Characterisation of the ATLAS ITK Strips Front-End Chip and Development of EUDAQ 2.0 for the EUDET-Style Pixel Telescopes

Richard Peschke

Fakultät für Mathematik, Informatik und Naturwissenschaften
Fachbereich Physik
Universität Hamburg

Dissertation
zur Erlangung des Doktorgrades

December 2016

Gutachter der Dissertation:

Prof. Dr. Erika Garutti	Universität Hamburg
Prof. Dr. Joachim Mnich	DESY/Universität Hamburg

Gutachter der Disputation:

Prof. Dr. Erika Garutti	Universität Hamburg
Prof. Dr. Joachim Mnich	DESY/Universität Hamburg
Dr. Ingrid-Maria Gregor	DESY
Prof. Dr. Caren Hagner	Universität Hamburg
Prof. Dr. Sven-Olaf Moch	Universität Hamburg

Datum der Disputation: 09.12.2016

Kurzzusammenfassung

Als Teil des ATLAS Phase-II Upgrades wird ein neuer Spurdetektor gebaut, der vollständig auf Siliziumtechnologie basiert. Der neue Spurdetektor wird aus einem Siliziumpixelsensor und einem Siliziummikrostreifensensor bestehen. Zur Auslese des Mikrostreifensensors wurde auf Basis der 130 nm CMOS Technologie ein neuer Auslesechip entwickelt: der sogenannte ATLAS Binary Converter 130 (ABC130). Der Chip besteht aus einer analogen Eingangsstufe mit 256 Kanälen.Jeder Kanal besitzt einen Vorverstärker und einen Diskriminator zur Umwandlung der analogen Sensorauslese in ein binäres Signal. Der Vorverstärker wurde geplant mit einer Verstärkung von $90-95\ \frac{\mathrm{mV}}{\mathrm{fC}}$. Erste Labormessungen mit der eingebauten Kontrollschaltung haben eine Verstärkung von $<75\ \frac{\mathrm{mV}}{\mathrm{fC}}$ ergeben. Im Laufe dieser Dissertation wurde eine Testbeamkampagne unternommen, um die Verstärkung unverfälscht und unter realistischen Bedingungen messen zu können. Die gemessene Verstärkung variiert von $\approx 90\ \frac{\mathrm{mV}}{\mathrm{fC}}$ bis $\approx 100\ \frac{\mathrm{mV}}{\mathrm{fC}}$. Damit liegen die von der Testbeamkampagne erhalten Werte innerhalb der Spezifikationen.

Um die Testbeamkampagne mit der größtmöglichen Effizienz durchzuführen, war es notwendig, das Data-Akquisition Framework, welches für die EUDET Type Testbeam-Teleskope verwendet wird, komplett zu überarbeiten. Die neue Version wird EUDAQ 2.0 genannt. Es wurde zur Handhabe von Geräten mit unterschiedlichen Integrationszeiten entwickelt, wie zum Beispiel LHC typische Geräte mit einer Integrationszeit von nur 25 ns, und Geräte mit einer langen Integrationszeit wie zum Beispiel der MIMOSA26 mit einer Integrationszeit von $114.5\,\mu$s. Um das zu erreichen, wurde ein neuer Synchronisationsalgorithmus entwickelt. Dieser gibt den Benutzern volle Freiheit über die Art und Weise, wie sie ihren Datenstrom mit dem System synchronisieren. Des Weiteren erlaubt EUDAQ 2.0 auch das benutzerdefinierte Encoding und Decoding von Datenpaketen. Damit kann der Benutzer den Datenoverhead verringern und mehr Rechenzeit zum Offlinestate verlagern. Um den Netzwerk-Overhead zu reduzieren, ermöglicht es EUDAQ 2.0 auch, dass Benutzer Daten lokal speichern können. Das Zusammenführen der Daten geschieht dann Offline.

Abstract

As part of the ATLAS phase-II upgrade a new, all-silicon tracker will be built. The new tracker will consist of silicon pixel sensors and silicon microstrip sensors. For the readout of the microstrip sensor a new readout chip was designed; the so called ATLAS Binary Converter 130 (ABC130) which is based on a 130 nm CMOS technology. The chip consists of an analog Front End built up of 256 channels, each with a preamplifier and a discriminator for converting the analog sensor readout into a binary response. The preamplifier of the ABC130 was designed to have a gain of $90-95\ \frac{\mathrm{mV}}{\mathrm{fC}}$. First laboratory measurements with the built-in control circuits have shown a gain of $< 75\ \frac{\mathrm{mV}}{\mathrm{fC}}$. In the course of this thesis a test beam campaign was undertaken to measure the gain in an unbiased system under realistic conditions. The obtained gain varied from $\approx 90\ \frac{\mathrm{mV}}{\mathrm{fC}}$ to $\approx 100\ \frac{\mathrm{mV}}{\mathrm{fC}}$. With this, the values obtained by the test beam campaign are within the specifications.

In order to perform the test beam campaign with optimal efficiency, a complete overhaul of the data acquisition framework used for the EUDET type test beam telescopes was necessary. The new version is called EUDAQ 2.0. It is designed to accommodate devices with different integration times such as LHC-type devices with an integration time of only 25 ns, and devices with long integration times such as the MIMOSA26 with an integration time of $114.5\,\mu$s. To accomplish this a new synchronization algorithm has been developed. It gives the user full flexibility on the means of synchronizing their own data stream with the system. Beyond this, EUDAQ 2.0 also allows user specific encoding and decoding of data packets. This enables the user to minimize the data overhead and to shift more computation time to the offline stage. To reduce the network overhead EUDAQ 2.0 allows the user to store data locally. The merging is then postponed to the offline stage.

Contents

List of Figures

Chapter 1

Introduction

The goal of physics is the understanding of the universe at its most fundamental level. To accomplish this goal physics has been subdivided into a variety of different specializations. The actual phenomena of interest can reach from the largest galaxy super cluster, down to the electron structure in solid states, to the tiniest fundamental particle which only exists for an inconceivably short time. In order to make new observations about nature, gigantic pieces of apparatus are built. One of which is the "A Toroidal LHC ApparatuS" (ATLAS) [**?**] detector at the Large Hadron Collider (LHC) [**?**] particle accelerator at the "European Organization for Nuclear Research" (CERN[1]) in Geneva (Switzerland). It was set up to answer some of the most fundamental questions about particles and their interactions. Among the list of physics phenomena the ATLAS detector was set up: to find the origin of mass (the Higgs boson [**?**]), to discover a hidden symmetry between bosons and fermions (Supper-Symmetry [**?**]) and to search for a possible gauge boson for gravity. In addition to the discovery of new particles the ATLAS detector was also set up to provide high precision measurements of Standard Model (SM) [**?**] physics processes.

After the discovery of the Higgs boson [**?**] it is desired to continue this path of success. After the end of the life span of the current Inner Detector (ID) [**?**] a significant upgrade to both the ATLAS detector, as well as to the LHC, is planned. The upgrade to the accelerator will increase the luminosity from 1×10^{34} cm^{-2} s^{-1} to 7×10^{34} cm^{-2} s^{-1}. The replacement for the Inner Detector (ID), the Inner TracKer (ITK) [**?**], will take full advantage of new technologies like radiation hard 130nm CMOS Application-Specific Integrated Circuit (ASIC) for the readout of the sensors. With the new technology the ATLAS detector will be able to process the dramatically increased luminosity.

The upgrade of the ATLAS detector is a massive operation where every part has to be investigated in high detail. One of the central components of the new ITK-strip tracker is

[1] CERN - Conseil Européen pour la Recherche Nucléaire

the sensor readout chip which converts the analog charge information from the silicon strip sensor to digital information. This chip is the ATLAS Binary Converter 130 (ABC130) [**?**]. It is a specially designed ASIC chip based on a 130 nm CMOS technology for the ATLAS detector. An important feature of every readout is the Signal to Noise Ratio (SNR) which depends, among others, on the quality of the preamplifier used in the analog input of the readout chip.

In order to test the readout chip under most realistic conditions a test beam campaign was set up. This thesis describes the test beam campaign on two levels. On one level it describes the actual test beam performed to characterize the new ABC130 readout chip. On the other level it describes the changes to the telescope data acquisition system (EUDAQ) [**?**] which were necessary to perform the test beam campaign. The result of the changes to the EUDAQ system is a complete overhaul of the DAQ concept, trigger scheme and software leading to a new release of the framework called EUDAQ 2.0. The results from the test beam gave vital input for the design of the next iteration of the readout chip, while EUDAQ 2.0 allows a whole new generation of devices to use the framework.

Chapter 2

ATLAS Experiment

The ATLAS collaboration was setup to answer some in the most fundamental questions of particle physics. With the unprecedented energy and luminosity delivered by the LHC [**?**], ATLAS is able to study physics beyond the Standard Model (SM) [**?**] and make new precise measurements of known phenomena. The ATLAS detector started with the goal (among others) to find the Higgs boson, search for Super Symmetric (SUSY) [**?**] particles and discover extra dimensions[**?**]. It was also set up to improve the knowledge of rare Standard Model (SM) phenomena like boson-boson scattering, the top quark and asymmetries between matter and anti-matter. To perform this wide variety of tasks the ATLAS detector was designed as a multi-purpose, 4π detector. The first milestone was reached by discovering the Higgs boson in 2012 [**?**]. The layout of the ATLAS detector is shown in figure 2.1. Its constituents are described in the following sections.

2.1 Overview

Figure 2.1 shows the layered structure of the ATLAS detector. It consists of three basic types of detector. The Inner Detector (ID) (see section 2.2) measures the transverse momentum of charged particles as well as the sign of the charge. The next detector outside the ID is the calorimeter (see section 2.3). It measures the total kinetic energy of all particles (except muons and neutrinos) by stopping them. The muon system is placed outside the calorimeter (see section 2.4). It is another tracking detector specially designed to precisely measure the total momentum of muons.

At the center of the detector the beam pipe is situated. In the vacuum of the beam pipe the hadrons travel through the detector. At the interaction point, the particles from both sides

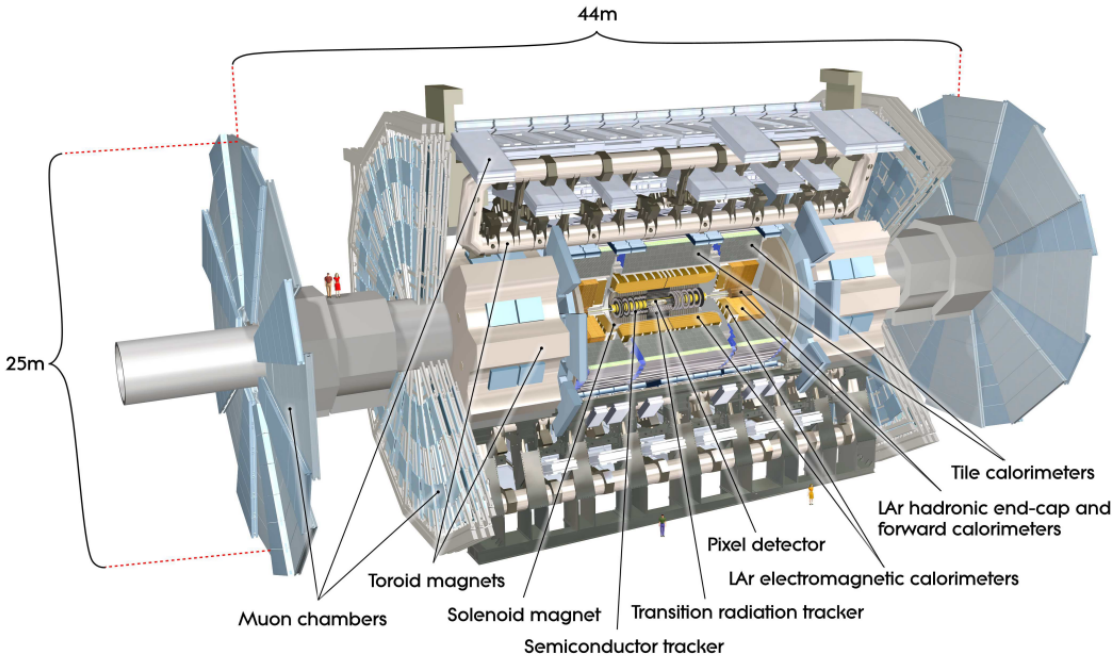


*Figure 2.1 The ATLAS detector with the subcomponents [**?**]*

collide with a center of mass energy of $\sqrt{s} = 14$ TeV[1] and new particles are created in the interaction point. Depending on the kind of interaction, the newly created particles have certain preferences in which directions they are emitted. Since ATLAS is a multi-purpose detector, it is important to cover as much spacial angle as possible.

2.2 Inner Detector

The Inner Detector (ID) [**?**] as pictured in figure 2.2 is directly outside the beam pipe. It has a diameter of 2.1 m and a length of 6.2 m. It consists of three tracking detectors and is placed inside a superconducting solenoid magnet which provides a magnetic field of 2 T. With the three tracking detectors the ID records the hits of charged particles along their trajectories. This information is used to reconstruct the particle tracks. Due to the magnetic field, the charged particles move on bent tracks. From the bending radius the particle transverse momentum can be obtained. From the particle trajectory, the primary and possible secondary vertex can be extrapolated.

All detector components of the ID are subdivided into a central region, with a barrel geometry, and a forward/backward region, with a disk geometry (the end cap region), to

[1] Design energy not yet reached.

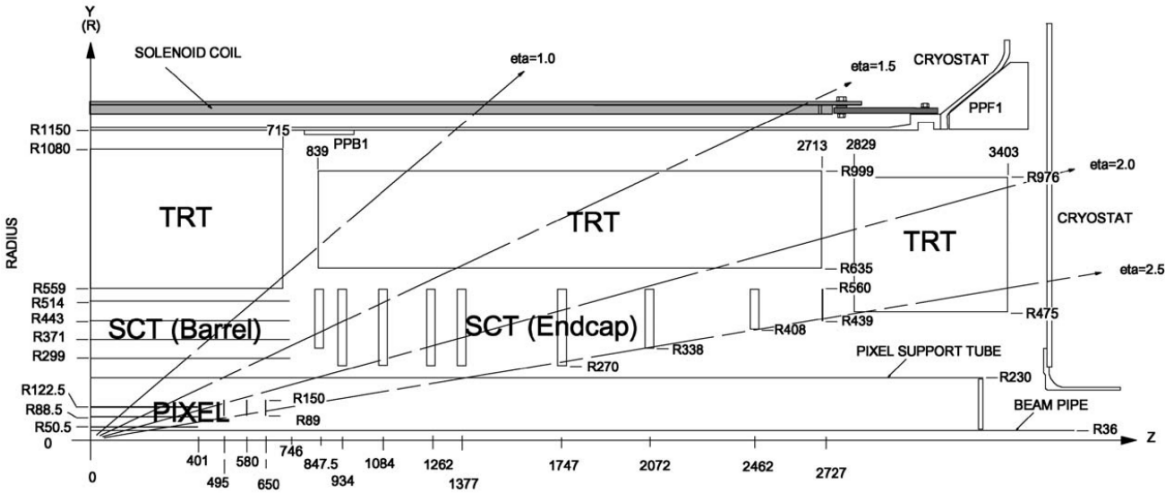


*Figure 2.2 Quarter segment of the ATLAS inner detector. The various layers are located at different radii and Z positions along the beam direction [**?**]*

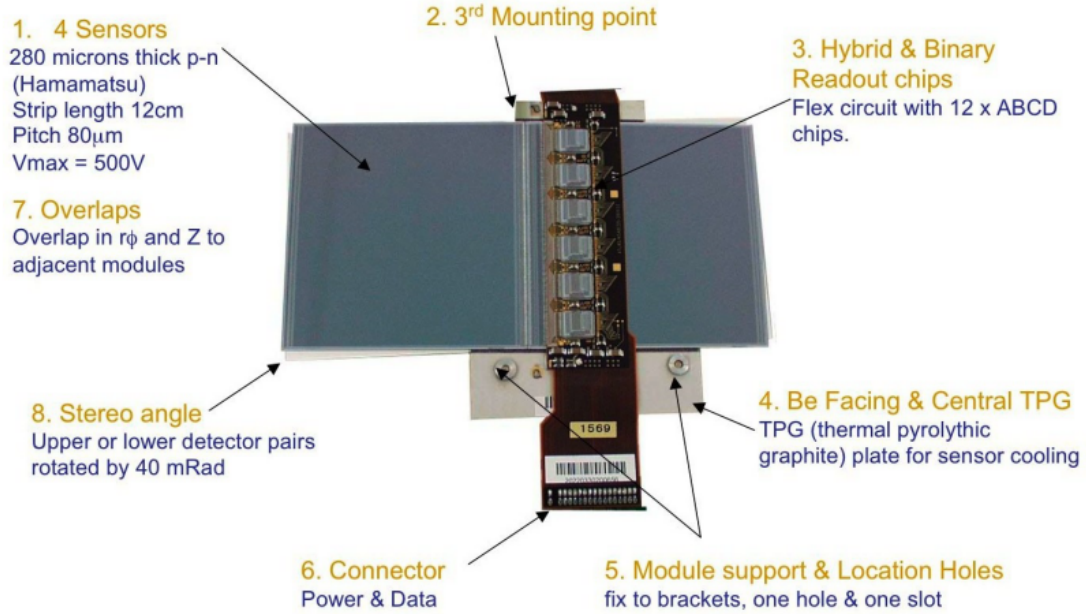


*Figure 2.3 SCT module [**?**]*

also detect particles with a shallow angle. The most inner layer of the ID is the pixel tracker (PIXEL). It provides the most precise hit information with the finest granularity of all three tracker components. With a pixel size of 50×400 μm^2, it provides three precise measuring points, with a resolution of 10 μm for $R\phi$ direction and a resolution of 115 μm in η direction. It covers the full acceptance range up to a pseudo rapidity of $|\eta| < 2.5$. The barrel consists of three layers while the end cap consists of four disks at each side. In 2014, during the long shutdown of the LHC, a new innermost barrel pixel layer was inserted into the ATLAS detector (Insertable B-Layer (IBL) [**?**]).

The pixel tracker is followed by the SemiConductor Tracker (SCT) [**?**]. The SCT is built up of individual silicon modules. Figure 2.3 shows the layout of a barrel module. It consists of two microstrip sensors with a strip length of 126.09 mm and a pitch size of 80 μm. To provide z information each module consists of two strip sensors glued with a small stereo angle of 40 mrad back to back on a cooling/support frame. With this technique the SCT modules provide a spacial resolution in $R\phi$ direction of 17 μm and in z direction of 580 μm. The data connection to the electronics outside the detector is done via an optical link. The modules are individually powered via a cable link. The central region consists of four barrel layers. The end cap region consists of nine disk wheels [**?**] on each side.

The most outer part of the ID is the Transition Radiation Tracker (TRT) [**?**]. Transition radiation is produced when a charged particle passes, at relativistic velocities, thin layers of materials with different dielectric properties. The radiated energy is proportional to γ (the Lorentz factor[2]). The TRT is a gaseous straw tube detector. It consists of about 300000 thin walled proportional mode drift tubes. It contributes to the tracking with an average of 30 two dimensional hits with an intrinsic resolution of 120 μm for $|\eta| < 2$ and $p_t > 0.5$ GeV. Since the radiated energy is proportional to γ the transition radiation can also be used to separate electrons from hadrons.

2.3 Calorimeter

Directly outside of the solenoid, the calorimeter is placed. Its purpose is to measure the total energy of the particles. For this, the particles are stopped inside the calorimeter (except muons and neutrinos). From the energy deposited in the calorimeter the energy of the particle can be calculated. As shown in figure 2.4, the calorimeter is subdivided into an electro magnetic calorimeter and a hadronic calorimeter.

[2] $\gamma = \frac{1}{\sqrt{1-v^2/c^2}}$ with v the particle speed and c the speed of light

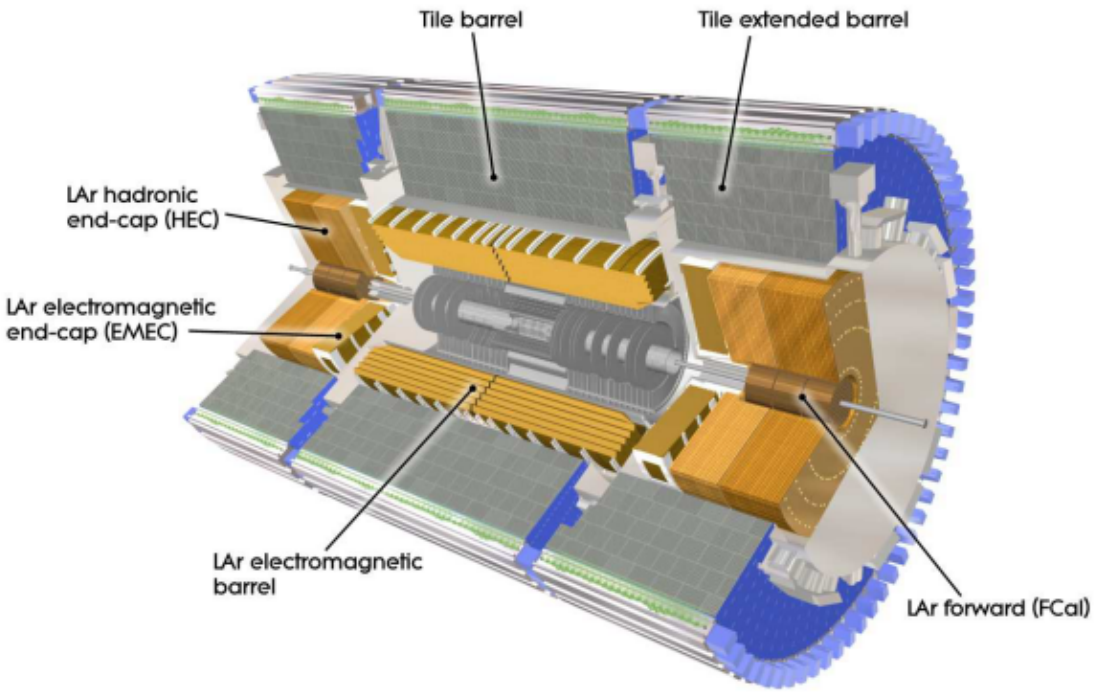


*Figure 2.4 ATLAS Calorimeter system [**?**]*

The ATLAS electromagnetic calorimeter is a liquid argon sampling calorimeter [**?**]. This means it consists of alternating layers of absorbing material (lead) and sampling material (liquid argon). It provides energy information for electrons, positrons and photons and covers a pseudo rapidity of $|\eta| < 2.5$. In addition to providing energy information it also helps distinguish between photons originated from the primary vertex and photons created by the decay of a π^0. Photons originating from π^0 decay hit the detector very close to each other and are therefore hard to distinguish from a single photon. To enhance the separation power between between photons from π^0 decay and single photons, the first layer of the calorimeter has a finer granularity than the rest of the calorimeter.

The hadronic calorimeter is built up of two different detector components. For the barrel and extended barrel region, a tile calorimeter [**?**] is used. The tile calorimeter consists of alternating layers of steel as an absorber and scintillator tiles as active material. For the Hadronic End Cap (HEC) calorimeter and the Forward Calorimeter (FCal) a liquid argon calorimeter is used [**?**]. The hadronic calorimeter without the FCal covers a pseudo rapidity up to $|\eta| < 3.2$.

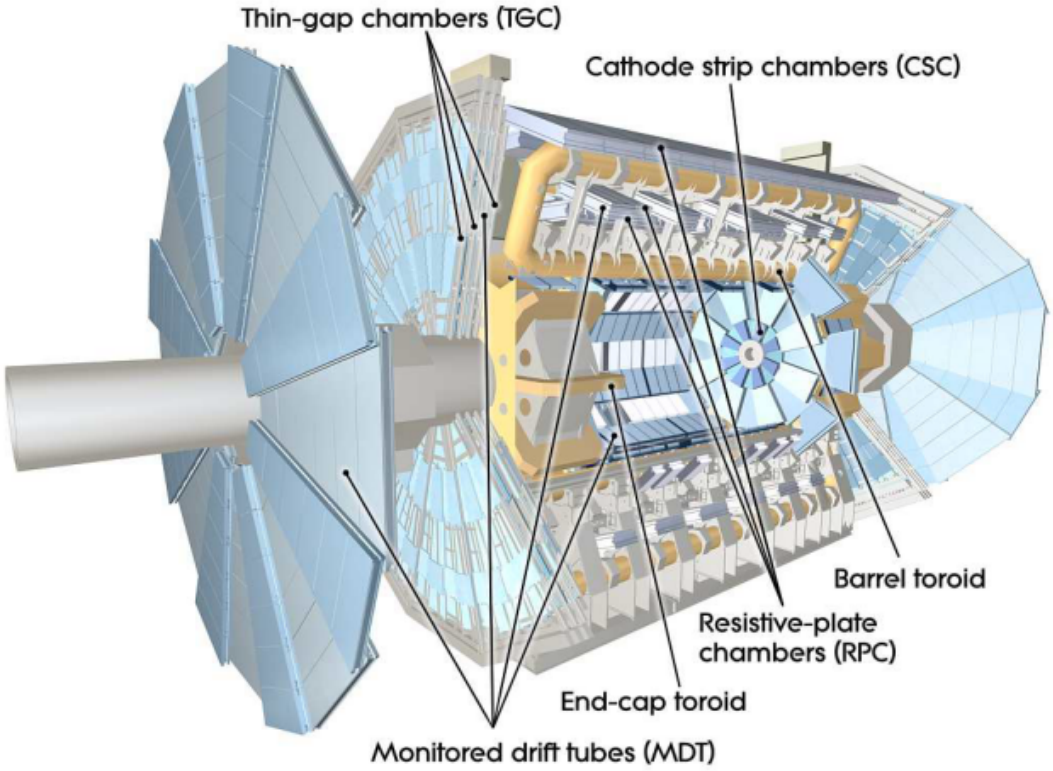


*Figure 2.5 ATLAS muon system [**?**]*

2.4 Muon System

The only charged particles which are able to pass through the calorimeter are muons. Since muons can give a very clean signature of certain physics processes, a large muon system [**?**] was built around the calorimeter to measure them very precisely. As shown in figure 2.5 the muon system consists of an array of tracking detectors built inside a toridal magnetic field. The toridal magnet has the benefit that the particles trajectory and the magnetic field are almost perpendicular [**?**]. This allows the measurement of the total momentum of the muons instead of only the transverse momentum.

Chapter 3

ATLAS Phase II Upgrade

The operation of the LHC and the ATLAS detector have already been very successful with the discovery of the Higgs boson in 2012 [**?**]. This discovery was an enormous success for the particle physics community. It demonstrated a high level of understanding of the fundamental forces of nature and by comparing the measured results to the theoretical predictions, it is truly astonishing how well the theory resembles reality. With the discovery of the Higgs boson, the fundamental interaction which lead to the masses of elementary particles is understood like never before. To continue this story of success a major upgrade of the LHC and the ATLAS detector is planned. The, so called, high luminosity LHC will provide an instantaneous luminosity of 7×10^{34} cm^{-2} s^{-1}, which will accumulated over its ten years of operation to an additional integrated luminosity of about 2500 fb^{-1} [1]. This unprecedented amount of data will open a new door to the understanding of particle physics at its most fundamental level. To achieve this goal the ATLAS detector needs to be upgraded to withstand this new radiation environment. It is foreseen to increase the number of interactions per bunch crossing to > 140. To handle this extreme environment many systems of the ATLAS detector have to be completely renewed. One central part of this upgrade is the new, all-silicon Inner Tracker (ITK) which will be described in this chapter. It starts with a description of the layout of the new ITK and then continues by describing the individual building blocks of the ITK. The last part of this chapter will then introduce the newly designed readout chip, the ATLAS Binary Converter (ABC130), for the strip detector and will shortly discuss means the through which a binary chip can be used to create charge spectra of the analog input signal.

[1] The discovery of the Higgs boson was made with two datasets with an integrated luminosity of 4.8 fb^{-1} at an energy of $\sqrt{s} = 7$ TeV in 2011 and 5.8 fb^{-1} at $\sqrt{s} = 8$ TeV in 2012.

3.1 Motivation for the Inner Detector (ID) Replacement

The current tracker (ID) was designed to withstand the radiation dose equivalent to ten years of operation with a peak luminosity 1×10^{34} cm^{-2} s^{-1} ($\sim 2 \times 10^{14}$ n$_{eq}$ cm^{-2}). During this time it is expected to collect 350 fb^{-1} of data. After this, the current tracker will have reached the end of its life time. With the phase-II upgrade, the requirements on the tracker will increase dramatically. The peak instantaneous luminosity will increase to 7×10^{34} cm^{-2} s^{-1}. With this, the pile up in the detector will increase from ~ 23 proton-proton interactions per bunch crossing to ~ 200. This harsh environment makes is necessary to completely exchange the current tracker with a newly designed all silicon tracker. The new tracker will not only to have to withstand the five times higher radiation doses of $\sim 10^{15}$ n$_{eq}$ cm^{-2} [**?**], but also has to provide enough bandwidth to forward the data to the off detector electronics while holding the overall occupancy below 1 %. To meet these harsh conditions, a new tracker is being developed with a much higher granularity.

3.2 Inner Tracker (ITK) Setup

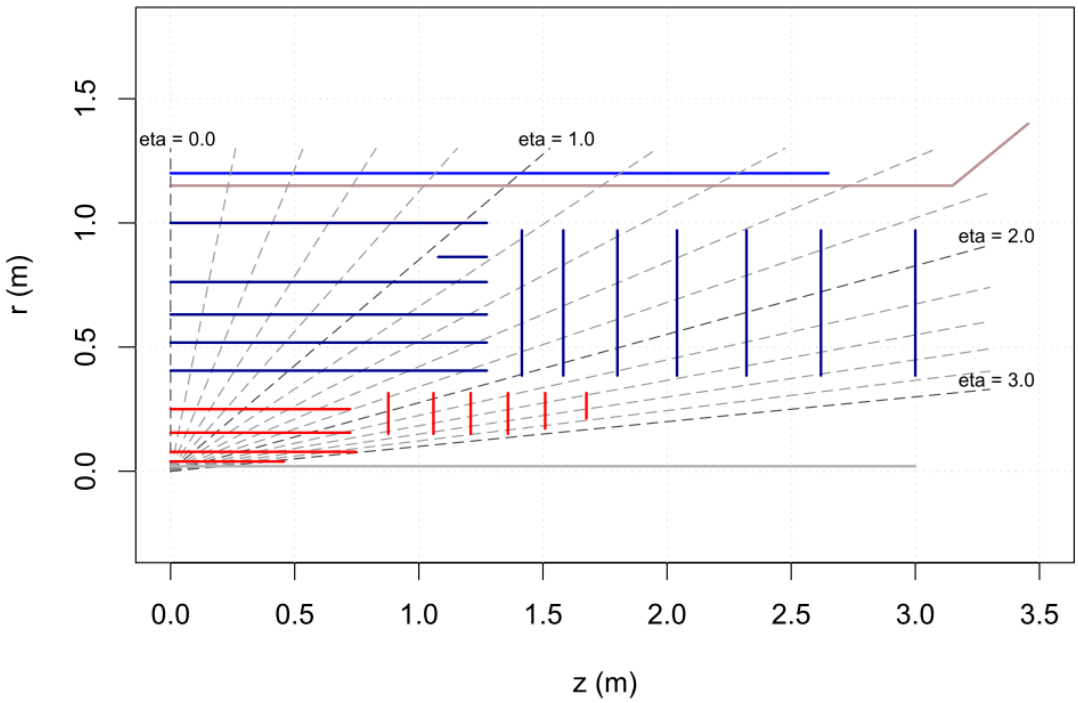
In contrast to the current Inner Detector (ID) the phase-II Inner Tracker (ITK) will consist of an all-silicon tracking detector with a pixel and a strip system figure 3.1a. The Transition Radiation Tracker (TRT) will be completely replaced by the silicon strip tracker. This was necessary since the TRT would not have been able to cope with the higher track rate of the High Luminosity LHC.

