[36] PCI-SIG, *PCI Express® Base Specification Revision 3.0*, November 2010.

[37] Wikimedia Commons, DDR3 SDRAM, 2014, [Online; accessed 11-April-2014].

Acknowledgments

I would like to express my thanks to everyone who supported me and helped carrying out this thesis.

This includes the whole Mu3e group for the nice and funny working atmosphere while performing challenging tasks. This includes all the test beams that have been performed, the DPG spring meeting, the Christmas party and the 'big GPU stress test'. Furthermore I would like to thank Dr. Niklaus Berger and Prof. Dr. Norbert Herrmann for the survey of my thesis.

A less formal and more personal acknowledgment (in no specific order):

Heiko Augustin You always helped me understanding fundamental problems by talking about them. Also thanks for the help with the MuPix setup.

Niklaus Berger Thanks for all the information you gave me in statistics, computing and physics. I hope that the numerous FPGA versions you made for me did not consume too much time.

Simon Corrodi Showed me how much relentless effort can be put into work, while still motivating everyone in the working group. Thanks for all the conversations, swiss gifts and the introduction to climbing.

Lennart Huth Always helping when mechanical tasks have to be done. Thanks for the quick reading of my thesis, all the conversations and the cheering of everyone during festivities.

Moritz Kiehn Showed me that coding can be an art. Furthermore thanks for all the interesting discussions. You have the rare gift to explain complex subjects in a reasonable way.

Raphael Philipp You usually questioned everything and sometimes revealed overlooked details. Thanks for the conversations and the work together, starting from the start of my study.

André Schöning Turning my interest from theoretical physics to experimental particle physics through the PSI practical course and the diversified lecture on accelerator physics.

Dirk Wiedner Thanks for all the patience you showed when explaining the electronics of the MuPix setup. Also thanks for all the conversations and the organization of my summer student stay and BINP, Novosibirsk.

Erklärung:

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

Heidelberg, den 5. Mai ....