For this thesis only the strip tracker part of the ITK is discussed. It consists of a barrel region where the detector planes are arranged in five concentric barrel layers around the beam pipe and an end cap region which covers the forward/backward areas up to $\eta = |2.7|$ [2]. It consists of seven individual disks for each side which are arranged as shown in figure 3.1b. More details can be found in the Letter of Intent (LoI) [**?**].

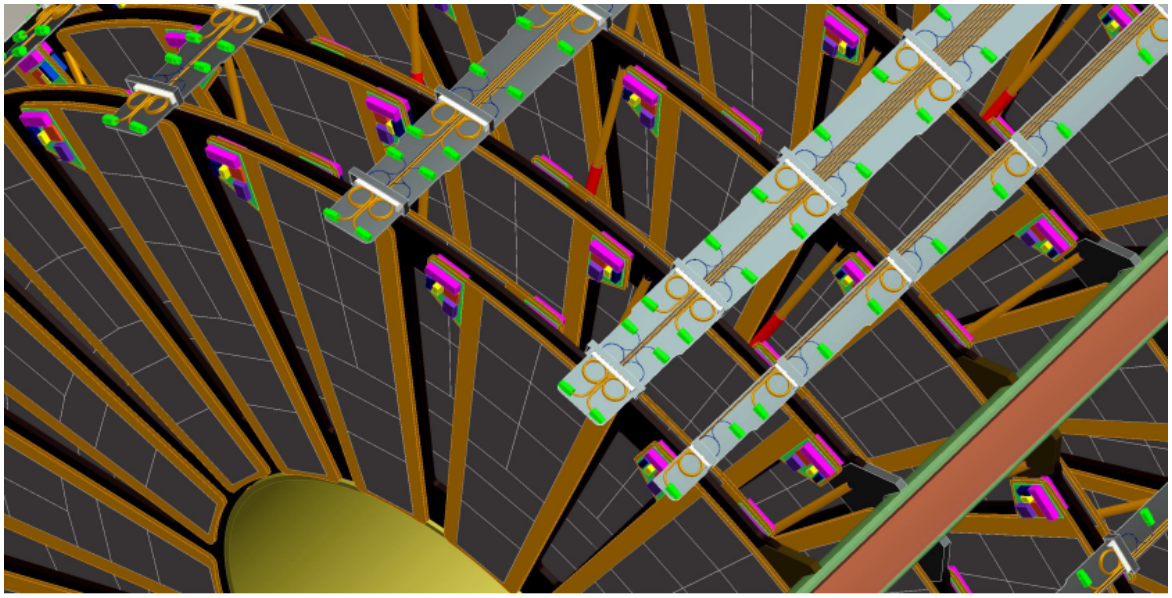
3.3 Stave and Petal

Both the barrel and end caps are built up of smaller substructures called staves, for the barrel, and petals, for the end cap (see figure 3.2). Both substructures share certain similarities like the electrical design, therefore they are developed via a common effort of the barrel and the end cap community. The major difference between the two types of substructures are the

[2]The pseudorapidity (η) is defined by $\eta = ln\left[tan\left(\frac{\theta}{2}\right)\right]$. With θ being the angle between the beam axis and the particle momentum [**?**].



*(a) Quarter segment of the base line layout of the Inner Tracker (ITK) [**?**]. The pixel tracker is colored in red. The strip tracker is colored in blue.*



*(b) End cap design: Castellated layout as described in [**?**]*

Figure 3.1 LOI layout of the phase II tracker.

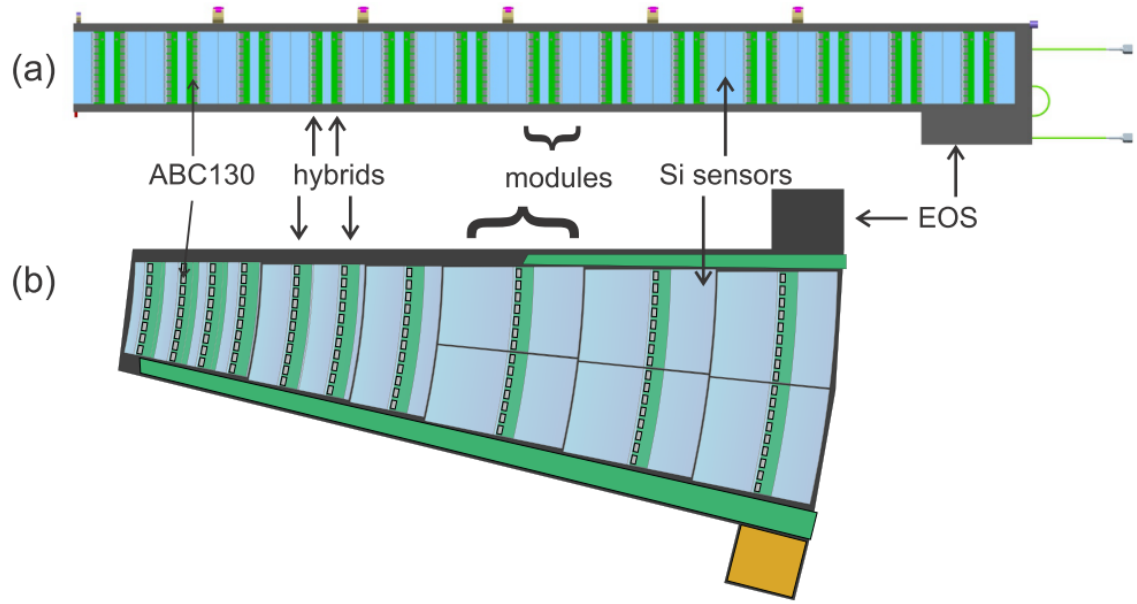


*Figure 3.2 Comparison between the substructure of the barrel, called a stave (a), and the substructure of the end cap, called a petal (b), as explained in [**?**]*

geometries, but both of them are equipped with compatible silicon strip sensors [**? ? ? ? ?**], with a nominal pitch size of 75.5 μm. The term "nominal pitch size" is used since the petal sensors have a radial strip layout, which means the actual pitch size varies along the length of the strips. On top of the sensor, a readout board, a hybrid, is glued on. The hybrids for petal and stave differ in their geometrical layout as well as in the number of readout chips they house. The stave hybrids are all identical whereas the petal hybrids have a different layout depending on the radius. The ABC130 readout chip is mounted ontop of the hybrid. The readout chip is connected via wire bonds to the sensor. The information produced by the ABC130 is transferred to the Hybrid Control Chip (HCC). From the HCC the data are transferred to the End Of Substructure module which connects the individual stave via optical and electrical link to the off-detector electronics (see figure 3.3).

3.4 Trigger

Due to the vast amount of data the ATLAS detector produces, it is necessary to perform an event selection during the data taking. Among other parameters, the trigger checks for the energy of the particles to decide whether or not to store an event. ATLAS uses a multi-stage trigger system as described in [**?**]. The first trigger is the L0 trigger which uses only information from the muon system and the calorimeter to decide whether the event is

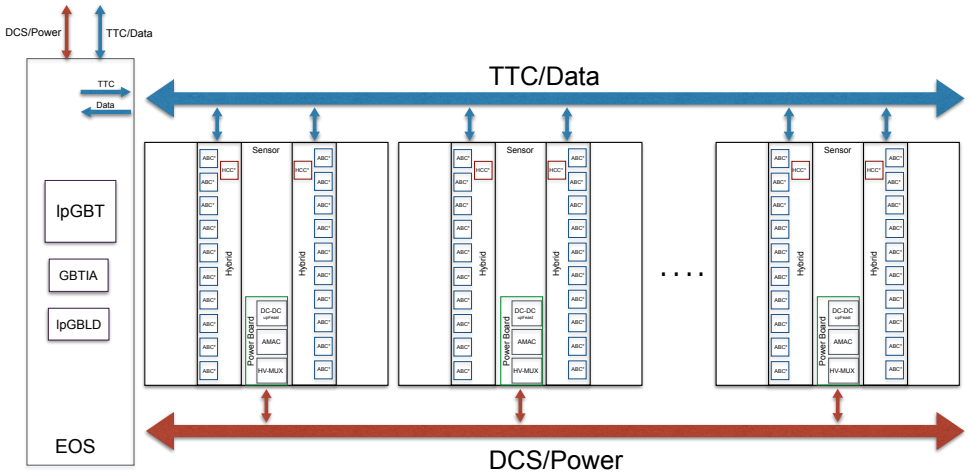


Figure 3.3 Stave: The stave is composed of modules which consist of a 10×10 cm^2 *silicon strip sensor and two readout hybrids glued on top of the sensor. The individual modules are chained together with a data bus. The End Of Substructure block collects the data from the stave and sends them to the off-detector Data Acquisition System (DAQ) system [***?** *].*

interesting or not. It will reduce the trigger rate to 500 kHz while having a delay of less than $6\mu s$. The next trigger is the L1 trigger which can perform more complicated operations. It will reduce the trigger rate to 200 kHz while the delay stays below $20\mu s$. The next step of event selection is done with high level software trigger. This sections describes the trigger as it was introduced in the Letter of Intent (LoI) [**?**] which was the baseline for the design of the ABC130. Since then the design has been modified resulting into higher anticipated trigger rates and a new layout for the readout chip (ABCSTAR).

3.5 ATLAS Binary Chip ABC130

3.5.1 Overview

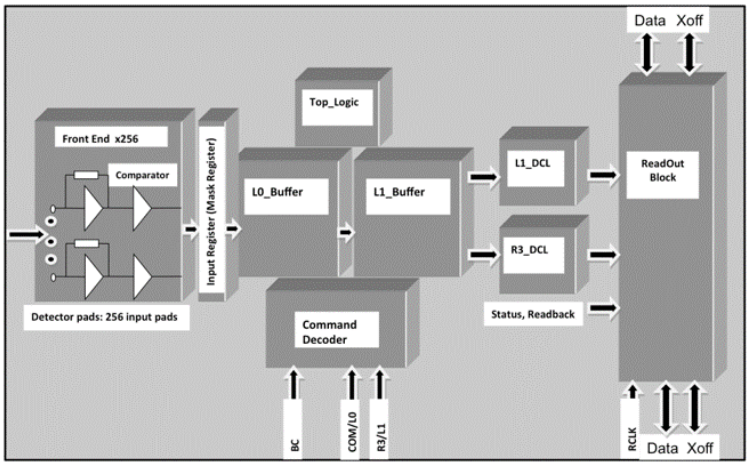


*Figure 3.4 Overview of ABC130 chip [**?**].*

The readout of the strip sensor is done with an Application-Specific Integrated Circuit (ASIC) based on a 130 nm CMOS [**?**] technology. This ASIC is designed to convert the analog charge signal into binary hit/no-hit information. As shown in figure 3.4 the ABC130 [**? ?**] consists of an analog front end with 256 channels (see section 3.5.2). From each channel the analog input signal is transformed into a binary signal. The next step is an input/mask register. It is used to either mask out certain channels (for instance noisy strips) or to input a certain hit pattern to the readout chain. After this the signals are stored in a two stage signal buffer. The buffer is necessary to preserve the signal data until the arrival of the ATLAS triggers. On arrival at the L0 trigger, the data corresponding to this trigger are transferred from the first buffer stage to the second. The first buffer stage (L0-buffer) can store the data for 6.7 μs. The data corresponding to the L1 trigger are transferred from the

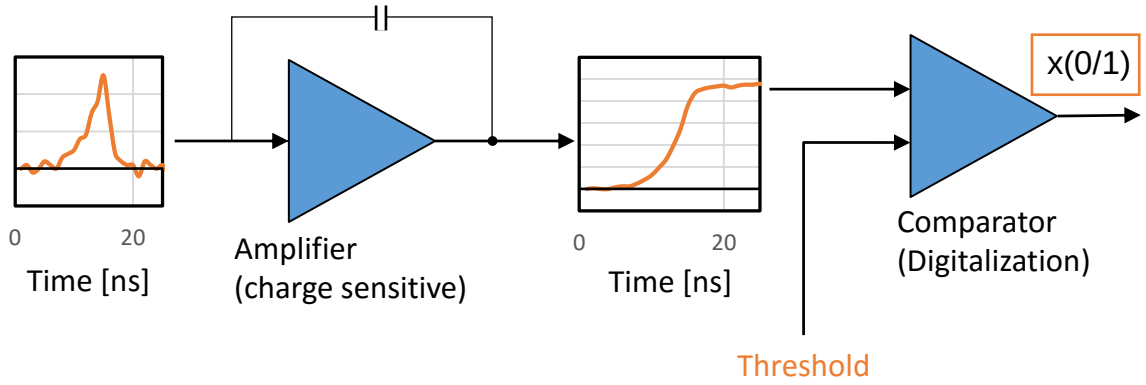


Figure 3.5 Individual channel of the analog Front End of the ABC130 chip

second buffer stage (L1-buffer) to the readout block. The L1-buffer can store the data up to 512 μs. Both buffers are designed to match the current design of the ATLAS phase-II trigger concept. All of the ABC130s on one hybrid are connected via a daisy chain which collects the data from the readout blocks of the individual ABC130. The data are transferred from the daisy chained ABC130 via the Hybrid Control Chip (HCC) to the End Of Substructure (EOS) block, from which they are sent via optical fiber to the outside DAQ system.

3.5.2 Analog Front End Input Block

The ABC130 possesses an analog Front End block with 256 channels. Figure 3.5 shows a simplified schematic diagram of an individual channel of the Front End block. It consists of two parts. The first stage consists of a charge sensitive preamplifier. It converts the input charge signal from the silicon into a voltage signal which is held for 20 ns. The total time of the signal peak including rise and fall is designed to be smaller than 25 ns (one bunch crossing). In the second stage, the voltage signal is compared to a threshold voltage. The actual circuit is more complex and also includes the means to trim the individual threshold values to take channel specific variations into account. These trim values have to be determined before the measurements and do not change over the course of the operation, as long as the setup does not change. Therefore they are not detailed further in this thesis. The output from the comparator is simple binary information showing the voltage value obtained from the preamplifier was above the threshold or below.

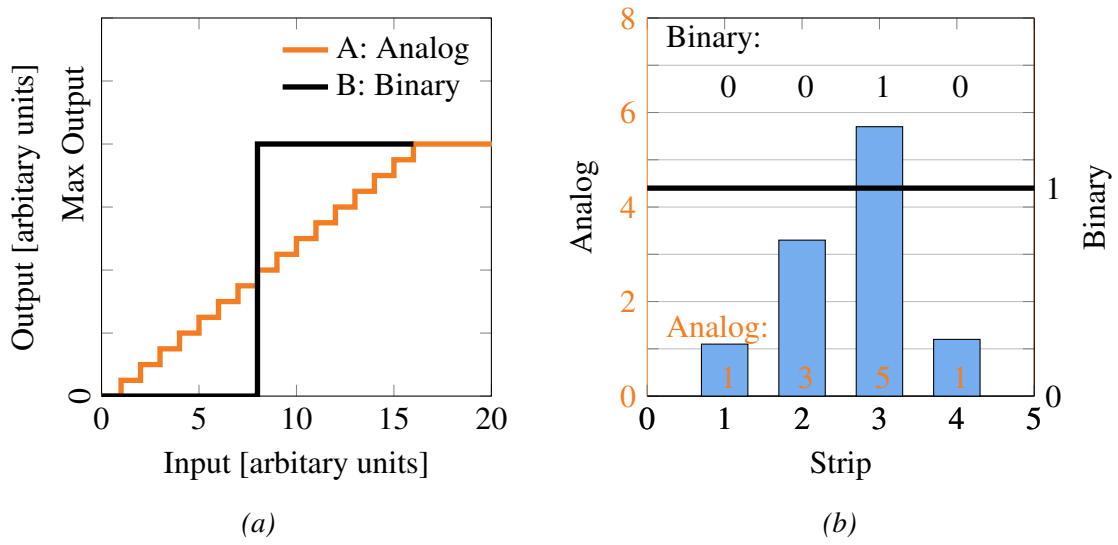


Figure 3.6 (a) Exemplary comparison between analog and binary readout. The analog readout [A] digitizes the input in one of 16 output values. The binary system [B] separates only in above and below threshold.
(b) Output of a binary and of an analog digitalization step. The analog readout preserves an approximation of the charge information.

3.5.3 Limitations of a Binary Readout System

For use in the actual ATLAS detector, a binary system is fully sufficient, but at the R&D stage it has some limitations. To understand the limitations of a binary system it is necessary to compare the binary readout system with an analog readout system. In this thesis the term "analog readout system" refers to a digital readout system with a finite Analog Digital Converter (ADC) register depth and a "binary readout system" refers to a digital readout out system with only two states (above/below threshold). Figure 3.6a shows a comparison between the two readout systems. For an analog system the input signal gets digitized in steps of the buffer granularity; in the example shown, the analog system has a granularity of 16 steps. For a binary system the granularity is one step. Figure 3.6b shows the output of these two readout systems when applied to a strip sensor. While the analog system preserves an approximation of the input charge, the binary system does not. Therefore it is possible with an analog system to count the charge inside a cluster while for a binary system it is only possible to count the number of strips above threshold. It does not provide any information about strips below threshold. With this limitation, charge information can only be obtained over a statistical method like a threshold scan.

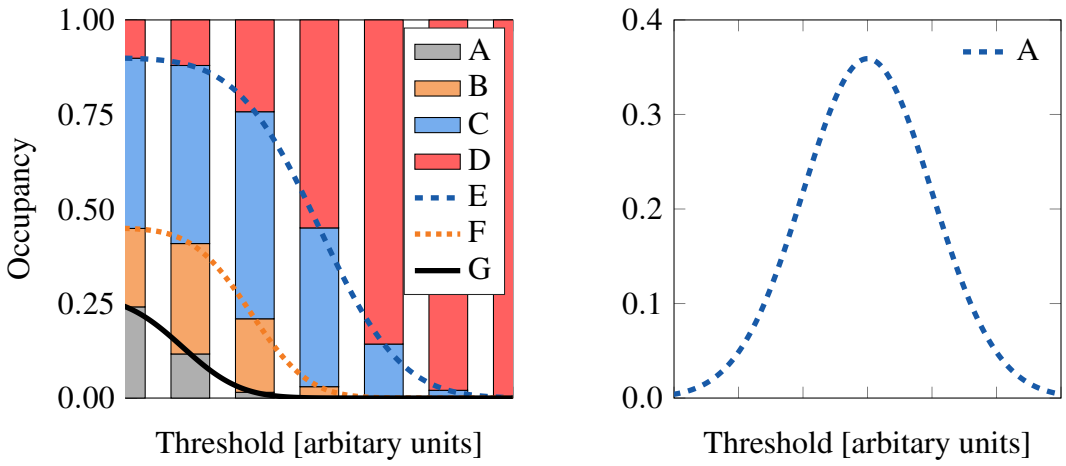


Figure 3.7 (a) Example threshold scan. [A] fraction of events with cluster size cl ≥ 3*. [B] fraction with cl* $= 2$*. [C] fraction with cl* $= 1$*. [D] no strips above threshold. From this the S-Curves for the n-th highest strip can be extracted. [E] S-Curve of the leading strip. [F] S-Curve of the second leading strip. [G] S-Curve of the third leading strip.*
(b) Charge spectrum for the n-th highest strip obtained by differentiating the corresponding S-Curve from (a). [A] charge spectrum of the leading strip. [B] second leading strip. [C] third leading strip

3.5.4 Threshold Scan

Figure 3.7a shows an example threshold scan of an ABC130 Front End chip. To perform a threshold scan, the signal efficiency at a certain threshold is measured N times. Each event is categorized by whether it has strips above threshold and whether they form a cluster. A cluster is a group of neighboring strips which have a signal above threshold initiated by the same input signal. [A] - [C] correspond to the fraction of events with: [A] cluster size larger than two, [B] cluster size equal to two and [C] cluster size equal to one. [D] corresponds to events with no strip above threshold. By combining all events with at least cluster size m, the S-Curve for the m-th highest strip can be extracted [E]-[G]. The leading strip occupancy [E] corresponds to the signal efficiency of the detector at a certain threshold. This is equivalent to the integrated charge spectrum of the leading strip. From the threshold scan, the charge spectrum of the strips can be obtained by differentiating the corresponding S-Curve. Figure 3.7b shows the charge spectrum obtained from the S-Curves. Each line corresponds to the charge spectrum of the m-th highest strip. [A] corresponds to the charge spectrum of the highest strip. [B] corresponds to the charge spectrum of the second highest strip. [C] corresponds to the charge spectrum of the third highest strip. This method allows to study the collected charge spectrum of the m-th highest strip. Since all information about the charge

spectra are obtained indirectly, it is not possible to directly extract the total charge spectrum. For this thesis, only the charge spectrum of the leading strip is investigated. The error for the individual points of the threshold scan (σ_p) is calculated from the variance (σ_n^2) given by the binomial distribution:

$$\sigma_n^2 = N \times p(1-p) \tag{3.1}$$

$$p = \frac{n}{N} \tag{3.2}$$

$$\sigma_p = \frac{\sigma_n}{N} \tag{3.3}$$

$$\sigma_p = \sqrt{\frac{n}{N^2} \times \left(1 - \frac{n}{N}\right)} \tag{3.4}$$

Where N is the number of measurements, n is the fraction of events with a charge above threshold and p is the probability of an event with a charge above threshold.

Chapter 4

Silicon Detectors

Semiconductors play an important role in the development of highly granular charged particle tracking detector systems. This chapter starts with a brief overview about the working principle of a semiconductor tracking detector. It continues with the charge deposition in semiconductors and corresponding spectrum. Following this, it progresses by describing the Readout of the device. This will cover the individual steps the signal has to take until it reaches the measuring computer. The last part will describe the function used to fit the data obtained by the device.

4.1 Working Principle of a Semiconductor Detector

Even though there is a huge variety of semiconductor particle tracking detectors, they rely on very similar physics processes. This section gives a brief overview over semiconductors, the pn-junction and the sensor layout. More detailed descriptions are given in [**? ? ? ?**].

4.1.1 Semiconductor

Semiconductor physics is a subbranch of solid state physics. As opposed to atomic or molecular physics it has to deal with a large number of atoms $\sim 10^{23} \frac{\text{atoms}}{g}$. Similarly to molecular physics the bonding of solid states depends on the sharing of valence electrons. Since electrons are fermions, every electron has to have its own state which differs from the other states by at least one quantum number. For molecules, this manifests in new and shifted energy states for the electrons. The number of constituents in a solid state material makes it necessary to describe the bonding of the atoms in the context of an effective many body theory. One result from this is that the electron states become so dense that they can be treated as a continues band of states as shown in figure 4.1. The blue band shows the

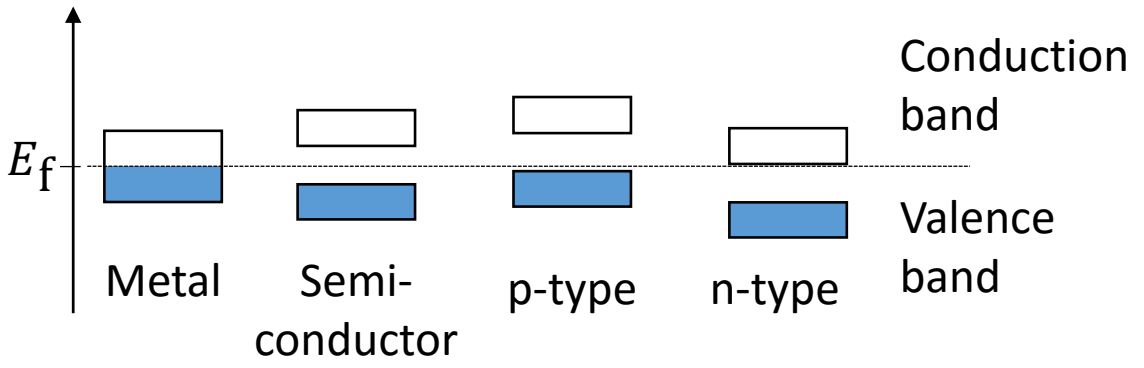


Figure 4.1 Comparison of the band structure of metals, intrinsic semiconductor, p-type and n-type semiconductor. The blue areas mark regions with filled electrons states (valence band). The blank areas mark regions with free electron states (conduction band). E_f marks the Fermi energy.

filled states and the blank band shows the free states. In order to have charge transport in a solid state it is necessary for the electron to have free states available. In the figure, E_f marks the Fermi energy which is the energy of the highest occupied state at 0 K. For metal, the Fermi energy is in the energy band. This means that the energy band has free energy states left which allow the electrons to move freely through the metal, this makes metals excellent conductors.

As opposed to metals the charge carriers in a semiconductor require a certain amount of energy to overcome the energy gap between the valence band and the conduction band. Whereas for metals, the charge is only carried by the negative charged electrons a semiconductor also has positive charge holes which carry charge and contribute to the current[1]. In a pure semiconductor (intrinsic semiconductor) the number of positive and negative charge carriers are equal. By implanting atoms into a semiconductor (doping) it is possible to change its properties. By doping a silicon (fourth group) semiconductor with an atom from the third group (like boron), which only has three valence electrons additional holes are added to the semiconductor. This changes the contribution of negative and positive charge carriers and makes it a p-type semiconductor. By adding an atom from the fifth group (like phosphor) to the silicon, additional free electrons are added. This leads to an majority of negative charge carriers and therefore it becomes an n-type semiconductor.

[1] A "hole" is an empty electron state. In an effective theory this empty states are treated as an particles of the opposite charge than the missing particles.

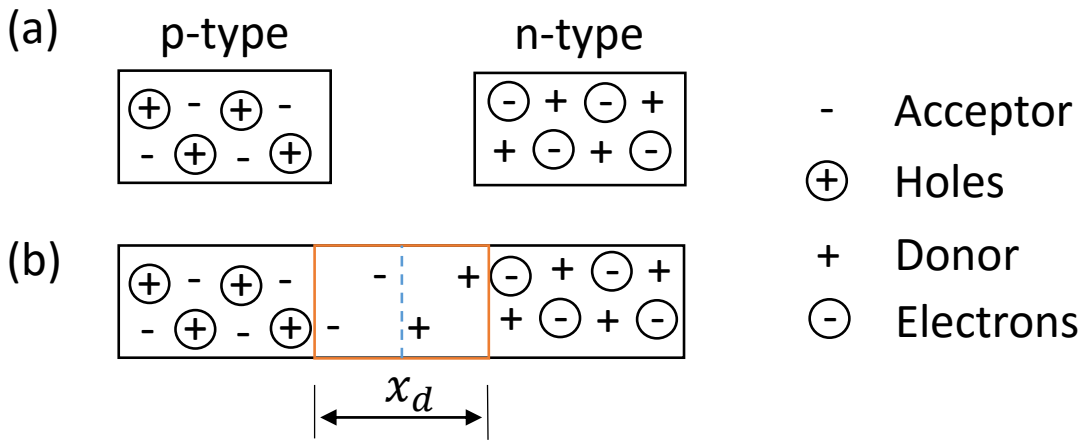


Figure 4.2 pn-junction.
(a) Separated p-type and n-type semiconductor. The p-type is doped with acceptor atoms. The acceptor catches electrons and becomes negatively charged. The missing electron leaves behind a hole which is free to move. For the n-type this effect is reversed and the donor atom contributes one electron to the conduction band of the semiconductor leaving the donor with a positive charge behind.
(b) At the interface of a p-type and an n-type semiconductor, a pn-junction is formed. The electrons and the holes can recombine at the junction leaving the semiconductor with a insulating region with no free charge carriers. The width of the depletion zone is given by x_d.

4.1.2 The pn Junction

Figure 4.1 shows that the additional atoms in p-type and n-type semiconductors do not only change the ratio between positive and negative charge carriers. They also shift the Fermi energy from the middle of the band gap to the acceptor and donor states next to the valence band and the conduction band.

By combining n-type and p-type semiconductors a pn-junction is formed. The Fermi energy for both types has to be equal. This results in a lower energy for the bands of the n-type and a higher energy for the bands of the p-type. This causes the free charge carriers to move away from the pn-junction. This leaves the semiconductor with a region of no free charge carriers (the depletion zone), as shown in figure 4.2. The pn-junction can be biased in two different configurations. First: the forward bias in which a positive bias voltage is applied from the p-type to the n-type. Second: the reverse bias in which a positive bias voltage is applied from the n-type to the p-type.

For the case of a forward bias, the depletion zone gets smaller with larger voltage and above a certain threshold the depletion zone vanishes and the pn-junction becomes conductive.

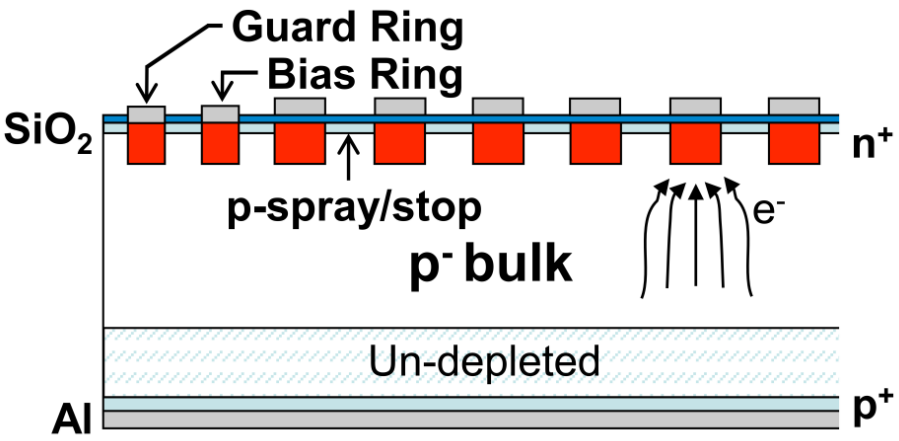


*Figure 4.3 Front cross section view of an ATLAS12 sensor [**?**].*

In the case of reverse bias the free charge carriers are tracked away from the pn-junction, which widens the depletion zone. The pn-junction remains insulating and only a small leakage current goes through the semiconductor. Above a critical voltage (the break through voltage) the leakage current increases dramatically. Depending on the device, the break through voltage can range from tens of volts up to kilo volts.

The depletion zone is interesting for the development of detector systems because it is a defined region inside a solid state which has almost no free charge carriers. This makes it possible to detect even small amounts of charge deposited from an outside effect such as a charged particle passing through it. At the same time the charge created inside the depletion zone is free to move and therefore can be measured electronically.

4.1.3 Sensor Layout

Virtually all semiconductor detectors for the observation of charged particles consist of a pn-junction in reverse bias mode. This means the pn-junction has no free charge carrier and therefore is not conductive. The charge deposition in semiconductors will be described in section 4.2. For the understanding of the sensor layout it is only important that when a charged particle passes through a semiconductor it continuously transfers a small portion of its energy upon the electrons of the semiconductor. With this extra energy the electrons can leave the valence band and can jump to the conduction band. In this process, pairs of free charge carriers are created. The sudden appearance of free charge carriers makes the depletion zone conductive until the charge has moved out. This charge can be measured with sensitive readout electronics.

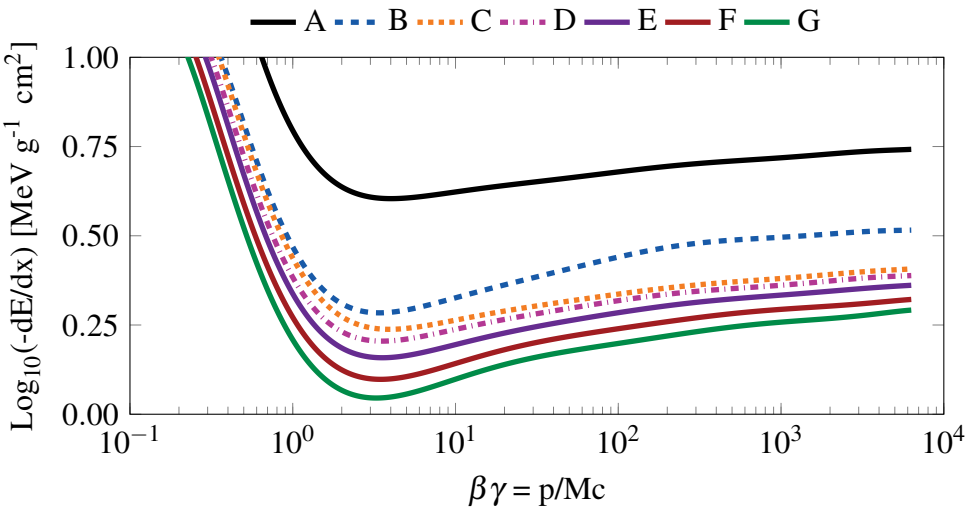


*Figure 4.4 Relative stopping power for a set of materials. [A] H_2 liquid. [B] He gas. [C] C. [D] Al. [E] Fe. [F] Sn. [G] Pb. [**?**]*

A conventional sensor layout consists of a bulk material of a given type and implants of the opposite type. For this work the so called ATLAS07 and ATLAS12 sensors were used [**?**]. Figure 4.3 shows a sketched cross-section of an ATLAS12 sensor. It consists of a large p-type bulk with small n-type strip implants. The aluminum readout strips are AC coupled to the n-type implants. The DC insulation is done with a silicon-oxide layer. The pitch size of the sensor is 74.5 μm. During operation the sensor will be exposed to a high dose of radiation. This makes it likely that after years of operation the sensor can not be fully depleted anymore. Due to the n in p layout the depletion zone starts at the readout strips. This allows for sensor operation even if the bulk is not fully depleted.

4.2 Charge Deposition in Silicon

This section will give a short overview on the charge deposition in semiconductors as it is described in [**?**] and [**?**]. It will first describe how the energy is transferred from the charged particle to the semiconductor. It then continues by giving an estimation for the energy spectrum deposited in a given semiconductor. It concludes by calculating the amount of charge deposited by a Minimum Ionizing Particle (MIP) passing through a thin silicon sensor. Since MIPs deposit the least amount of charge in a material, they are usually taken as a reference to qualify a given sensor.

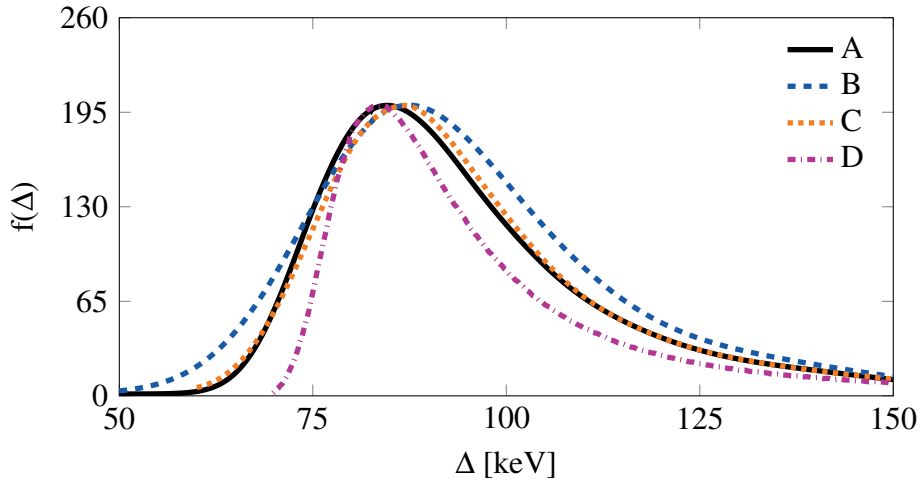


Figure 4.5 Calculated spectrum of the total energy deposition of a charged π with an energy of 45 GeV *in* 300 µm *silicon. The MP-Values are: [A]* 84.8 keV, *[B]* 87.7 keV, *[C]* 86.4 keV, *[D]* 83.5 keV. *[?]*

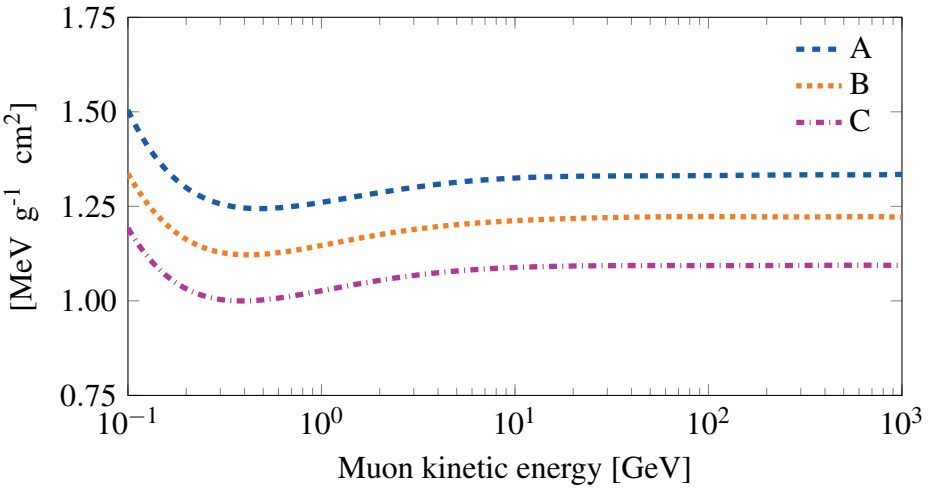


Figure 4.6 MP-Value of the relative energy deposition (electronic losses only) in silicon for various material thicknesses and a range of particle momenta [?]. The relative energy loss increases with the thickness of the material. [A]-[C] show the MP-Value for different thicknesses of silicon. [A] 1600 µm. *[B]* 320 µm. *[C]* 80 µm.

4.2.1 Energy Loss Rate in Material

For the charge deposition in silicon, there are two main contributing processes. The first one is the homogeneous ionization of atoms along the trajectory. The second process is the "radiation" of delta electrons, which is a hard interaction with one of the electrons of the shell. In this process the δ electron gets a high enough energy to travel a noticeable distance.

The total energy deposit in the material is given by the Bethe-Bloch Formula [**?**] (see equation (4.1)). It is sensitive to the different materials the charged particle is passing through. It is dependent on the atomic number A, the charge number Z and the ionization energy I. But within the limitations of this formula it only depends on the β and the $\beta\gamma$ of the charged particle passing through the material. This effect can be studied in figure 4.4. It shows the relative energy loss of charged particles in different materials. This picture shows three properties of the energy loss that are relevant to this thesis. First; for different particles only the x axis has to change with respect to their mass. Second; the rise of the energy loss for high energies is very modest. Finally; the relative energy losses for different materials stay within the same order of magnitude.

$$-\frac{dE}{dx} = Kz^2\frac{Z}{A}\frac{1}{\beta^2}\left[\frac{1}{2}ln\frac{2m_ec^2\beta^2\gamma^2T_{max}}{I^2} - \beta^2 - \frac{\delta_{dec}}{2}\right] \tag{4.1}$$

4.2.2 Total Energy Loss in Material

Figure 4.5 shows the deposit energy spectrum for 45 GeV π in 300 μm silicon. Besides the different particles and energies the setup is compatible with the setup used in this thesis. The different lines show means of calculating the spectrum as shown in [**?**]. The spectrum has a Landau shape (the landau distribution is discussed in section 4.4). It is important to note that even though the width of these distributions do differ, the MP-Values for all distributions are very close to each other.

Figure 4.6 shows the MP-Value of the energy loss spectrum normalized to the specific weight for a large range of particle momentum. It is remarkable that for high momentum the MP-Value stays, for the Landau/Vavilov/Bichsel model, almost constant ($\Delta E_{mpv} = 1.2\ \frac{\mathrm{MeVcm^2}}{\mathrm{g}}$). From this figure the total energy deposited in 300 μm can be calculated to be:

$$E_d = \Delta E_{mpv} * \rho_{Si} * d = 84.1\ keV \tag{4.2}$$

With $\rho_{Si} = 2.3\frac{\mathrm{g}}{\mathrm{cm^3}}$ defined as the specific weight of silicon. This value is within the MP-Values from figure 4.5. The energy deposited in the sensor will create electron hole pairs. The creation of a single pair takes 3.63 ± 0.03 eV [**?**]. With this the total charge can be calculated to be:

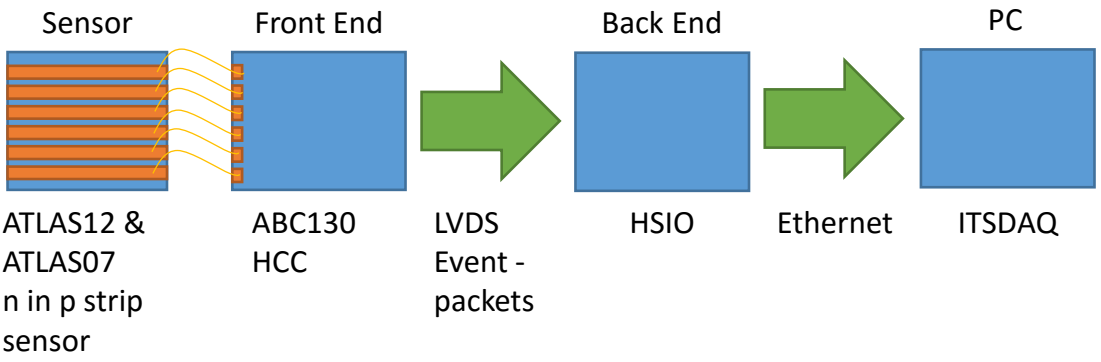


Figure 4.7 Overview of the four base components in which the readout is modulized.

$$Q = \frac{E_d}{3.63\ eV} = 23.2\ \text{ke}^- = 3.7\ \text{fC} \tag{4.3}$$

4.3 Readout Chain

This section will give a brief overview of the readout chain used in this thesis, as sketched in figure 4.7. It uses a conventional approach by splitting the task into four components. It starts with the sensor itself. It consists only of passive components like resistors and capacitor. At the surface the sensor provides aluminum pads on which small wires can be bonded. These so called bond pads are connected to the individual strips. The actual layout and routing of the connection from bond pads to the strips is a current topic of research.

Closest to the sensor itself is the Front End. The task for the Front End is only to capture the information from the sensor and send it as packets to the back end electronics. The actual Front End consists of more parts than just the ABC130. It also consists of a Hybrid Control Chip (HCC), which handles the data streams to and from the Front End. Since space close to the sensor and therefore close to the interaction point in the detector is very limited the Front End does not perform any sort of high level computation.

After the readout information is packed and converted it is sent to the High-Speed-Input-Output (HSIO) board [**?**]. The HSIO is a custom Field Programmable Gate Array (FPGA) [**?**] board with a specially designed firmware for the readout of the Front End. It handles the triggering of the readout as well as the transferring of the data to the PC. The HSIO is able to do a variety of computation such as making histograms of strip hits. However for this

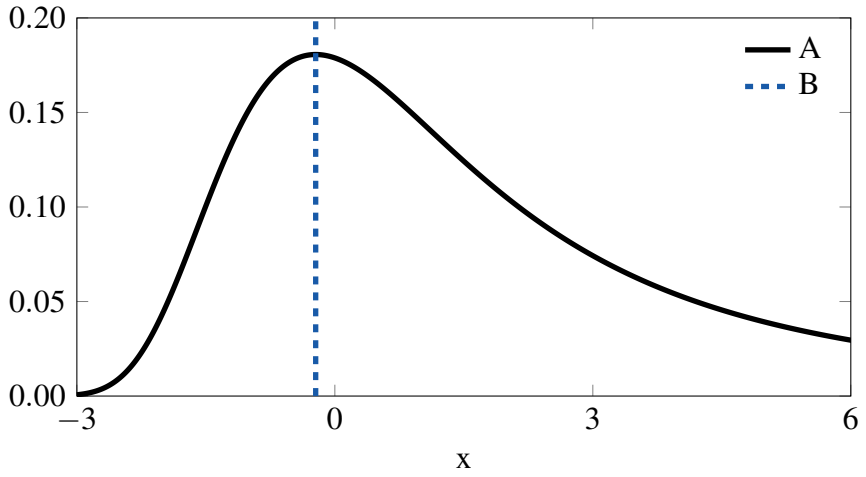


Figure 4.8 Landau-Function [A] Probability density function. [B] MP-Value

thesis it is mainly used to handle the triggering and to provide a link between the PC and the readout.

The last part in the chain is the PC. It provides, through the software ITSDAQ, a user interface to control, receive and visualize the readout. It provides the possibility to store the data on disc. In the course of this work an interface has been created that allows this software to interact with the telescope data acquisition framework, EUDAQ (see section 8).

4.4 Landau Gauss Fit Function

The spectrum of the deposited charged can be described by a Landau-Function [**?**]. It can be described by the two equivalent equations:

$$p(x) = \frac{1}{2\pi i} \int_{c-i\infty}^{c+i\infty} e^{s\log s + xs}\, ds \tag{4.4}$$

$$p(x) = \frac{1}{\pi} \int_{0}^{\infty} e^{-t\log t - xt} \sin(\pi t)\, dt \tag{4.5}$$

As shown in figure 4.8, the Landau-Function peaks at $x_{mpv} \approx -0.22$. Compared to a gaussian function it shows a strong asymmetry towards higher x values. For the Landau-Function the mean and the variance is undefined and it has no free parameter. In order to use this distribution as a fit function L two free parameters are introduced. Firstly; the most probable value (short: MP-Value or mpv) which gives the position of the distribution. Secondly; a parameter which factorizes the spread of the distribution; the parameter is called σ_L. It must not be confused with an actual sigma derived from the second moment

of the distribution. It is a parameter which describes the difference in spread to a pure Landau-Function. The function is described by:

$$L(\mathrm{x},\mathrm{mpv},\sigma_\mathrm{L}) = \mathrm{p}\left(\frac{\mathrm{x}-\mathrm{mpv}}{\sigma_\mathrm{L}} - \mathrm{x}_\mathrm{mpv}\right) \tag{4.6}$$

The electronic readout of the charge introduces gaussian noise. The Gauss function G has two free parameters; the mean value μ and the standard deviation σ_G.

$$G(\mathrm{x},\mu,\sigma_\mathrm{G}) = \frac{1}{\sqrt{2\sigma_\mathrm{G}^2\pi}}\,\mathrm{e}^{-\frac{(\mathrm{x}-\mu)^2}{2\sigma_\mathrm{G}^2}} \tag{4.7}$$

In order to model the signal readout by the device both functions have to be combined. The Landau-Function from the charge distribution itself and the Gauss function from the noise of the readout electronics. The two distributions are mathematically combined by applying a convolution. The convolution is defined by:

$$con(x) = (f*g)(x) = \int_{-\infty}^{\infty} f(y)\,g(x-y)\,dy \tag{4.8}$$

The result of the convolution of the Landau-Function L with the Gauss function G gives the Landau-Gauss function LG:

$$LG(x,\mathrm{mpv},\sigma_\mathrm{L},\sigma_\mathrm{G}) = \int_{-\infty}^{\infty} \mathrm{L}(\mathrm{y},\mathrm{mpv},\sigma_\mathrm{L})\,\mathrm{G}(\mathrm{x}-\mathrm{y},0,\sigma_\mathrm{G})\,\mathrm{dy} \tag{4.9}$$

It is noticed that the mean value of the Gauss function has been set to zero. This simplification is justified since both distribution obey the following transformations.

$$G(\mathrm{x},\mu,\sigma_\mathrm{G}) = \mathrm{G}(\mathrm{x}-\mu,0,\sigma_\mathrm{G}) \tag{4.10}$$

$$L(\mathrm{x},\mathrm{mpv},\sigma_\mathrm{L}) = \mathrm{L}(\mathrm{x}-\mathrm{mpv},0,\sigma_\mathrm{L}) \tag{4.11}$$

With this, the convolution depends only on the sum $\mathrm{mpv}+\mu$ and not on the individual values. With this, the Landau-Gauss function has three free parameters. The MP-Value of L, the spread σ_L of L and the standard deviation of G (σ_G).

Due to the limitations of a binary readout system only the S-Curves from the threshold scans can be investigated. The spectrum can only indirectly extract by (numerically) differentiating the S-Curves. The differentiation for sparse data points can lead to increased errors. Instead of differenting the data points the Landau-Gauss function LG gets integrated to fit the S-Curves directly. Since there is no analytic form for describing LG the integration is also done numerically. The final fit function ILG is given by:

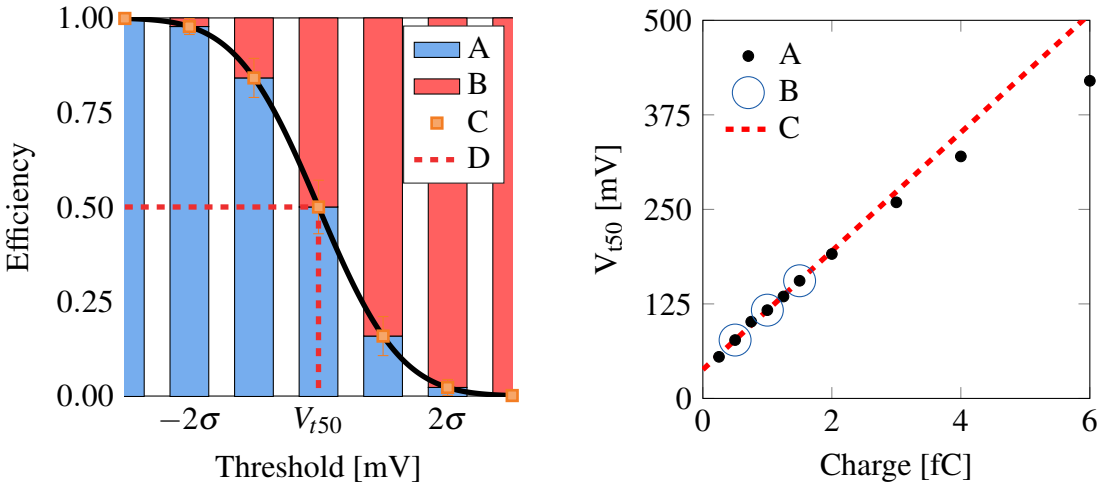
$$ILG(\mathrm{x}, \mathrm{mpv}, \sigma_{\mathrm{L}}, \sigma_{\mathrm{G}}) = 1 - \int_{\mathrm{x}}^{\infty} \mathrm{LG}(\mathrm{x}, \mathrm{mpv}, \sigma_{\mathrm{L}}, \sigma_{\mathrm{G}})\,\mathrm{dy} \tag{4.12}$$

Chapter 5

Laboratory Measurements of the ABC130

The of the main achievement of this thesis is the newly developed ABC130 readout chip introduced in section 3.5. The understanding of its functionality is not only essential for the operation of the actual tracking detector but also for the Monte Carlo simulation of the reconstruction performance of the new tracker. The key attributes of the readout chip are signal detection efficiency and the noise occupancy for a given input distribution. These attributes are, among others, determined by the gain of the preamplifier. The ABC130 provides a means to measure the gain by itself, however it is mandatory to have an unbiased possibility to measure the gain and also it is necessary to perform measurements under realistic conditions. Therefore, it is indispensable to investigate the performance of the ABC130 mounted to a sensor reading out signals induced by MIPs. The first choice would be to perform a test beam study as shown in section 6. However, since a test beam campaign is a major undertaking, it is indispensable to test individual components of the setup such as the triggering, the readout chain and the data analysis first on a smaller scale with a laboratory setup. This section gives an overview about the built-in tests of the ABC130, then progresses by describing the setup which was used to test the synchronization between the Trigger Logic Unit (TLU) and the Device Under Test (DUT) and finishes with a beta source measurement of an ABC130 single chip module.

In this chapter three different DUTs were used. For the built-in tests the same DUT was used as for the test beam studies. It consisted of a barrel hybrid board equipped with three ABC130 readout chips reading out two individual ATLAS12 mini sensors. For the synchronization test a 10×10 cm^2 barrel module was used. The full barrel module was equipped with an ABC250 [**?** **?** **?** **?**] which is the predecessor of the ABC130. The beta source measurements were done with a single chip module housing an ABC130.



(a) *(b)*

Figure 5.1 (a) Example threshold scan. [A] corresponds to the fraction of measurements above threshold. [B] corresponds to the fraction of hits below threshold. [C] Extracted data point for the given threshold. The uncertainty is given by the binomial distribution. The current example is calculated with N = 50 *measurements per point. [D] The* V_{t50} *value is defined by the* 50 *% signal efficiency threshold value.*
(b) Response Curve. To create this response curve ten threshold scans over a wide range of input charges are performed. [A] the response curve shows the V_{t50} *values as function of the input charge. [B] 3PointGain at* 1 fC. *To perform a 3PointGain, three threshold scans with input charges around a nominal value are done. The concrete example shows a 3PointGain for a nominal threshold of* 1 fC. *The data points [B] are fitted with a linear function [C]. From the slope of the function [C] the small signal gain is calculated. By comparing the response curve [A] with the extrapolation from the 3PointGain [C] the non linearity becomes visible.*

5.1 Built-in Tests

The important parameter for the qualification of the ABC130 is the gain of the preamplifier of the ABC130. For this the ABC130 has a built-in test circuit which allows the injection of a defined charge at the analog input of the ABC130. The ABC130 is a binary readout chip, therefore it has no possibility to directly measure the output of the preamplifier. The output can only be investigated through statistical evaluations such as a threshold scan (see section 3.5.4). For a threshold scan the signal efficiency of a given threshold is measured N times. From the N measurements N1 events have a value above threshold. From this a signal detection efficiency for this threshold can be obtained with $\varepsilon_{thr} = \frac{N1}{N}$. To obtain the S-Curve this procedure is repeated for several different input charges. From the S-Curve the so called V_{t50} value can be obtained. This value corresponds to the threshold at which the

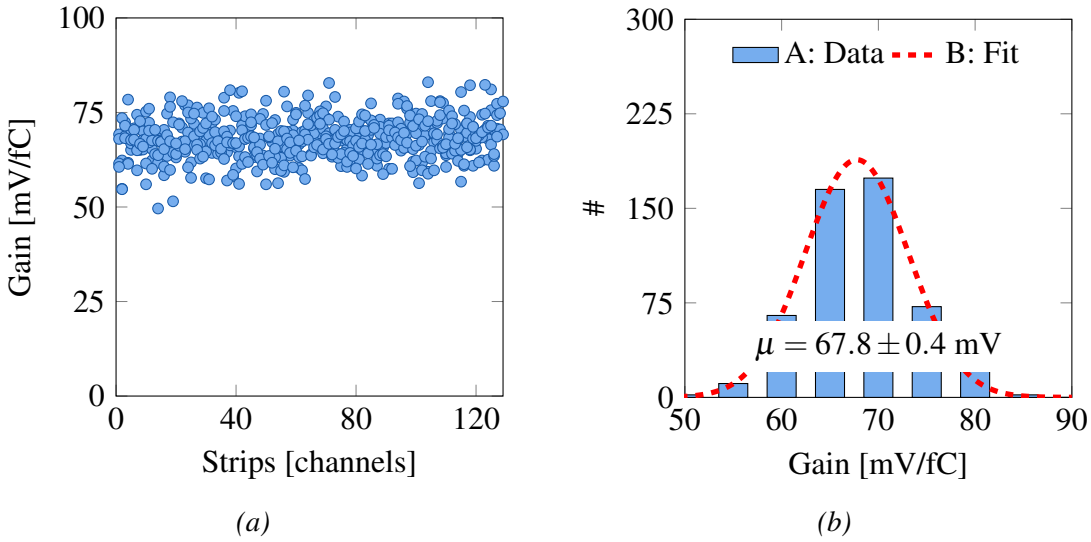


Figure 5.2 3PointGain measured for all strips of a given ABC130. Only bonded channels are investigated. (a) Gain as function of the strip number. (b) Gain histogram [A] with Gaussian fit [B]. Mean $= 67.8 \pm 0.4$. Sigma $= 5.37 \pm 0.41$ *with 95%CL.*

signal detection efficiency (ε) is at 50 % (see figure 5.1a). The response curve is shown in figure 5.1b. From this the gain can be calculated. It is important to note that the response curve is non-linear. This implies that for a given threshold point there are always two gains to consider. The so called working point gain and the small signal gain. The small signal gain describes the slope of the response curve at a given working point while the working point gain describes the slope of the linear function connecting the working point with coordinate origin (pedestal corrected).

Instead of a full response curve, a so called 3PointGain can be useful. For this only three thresholds are investigated (see figure 5.1b) Typical 3PointGain used are at 1 fC [0.5 fC, 1 fC, 1.5 fC] and 2 fC [1.5 fC, 2 fC, 2.5 fC]. From these measurements the small signal gain of the preamplifier can be calculated. In figure 5.2a the result of the 3PointGain is shown on a strip by strip basis and a histogram can be created from this (see figure 5.2b). It shows that the average small signal gain of the preamplifier of the given ABC130 is below 70 $\frac{\text{mV}}{\text{fC}}$. This observation is consistent with the results from various different ABC130 readout chips measured by other groups within the collaboration. This consistently derived value is not within the specification of the ABC130 which is $90 - 95$ $\frac{\text{mV}}{\text{fC}}$. A possible explanation for the observed problem could be the built-in test itself. The control circuit which injects the charge to the analog input could be functioning incorrectly be not working correctly. The built-in test can be likened to trying to measure the accuracy of a ruler by measuring it with itself. Therefore it is necessary to create a test stand where the input to the DUT is not created by

the DUT itself. In this thesis the well understood charge spectrum induced by MIPs in a silicon strip sensor is used as the input charge to the ABC130.

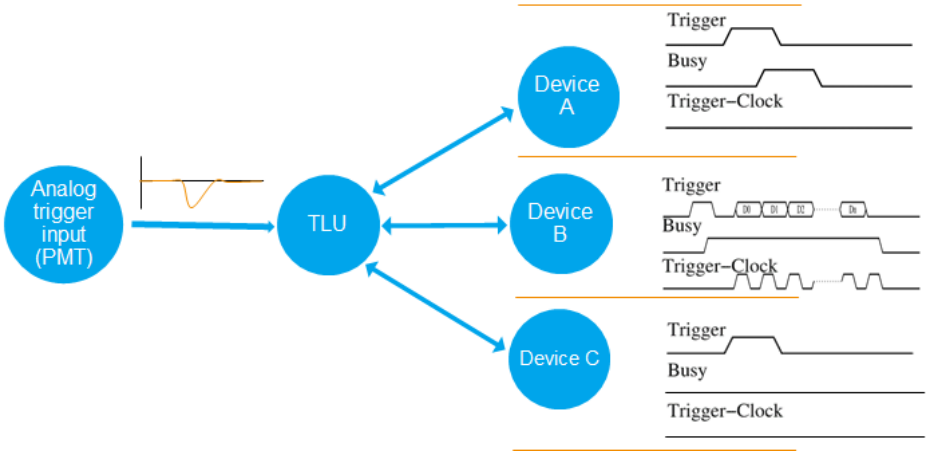
5.2 Synchronization with Trigger Logic Unit (TLU)

For every group which wants to perform a test beam campaign it is a challenge to synchronize their DUT with the EUDET telescope. This section describes two approaches on how to measure the synchronization of a new DUT with the telescope setup under laboratory conditions without a particle beam. In the first part of this section the TLU, a central part of the EUDET type telescopes, is described; the telescope itself is described section 6.1.2. The TLU handles the distribution of the trigger signals, therefore synchronizing the DUT with the telescope is equivalent to synchronizing the DUT to the TLU. During a test beam the synchronization can easily be checked by creating correlation plots (see section 6). Under laboratory conditions without a second particle detector and without a particle beam this approach is not possible, instead, other means have to be found. The second part of this section describes a timestamp based approach. In the third part of this section a more elaborate method is described. It follows the idea that the individual DUT events have to be made "different" to spot a desynchronization in the analysis. For this, the light sensitivity of the sensor is used. Therefore, it is possible to introduce a signal on the sensor with a pulsed LED. These two techniques have been developed in the scope of this work.

5.2.1 The EUDET type Trigger Logic Unit (TLU)

The TLU, as shown in figure 5.3a, is a central part of the EUDET type beam telescopes. It is the central authority and provides the trigger signal to all devices. Figure 5.3b shows a technical sketch of the functionality of the TLU. The TLU provides four analog trigger inputs (Chan0 - Chan3). The trigger inputs are connected via a programmable logic gate. From this gate the trigger is created. The TLU is readout via USB by a computer. It provides information about the trigger inputs of a given event as well as the timestamp of it. The trigger is then distributed to the individual devices. For this it provides two different connectors (LEMO/RJ45). The TLU supports three different means of communication to the individual DUT. These are called handshakes. Device A shows a device which uses a simple trigger busy handshake. The TLU sends a trigger signal to the DUT and the DUT responds with a busy signal which has the length the DUT needs for processing the trigger and transferring the signal. The TLU is blocked during this time and does not issue new triggers. Device B corresponds to a handshake where in addition to the trigger / busy handshake the TLU

(a) TLU as used with the EUDET type beam telescopes.



(b) Visualization of the trigger distribution and the different handshake modes of the TLU.

*Figure 5.3 The Trigger Logic Unit TLU [**? ?**].*

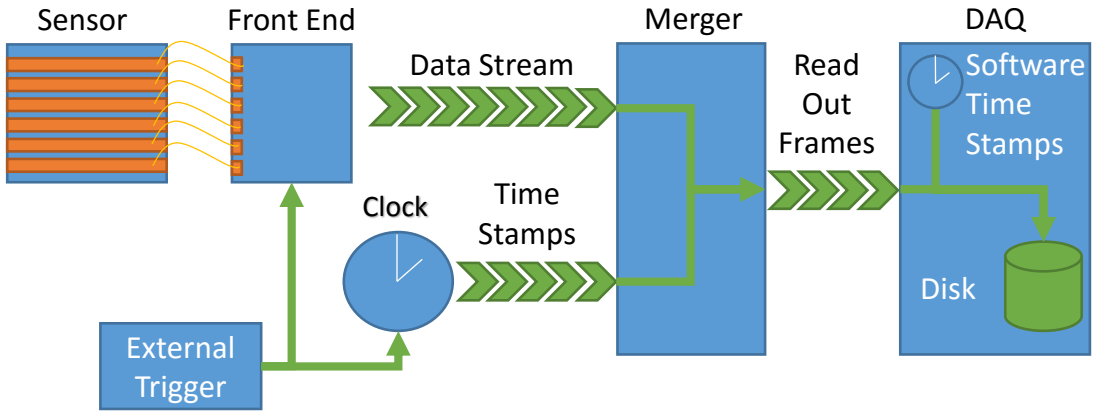


Figure 5.4 Shown are the different components of generic readout system.

trigger ID is also transmitted. The trigger ID can be used to resynchronize the DUT with the TLU in the DAQ framework. The last handshake mode shown by device C corresponds to no handshake. Only a trigger is sent but the TLU does not expect a busy signal from the DUT. This way the TLU is not blocked by the readout of the DUT.

5.2.2 Timestamp based Approach

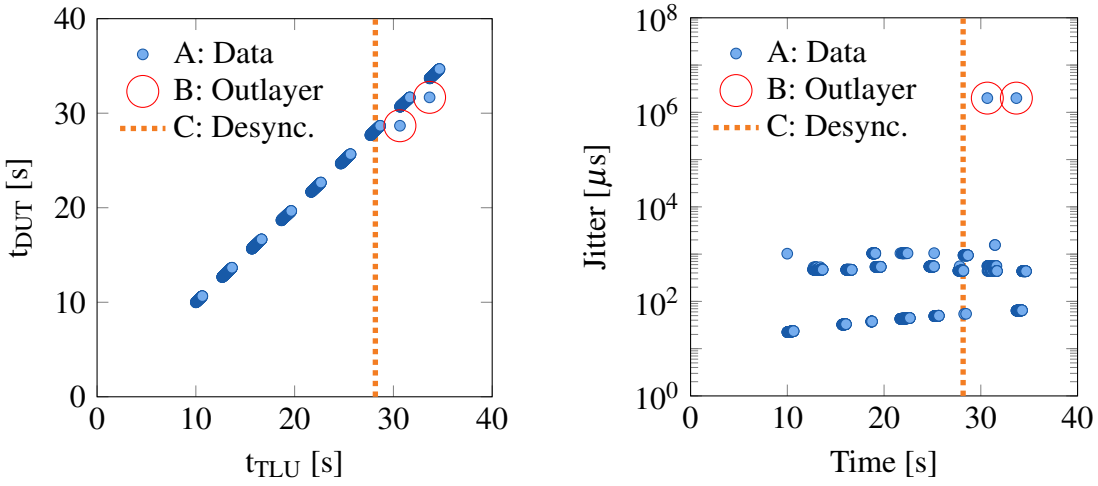
The idea behind this method is that both devices record a timestamp for each event. The time difference between the timestamps can be used to check whether both devices are still synchronized. For this, the timestamps from both devices for a given event have to be smaller than a certain maximum value Δt_{max}:

$$|t^j_{DUT} - t^j_{TLU}| = \Delta t^j < \Delta t_{max} \tag{5.1}$$

With t^j_{DUT} being the DUT timestamp of the given event, t^j_{TLU} the TLU timestamp of the given event and Δt_{max} the maximal length of time allowed between the two. Δt_{max} has to be chosen to be significantly larger than the jitter of the TLU timestamps and the DUT timestamps combined. It has also to be chosen to be smaller than the time distance between two consecutive events:

$$Jitter << \Delta t_{max} << t^{j+1} - t^j \tag{5.2}$$

This implies that there is an upper rate limit to which this method can be used. This is problematic since certain effects only appear at higher rates. Figure 5.4 shows a generic readout system. In this, the external trigger was used to trigger the Front End Chip and



(a) *(b)*

Figure 5.5 (a) Correlation between the timestamps is recorded by the DUT and the timestamps are recorded by the TLU [A]. Due to the FPGA timing board the timestamps are clustered. The gaps between the cluster originate from the long busy signal. At the time [C] the two devices are losing synchronization. The desynchronization results in off diagonal elements between two clusters [B].
(b) Absolute time difference Δt between DUT and TLU timestamps [A] ($\Delta t^j = |t^j_{DUT} - t^j_{TLU}|$). At the time [C] the two devices are losing synchronization. Due to the large jitter the desynchronization is not directly observable. Between the cluster the desynchronization results in a Δt which are of the order of the long busy signal [B]. This measurement was done with a $10 \times 10\text{cm}^2$ barrel model equipped with ABC250 readout chips.

a clock which generated timestamps. At the time of these measurements the module to generate the timestamp was not implemented. It was added later to the HSIO firmware. Due to this limitation the software timestamps recorded by the DAQ system were used. With this limitation, the jitter on the DUT timestamps was of the order of milliseconds. This limits the maximal rate which can be investigated to $100 - 200\ s^{-1}$. The limit is given in average trigger rates. Due to the readout scheme of the TLU the instantaneous rates are higher.

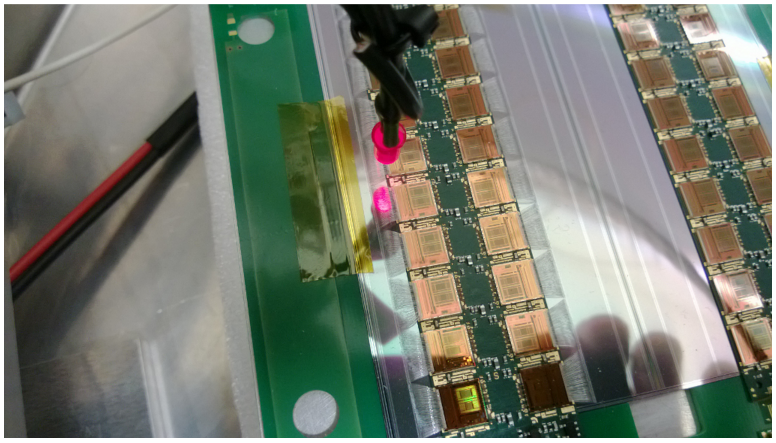
Equation (5.2) gives a limitation on whether this method can detect desynchronization for a given rate and jitter. Fortunately this condition only needs to be true for events where the synchronization is actually checked. With this events can be separated at to those where the synchronization is checked and those for where it is not checked. Only for events where the synchronization is checked does the time difference between two consecutive events needs to be larger than the jitter. To achieve this a "counting FPGA" board was introduced. It was tasked to count the arriving trigger and to produce, after every n-th trigger, a long busy signal. The long busy signal was chosen to be of the order of seconds. With the FPGA board the events were clustered in time into blocks with high trigger rate followed by large gaps. The events around the gaps can be used to check for desynchronization (see figure 5.5a). The long busy signal can be chosen with almost no constrains. This eliminates the limitation due to the jitter completely (see figure 5.5b).

As shown in figure 5.4 the timestamp originates from a different source than the data. Therefore it is not necessarily true that if the timestamps of TLU and DUT are in sync the data itself are in sync as well. No matter if the timestamp is generated on the readout board (HSIO) or in the software, it is additional information which is added at some point to the data. It does not follow the data stream all the way through the readout system. The LED setup (see section 5.2.3) was introduced to overcome this limitation.

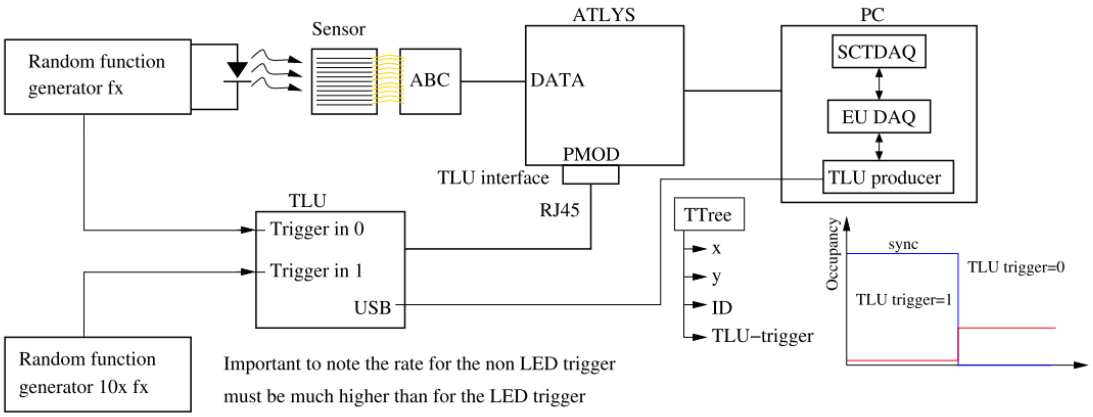
Once the timestamps are understood they can be used to resynchronize the data using the newly developed EUDAQ 2.0 framework (see section 8).

5.2.3 LED based Approach

The timestamp based approach can be useful to give initial insight, however it has its limitations. The most fundamental limitation is that the information used to check for the synchronization (the timestamp) does not necessarily follow the same readout line (or is added at a later state) as the actual sensor information (see figure 5.4). To overcome this limitation it is required to build a test stand which injects signals at the same input as the actual data. In this simplified context the silicon strip sensor which is mounted to the ABC130 works as the input to the ABC130. For this approach it is essential to create distinguishable events. Similarly to charged particles also photons can also induce charge in silicon sensors.



(a) LED setup. Image shows a barrel module consisting of a 10×10 cm^2 *silicon sensor mounted to a hybrid. The hybrid was equipped with ABC250 readout chips. The ABC250 is the predecessor of the ABC130 readout chip. It was used as a dummy before the ABC130 was available. On top of the sensor a red LED (635 nm) was mounted. The LED was connected to a pulse generator which also triggered the HSIO readout board.*



*(b) Schematic view of the latest LED setup [**?**]. Compared to the original setup this setup uses an ABC130 and is readout by the ATLYS [**?**] board instead of the HSIO. The ATLYS board is connected to the TLU with an specially designed RJ45 connector instead of the LEMO connector.*

Figure 5.6 LED Setup

This charge can be measured with the ABC130 as well. The charge distribution of signals induced by photons will be different from the charge distribution of charged particles. In the case of charged particles it is one particle that deposits a small fraction of its energy in the detector. In the case of photons from an LED there are many photons hitting the detector at the same time detector and the photons do not pass through the detector. However for testing the ABC130 this is irrelevant, and the processing in the readout chain is identical. The charge introduced by the LED is large enough such that at least the signal of one strips of the sensor is always above threshold. These events are called "N hit events" or $N-Hit$. Without the light of the LED none of the strips were above threshold. These events are called "zero events" or $0-Hit$. A picture of the first version of this setup is shown in figure 5.6a. It uses a 635 nm LED. The latest version of the test stand as it is currently used is shown in figure 5.6b. The central authority in this setup is the TLU. The TLU is triggered by two independent function generators. The function generator connected to trigger input 0 also provides a trigger signal to the LED. The LED is directly mounted above the sensor which is readout by the ABC130. The function generator connected to trigger input 1 does not trigger the LED. Therefore the events triggered by this trigger do not contain a signal. With this the events created by the different function generator are fundamentally different. The TLU stores the status of the input channels for every event. For the analysis the occupancy of the events is compared. For events corresponding to TLU input 0 every event is expected to have at least one strip above threshold. For events corresponding to TLU input 1 the events are expected to have zero strips above threshold (depending on the threshold). When the desynchronization happens the events corresponding to trigger 0 are now empty and the events corresponding to trigger 1 have now 1 in n events with strips above threshold (depending on ratio of the two frequency). Therefore it is in the analysis software possible to check for desynchronization by checking whether an event with TLU trigger 0 corresponds to a $N-Hit$ event and whether an event with TLU trigger 1 corresponds to an $0-Hit$ event.

The LED setup was used to determine the maximum trigger rate for the first test beam campaign performed with a full size 10×10 cm^2 barrel module. The sensor was readout with the predecessor of the current ABC130 the so called ABC250. Another result of this setup was the creation of an RJ45 interface board between the TLU and the HSIO which was able to handle the trigger signal, encode the TLU ID and send a busy handshake back to the TLU. The desynchronization is a common problem for the tests of different DAQs. Here described is a setup to detect this problem before the actual test beam campaign and is now commonly used by users of the telescope, also outside the ATLAS collaboration.

5.3 Beta Source Test

While the LED was a suitable tool for testing and debugging the trigger and readout chain, the signals induced on the sensor are vastly different to the signals of particles passing through the sensor. In the next step, the system, consisting of module and readout, was tested using a beta source. Compared to the LED, the beta source[1] has the advantage of depositing a similar amount of charge in the sensor to the 5 GeV electrons from the DESY test beam. With the beta source it was possible to study the charge deposition in the sensor as well as the amplification and the digitization of the recorded events. To obtain first insights into the gain of the preamplifier it is essential to have events originating from real particles passing through the sensor. As a reference, a compatible ATLAS12 sensor was measured with an ALiBaVa [**?**] readout system. Both measurements are compared to obtain the gain of the preamplifier of the ABC130.

5.3.1 Setup

After understanding the trigger chain and the readout, the next step is to test the device with a Sr-90 source. The setup, as shown in figure 5.7a, consisted of the following components: The electrons from Sr-90 are collimated to maximize the fraction of electrons in the sample impinging the sensor. While passing through the sensor, the electrons deposit charge. The DUT used in this test consisted of single chip module housing an ABC130 which was mounted to an ATLAS07 [**? ? ? ?**] mini sensor (10×10 mm^2 silicon strip sensor with a pitch size of 74.5 μm). The setup was housed in a metal box which was used to control the environment and to keep the DUT in the dark. As shown in figure 5.7b[2], inside the box was a frame on which the DUT, the beta source and two scintillators were mounted. As a readout device for the ABC130 a High Speed Input Output (HSIO) board was used. The board is connected via a network to a computer which controls the measurements and stores the data. The HSIO board consists of an FPGA which handles the data processing and data packaging. This board is the bridge between the DUT and a computer. The board can be triggered with an external signal with a LEMO connector, as well as with a specially designed RJ45 connector interface board to interact with the TLU. The triggering system consisted of two scintillator planes, each of which were readout by a photomultiplier tube (PMT). The signal from the PMTs was digitized. The trigger signal was defined by a logical AND between the two input signals. The trigger signal was used as input for the HSIO. The area covered from

[1] As beta source a Sr-90 source was used which emits electrons with a maximum energy of ≈ 2.3 MeV [**?**].

[2] The setup was located at the University of Freiburg and measurements were conducted during a research stay.

the two scintillators was 4×4 mm^2. The coincidence of two scintillators was used to trigger only on electrons with higher energy. With the beta source used in this setup a trigger rate of 2 s^{-1} was achieved. Since only a small fraction of the electrons triggered the readout, it was not possible to use a stronger source without increasing the number of double hits on the sensor.

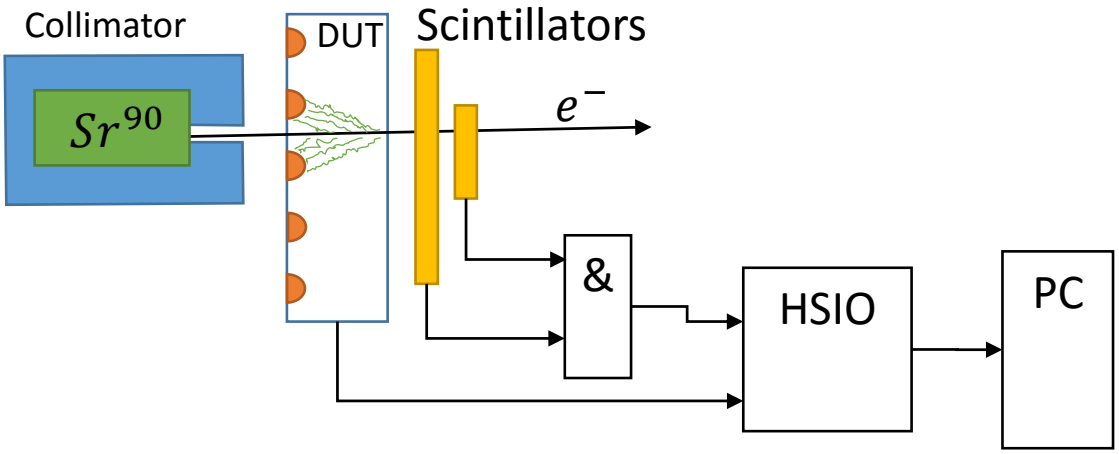
The relative statistical error should be smaller than 0.03, therefore the statistics needs to be higher than $\frac{1}{0.03^2} \approx 1.1$ k events. To be on the safe side, 2 k events were recorded. With an event rate of 2 s^{-1}, and a measurement goal of 14 bias voltages with 48 thresholds each, the bare measurement time would be seven days. The actual measuring time in the end was approximately two weeks. To perform such a long measuring campaign it was necessary to have the setup fully automatized. The automatization included starting runs and setting the bias voltages. It also included recording environmental parameters such as temperature and humidity. In addition, the automatization was also able to monitor the running of the ITSDAQ software and restart it after a crash. Once the system was configured it was able to perform all measurements without human interaction.

5.3.2 Event Categorization

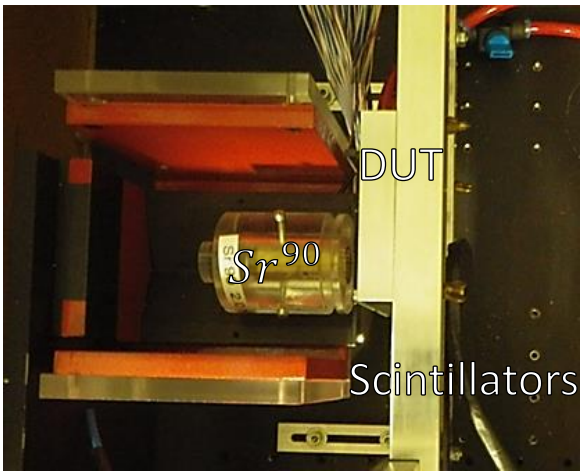
The events recorded from the DUT were categorized in the following two main categories: 'No hit' and 'any hit'. In the first category none of the investigated strips were above threshold ($0-$Hit). In the second, at least one strip is above threshold (N $-$ Hit). This category is divided into three subcategories. First: events with cluster size one. Exactly one strip was above threshold (CL1). Second: events with cluster size larger one. Two or more strips were above threshold and all strips above threshold were in one cluster (CL2$^+$). Third: events with double hits. Two or more strips have been above threshold and they are separated by at least one strip below threshold (2^+-Hit). The difference between (CL2$^+$) events and (2^+-Hit) events is visualized in figure 5.8.

5.3.3 DUT Pedestal Measurement

The pedestal is an artifact of the digitization step. It is common for a digitizer to have only a positive voltage range for the input; therefore it is necessary to elevate the base line. The new base line is now on top of a "pedestal" which is a constant offset of the measured quantity. The pedestal can be measured by performing a threshold scan (see section 3.5.4) at the base line. Below the pedestal the occupancy for all strips is 100 %, above the pedestal it is close to zero. The pedestal is defined as the 50 % value of this threshold scan (V_{t50}). The pedestal

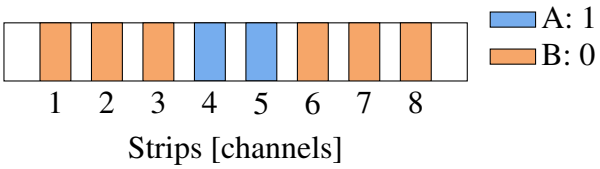


(a) Technical sketch of the measurement setup.

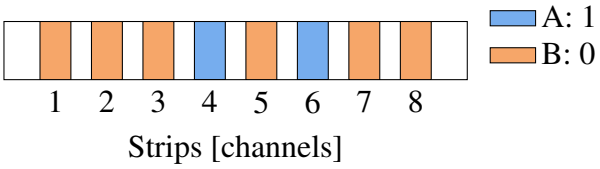


(b) Picture of the actual measuring setup.

Figure 5.7 Schematic diagram of the measuring setup and photograph of the realized setup.

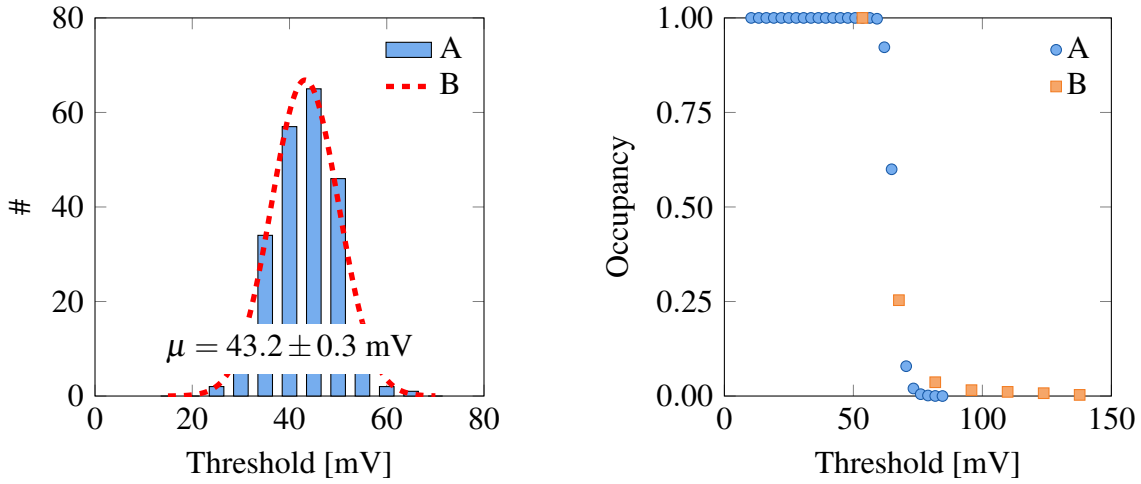


(a) $CL2^+$*: adjacent strips above threshold are counted as one hit with the corresponding cluster size.*



(b) $2^+ - \text{Hit}$*: strips above threshold with at least one strip below threshold in between are counted as independent hits.*

Figure 5.8 Illustration of the difference between $2^+ - \text{Hit}$ *events and* $CL2^+$ *events. [A] Strips above the threshold. [B] Strips below threshold.*



(a) *(b)*

Figure 5.9 (a) [A] Histogram of all pedestal positions (V_{t50}) *of the ABC130 readout chip used in this setup. [B] Gauss fit*
(b) Comparison between the sensor occupancy of $N - \text{Hit}$ *events without beta source [A] and the occupancy of* $2^+ - \text{Hit}$ *events beta source [B]. It is visible that the probability for a double hit* ($2^+ - \text{Hit}$) *is almost identical to the probability of a noise hit. This indicates that the activity of the source is low enough to only have single electrons hitting the sensor.*

has to be extracted for all channels. Figure 5.9a shows a histogram of the pedestals positions (V_{t50}) [A] of all channels and a Gaussian fit [B]. The average pedestal is 43.2 ± 0.3 mV. It has to be taken into account for the calculation of the MP-Value.

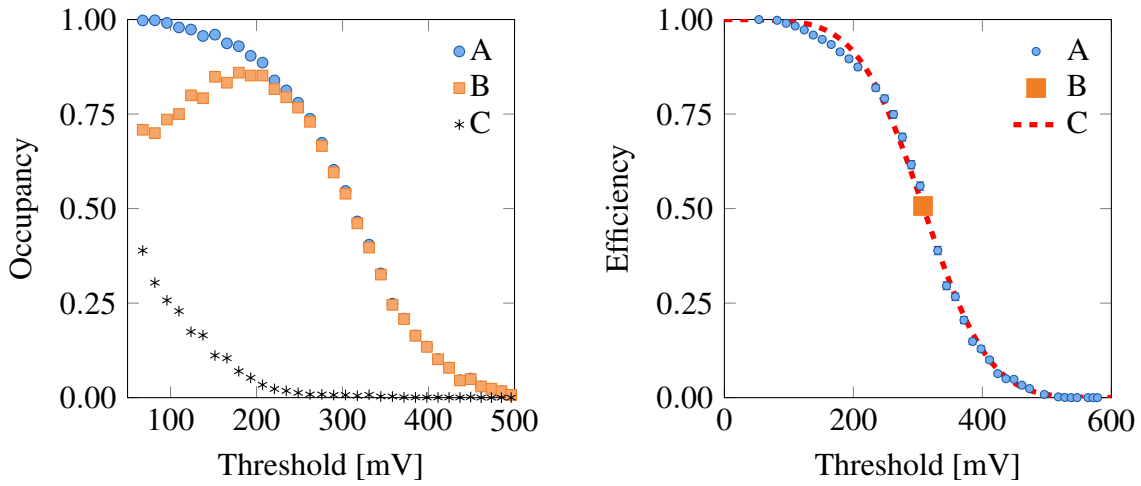
For the analysis it is important to understand the composition of the events recorded. In contrast to the test beam, where every electron can be traced and the predicted impact position can be estimated to down to $5-10$ μm, with a beta source measurement it is much more important to understand every signal from the DUT. While for test beam analysis only strips in a narrow window around the extrapolated hit positions are investigated, at a beta source setup, due to the missing spacial resolution, all strips have to be taken into account. Therefore it is important to understand the occupancy with double hit events ($2^+ -$ Hit). Double hit events can occur from noise hits, as well as from real hits. If the origin of the double hits is the noise of the strips, the double hit occupancy behaves exactly the same as the any hit (N – Hit) occupancy for an auto trigger run. If the double hit events originate from two electrons hitting the sensor, the two hit occupancy would stay constant over a wide threshold range and would decrease like the any hit occupancy with higher thresholds. The beta source has been chosen to be weak enough to not have two electrons hitting the sensor at the same time (see figure 5.9b).

5.3.4 Results

The outcome of the measurements are summarized in this section. This chapter was dedicated to the goal of preparing for the test beam campaign. Therefore as a first result it has been demonstrated that the DUT can be operated for several days continuously.

The second result is that with the setup it was possible to measure not only the S-Curve (N – Hit occupancy), which corresponds to the charge spectrum of the leading strip, but also the $CL2^+$ occupancy, which corresponds to the charge spectrum of the second leading strip (see figure 5.10a). For large thresholds, N – Hit occupancy and CL1 occupancy are exactly on top of each other. This means that above a certain threshold all events recorded are CL1. At high threshold only the charge spectrum of the leading strips can be studied. For lower thresholds the effect of charge sharing also can be studied. $CL2^+$ occupancy corresponds to the charge spectrum of the second highest strip. As described in section 3.5.4, with a binary system it is possible to investigate the charge spectra of the individual n-th highest strips, but not the charge spectrum of the entire cluster.

From the S-Curves, the MP-Values have been extracted with a Landau-Gauss fit (see figure 5.10b). In addition figure 5.11 shows that the MP-Values follow the expected trend of the width of the depletion zone as a function of the bias voltage. The MP-Value increases



(a) Signal Distribution *(b) Fit*

Figure 5.10 (a) Typical signal distribution obtained by a threshold scan. [A] N − Hit *occupancy. [B]* CL1 *occupancy. [C]* CL2$^+$ *occupancy.*
*(b) [A] Signal efficiency (*N − Hit *occupancy) versus threshold scan with Landau-Gauss fit [C]. The maximal errors for [A] is with* 1.1 *% smaller than the data marker. From this fit the MP-Value (*$= 307.8 \pm 2.0$mV*) can be extracted [B].*

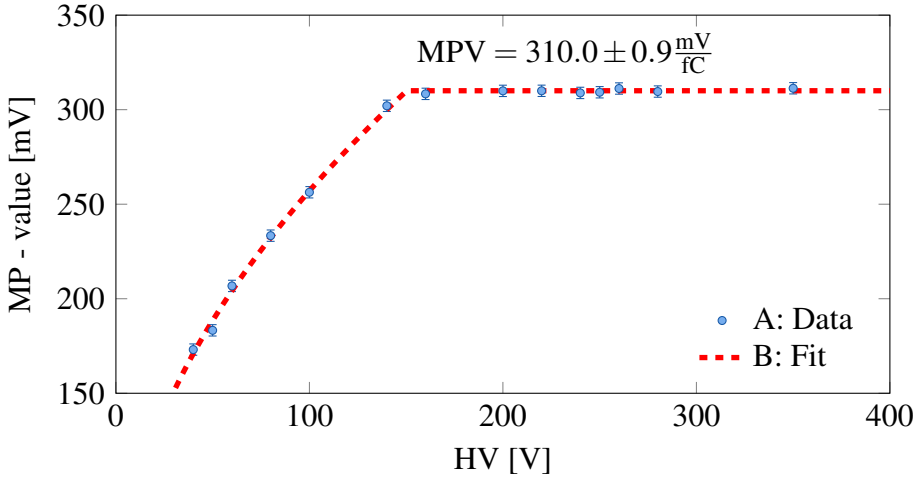


Figure 5.11 Shown here is the MP-Value obtained from a Landau-Gauss fit of threshold scans versus the bias voltage. The charge deposition is proportional to the width of the depletion zone, therefore the measured MP-Value shares the same trend as the width of the depletion zone itself.

with $\sqrt{V_{bias}}$ until the sensor is fully depleted. Above the depletion voltage a further increase of the voltage does not effect the MP-Value.

The third result is the gain calculated from the MP-Values, obtained from the threshold scans. The average MP-Value for all data points above the depletion voltage has been calculated to be $MPV = 310.0 \pm 0.9$ mV. From this value the pedestal value of $P = 43.2 \pm 0.3$ mV has to be subtracted. With the ALiBaVa system (section 7.1) a charge for the leading strip of $Q = 2.85 \pm 0.13$ fC has been measured. From this the gain can be calculated with:

$$gain = \frac{MPV - P}{Q} = 93.6 \pm 4.3 \ \frac{\text{mV}}{\text{fC}} \tag{5.3}$$

With this the gain obtained from the beta source measurement is significantly higher than the one obtained from the built-in test. The discussion of the results will follow in section 7.

Chapter 6

Test Beam

Even though the beta source measurement was giving some useful insights, it still has its limitations. The most important ones are the lack of spacial information regarding where the particle was impinging, the low trigger rate of only two events per second and the low particle energy (≈ 2 MeV) of the beta electron. A test beam campaign at the DESY-II test beam facility[**?**] was carried out to test the system with highly energetic particles. For this, the DESY beam test telescope DURANTA was used [**? ?**]. The EUDET style beam telescopes are used by many groups over the years and they have become a de facto standard tool for tracking detector development at DESY and CERN.

6.1 Test Beam Setup

This section describes the individual components of the test beam setup. This includes the DESY test beam facility as well as the beam telescope and the Device Under Test (DUT) used for these studies.

The DESY test beam facility has some unique qualities which make it an outstandingly powerful tool for the developing of particle tracking detectors. For the test beam a new copy of the very successful EUDET type telescope were used: the DURANTA telescope as shown in figure 6.1. It consist of six MIMOSA26 senor planes and one FEI4 APIX single chip hybrid plane [**?**]. In the middle the DUT was mounted inside a box to control the environment, especially the temperature and humidity, but also to keep it dark. The box was mounted on a movable stage which could be moved in the x and y directions. The MIMOSA26 sensor planes with their combination of low material budget and small pixel size allows one to perform test beam campaigns which obtain very high quality data. Nevertheless the telescope has certain limitations concerning the time resolution of the particle hits (as shown in section 6.1.2). To overcome this problem the DURANTA telescope was combined

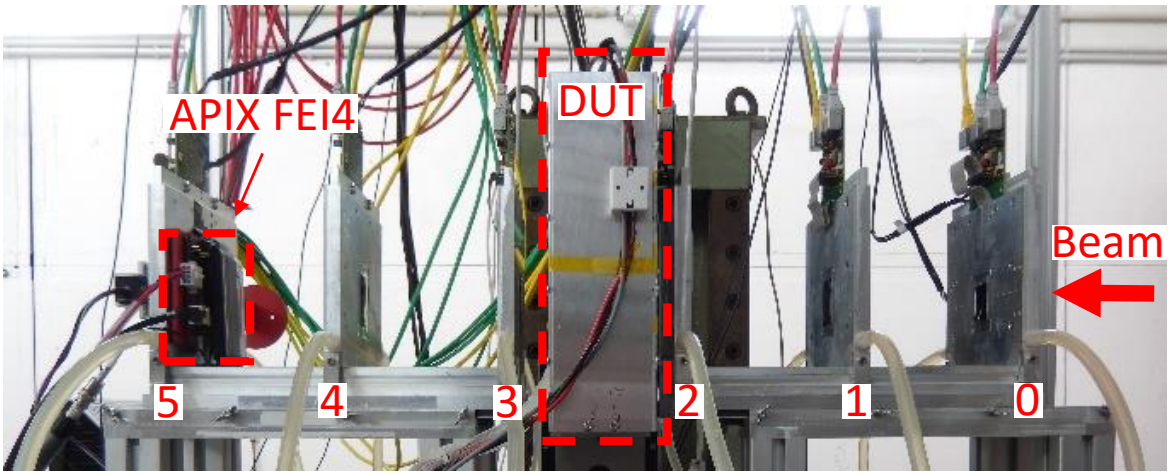


Figure 6.1 Test beam setup. It shows the DURANTA telescope including an additional FEI4 APIX plane.

with a FEI4 APIX. For technical reasons the FEI4 APIX had to be mounted in front of the last MIMOSA26 plane instead of behind. The relative distance between the two planes was so small that the effect of multiple scattering was minimal. With this addition the telescope setup combines the best of both worlds. It has the high spacial resolution of the EUDET type telescopes and the precise time resolution of the FEI4 APIX. The DUT was a full module hybrid with two ATLAS12 mini sensors mounted. It was hosted in a light and air tight box to control the environment of the sensor.

6.1.1 DESY-II Test Beam Facility

The measurement was done at the test beam facility at DESY. It uses the DESY-II accelerator, which is currently used as pre-accelerator for PETRA3 electron/positron storage ring. The beam is extracted with a thin carbon fiber (radius $< 10\ \mu m$) which is placed inside the electron beam. The electrons hitting the carbon fiber create "Bremsstrahlung" photons which continue straight and leave the ring accelerator. Outside the ring the photons hit a converter target made of copper, aluminum or other materials. On this target the photons interact with the material and create electron positron pairs. Directly behind the converter target is a dipole magnet which is used for energy selection [**?**]. The user can vary the beam energy from 1 GeV up to 6 GeV. Since there is a close relationship between beam energy and beam rate one has to find a compromise. The rate is highest for energies around 3 GeV and continuously declines for higher and lower energies[**?**]. To reduce the effect of multiple scattering it is beneficial to use higher beam energies [**?**]. For the test beams studies presented here a beam

energy of 4.8 GeV was used, which gave a rate of several hundred particles per second while the multiple scattering is still small.

6.1.2 DURANTA Telescope

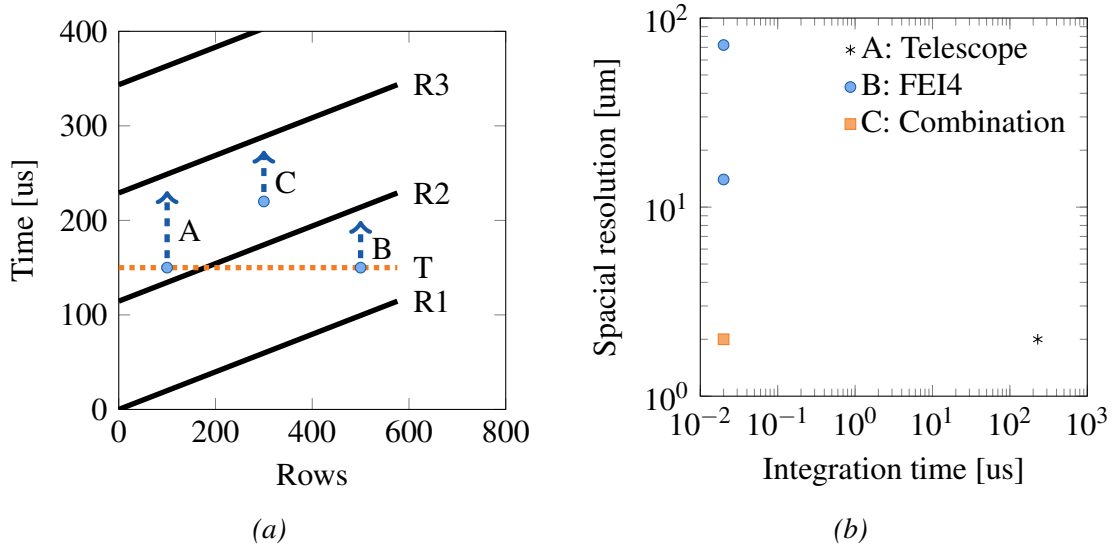
The EUDET style test beam telescopes [**?**] are developed as part of the EUDET project. They are focused on providing accurate position resolution for charged particles. They consist of six MIMOSA26 planes, which have an active area of $1 \times 2\ cm^2$ and a pitch size of 18 μm. Under optimal conditions they achieve a pointing resolution of below 2 μm [**?**].

The readout of the MIMOSA26 is done using a rolling shutter principle (see figure 6.2a). Every pixel stores the hit information until it gets readout. During the readout the pixel is reset. One full readout cycle of the continues readout takes 114.5 μs and after the cycle the readout starts again from the first row. Thus each pixel integrates over the full readout time. This means that the time coordinate of the hit cannot be determined more precisely than 114.5 μs and that it is very likely for two particles to hit the MIMOSA26 in the same readout frame. The individual readout cycles are indicated with diagonal lines. The horizontal line indicates when a particle passes through the telescope and the TLU issues a trigger. Hits are indicated with circles. The x coordinate of the hit shows on which row the hit appeared and the y coordinate shows at what time it occurred. Hits [A] and [B] occurred at the same time but due to their different row position they end up in two different readout frames. Therefore the time resolution of the MIMOSA26 sensor is at least two readout frames of 114.5 μs. Hit [C] happened at a later time but it ends up in the same readout frame as hit [A] even though it is not related to the particle that issued the trigger. For efficiency studies this can lead to problems since the integration time of an LHC type device is typically of the order of 25 *ns*. To overcome this limitation the telescope was combined with another LHC type device: the ATLAS FEI4 APIX detector, which has an equally short integration time.

6.1.3 FEI4

The FEI4 APIX hybrid sensor is an ATLAS pixel detector and it is currently used in the insertable B-layer (IBL) [**?**]. It has a pixel size of $50 \times 250\ \mu m^2$, an active area of $\approx 20 \times 20\ mm^2$ and a bulk thickness of 300 μm. The FEI4 APIX was mounted directly in front of the last telescope plane.

By combining the FEI4 APIX with the MIMOSA26 one can combine the benefits of both worlds by having a very short integration time and also a very precise spacial resolution, as shown in figure 6.2b. The MIMOSA26 sensor has a pixel size of only $18.5 \times 18.5\ \mu m^2$ which results in a spacial resolution for a setup of six planes of $\approx 2\ \mu m$, but it has a poor time



(a) *(b)*

Figure 6.2 (a) Rolling shutter readout of the MIMOSA26 sensor. [R1,R2,R3] indicate the readout cycles. Hits are indicated with [A,B,C]. [T] trigger time.
(b) Pixel size versus the integration time of the specific readout system. [A] Performance of the telescope with six MIMOSA26 planes. Integration time $\approx 2 \times 114.5\,\mu$s. *Spacial resolution* $\approx 2\ \mu$m. *[B] ATLAS FEI4 APIX sensor. Integration time* 25 *ns. Spacial resolution* $\approx 14 \times 72\ \mu m^2$. *[C] combination of both system into one setup.*

resolution of $2 \times 114.5\,\mu$s. On the other hand the ATLAS FEI4 APIX sensor can be readout with 25 *ns* time resolution but has the significant larger pixel size. From the pixel size the spacial resolution can be calculated to be $\approx 14 \times 72\ \mu m^2$. During the test beam these sensors are combined to use the strength of both. The tracking is done with six MIMOSA26 planes. The time stamping of the individual tracks is done with an ATLAS FEI4 APIX detector.

6.1.4 Device Under Test (DUT)

The DUT for this test beam is shown in figure 6.3. It consists of a barrel module hybrid which is equipped with three ABC130 readout chips. The readout chips at the position 5 and 7 are wire bonded to two ATLAS12 mini sensors. The mini sensors have an overall size of 1×1 cm^2 and a pitch size of $74.5\,\mu$m. The hybrid was glued on a Printed Circuit Board (PCB) with custom designed conductive tracks for data, low voltage and high voltage. The whole device was placed in a light-tight box. The box was mounted on an x-y stage which allowed the DUT to move between the runs. The device was readout with an HSIO board which processed and packed the data to be transferred to a Windows 7 PC running ITSDAQ. The ITSDAQ software is used to handle the communication between the PC and the device. As result of this work the software can directly interact with the EUDAQ framework which is

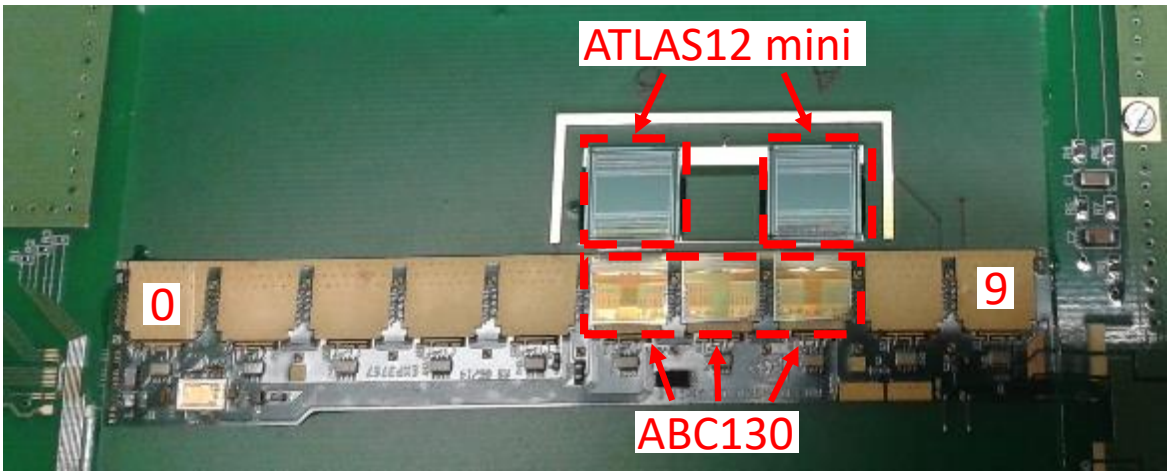


Figure 6.3 DUT. Barrel module hybrid with three ABC130 readout chips wire bonded to two ATLAS12s. Only the readout chips at position 5 and 7 are connected to the ATLAS12. The numbering of the ABC130 slots on the hybrid starts at the left hand side with position 0.

used for reading out the telescopes (see section 8.5). For the test beam two identical devices were tested. They are referred to as "Device 1" and "Device 2". An identical device was also investigated at an x-ray test beam in [**?**].

6.2 Beam Test Analysis

The evaluation of the test beam data is accomplished in two steps. In the first step there is the reconstruction, which transforms the raw data from pixel/strip hits to tracks and on the other side the analysis, which takes tracks and raw hits as input. The reconstruction is mainly done with the software frame work EUTelescope[**?** **?**]. It was developed as part of the EUDET project and is now the standard tool for the track reconstruction of pixel telescope data. EUTelescope uses for the track reconstruction the "General Broken Line Fitter" [**?**]. The output from EUTelescope is a collection of estimated hit positions and the raw hit information from the DUT. With this as the input the analysis can determine the efficiency of the DUT. This can either be done for the entire DUT or just for sub-regions like individual strips.

The reconstruction (as described in [**?** **?** **?**]) is the process of calculating the particle tracks from the detector hit data to obtain their extrapolated hit position on the DUT. The reconstruction was performed with the EUTelescope framework [**?** **?**] which was developed as part of EUDET project. It is built on top of the ILCSoft framework [**?**]. It is designed to calculate particle interaction for a possible future International Linear Collider (ILC) [**?**] experiment, therefore it is well suited for the needs of the high energy community by

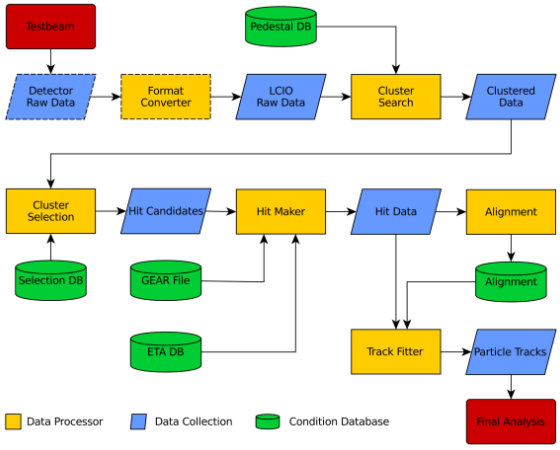


*Figure 6.4 Reconstruction Chain [**?**]*

providing a collection of high level classes to handle detector hits and particle trajectories. It provides the possibility to store these objects in a persistent file format called LCIO (Linear Collider Input Output) [**?**]. The individual tasks of EUTelescope are subdivided into individual processors. The entire reconstruction chain is shown in figure 6.4. For the reconstruction of the particle tracks the physical position of the individual detector components is needed. This information is stored in a so called GEAR [**?**] (GEometry API for Reconstruction) file. It contains all geometrical information about the individual detector components like position, rotation, pixel size, and material budget. It is used to convert the hit information from the local coordinate system of the individual detector plane to a common coordinate system for the entire telescope setup.

6.2.1 Components of EUTelescope

It starts with the conversion from detector raw data collected at test beam. Since all test beam setups are always very experimental it is very common to store the hardware buffer readouts without any conversion to, for example, pixel hits. This gives the groups the freedom to change the conversion after data taking. The downside to this approach is that the data is not stored in a standardized fashion. This data is transformed by the converter to a standardized pixel hit information format within EUTelescope. After the converter the data is stored as zero suppressed pixel information. For this test beam setup raw data of the DUT are stored in the same data file as the telescope data. The conversion of the DUT happens simultaneously with the conversion of the telescope data. Inside EUTelescope, data from the strip sensor is stored as pixel hits. For the further processing of data EUTelescope does not differentiate between telescope data and DUT data.

In the next step the pixel information is used to find adjacent active pixels and combine them into clusters.

In the following step all clusters containing noisy pixels are removed. A pixel is marked as noisy when it is active for a certain fraction of the events, the fraction is chosen by the user.

The hit maker transfers the cluster positions, measured in pixels to hits, measured in millimeters, within the local coordinate system of the individual detector. It uses the detector geometry stored in a so-called GEAR file.

To convert the data from the local to the global coordinate system the GEAR file needs to have the precise position information of the individual detector. Usually the first version of the GEAR file is created by measuring the telescope setup by hand. With this the precision is limited to $\approx 1\ mm$. To obtain a more precise gear file the alignment step is performed. This processor uses the hit information to produce an optimized GEAR file which has a precision of a few μm. This process can be done iteratively but usually does not improve much after two iterations.

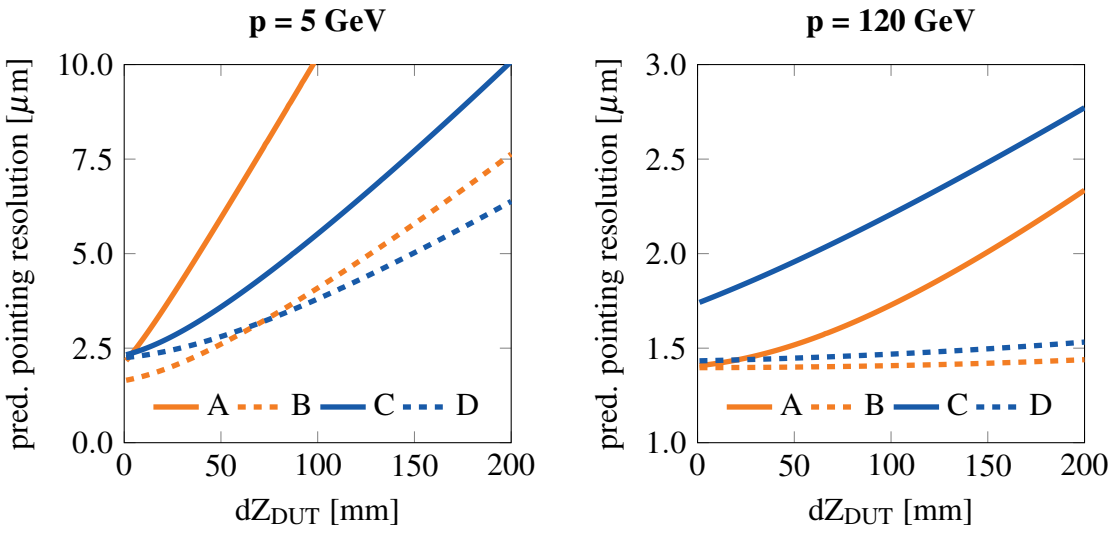
The last step of the reconstruction is the track fitter. It takes the hit data and the GEAR file to recreate the particle tracks. From this, the hit position on the DUT can be extrapolated. Within EUTelescope a variety of track fitters are possible but in this work only the GBL fitter was used. The GBL fitter is global least square track fitting algorithm. It was developed for the track reconstruction in large scale high energy physics experiments. It takes the multiple scattering into account by fitting the tracks not with straight lines but with broken lines. For this the material is modeled with thin scattering planes. The fit allows the particle trajectory to be broken at the intersection points of the planes with the trajectory. With this the GBL fitter is able to achieve a very high precision for the Track reconstruction.

6.2.2 Residual and Pointing Resolution

The residual is defined as the distance between extrapolated hit position and measured hit position. There are two different kind of residuals: the biased residual and the unbiased residual. The difference between them is that for the biased residual the measured hit position for the DUT takes part in the track fitter while for the unbiased residual the DUT hits are completely ignored for the track fit. In this work only unbiased residuals are investigated.

The residual consists of the following components: Intrinsic DUT resolution, telescope resolution and the effect of multiple scattering.

$$\sigma^2_{meas} = \sigma^2_{DUT} + \sigma^2_{Tel} + \sigma^2_{MS} \tag{6.1}$$



(a) Low energy beam. *(b) High energetic beam.*

Figure 6.5 Predicted pointing resolution for different telescope setups at different beam energies. (a) 5 GeV *low energy beam. (b)* 120 GeV *high energy beam.*
Narrow setup with $dz = 20$ mm*: [A]* $\varepsilon_{DUT} = 0.03$*, [B]* $\varepsilon_{DUT} = 0.001$*,*
Wide setup with $dz = 150$ mm*: [C]* $\varepsilon_{DUT} = 0.03$*, [D]* $\varepsilon_{DUT} = 0.001$*. [**?**]*

The spacial resolution of the DUT is determined by the pitch size, cluster size and the buffer depth of the digitization step especially whether or not the device has binary or analog readout. For a binary system with cluster size one the spacial resolution can be determined to be:

$$\sigma^2_{DUT} = \frac{1}{d}\int_{-d/2}^{d/2} x^2\,dx = \frac{d^2}{12} \tag{6.2}$$

With d being the pitch size. The telescope resolution and the influence from multiple scattering can be combined to the pointing resolution.

$$\sigma^2_p = \sigma^2_{Tel} + \sigma^2_{MS} = \sigma^2_{meas} - \sigma^2_{DUT} \tag{6.3}$$

The individual contributions to the pointing resolution are described in [**?**]. To obtain an optimal pointing resolution it is important to choose the telescope geometry according to the material budget and beam energy used at the test beam facility.

Figure 6.5 shows the pointing resolution for different material budgets and different beam energies as a function of the telescope setup. Investigated are two different telescope setups at two different test beam facilities: a wide setup (blue), a narrow setup (orange), a low

energetic beam with 5 GeV and a high energetic beam with 120 GeV. For these setups a collection of different material budgets ε_{DUT} are inspected.

$$\varepsilon_{DUT} = \frac{\Delta z_{DUT}}{X_0(DUT)} \tag{6.4}$$

Where Δz_{DUT} is the material thickness in z direction and $X_0(DUT)$ is the radiation length of the material the DUT is made of. dz_{DUT} is the symmetric distance between the telescope planes closest to the DUT and the DUT. It is recommended to choose this distance as small as possible. For a low energy beam, understanding the multiple scattering is key to obtain good pointing resolution. For more material budget it can be beneficial to use a wider spacing on the telescope plains. For a high energy beam the pointing resolution is less dependent on the telescope setup and the material budget.

The DUT consisted of 300 μm of silicon which corresponds to a relative radiation length of $\varepsilon > \frac{300\ \mu m}{93.7 mm} = 0.003$. The distance between telescope planes and DUT was limited by the size of the box for the DUT, it was roughly $dz_{dut} \approx 6\ cm$. With these two parameters a telescope geometry of $dz = 150\ mm$ was used.

6.2.3 Extracting of the Fiducial Area

Since the actual strip sensor area of $1 \times\ 1\ cm^2$ is smaller than the MIMOSA26 planes with $1 \times\ 2\ cm^2$, it is crucial to define a fiducial area. Only tracks that are inside the fiducial area are accepted and tracks outside this area get rejected. The fiducial area can be extracted by investigating the hitmap for a low ABC130 threshold of 150 mV. The hitmap is created by combining the extrapolated track position and the hit position on the DUT. In this analysis the collection of extrapolated track positions has only one track per event. The DUT can have multiple hits per event. In the next step the distance between the hits and the tracks is calculated. Since a strip detector has only one coordinate the distance is only calculated along this axis. Figure 6.6 shows an exemplary hitmap. The hitmap shows that inside the fiducial region the efficiency is rather constant at one threshold. For the efficiency it is more important to have a precise value than to have more statistic, therefore the edge regions are cut out completely. The fiducial area is shown in figure 6.6.

6.2.4 Efficiency Calculation

After the fiducial area is defined the hit efficiency of the sensor can be calculated. In the first step of the reconstruction EUTelescope uses only the MIMOSA26 hits to create tracks. As discussed in section 6.1.2 the MIMOSA26 sensor has a large integration time therefore many

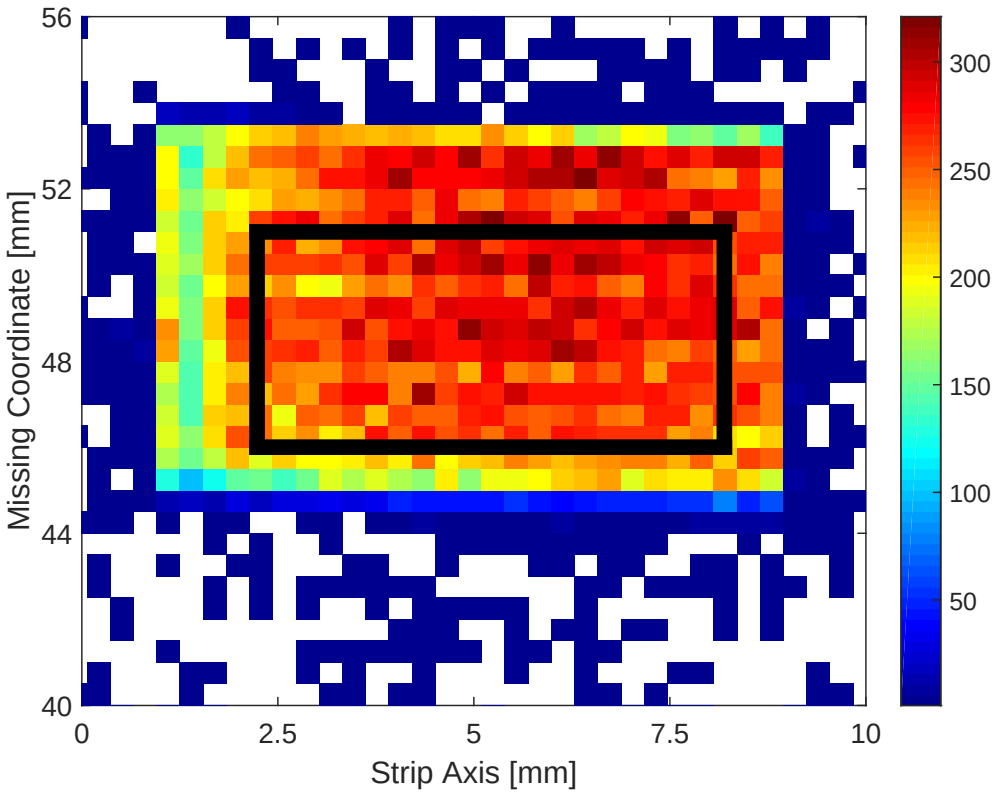


Figure 6.6 Hitmap with fiducial area. Histogram of track position extrapolated to the DUT, for tracks that matched to a hit on the DUT. The black rectangle confines the fiducial area.

of the tracks that are reconstructed are not connected to the trigger. To remove these tracks the information of the FEI4 APIX is used. After this only hits with the right timing remain. In the next step all tracks that are outside the fiducial area are removed. The collection obtained by these two cuts contains only hits that hit the DUT and have correct timing. This collection is called "true hits". The next step is to obtain the hits with corresponding DUT hits. For this, the distance between the "true hit" and the DUT hit is calculated. If the distance is smaller than the cut off parameter of $\pm 5 \times$ pitch size the hit is used as a "true DUT hit". The "true DUT hits" use the position coordinate of the "true hits" this allows one to investigate also in-strip effects. In the last step the two hit positions are filled in a histogram with equal binning and the histograms are divided by each other to obtain the efficiency. For the overall efficiency the histograms have only one bin.

6.3 Data Selection

This section describes the steps necessary to evaluate the quality of the data and the precision of the reconstruction. The first step is always to evaluate the device without beam and only with a random trigger. From this, not only the pedestal and the input noise can be determined but also the variation of these values for different channels and chips. Also, it gives some insights to the operational stability of the setup. This is necessary since the test beam environment is vastly different from the clean and controlled laboratory environment.

The first test with the particle beam is to create correlation plots. These plots are central for the data quality monitoring during the test beam. They show not just the correlation between the DUT and telescope but also show any sort of misalignment. These plots are part of the online and offline monitor of the EUDAQ framework.

After the reconstruction the first plot to investigate is the residual plot. It gives information about the precision with which the particle tracks can be reconstructed. It is a key quantity to describe the quality of the test beam campaign.

In the next step the residual can also be used to check for errors which can occur during test beam. For example, the DUT can lose synchronization to the TLU (see section 5.2). By plotting the residual versus time the loss of synchronization can easily be detected. This plot is, in a simplified version, part of the offline monitor of EUDAQ and is therefore already available during a test beam.

A special feature of strip sensors is that, compared to pixel sensors, they only measure one coordinate of a particle hit. Since most of the reconstruction algorithms used in EUTelescope are designed to handle pixel data it can happen that the alignment step misplaces the DUT.

Rotations are particular troubling. With the residual versus missing coordinate plot these rotations can be spotted.

6.3.1 DUT Auto Trigger Runs

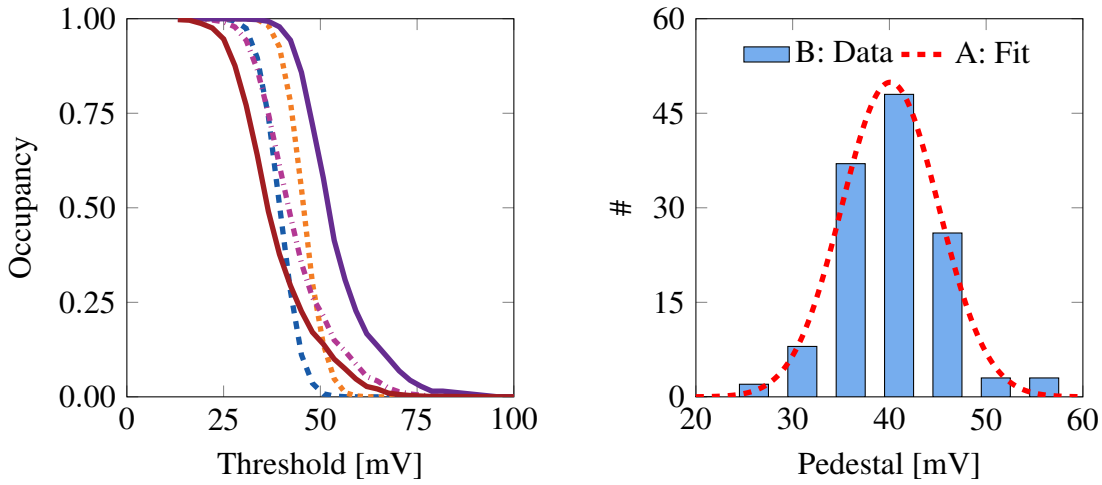
To extract the pedestal for each of the devices, auto trigger runs were performed. The auto trigger was created by the internal clock of the TLU. The auto trigger runs are done with the exact same setup as the actual test beam, therefore it is certain that the pedestal position did not change afterwards. Each S-Curve consists of 25 individual runs each with a sample size of 10^4 *trigger*. Figure 6.7a shows the S-Curves obtained from these threshold scans measured at 25 thresholds. The underlying distribution is mostly gaussian noise therefore the S-Curves can be fitted with an integrated gauss function. It is noticed that some of the S-Curves show a pure gaussian form while other show a longer tail. The pure gaussian lines correspond to unbonded channels while the bonded channels show long tails. From the fit, the pedestal and the input noise can be obtained. The pedestal corresponds to the mean value of the gauss function while the input noise corresponds to the standard deviation. Figure 6.7b shows a histogram of the pedestal value of only bonded channels. The averaged pedestal position at 40.1 ± 0.2 mV and the standard deviation of 5.0 ± 0.4 mV are obtained by the gauss fit. Table 6.1 shows the pedestal and input noise values extracted from the S-Curve fit of the noise runs. It shows that the different chips behave rather homogeneously. The input noise as a function of the channels is shown in figure 6.8a. As expected the input noise decreases with higher bias voltage and the input noise is higher for bonded sections. The regions are marked with the different chip number. Chip 6 was completely unbonded. In figure 6.8b the pedestal as a function of the channel number is shown. It is clearly visible that the pedestal does not depend on the bias voltage. It shows that the pedestal is lower for bonded chips.

6.3.2 Correlations

The first plot to check for data quality of the test beam data is the correlation plot as shown in figure 6.9a between two different telescope planes (including DUT). For this, each hit from one plane is combined with all hits from the other plane. This results in a two dimensional histogram for each coordinate. Hits originating from the same particle form a solid line in the histogram. Hits corresponding to different particles form a homogeneous distributed cloud of hits in the histogram. Noise pixel/strips form solid horizontal or vertical lines. The concrete example shows the correlation between telescope plane 2 and the DUT. The Correlation plot provides information about synchronization issues, misalignment and orientation of the

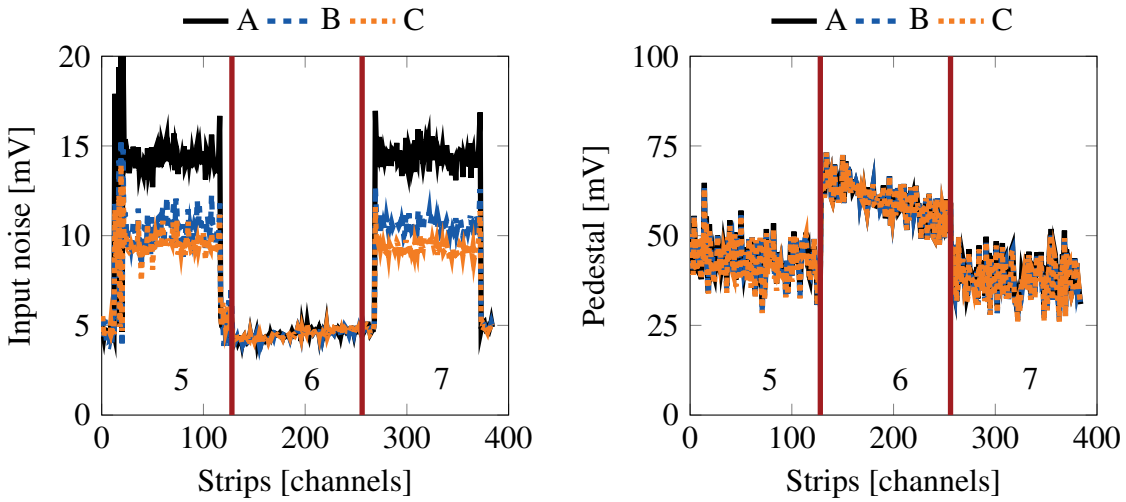
Table 6.1 Pedestal and input noises for various bias voltages obtained from auto trigger runs

ASIC	Bias Voltage [V]	Pedestal [mV]	Input Noise [mV]
Device 1 POS 5	150	43.2 ± 5.0	14.3 ± 1.9
Device 1 POS 5	250	43.0 ± 5.2	10.3 ± 1.5
Device 1 POS 5	350	41.9 ± 5.6	9.5 ± 1.1
Device 1 POS 5	400	41.4 ± 5.1	9.3 ± 1.0
Device 1 POS 7	150	39.1 ± 5.3	14.4 ± 0.6
Device 1 POS 7	250	37.8 ± 5.4	10.6 ± 0.3
Device 1 POS 7	350	37.6 ± 5.4	9.3 ± 0.3
Device 1 POS 7	400	37.4 ± 5.5	9.3 ± 0.4
Device 2 POS 5	150	46.2 ± 8.1	13.4 ± 0.7
Device 2 POS 5	250	44.7 ± 8.2	10.5 ± 0.5
Device 2 POS 5	350	44.3 ± 8.1	9.2 ± 0.4
Device 2 POS 5	400	44.1 ± 8.2	9.0 ± 0.4
Device 2 POS 7	150	58.2 ± 3.1	12.6 ± 0.4
Device 2 POS 7	250	57.2 ± 3.1	10.1 ± 0.4
Device 2 POS 7	350	56.7 ± 3.1	8.9 ± 0.3
Device 2 POS 7	400	56.6 ± 3.1	8.8 ± 0.3



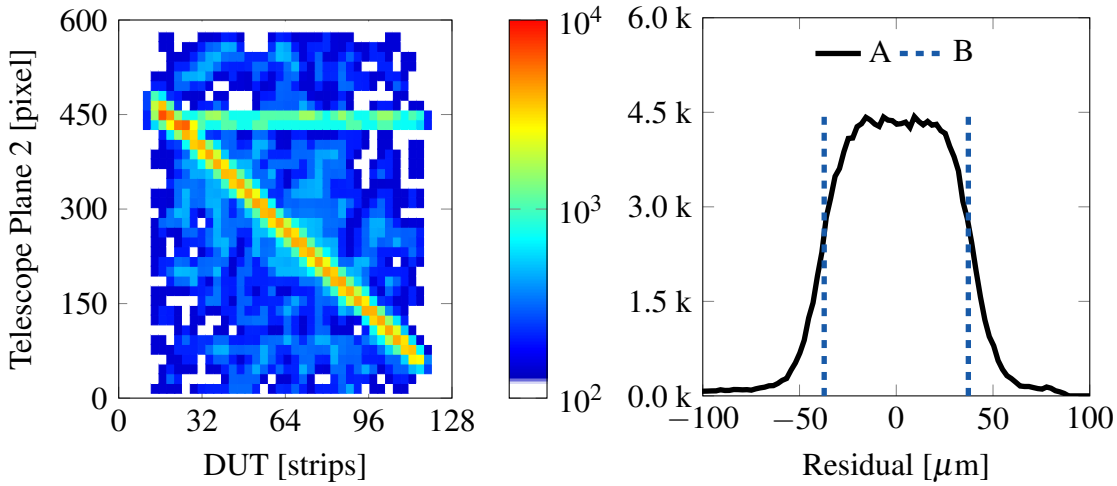
(a) *(b)*

Figure 6.7 (a) Example S-Curves obtained from an auto trigger run for different readout channels of a ABC130.
(b) Histogram of pedestal positions for one ABC130 readout chip [B]. [A] Gauss fit of [B].



(a) Device 1: Input noise *(b) Device 1: Pedestal*

Figure 6.8 [A] HV = 150 V, [B] HV = 250 V, [C] HV = 350 V.
(a) Input noise as function of the channel number for device 1. The areas with significantly lower noise correspond to the unbonded channels of the chip. The regions are marked with the chip number. Chip 6 was completely unbonded. It is noticeable that the input noise decreases with increasing bias voltage.
(b) Pedestal as a function of the channel number for the same device. It is noticeable that the pedestal stays constant for different bias voltages. Like for the input noise, the unbonded chip is easy to spot by its significantly different pedestal value.



(a) *(b)*

Figure 6.9 (a) Shown is the correlation histogram between neighboring planes of the telescope setup; plane 2 of the telescope and the DUT.
(b) Residual plot for the DUT [A]. [B] indicates the pitch size (74.5 μm).

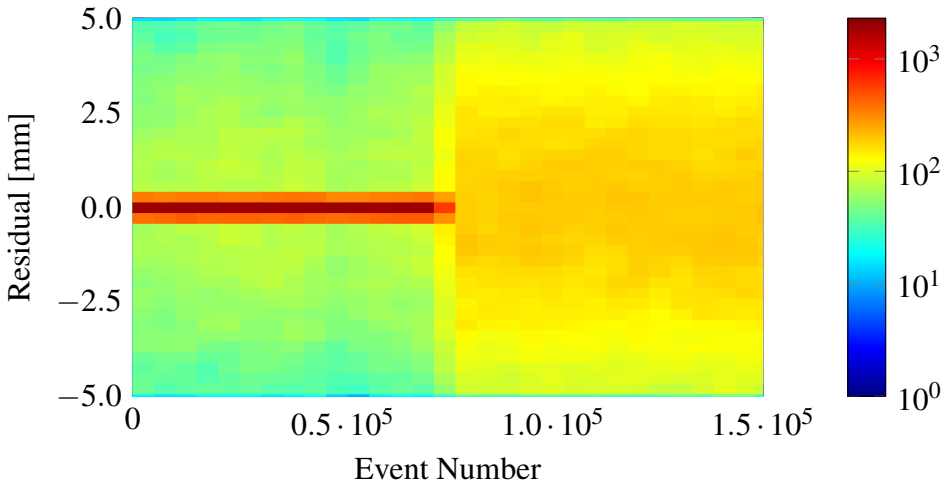


Figure 6.10 Example Residual versus Time (RvT) plot.

detector planes. Correlation plots are key component of every data quality monitoring and they are produced at almost any stage of the reconstruction.

6.3.3 Residual

The residual is used to check the quality of the reconstruction (see figure 6.9b). Since the pitch size of the DUT is (at 74.5 μm) large compared to the pointing resolution of the telescope the residual plot has an almost rectangular shape. It shows a plateau region at the center and steep edges. The slope of the edges is determined by the pointing resolution of the telescope. The standard deviation measured from a residual plot is the combination of pointing resolution and spacial resolution of the detector (see eq. 6.1). The expected standard deviation for a given DUT with binary readout can be calculated with eq. 6.2. For the given DUT a standard deviation of $\sigma_{DUT\ exp} = \frac{74.5\ \mu\text{m}}{\sqrt{12}} = 21.5\ \mu$m is expected. The observed width of the residual is $\sigma_{mes} = 24.0\ \mu$m. This results in a pointing resolution of:

$$\sigma_p = \sigma^2_{mes} - \sigma^2_{DUT\ exp} = 11\ \mu\text{m} \tag{6.5}$$

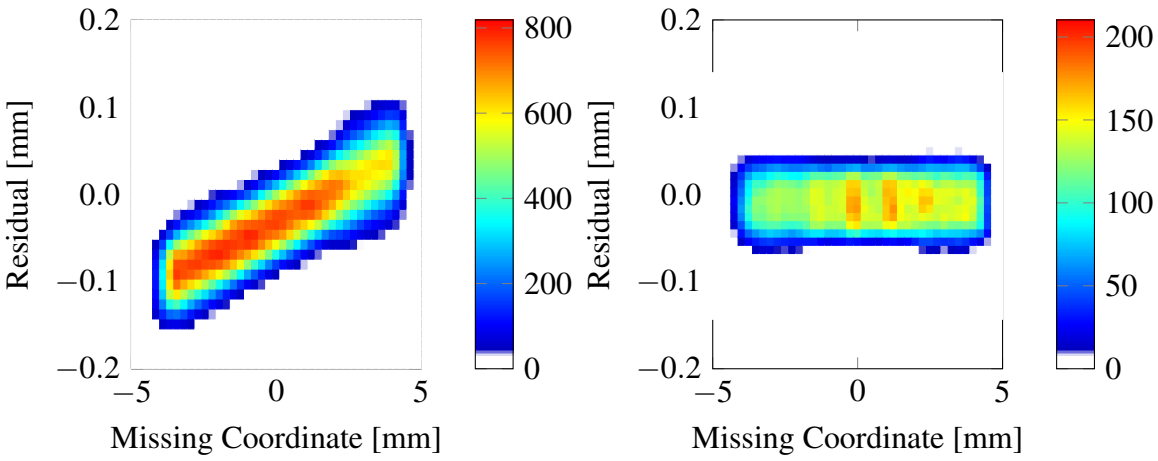
Thus the pointing resolution is higher than what was expected from the considerations done in section 6.2.2. It is still a matter of discussion how to improve the pointing resolution further. For the analyses performed in this work a pointing resolution of 11 μm is sufficient.

6.3.4 Residual versus Time (RvT)

The residual can also be plotted versus time (or event number). This is used to check for any sort of de-synchronization. It is a common problem in test beam setups using different DAQs that devices lose synchronization after some time. With a correlation plot this is difficult to observe, especially if the de-synchronization happened after a longer period of time when a large fraction of the total data is actually synchronized. With a RvT plot this is easy to spot. Since the event number is roughly proportional to the time, it is sufficient to use the event number instead of the time for the RvT plot. During data taking it is common to create the RvT plots from the residual distance between hits on neighboring plains, similarly to the correlation plots. This is why the RvT plots are also called "correlation over time/event number" plots. Compared to the correlation plot the RvT plot requires more information about the devices involved. It needs to know at least the pixel sizes as well as the relative orientation of the devices involved. Figure 6.10 shows an RvT plot for a run which loses synchronization. It starts on the left hand side being synchronized. In this region it shows the typical concentration of entries at a certain residual position. These entries belong to hit combinations from the same particle track. These entries are surrounded by a homogeneous spread of entries which belong to a hit combination of uncorrelated hits; that means hits from two different tracks or hit combinations from which at least one hit comes from a noisy pixel. After about $0.7 \cdot 10^5$ events the device loses synchronization. The entries are now scattered over a wide area. For this run the device does not regain synchronization. Since the device both counts trigger too many times and misses triggers completely it is possible for the device to regain synchronization. This plot is implemented in the offline monitor and the online monitor of EUDAQ. For further analysis, only runs that are synchronized from begin to end are used.

6.3.5 Residual over Missing Coordinate

A specialty of strip sensors is that they have a "missing coordinate", which means along the strip axis they do not provide position information. For the real experiment this is compensated by using two strip sensors with a certain angle in between (stereo angle). At test beam this is not a good solution because it doubles the complexity of the setup. It also doubles the material budget which leads to higher multiple scattering. Instead means have to be found to handle the missing coordinate. The missing coordinate is problematic, especially for the alignment step. At this stage it can happen that the algorithm misplaces the strip sensor plane. To check for any sort of misplacement it is important to check the "Residual over Missing Coordinate" plot. Figure 6.11a shows this plot for a poorly aligned device,



(a) Misaligned setup description *(b) Well aligned setup description*

Figure 6.11 Example residual vs missing coordinate plot.

which results in a steady slope of the center of the residual position. It is clearly visible that the residual has a slope as a function of the position along the strip. This indicates that the alignment algorithm was not able to determine the rotation of the sensor correctly. Figure 6.11b shows the plot for a correctly aligned sensor plane. The residual has the same center position for all positions along the strip. For the further analysis it was made sure that the alignment was correct for all runs.

Chapter 7

Results and Discussion of the Test Beam Campaign

In this thesis the new readout chip ABC130 for the ATLAS silicon strip tracker has been investigated. The key question was to determine the gain of its preamplifier. For this, the readout chip ABC130 was evaluated at the DESY test beam facility. After reconstruction of the tracks and the further data analysis, the so called S-Curves have been obtained. The S-Curves are defined as the detection efficiency as function of the threshold. This chapter investigates the S-Curves by dividing them into two regions of interest. The first region is the plateau region in which the efficiency is almost independent of the threshold. The second region is the slope region. This region originates from the charge deposition spectrum of the particle passing through the detector. In this chapter it will be shown how to obtain the gain of the preamplifier from this slope region.

7.1 ALiBaVa Measurements

As a control measurement, the charge distribution of the silicon sensor was measured with an analog readout system (ALiBaVa system [**?**]). The setup for these measurements was virtually identical to the beta source setup, described in section 5.3, with the main difference that this time an analog readout was used. From this measurement two quantities can be extracted: the charge distribution of the entire hit and the charge distribution of the leading strip. The charge distribution is fitted with a Landau-Gauss function to extract the Most Probable value (MP-Value). From literature (see section 4.2.2) it is known that the charge distribution for an entire cluster is Landau shaped and has an MP-Value around of 3.7 fC for 300 μm Si. The measurements with the ALiBaVa system were able to confirm this value.

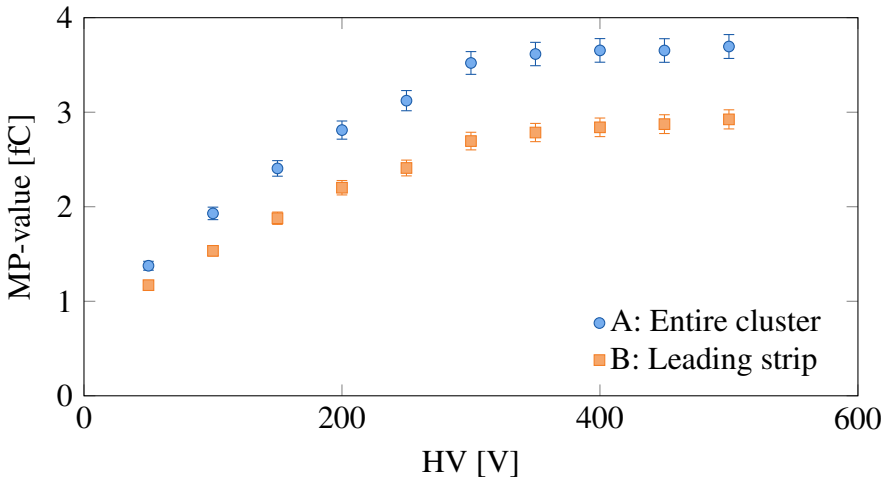


Figure 7.1 MP-Values of the charge distributions measured with the ALiBaVa system for different bias voltages. For these measurements an ATLAS12 sensor was used.[A] shows the MP-Values for the entire cluster. At full depletion it reaches a plateau at 3.6 ± 0.1 fC. *[B] shows the MP-Values only for the leading strip. It reaches the plateau around 22 % lower at* 2.8 ± 0.1 fC. *[**?**]*

Since a binary readout system cannot sum the charge inside a cluster it is crucial to extract the MP-Value of the leading strip. This measurement can also be done with the ALiBaVa system. The comparison is shown in figure 7.1. It shows the MP-Value as a function of the bias voltage for an ATLAS12 sensor. When examining the entire cluster, the plateau is reached at the expected 3.6 ± 0.1 fC, but when only investigating the leading, strip the plateau is reached around 22 % lower at 2.8 ± 0.1fC.

7.2 The Plateau Region

This is the region of the S-Curve (figure 7.2) for low thresholds, where the efficiency is almost constant. The end of the plateau, it is defined by the drastic change of the derivative of the S-Curves. It is an interesting region because it is most likely that the actual detector will be operated in this region during the HL-LHC period. From this region, the nominal efficiency of the detector can be obtained. Figure 7.3 shows a comparison between the efficiencies obtained in the plateau region for different thresholds, compared to the efficiency specification of the current ATLAS inner Tracker. The individual data sets show large outliers. The outliers can not be explained by the statistical uncertainty. The cause of the outliers is still a work in progress and might only be understood by taking additional measurements.

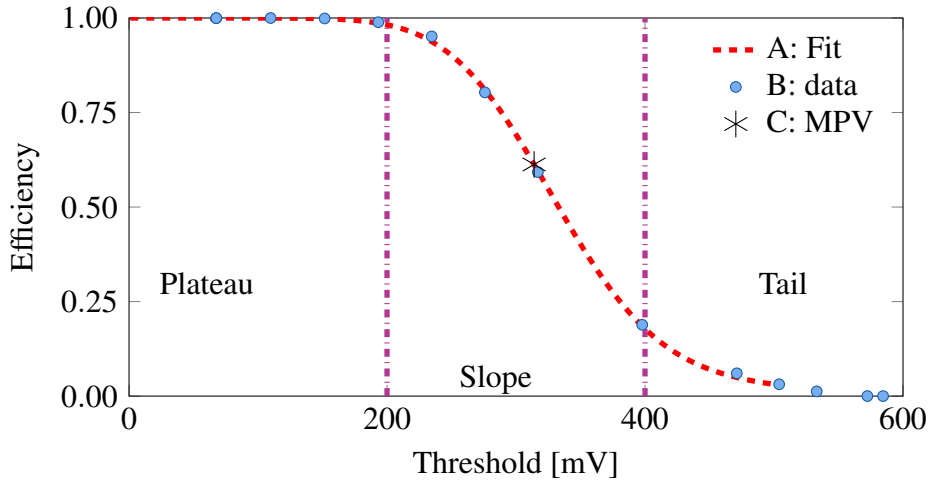
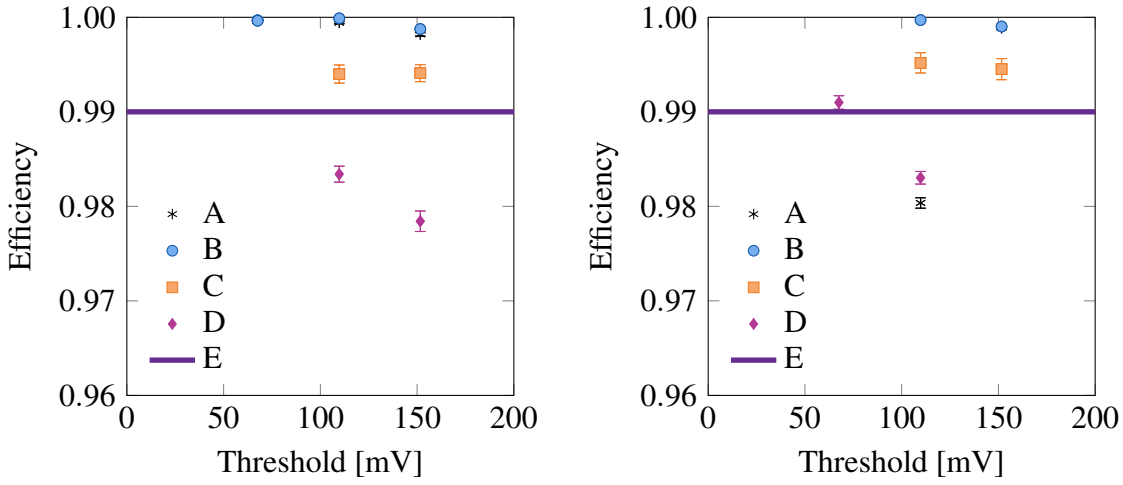


Figure 7.2 S-Curve plot: It shows the efficiency as a function of the threshold for test beam data [B]. The error on the efficiency is less than 0.3 *% and is therefore smaller than the marker for the data points. The S-Curve is split up into three regions, plateau, slope and tail. From the plateau region, the nominal efficiency can be estimated. From the slope region, the gain of the preamplifier can be calculated. The red line [A] shows a fit with an Integrated Landau-Function (see section 7.3.1). It is in good agreement with the data. From this function, the MP-Value is extracted [C].*



(a) Bias Voltage 350 V

(b) Bias Voltage 400 V

*Figure 7.3 Efficiency in the plateau region as function of the threshold. [A] Device 1 position 5, [B] Device 1 position 7, [C] Device 2 position 5, [D] Device 2 position 7. [E] efficiency specification from the current ATLAS tracker [**?**]. The spread between the individual ASICs is larger than expected from the statistical uncertainties.*

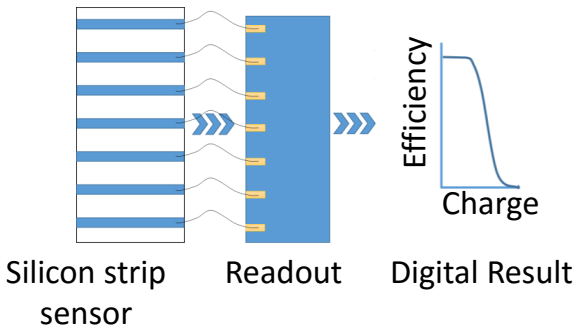


Figure 7.4 Schematic overview of the readout chain. Starting with the silicon sensor at the left. The individual strips are connected with wirebonds to the readout in the middle. The readout can either be performed by the ABC130 or the ALiBaVa.

7.3 The Slope Region

The slope region (as shown in figure 7.2) originates from the amount of charge deposited in the silicon. The setup is sketched in figure 7.4. On one side there is the silicon strip sensor which produces a certain amount of charge, on the other side there is the output of the readout chain either as charge spectrum or as S-Curve depending on using an analog (ALiBaVa) or a digital (ABC130) readout. From both readouts the efficiency as a function of a threshold can be extracted. For the ALiBaVa system the thresholds are given in input charge while for the ABC130 the thresholds are given in voltage, after the preamplifier. Since the input for both readouts is the same silicon strip sensor, the differences in the output are arising mostly from the readout chip and as such especially from the gain of the preamplifier. The important question of this work is how to quantify the S-Curves. In this work two approaches are investigated.

7.3.1 Method I: Fitting the S-Curves with an Integrated Landau Gauss Function

This method is based on the theory of charge deposition in semiconductors, as described in section 4.2. This theory describes the charge distribution as function of the material and the thickness of the material. The total charge deposition is independent of the strip sensor layout, and therefore it is independent of the pitch size, the inter strip resistance and inter strip capacitance. This approach is strongly theoretically motivated. Its benefit is that it is very generic and makes only a few assumptions about the actual sensor layout. The assumptions are that the charge is only split up into individual strip and the leakage current is so low that

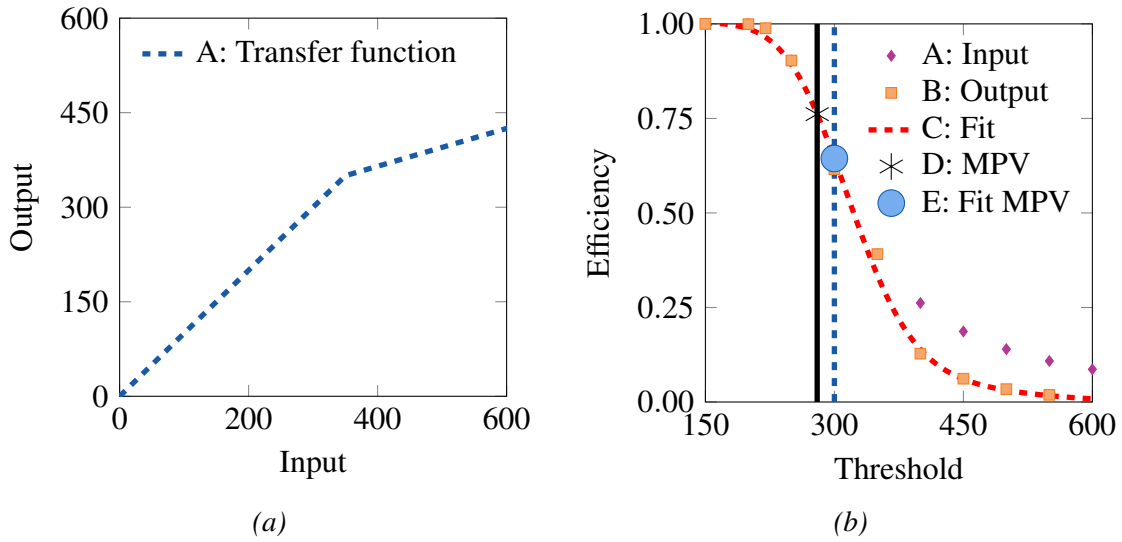
it can be ignored. Moreover, the summation of all strips in one cluster gives the total charge deposited by the particle.

As described in section 3.5.3, a charge summation over the entire cluster is not possible with a binary system, therefore it is not possible to extract the full charge but only the charge of the leading strip. The charge of individual strip is dependent on the actual strip layout especially the inter strip resistivity, inter strip capacitance and pitch size. The amount of charge sharing can either be calculated from the electrical connection between the individual strips or it can be directly measured with an analog readout system. In this work the response from the binary readout is compared to ALiBaVa measurements.

Another difficulty of modeling the S-Curves is that the underlying Landau-Gauss function is not symmetric, therefore the MP-Value is not exactly at 50 % efficiency but is shifted to lower thresholds. As such it is reasonable to extract the MP-Value with an Integrated Landau-Function (see section 4.2). The feature of the Integrated Landau-Function is its sensitivity to changes in the slope. Values at the edges of the slope region (low and high) which have an impact on the MP-Value will be modeled well. This is interesting since the preamplifier has a certain non linearity, which can have an effect on the MP-Value obtained by the fit. The values obtained by the fit are shown in table 7.1. The actual MP-Values are rather similar at around 315 mV and with this the gain of the preamplifier of the ABC130 can be calculated, with equation 7.1, to be $> 97\ mV/fC$. It is interesting to note that the variance of the Landau distribution is small compared to the variance of the convoluted gauss distribution as shown in table 7.1.

$$gain = \frac{\text{MPV} - \text{pedestal}}{\text{charge}} \tag{7.1}$$

Figure 7.2 shows a threshold scan with Landau-Gauss fit. It is interesting that it does not show the typical long tail, as expected from a Landau-Gauss function. One reason for this could be a rapid degradation of the gain after the MP-Value. The most prominent distinction between the two functions is the long tail of the Landau function. By truncating the tails, due to a strong non linearity in the preamplifier, it is possible that the function appears to the fit algorithm more like a gauss function than like a landau function. This would shift the MP-Value to higher threshold value; this effect can be studied in figure 7.5. The left hand side (figure 7.5a) shows a transfer function of an example amplifier with saturation. It is a strongly simplified model in which the amplifier has its nominal gain below a threshold, and above the threshold the gain significantly degrades. In this example the saturation gain is 30 % of the nominal gain. Input/output is given in arbitrary units. Figure 7.5b shows the S-Curve obtained by this amplifier. The input S-Curve was created with a long tail and an MP-Value of 280. The output S-Curve shows almost no remains of the long tail of the input



(a) *(b)*

Figure 7.5 (a) The transfer function of an example amplifier with saturation. (b) The effect of an amplifier with saturation on the shape of the S-Curve. [A] Input S-Curve with a long landau tail and a MP-Value of 280 [D]. [B] output of this amplifier. [C] fit with an Integrated Landau-Function of [B]. [E] MP-Value obtained from the fit [C] (296.3 ± 0.6).

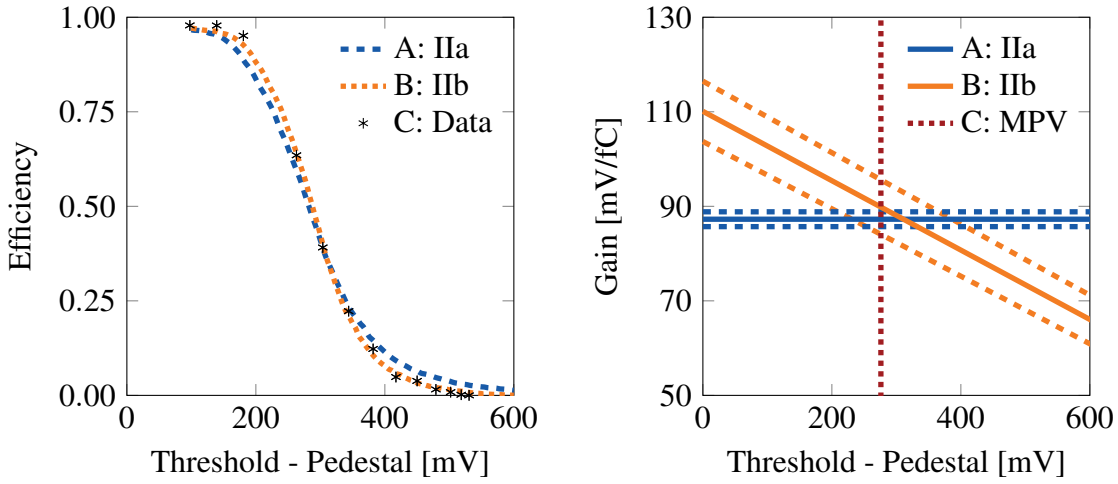
function. By fitting the output S-Curve with an Integrated Landau-Function an MP-Value of 296.3 ± 0.6 is obtained. It is noticeable that the MP-Value of the input function is at a significantly lower threshold than the MP-Value obtained by the fit. This example shows how the non linearity of an amplifier can change the interpretation of the resulting S-Curve.

7.3.2 Method II: comparing to ALiBaVa

The second method is a direct comparison of the ABC130 S-Curves with S-Curves obtained by ALiBaVa measurements. As described in section 7.1 a control measurement was done

Table 7.1 Results of the S-Curve fits obtained with method I

ASIC	MP-Value [mV]	σ_{Landau} [mV]	σ_{Gauss} [mV]	χ^2	gain [mV/fC]
Device 1 pos 7 350 V	312.0 ± 1.3	8.1 ± 0.5	58.7 ± 1.4	13.6	97.0 ± 3.5
Device 1 pos 5 350 V	315.1 ± 1.5	8.9 ± 0.6	63.8 ± 1.8	23.5	98.1 ± 3.5
Device 2 pos 5 350 V	310.2 ± 2.2	9.8 ± 0.9	52.5 ± 3.0	77.3	96.3 ± 3.5
Device 2 pos 7 350 V	321.4 ± 1.4	7.9 ± 0.5	59.5 ± 2.0	23.1	100.3 ± 3.6
Device 1 pos 7 400 V	312.2 ± 1.2	9.4 ± 0.5	55.1 ± 1.8	25.2	97.0 ± 3.5
Device 1 pos 5 400 V	320.8 ± 1.5	9.5 ± 0.6	55.5 ± 2.3	27.3	100.1 ± 3.6
Device 2 pos 5 400 V	312.9 ± 1.5	9.1 ± 0.6	55.8 ± 2.0	34.2	97.3 ± 3.5
Device 2 pos 7 400 V	321.6 ± 1.2	8.9 ± 0.5	54.4 ± 1.5	18.6	100.4 ± 3.6



(a) *(b)*

Figure 7.6 Gain calculation with method II.
(a) [C] Exemplary S-Curve obtained for device 2 at full depletion with a bias voltage of 350 V. [A] shows a fit to the ALiBaVa data with method II_a with one free parameter, the gain of the preamplifier. [B] shows a fit to the ALiBaVa data with method II_b with two free parameters, the gain and the degradation of the gain.
(b) Comparison between the gains obtained by methods II_a and II_b. [A] Gain obtained from method II_a (gain $= 87.3 \pm 1.6 \frac{\text{mV}}{\text{fC}}$). [B] corresponds to method II_b depicting a linear degradation of the gain. The gain at the MP-Value is $gain_{mpv} = 89.9 \pm 6.1 \frac{\text{mV}}{\text{fC}}$. The dashed lines around [A] and [B] show the 95% confidence level. [c] Indicates the MP-Value.

with the ALiBaVa system. From the control measurement the charge distribution for the individual strips can be obtained.

It is assumed that the input charge spectrum is identical for both readouts. It is also assumed that the ALiBaVa system does not introduce any bias to the system, particularly that the amplification is completely linear and that it does not introduce any more charge sharing. With these assumptions all variations between these two readouts are arising from the preamplifier of the ABC130.

A difference between the two readout systems is that the ALiBaVa system measures the actual charge spectrum, while with the ABC130 readout can only obtain S-Curves. The S-Curve is the integration of the charge distribution of the leading strip. To compare both data sets it is possible to either integrate the ALiBaVa charge distribution, or to derive the S-Curve obtained by the ABC130. Since the data points of the ALiBaVa system are very dense, the ALiBaVa charge spectrum gets integrated to obtain the S-Curve. The S-Curve of the ALiBaVa system is given in data pairs of charge and efficiency. For further processing it is necessary to have a continuous function. Since the ALiBaVa system gives very dense data points a spline interpolation is sufficient. In this text the spline function of the ALiBaVa data is referenced as $S_{ALiBaVa}(charge)$. Since, for the ABC130 the threshold is set after the preamplifier, it gives data pairs of threshold voltages and efficiencies.

The effect of the preamplifier can be modeled with a transfer function T:

$$T : \text{charge} -> \text{Voltage}. \tag{7.2}$$

Assuming the transfer function T is known, and the inverse function of T exists, then the efficiency for a certain threshold can be predicted with:

$$\mathrm{S_{ALiBaVa}}(\mathrm{T^{-1}}(\text{threshold})) = \text{Efficiency}. \tag{7.3}$$

This can be used to calculate the transfer function by combining the least square method with the variational method. For this the following functional is defined:

$$F\left[T^{-1}\right] = \sum_{i=0}^{N} \left(\mathrm{S_{ALiBaVa}}(\mathrm{T^{-1}}(\mathrm{threshold_i})) - \mathrm{Efficiency_i}\right)^2, \tag{7.4}$$

with the pairs $(\mathrm{threshold_i}, \mathrm{Efficiency_i})$ being the measurements obtained with ABC130. The function T^{-1} can be calculated as minimum of the functional equation 7.4. To simplify the calculation the T^{-1} is approximated by two models. II_a: a constant gain which does not depend on the threshold. II_b: a linear decreasing gain.

Table 7.2 Results of the S-Curve fits obtained with method II

	II_b		II_a	
ASIC	χ^2	gain [mV/fC]	χ^2	gain [mV/fC]
Device 1 Pos 7 HV = 350 V	40.2	90.8 ± 5.7	283.5	88.8 ± 2.1
Device 1 Pos 5 HV = 350 V	26.1	91.7 ± 4.6	146.4	89.4 ± 1.3
Device 2 Pos 5 HV = 350 V	93.5	91.1 ± 9.3	317.5	88.2 ± 1.9
Device 2 Pos 7 HV = 350 V	31.3	93.4 ± 5.7	285.3	89.4 ± 1.9
Device 1 pos 7 HV = 400 V	37.3	90.7 ± 5.4	260.3	88.6 ± 1.6
Device 1 pos 5 HV = 400 V	42.2	94.1 ± 7.3	244.2	90.9 ± 1.7
Device 2 pos 5 HV = 400 V	53.9	91.4 ± 6.9	281.8	88.5 ± 1.8
Device 2 pos 7 HV = 400 V	52.6	95.0 ± 7.6	430.6	90.8 ± 2.1

$$T^{-1}(Threshold) = \frac{Threshold}{gain} \tag{7.5}$$

$$T^{-1}(Threshold) = \frac{Threshold}{gain_0(1 - deg \times Threshold)} \tag{7.6}$$

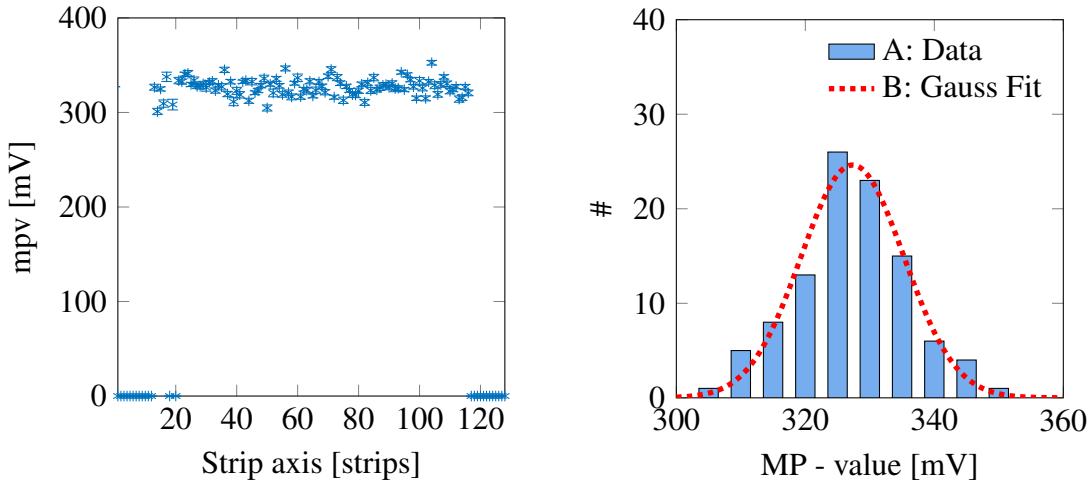
The result of the two different fit models can be studied in figure 7.6a. It shows an exemplary S-Curve obtained for device 2, at full depletion, with a bias voltage of 350 V. The x axis shows the threshold reduced by the average pedestal of 42 mV. It is clearly visible that the additional free parameter significantly improves the fit quality.

But this additional parameter comes at the cost of a higher uncertainty, as seen in figure 7.6b. It shows a comparison between the gains obtained by methods II_a and II_b. Method II_a shows a constant gain with respect to the threshold of $gain = 87.3 \pm 1.6 \frac{\text{mV}}{\text{fC}}$. Method II_b shows a linear degrading gain with $gain_0 = 110.1 \frac{\text{mV}}{\text{fC}}$ at 0fC input charge and a degradation of the gain $deg = 0.073 \frac{1}{\text{fC}}$. Since the fit is mostly sensitive to regions around the MP-Value the nominal value for this fit is the gain around the MP-Value. With $gain_{mpv} = 89.9 \pm 6.1 \ \frac{\text{mV}}{\text{fC}}$ being the gain obtained by method II_b it is compatible with the values obtained by method II_a. To quantify the quality of the fit, the reduced χ^2 equation 7.7 is used. The results are given in table 7.2.

$$\chi^2_{red} = \frac{1}{f} \sum \frac{(O_i - E_i)^2}{\sigma_i^2} \tag{7.7}$$

7.3.3 Strip-by-Strip Analysis

For the later use as part of a more complex detector system it is important to know how strongly the individual channels differ from each other. In this study the actual number of



(a) *(b)*

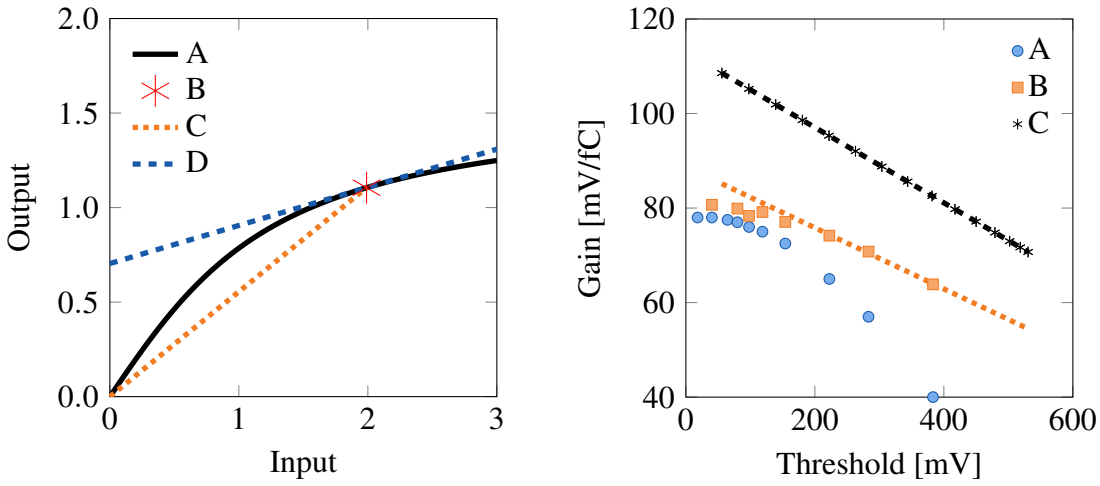
Figure 7.7 (a) The MP-Values at full depletion for the individual strip. The MP-Values are uniformly distributed and there is no systematic drift visible.
(b) Histogram of the MP-Values with a mean value of 327.3 ± 0.65 mV *and a standard deviation of* $\sigma = 8.0$ mV

the MP-Value is not as important as its uniformity, therefore it is reasonable to use only one model to fit the S-Curves. For the strip by strip analysis only the fit with the Integrated Landau-Function is used. As seen in figure 7.7a the MP-Value is rather homogeneous among all strips.

7.4 Conclusion of the Gain Estimations

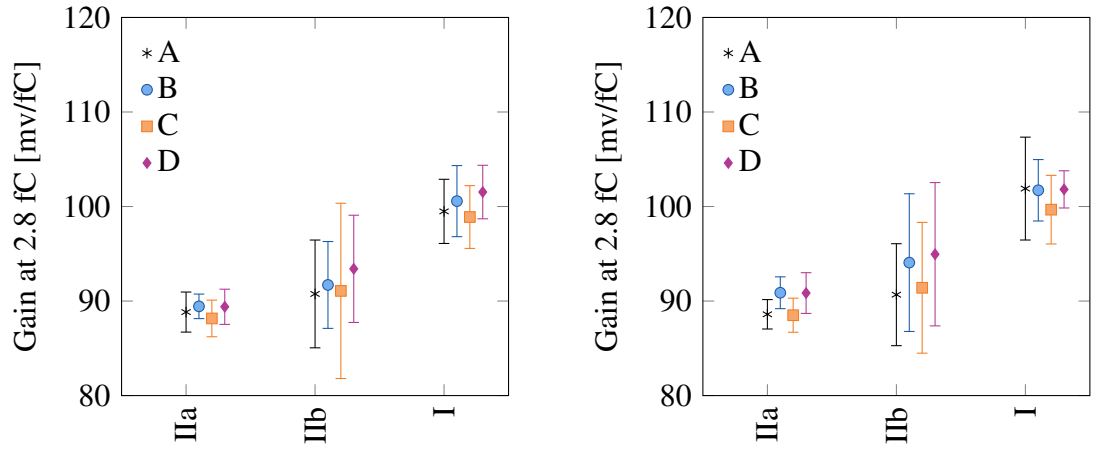
In this section, the gains obtained by the different methods are investigated. There are two strong differences between the methods. Method *I* is motivated by an actual theory which gives it the appearance of being independent of the actual sensor which is used. This idea would be correct if one would be able to extract the full charge deposit in the silicon. Since the charge is shared between neighboring strips, and there is no possibility in a binary system to fully recover the full charge of an cluster, the actual charge measured by the readout system is very specific to the individual sensor design. Method I which appears to be an unbiased method gets its bias through the charge sharing. This value depends on the actual design of the sensor. Which means it is dependent on the pitch size, the inter strip resistance, and inter strip capacitance.

If the measurement can be understood as being only complementary to a measurement with an analog system, then the idea naturally comes to just compare the two distributions



(a) *(b)*

Figure 7.8 (a) Shown is a generic response curve of a non-linear amplifier. [A] Transfer function. [B] Working point. The working point gain is given by the slope of [C]. The small signal gain is given by the slope of [D].
(b) Shows the comparison between the build in test of the ABC130 and the test beam measurements. [A] small signal gain, [B] working point gain. [A] and [B] are extracted from the built-in response curve. [C] shows the results of the test beam measurements analyzed with method II_b.



(a) Bias voltage 350 V *(b) Bias voltage 400 V*

Figure 7.9 Working point gain comparison for two voltages at full depletion. [A] Device 1 Position 7. [B] Device 1 Position 5. [C] Device 2 Position 5. [D] Device 2 Position 7.

directly with one another. This is done in method II_a and II_b. The second method comes down to the fact that the gain can be dependent on the charge. Only method II_b allows the preamplifier to have non linearity.

It is important to define precisely which gain is investigated. A preamplifier with non linearity has two fundamentally different gains as shown in figure 7.8a. It shows for small input charge the amplifier behaves almost linear but for larger input charges the amplifier shows a saturation effect. One gain is called the large signal gain or the working point gain. It describes the translation between the input signal and the output signal.

$$gain_{working\ point} = \frac{output}{input} \tag{7.8}$$

The other gain is the small signal gain or the gain at the working point. This gain is defined in terms of small changes at the working point.

$$gain_{small\ signal} = \frac{\Delta output}{\Delta input} \tag{7.9}$$

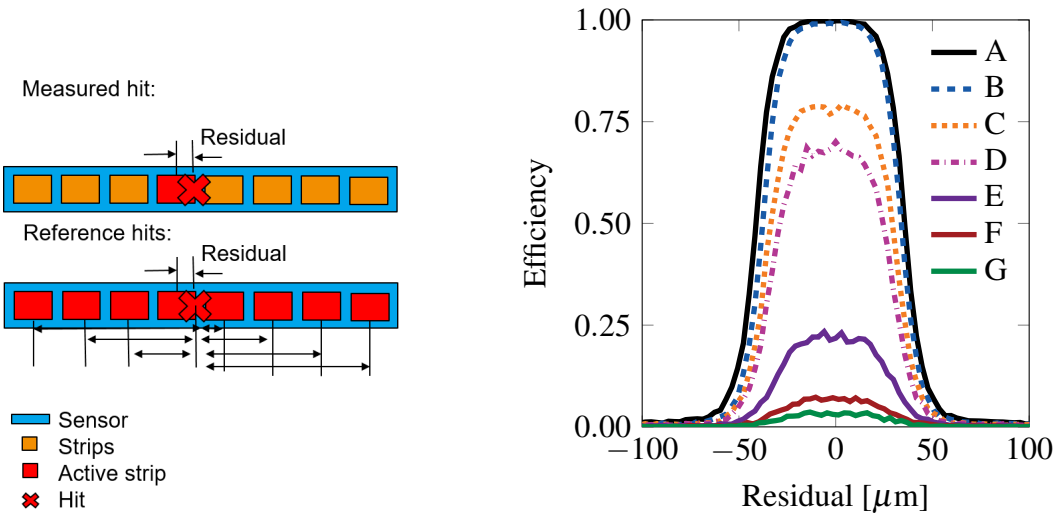
This two gains can be significantly different. With the methods described in this work only the working point gain can be investigated.

Since the gain is different for every working point, the gain is only calculated for an input charge around the MP-Value. Figure 7.9 shows the gains at the MP-Value for all devices and positions and full depletion. It shows that the gain depends strongly on the different methods used but is rather homogeneous for the different chips investigated. Also the gain varies only a small amount between the 350 V and 400 V bias voltage. While both methods II_a and II_b, which are using the ALiBaVa data as reference distribution, show a rather similar gain, method I, which is using the Landau-Gauss fit, is systematically higher.

All three methods give a value that is dramatically higher than the value obtained by the built-in tests. Figure 7.8b shows the comparison between the built-in test and test beam results. By comparing the working point gain obtained from the built-in tests and the test beam results in figure 7.8b, it is noticeable that the slope of the gain from the built-in test, at the $MP-Value$ ($\approx$ 300 mV) at $-0.065\frac{1}{\mathrm{fC}}$ is similar to the slope of the gain obtained from the test beam with $-0.08\frac{1}{\mathrm{fC}}$. However, it is also noticeable that for the actual gain value there is a large gap between the two.

7.5 In Strip Characteristics

In this section the strips are subdivided into smaller regions. This allows the investigation of inter strip effects, like efficiency or cluster size, as function of the in-strip position. These



(a) *(b)*

Figure 7.10 (a) Definition of residual efficiency.
(b) Residual efficiency. [A] Threshold = 150 mV, [B] Threshold = 234 mV, [C] Threshold = 275 mV, [D] Threshold = 316 mV, [E] Threshold = 397 mV, [F] Threshold = 471 mV, [G] Threshold = 503 mV.

can give some insights into the charge sharing between strips and the spacial extent of the charge cloud produced by charged particles. In addition to the traditional in strip plots, a new residual efficiency plot was introduced. This plot shows the efficiency of the strips as function of the hit distance from the strip.

7.5.1 Residual Efficiency

The idea behind the efficiency as function of the residual is to have a residual plot that is normalized to the amount of tracks. It is impossible to define a residual of a track without having a hit on the DUT. To overcome this problem two histograms are created. The first histogram is filled with the ordinary residual. The second histogram is filled with the residual of a "perfect" sensor. "Perfect" in this context means that all strips are always hit. It is filled with all possible combinations of extrapolated hits and strips. This will be used as the reference histogram. To calculate the efficiency, the residual histogram is divided by the reference histogram. The definition is shown in 7.10a.

The result is shown in 7.10b. It is similar to the residual plot but with the difference that it is normalized. Since it is normalized it is easier to compare different runs. It shows the actual detection efficiency as function of the hit distance from the strip. It is an interesting quantity to study because it gives some insight into at what distance the strip picks up charge.

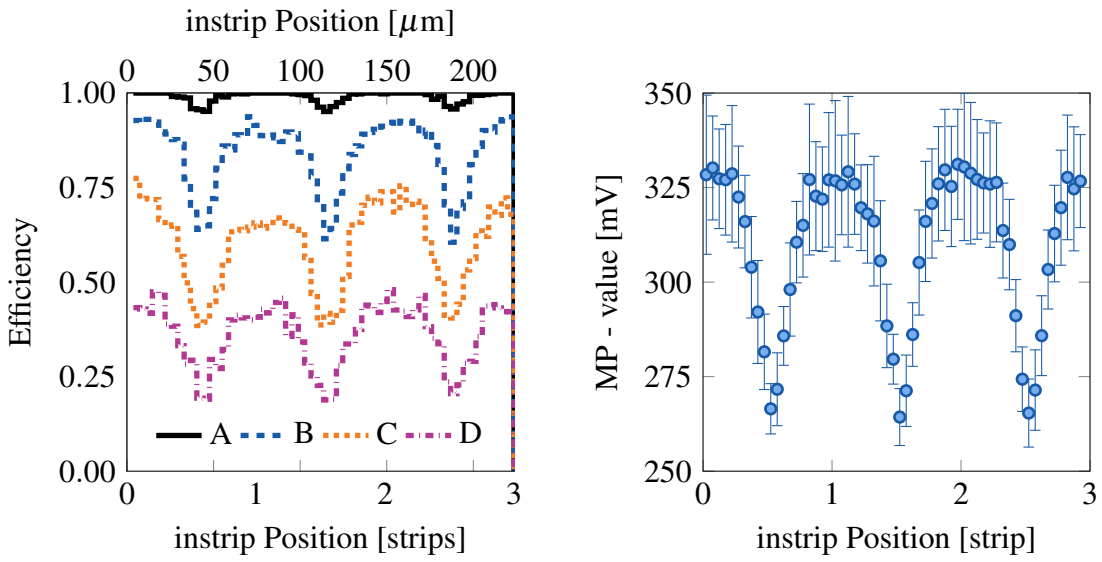


Figure 7.11 (a) Efficiency as function of the in-strip position for a variety of thresholds at full depletion. The position is calculated by using modulo 3 on the extrapolated hit position to show 3 inter strip regions at once. in-strip position (0, 1, 2, 3) correspond to hits directly on the strip while hit position (0.5, 1.5, 2.5) corresponds to hits exactly in the middle of two strips. [A] Threshold = 193 mV, *[B] Threshold* = 275 mV, *[C] Threshold* = 316 mV, *[D] Threshold* = 357 mV
(b) MP-Value as function of in-strip position calculated from (a).

For strips with a pitch size of 74.5 μm is expected that the residual efficiency is almost a two-sided step function. However, since there is a small amount of charge sharing between the individual strips, and the hit position is not defined with infinite precision, the actual residual efficiency is a two sided step function convoluted with a Gaussian. It is even possible to extract S-Curves as function of the residual from this plot. This can give insights into the charge collected by the strip as function of the hit distance. But this plot is also sensitive to the pointing resolution of the telescope setup. With the current setup it was possible to achieve a pointing resolution of 11 μm which is enough for strip by strip measurements, but measurements like this which investigate substructures inside the pitch would actually benefit from having a better pointing resolution.

7.5.2 Modulo Efficiency

To investigate sub-regions the senor is treated as if every channel is equal. This can be done since the individual channels behave rather similarly. For an easier interpretation of the resulting plots, the modulo of three strips is used instead of the modulo of one strip. This

allows to see not only one strip, but also its neighbors. Figure 7.11a shows the efficiency as function of the in-strip position for different thresholds. As expected for low thresholds the efficiency is a flat line, but for higher thresholds the efficiency starts to drop inbetween the strips. Even for thresholds which are still part of the plateau, the efficiency can drop to lower than 99 %, and in some cases even below 95 %. In future it might be interesting to investigate how these regions change as a function of radiation dosage. From the modulo efficiency, the S-Curves can be extracted and the MP-Value as a function of the in-strip position can be estimated as shown in 7.11b. It is interesting to note how vastly different the MP-Values are for a different region on the sensor. The maximum MP-Values at the center of the strips are about 330 mV, while for the regions between the strips, the MP-Value goes down to 270 mV.

7.5.3 Cluster size

From the ALiBaVa measurements it is known that the charge is not collected on only one strip but is shared between neighboring strips. From theoretical considerations, it is known that there are two main sources of charge sharing. Firstly, the spacial dimension of the charge cloud in the silicon. For 300 μm of *Si* a charge cloud of 10 μm is expected [**?**]. Secondly, the electrical cross-talk due to the finite resistivity and capacitive coupling between neighboring strips. The first cross-talk is very sensitive to the actual hit position alongside the strip, while the second one is rather agnostic to this. Figure 7.12 shows the cluster size as a function of the in-strip position for a low threshold of 68 mV and a fully depleted sensor. With this low threshold both of these charge sharing effects are visible. Not only is the strong increase in the cluster size between two strips clearly visible, but also that the cluster size does not go back to one in the center of the strips. By increasing the threshold, the effects of the charge sharing are less pronounced and the cluster size is mostly one, except for the regions right between two chips as seen in figure 7.13a. It also shows that at the center of the strips the cluster size is very small and almost constant.

Figure 7.13b shows the average cluster size as a function of the threshold. It is clearly visible that the cluster size asymptotically decreases towards cluster size one. Since for this analysis only events that have at least one hit are used the average cluster size cannot be smaller than one.

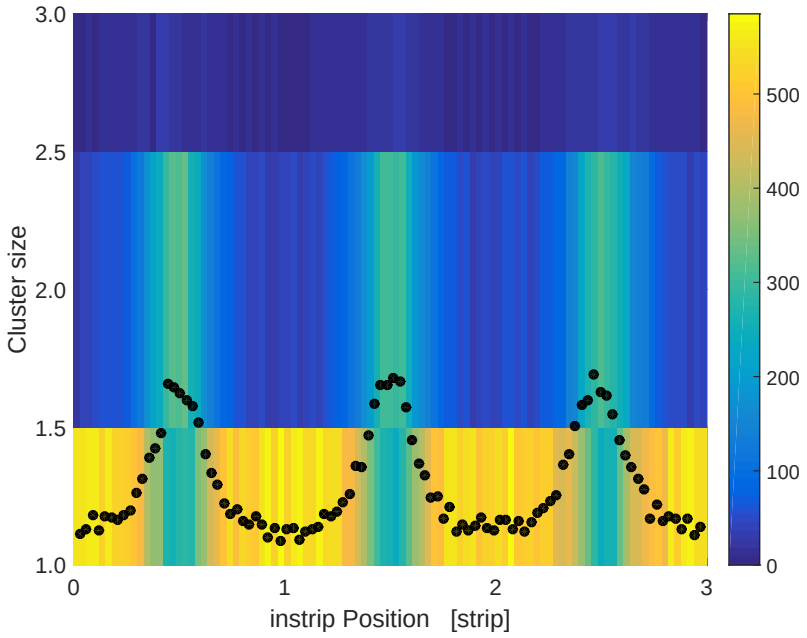


Figure 7.12 Cluster size histogram as function of the in-strip position. The color code shows the abundance of hits with a certain cluster size at this in-strip position. The doted line shows the average cluster size at a given in-strip position. The measurement were done with: Bias voltage = 350 V, Threshold = 68 mV, Pedestal = 42 mV

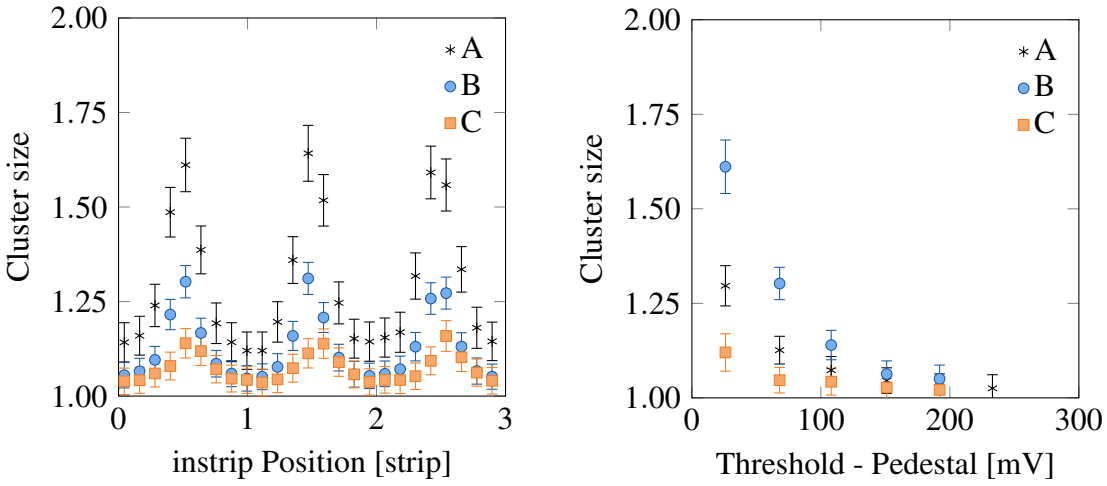


Figure 7.13 (a) Cluster size as a function of in-strip position for different thresholds. [A] Threshold = 68 mV, [B] Threshold = 109 mV, [C] Threshold = 151 mV. Pedestal = 42 mV *(b) Cluster size as a function of the effective threshold. [A] average cluster size. [B] cluster size in between two strips. [C] cluster size at the center of one strip.*

Chapter 8

Development of Test Beam Infrastructure

During the extensive development phase of a new particle detector it is necessary to investigate the detector properties, under controlled conditions, throughout the development cycle. Typically, this is done at facilities that provide energetic beams of charged particles. To take full advantage of the test beams, so-called test beam telescopes (short: telescopes), which can track the trajectory of test beam particles precisely are deployed to make measurements. This information can be used to calculate where the Device Under Test (DUT) is hit, and can compare it to the response of the DUT. The telescopes were developed within the EUDET project [**?**]. Copies of these telescopes are in use worldwide. As the purpose of these telescopes is to give track information for the investigation of new DUTs, the easy integration of the new DUTs specific readout systems is a mandatory feature. Therefore the software framework was written with the goals to be flexible, extensible and portable.

For the integration of the strip detector, developed for the phase II upgrade of the ATLAS detector, some modifications where necessary. Firstly, providing a clear interface between the EUDAQ framework [**? ?**] and the ITSDAQ (the ROOTProducer see section 8.5). Secondly, having the possibility to resynchronize events based on timestamps and other characteristics. For this, the development of a more sophisticated synchronization algorithm was mandatory. It turned out that this algorithm can also be used to handle the asynchronous data streams of EUDAQ 2.0 as described in section 8.7. Thirdly, to make the EUDAQ framework easier to understand and develop it was necessary to modularize it into smaller parts, the so-called processors (see section 8.6).

The development of packets (see section 8.4) and multiple Data-Collector (see section 8.8) was a request from the test beam community, and were not primarily done to support the ATLAS upgrade strip detector development. Nevertheless the packets and multiple

Data-Collector can become useful for ATLAS strip detector development for reaching trigger rates beyond 10^4 s^{-1}.

8.1 Data Acquisition System (DAQ)

The purpose of a DAQ system is to capture physical observables as digital quantities for later processing [**?**]. It connects the physical world with the digital world. Thus a DAQ system always consists of two layers. The hardware layer, in which we have the actual measuring devices, and the software layer, in which the data handling and data storage take place. This part of the work focuses completely on the software part.

8.1.1 Telescope DAQ system

Depending on the environment every DAQ system has its own requirements. In the context of detector R&D a DAQ system has very specific requirements. It needs to be easy to integrate already existing DAQ systems; for this it must run on any platform with as little extern dependencies as possible. To achieve this it must have a modular design so that every component can run independently from each other. The Application Programming Interface (API) "must be easy to use correctly and hard to use incorrectly" [**?**]. The files must be stored for a very long time, therefore it is mandatory to have a robust file format which is supported over all generations of this software.

8.1.2 Telescope DAQ Setup

The telescopes consists of a variety of different hardware and software components, as shown in figure 8.1. On the hardware side the Trigger Logic Unit (TLU) works as the central authority. It receives an analog trigger signal from a fast triggering detector, like a scintillator with photomultiplier (PMT) or an ATLAS FEI4 detector. The TLU can take up to four different devices as a trigger input. It is possible to combine the trigger signal in the TLU with the firmware in a variety of combinations of AND, OR and NOT. On the software side (EUDAQ) the data taking process is controlled by a central authority called Run-Control. Its purpose is to configure the producer and tell them when to start and when to stop. The hardware is represented on the software side by so-called “producers” which act as an interface between the DUT’s readout systems and the EUDAQ framework. In the current scheme it is expected that every producer sends one event to the central Data-Collector for each trigger. The Data-Collector makes sanity checks such as that every device sends an event and that run number and event number are correct. The complete communication

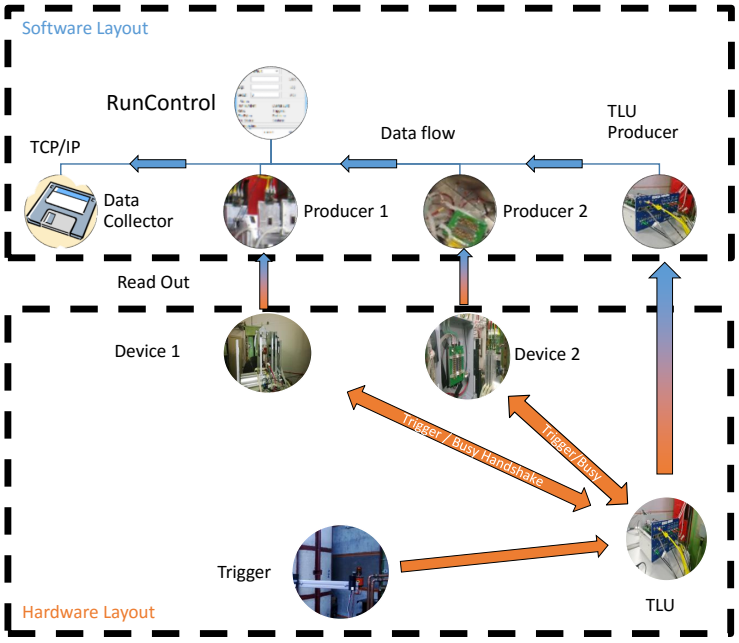


Figure 8.1 Overview of the hardware and software layout of the EUDAQ framework.

between individual components of the framework is done via TCP. Therefore it is possible to run every part of EUDAQ on a different computer with a different operating system. This allows for the easy integration of user's hardware to the test beam infrastructure.

8.2 Limitations of EUDAQ 1.x

EUDAQ 1.x makes certain presumptions about the DUTs. It assumes that all DUTs can be externally triggered, that they give exactly one data set for every trigger and that the integration time of all DUTs is roughly identical. Figure 8.2a shows the typical working scheme of EUDAQ 1.x. It includes two devices with different integration times and the TLU. The TLU issues a trigger as soon as the particle passes through the devices. DAQ1 shows a device with a long integration time (for example the MIMOSA26 with $2 \times 114.5\,\mu$s). DAQ2 shows a device with a short integration time (for example the FEI4 APIX with 25 ns). In order to fit both devices in the EUDAQ 1.x readout scheme, the issuing of a new trigger from the TLU has to be blocked for the entire ReadOut Frame (ROF) (see section 8.4.1) of DAQ1. With this scheme the device with the longest integration time determines the maximum trigger rate of the entire setup. To increase the data rate it is necessary to let each device run at its own speed and provide a sophisticated synchronization algorithm. This requires the implementation of asynchronous data streams. Figure 8.2b shows the EUDAQ

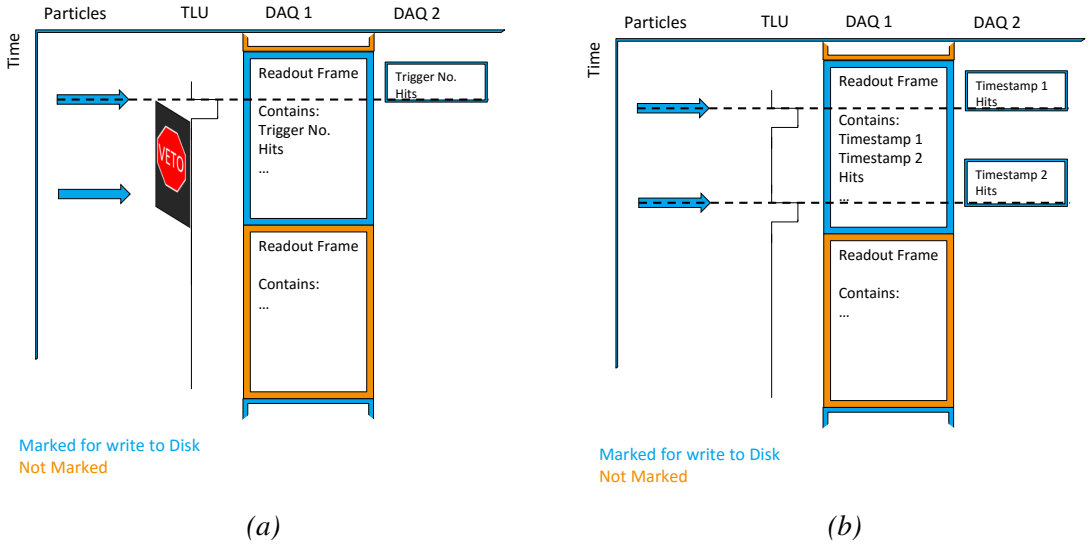


Figure 8.2 Comparison between EUDAQ 1.x and EUDAQ 2.0.
(a) EUDAQ 1.x readout scheme. The TLU is vetoed for the time the DAQ1 needs to finish its readout chicle.
(b) EUDAQ 2.0. The TLU is not blocked by DAQ1. The second particle can also be recorded.

2.0 scheme where the TLU is not vetoed after the first trigger but is allowed to issue the next trigger as well. In this example, DAQ1 adds the second timestamp to the event which is later used for synchronizing events.

The notation of a data stream in EUDAQ 1.x is not defined as a central part of the framework. Data is transferred from one part of the framework to the next by vastly different interfaces. This makes it difficult to follow the data streams. In order to manage the increased difficulty of handling asynchronous data streams the framework needs to provide a common interface for handling data streams (see section 8.6).

In addition to these limitations EUDAQ 1.x does not provide a clear Application Binary Interface (ABI) [**?**] for the producer class. The producer class is the interface from the user-specific DAQ system to EUDAQ. Due to the lack of a clean ABI the user code and EUDAQ need to be compiled with identical compilers and compiler flags, which imposes a huge burden on the users code. The user DAQ system and EUDAQ are usually developed independently of each other, therefore the issue of connecting the two DAQ system arises very late in the development which makes it more difficult to integrate. In the community this problem is solved by, for example, writing the whole DAQ software inside the EUDAQ framework as is done with the TLU integration, or by writing their own interface like it is done with the NI producer, which is used to interact with MIMOSA26 telescopes. This producer is compiled as part of EUDAQ and provides TCP/IP connections for the actual

MIMOSA26 readout to connect to. Both of these solutions are not optimal since they either restrict the freedom of the developer, or force the developer to create their own interface. The first steps towards providing a clean ABI was done by implementing a language interface to python [**?**].

8.3 Requirements for EUDAQ 2.0

For the further development of the EUDAQ framework four goals were to be achieved. Firstly, improving the possible trigger rate by up to two orders of magnitude, from a few hundred trigger per second up to several ten thousand trigger per second. Secondly, easier integration of different types of devices. EUDAQ 1.x is limited to trigger based devices where one trigger corresponds to one ReadOut Frame (ROF) (see section 8.4 and section 8.7). Thirdly, Integration of the ITK Strip DAQ (ITSDAQ) software. Since the ITSDAQ is based on the CPP interpreter (CINT) it was necessary to have an interface for it (see section 8.5). Fourthly, Improve the overall structure of the software and provide useful interfaces for further development (see section 8.6).

EUDAQ 2.0 introduces several big changes to the way data is handled. All changes have the main focus on empowering the user to have more control over the data stream. For example in the scheme of EUDAQ 1.x, the user's DAQ has to be able to count every single trigger issued from the TLU. It has to preprocess every single buffer readout during data taking and extract the individual ROFs. All the data has to be sent to a centralized data collector which stores the data in one file. In the scheme of EUDAQ 2.0 these restrictions have to be loosened. For EUDAQ 2.0 the user's DAQ should not need to receive the trigger from the TLU. The user should be able to synchronize the data based on timestamps, for example (see section 8.7). With EUDAQ 2.0 the means of synchronization can be chosen by the users. The user's DAQ also should not need to decode the buffer readouts into individual ROFs. Therefore, EUDAQ 2.0 should support packets (see section 8.4) which can store raw data from several ROFs, and then the packets are only converted to ROFs when actually needed (online monitoring or offline analysis). To prevent bottlenecks from the impending network, EUDAQ 2.0 should be able to store data locally on the same PC that does the readout. Therefore EUDAQ 2.0 must supports multiple data collectors which allows each producer to store the data locally, without network overhead (see section 8.8). The merging should then be done completely offline.

For many years EUDAQ has been the backbone of a successful test beam campaigns, therefore it was crucial to break backward compatibility as little as possible. This means the switching from EUDAQ 1.x to EUDAQ 2.0 must not be more complicated than compiling

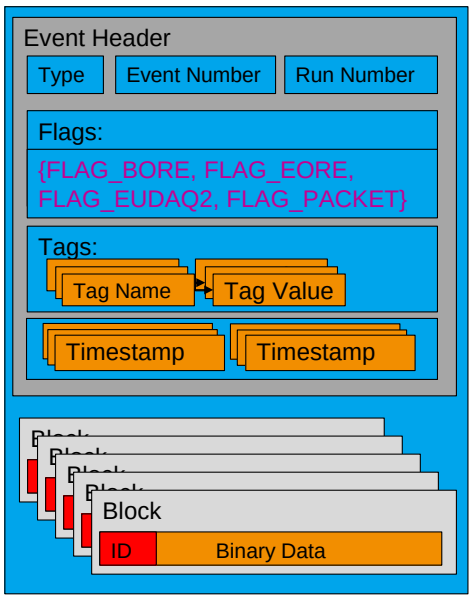


Figure 8.3 Visualization of the RawDataEvent. It is separated into the header part which is derived from the event base class and the body which contains the data blocks.

against the new library. Changes to the user's code will only be necessary when the user wants to use new features.

To empower more users to use the new features of EUDAQ 2.0, the code base needed to be modernized. Particularly, it was important to improve the handling of the data streams. It was desired for EUDAQ 2.0 to have a transparent path of the data through the framework. This required the introduction of a new base class and a new mechanism to handle data streams (see section 8.6).

The design and implementation of EUDAQ 2.0 is part of this thesis. The individual components are described in the following sections.

8.4 Packets

The readout of many devices is not done event by event but instead the entire hardware buffer is readout at once. In EUDAQ 1.x these packets have to be split into events which correspond to individual triggers. For some devices this includes a significant amount of overhead. A typical example of such a device is the TLU. The information connected to the individual trigger is 8 Bytes (one uint64), the size of one TLU event is 33 Bytes. That means the overhead of this event is more than three times the information itself. To reduce

this overhead, packets are designed to store the information of multiple triggers in one object, without any unnecessary overhead. To store a 1k triggers or even 1M triggers would have the same overhead as storing one trigger in the EUDAQ 1.x. The extraction of the data is moved to the offline reconstruction.

8.4.1 Definition ReadOut Frame (ROF)

A readout frame (ROF) is given by the finest granularity that can be retrieved from a device. For this definition it is not important whether the ROFs are transmitted to the DAQ one ROF at a time, or whether they are packed in blocks of ROFs.

For example; the TLU stores a large number of ROFs internally and they all get readout at the same time, but since the software is able to reconstruct the information to the individual trigger, the finest granularity of the TLU is the individual trigger. Therefore every trigger is one ROF. Another example is the MIMOSA26. A MIMOSA26 frame covers a time of $114.5\,\mu$s. Since the MIMOSA26 frames cannot be subdivided into smaller time units, one ROF is one frame of the MIMOSA26 rolling shutter.

This definition only asks about the information from a device but not how the information was readout.

8.4.2 The RawDataEvent

For most use cases the RawDataEvent is a suitable data structure. Its layout is sketched in figure 8.3. All event types share a common event header which stores meta information like type/subtype, event number, run number, several flags, tags and a vector of timestamps. The type and subtype is used to identify the corresponding Data-Converter-Plugin. The run number identifies the individual run the event was created from. It is checked during data taking to ensure that all producers are on the same run. The event number is a continuous number which can be used to synchronize the events from different producers. The flags are designed to mark certain events, for example, Begin Of Run Event (BORE) or to mark if it is a EUDAQ 1.x type event, or a EUDAQ 2.0 type event. It is also used to distinguish between events and packets (see section 8.4.4). Tags are used to store additional information to this event. Internally tags are always stored as strings. Compared to EUDAQ 1.x, the EUDAQ 2.0 RawDataEvent has a vector of timestamps instead of just one timestamp. This change was required to allow devices with a long integration time to store all trigger timestamps. The RawDataEvent adds a vector of data blocks to it. This blocks consist of an ID to identify the individual data block and a storage for binary data. The framework makes no assumption

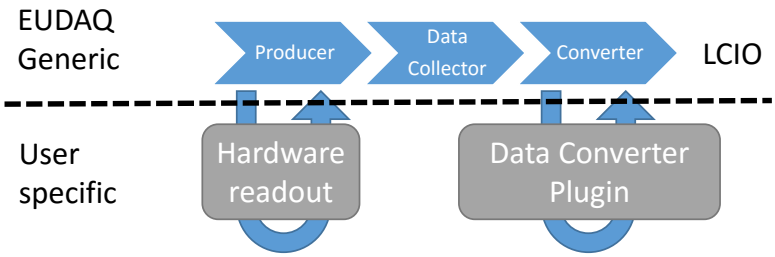


Figure 8.4 Visualization of the synergy between user specific code and library generic code through the means of the Data-Converter-Plugin.

about how to store the data inside this block. The encoding is done by the user's producer and the decoding is done by the user's Data-Converter-Plugin.

8.4.3 The Data-Converter-Plugin

The separation of the user's code from the library code is an important goal for the EUDAQ framework. This separation is achieved by using so called Data-Converter-Plugins. The propose of these plugins is to contain all user specific code. A typical example is illustrated in figure 8.4. It shows the data flow inside EUDAQ. The data is created on the producer side by reading out the given hardware. Since no two detectors are alike, this readout has to be specific to the hardware. The information is then stored in an EUDAQ event, for example the RawDataEvent. The framework knows how to transfer the data over the network and how to store it on disk but it does not know how to convert it to pixel information. For track reconstruction with EUTelescope it is necessary to convert the data to LCIO files containing standardized pixel information. Therefore every user group has to provide a Data-Converter-Plugin which is able to convert their raw data to pixel information. The interface of the Data-Converter-Plugin is sketched in listing 8.1. The Data-Converter-Plugin interface provides two virtual function for the user to overload. Firstly, "GetStandardSubEvent" which converts the data to hits on a plain which is then stored inside a StandardEvent. Secondly, "GetLCIOSubEvent" which stores the data as a collection in a LCIO event. This approach allows the users to store the data as it suits them best, and still ensures a clear separation between user code and library code.

Listing 8.1 shows three more functions (IsSyncWithTLU, ExtractEventN and GetNumberOfEvents) which are used for doing the synchronization with the TLU (see section 8.7), and for unpacking events from packets (see section 8.4.5).

Listing 8.1 Core functionality of the Data-Converter-Plugin.

```
using event_sp = std::shared_ptr < eudaq::Event > ;

class DataConverterPlugin {
public:

  virtual int IsSyncWithTLU(const eudaq::Event& ev,
                            const eudaq::Event& tluEvent) const;


  virtual bool GetLCIOSubEvent(lcio::LCEvent& /*result*/,
                   const eudaq::Event& /*source*/) const;


  virtual bool GetStandardSubEvent(StandardEvent& /*result*/,
                            const eudaq::Event& /*source*/) const;


  virtual event_sp ExtractEventN(event_sp pac, size_t EventNr);
  virtual size_t GetNumberOfEvents(const eudaq::Event& pac);
...
};
```

8.4.4 Implementation of Packets as an Extension of Events

Since EUDAQ 1.x already had a data structure (the RawDataEvent) which had almost all the properties needed for the new packet class, it was logical to think about packets as an extension to events. There are two major changes to the event class itself to make it usable in both contexts. The first one was to give it a new flag which indicates whether it is an event or a packet. The second one is to allow it to have multiple time stamps. Due to the multiple timestamps, EUDAQ 2.0 files are not compatible with EUDAQ 1.x files. Nevertheless EUDAQ 1.x can read EUDAQ 2.0 events but it will lose the information of the additional timestamps. There is also a possibility to compile EUDAQ 2.0 with a flag to ensure that it writes EUDAQ 1.x events. In this case the additional timestamps get lost.

8.4.5 Packing and Unpacking

In EUDAQ 1.x the internal layout of an event is completely up to the user. Only the header of the event stores some common data, like the event number, to do event building, for example. The reconstruction of, for example, the pixel information in EUDAQ was delegated to the Data-Converter-Plugin. This part of user's code was designed to handle all sorts of DAQ specific modifications. The benefits of having the event as a data structure separated from the Data-Converter-Plugin, which contains details about the individual event type, is that not all

Listing 8.2 Example code for the unpacking

```
using event_sp = std::shared_ptr < eudaq::Event > ;

void ExtractEvents(event_sp Ev){
  if (Ev->IsPacket()){
    const auto max_events = PluginManager::GetNumberOfEvents(*Ev);
    for (size_t i = 0; i < max_events; ++i){
      ExtractEvents(PluginManager::ExtractEventN(Ev, i));
    }

  } else {
    AddEvent(fileIndex, Ev);
  }
}
```

parts of the EUDAQ framework need to have all Data-Converter-Plugins. During data taking EUDAQ runs on many different computers reading out many different hardware components.

Since the packing of the packet is completely the user's responsibility the unpacking must also be under the user's control, therefore the the code for the unpacking is part of the Data-Converter-Plugin. The Data-Converter-Plugin has two new virtual functions to be overloaded by the user called "ExtractEventN" and "GetNumberOfEvents", as shown in listing 8.1. They are used as follows: first, the event has to be checked if it is a packet or not. This is done by calling the "IsPacket" function from the event class. It returns true if it is a packet, and false if not. After this the packet can be unpacked in a loop. "GetNumberOfEvents" gives the number of events stored inside the packet. "ExtractEventN" extracts the N-th event from the packet. An example for unpacking is given in listing 8.2. It should be noticed that the extracting happens recursively. This is necessary since packets are allowed to store other packets as well.

8.5 Interface between EUDAQ and the CPP Interpreter CINT of the CERN ROOT Framework

Since the ITSDAQ [**?**] software which is used for reading out the ITK strip modules is based on the CERN ROOT framework [**? ? ?**], it is desired to have an interface to ROOT. Even though its primary reason for existence is to handle the communication with the ITSDAQ, it is not desired to limit its compatibility to only this software, but instead it is designed to be more generic. The C++ interpreter of ROOT (CINT) has certain limitations which the interface had to compensate for. The three most problematic limitations are: firstly, the

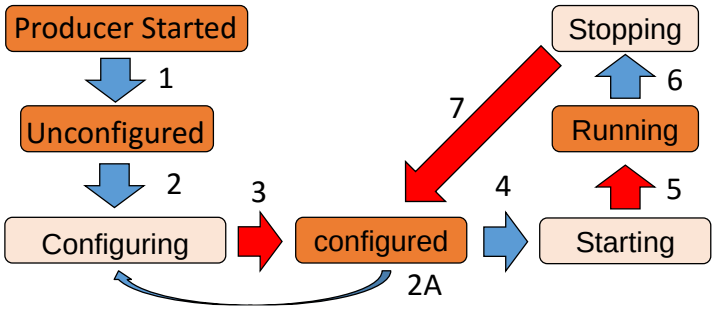


Figure 8.5 The states of the ROOTProducer

exceptions must not leak from EUDAQ to the CINT. Secondly, multi threading. Particularly when a EUDAQ thread calls into interpreted code, it has caused sometimes crashes which did not appear in compiled code. Thirdly, all classes have to be "exported" to make them usable inside CINT. ROOT also adds some features to the toolbox. One feature which is heavily used is the signal-slot feature.

Exporting classes for CINT: To make a class accessible from inside CINT a dictionary file needs to be created. There are certain limitations on what the parser can easily understand, therefore it is reasonable to use the Pointer to IMPLementation (PIMPL) idiom [**?**]. This is a technique used to provide a clean interface to the caller of the library. The idea is to create two classes, one interface class and one implementation class. The interface class contains the full interface used later. In addition it holds a pointer to the implementation class. It also manages the life cycle of the implementation class. Every call to the Interface class gets forwarded to the Implementation class. This technique is easy to implement but has the overhead of implementing each member function twice, and also each member function call requires a pointer dereference. Nevertheless the benefits of the clean interface is worth the effort.

Exceptions: The EUDAQ framework relies quite heavily on throwing exceptions as a mean to communicate errors to the users. Inside C++ exceptions are the superior means to communicate errors but to form a consistent ABI they are not allowed to leak the binary boarder. Therefore they need to be caught inside the library binary. If they are not caught inside the binary the behavior is undefined. For the ROOTProducer the exceptions are caught internally and an error code is returned. In addition an error messaged is displayed.

Multithreading: Each producer runs at least two threads that have a direct impact on the users. The first one is controlled by the user and handles the readout of the user's device. The other one is the internal thread used to receive commands from the Run-Control. The transition between the individual states of the producer is also done on these thread. To make this kind of transitions, the internal thread would need to call in the interpreted user's code,

Listing 8.3 The interface the state machine of the ROOTProducer

```
class ROOTProducer {
...
  // signals
  void send_onStart(int);
  void send_onConfigure();
  void send_onStop();

  //Slot
  void checkStatus();
  void setStatusToStopped();
...
}
```

which has been shown to not be reliable. Therefore it was mandatory to switch this work to the user's thread. The solution used for the ROOTProducer is to transform the transitions into states of the ROOTProducer. This way it is the user's responsibility to check for the current state of the producer and act accordingly. Figure 8.5 shows the ROOTProducer as a state machine. Compared to the producer class itself it has three more states (configuring, starting, stopping). The producer starts on the top left hand side with the "Producer Started" state. (1) From there the producer tries to connect to the Run-Control. When the producer is connected it is in the state "Unconfigured". At this stage it waits for the Run-Control to sent a configuration. (2) When the configuration is send the producer enters the "Configuring" state. At this state it waits for the user to configure the device. (3) The user has then to respond and set the producer in the configuring state. There are now two options: (2A) either the producer can be configured a second time or (3) the Run-Control can send the start command which leaves the producer in the "Starting" state. Now it is again up to the user to check for this status and set the device to start. (5) after this the user has to set the producer to the "Running" state. (6) On arriving of the stop command from the Run-Control the producer is in the "Stopping" state. Again the user has to make sure to check for this state and stop the device. (7) After this it has to change the state back to "Configured". Doing all of this by hand is a tiresome exercise but luckily the ROOT framework provides a possibility to stream line this work by using signals and slots.

Signals and Slots: To simplify the work with the ROOTProducer and to keep the interface clear between the producer and the ITSDAQ, the signal slot mechanism is used [**?**]. It is an elegant way to assign function pointers in the program. In this thesis not all parts of the interface are explained but the focus is on the state checking. Instead of checking the states by hand, the ROOTProducer has an automation for this. The part of the public interface which handles the state checking is shown in listing 8.3. The user's DAQ has to mirror this

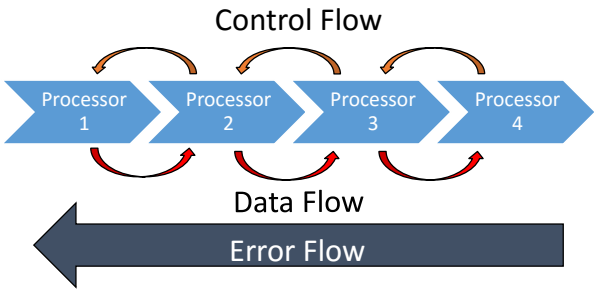


Figure 8.6 Information flow through a chain of processors

interface with the difference that all signals must become slots and all slots become signal. Once the connection is made, every time a signal from one class A is called it will call the slot from the class B it is connected to. The user's DAQ system must provide a function that checks periodically the status of the producer by calling the slot "checkStatus". If the state is none of the newly introduced states the function will return immediately. If the state is "Configuring" it will send the signal "send_onConfigure". The function from the user's DAQ which is connected to this signal will be called. This will happen on the same thread as the "checkStatus" call. After the function "send_onConfigure" returns the state is set to "Configured". Equally if the state is "Starting" it will send the signal "send_onStart" The argument given by the signal is the current run number. After the function returns the state will be set to "Running". Finally if the state is Stopping it will send the signal "send_onStop". The only difference for this state is that on the return of the function it will remain in the Stopping state until the slot "setStatusToStopped" is called. Since the switch of the status happens on the same thread as the readout the producer would not be able to finish running if it would be stopped immediately.

This kind of state switching is obviously not perfect. Therefore it is worth, as soon as EUDAQ has switched to ROOT 6.x, to try to address the multi thread issue again with the new interpreter (CLING) [**?**], which comes with ROOT 6.x. The solution shown in this section is easy to adapt if multi threading would be available. In this case the send signal can be called directly from the internal thread, and the extra step of periodically checking the state would be omitted.

Listing 8.4 The base class of the processors

```
  using event_sp = std::shared_ptr < eudaq::Event > ;

class ProcessorBase {
  public:
...
    virtual void init();
    virtual ReturnParam ProcessEvent(event_sp ev, ConnectionName con);
    virtual void end();
    virtual void wait();
...
  protected:
    ReturnParam processNext(event_sp ev, ConnectionName con);
...
};
```

8.6 Processor Class for the transparent Handling of Data Streams

The main goal of introducing the processors was to modulize the framework into individual blocks which can easily be linked together.

The EUDAQ processors take ideas from many different frameworks but three of them have the strongest impact on the actual design. First, the Marlin framework [**?**]. Second, the pipeline system of the bash shell [**?**] and third the pipeline system of Powershell [**?**]. The strength of the Marlin system is that it combines the processors only during run time with the help of an XML [**?**] based file, called the steering file. This allows to make various changes to the processors and their correspondents to each other without having to recompile the project. For an analysis framework this is a good feature, but not necessarily for a DAQ system. For a DAQ system this approach would have too much overhead. For a DAQ system it is beneficial to have as much as possible written in actual code. The usual work flow is that the project is built once and runs for a long time without any changes.

For simplicity reasons the processors are implemented by using a common base class (ProcessorBase). The interface is sketched in listing 8.4.

8.6.1 The Processor as a state machine

The processor can be understood as a state machine with two states. After creating of the processor, it is in the "uninitialized" state. In this state the processor can be modified, for example it can be chained to another processor but it cannot process data. The transition from the "uninitialized" state to the "running" state is done by the "init" method. As an example

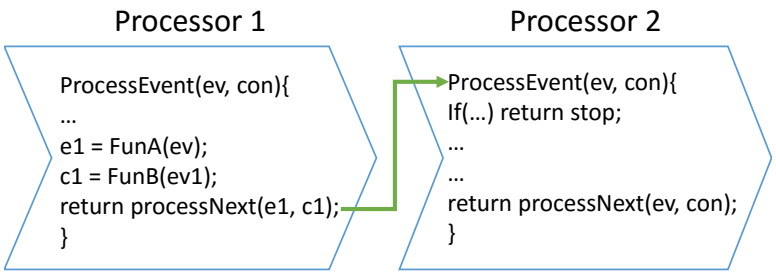


Figure 8.7 Connection between the processors in a chain. (pseudo code)

the file reader processor opens the file it is supposed to read with this method. For other processors this can mean resetting some internal buffers or just storing the new state it is in. After the processor is initialized it is in the "running" state and prepared to accept data. The initialization has always to start with the last processor in line first (back to front) since the processor is allowed to produce data directly after it is initialized.

The "wait" method allows for a given processor to keep the processor chain in the running state for as long as it needs to complete its work. This method is specially designed so the data stream runs on a different thread to the control thread. In this case the control thread can wait in the wait method until the run is finished.

After the run is finished the end function must be called. It transfers the processor back to the "uninitialized" state. This method is responsible for the clearing of buffers and the release of no longer required handles (for example: file handles, socket handles...). This must always happen from the first processor to the last processor (front to back).

8.6.2 Information Flows

Figure 8.6 shows four generic processors chained together. It also shows the three different information flows which flow through the processors.

First, the data flow: Every processor is directly connected to the next processor in line. It receives data from the previous processor and sends data to the next processor. The data is given to the next processor by shared pointers of the event base class. In addition to the event pointer also the stream number is given to the processor. Figure 8.8 shows an example in pseudo code of two connected processors. Processor 1 takes an event "ev" on the stream given by the variable "con" (short for connection). It then processes the event "ev" and produces "e1" which is then sent to the next processor on the stream "c1".

Second, the control flow: Every processor can respond to a given event with one of three return values: success, stop or skip. Success is the return value that indicates everything is

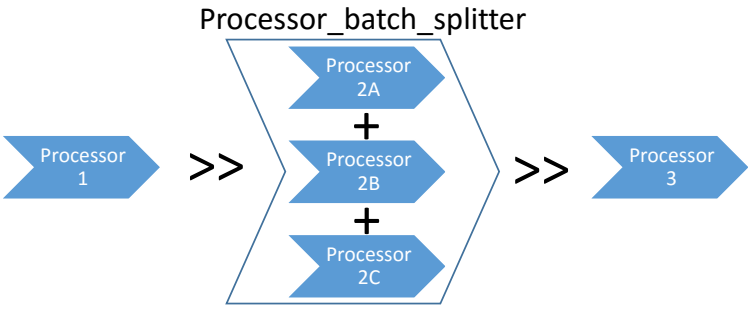


Figure 8.8 Visualization of the Processor_batch_splitter. It allows to split the data stream on multiple processors.

alright and the run should be continued normally. Stop is used to stop a run. It can be used for example to make an event selection processor which only uses the First N events. After the N-th event it returns stop and the run will be stopped. Skip tells the previous processor to skip this event. This is used for the batch splitter (see section 8.6.3).

Third, the error flow: Errors which are thrown by one processor can be caught by any processor in the line. For example, if processor 3 in figure 8.6 throws an exception then at first processor 2 has a chance to catch the exception. If processor 2 does not catch the exception it propagates backwards to processor 1. If the exception is not caught at all the program gets terminated. Exceptions in EUDAQ are used to communicate any sort of unwanted condition. For example, when a file writer tries to open a file but the file does already exist it will throw an exception. The approach to allow the exceptions to travel through all previous processor enables the user to write special processor which only deal with a certain kind of problem and add them to the processor chain. This approach makes the error handling very flexible. This can be visualized by assuming a system which needs the processors 1-3-4 to complete a certain task. Processor 3 can throw an exception from which the system cannot recover. To handle this exception processor 2 is introduced so that it catches the exception and sends a stop signal to processor 1, and in addition it produces a report to the user. If for some reason the report to the user must change, for example there is one version of the program which runs in the command line and another one which has a GUI, then only processor 2 needs to be replaced, but all the rest can stay as before. This approach gives a high amount of flexibility to the user and allows for maximum code reuse.

8.6.3 Processor Chains (Batch)

As mentioned earlier, processors can be chained together. For this a special kind of processor is used, the so-called batch. There are two kinds of batch processor.

The first one is Processor_batch. It links the processors horizontally together as seen in figure 8.6. For this batch, the output of one processor is the input for the next. This processor possesses a new member function, "run", which runs the processors in the chain by sending empty events[1] to the first processor in line.

The second one is Processor_batch_splitter. It is used to link processors vertically together as shown in figure 8.9b. All processors get the same input event pointer and are linked to the same output processor. To give an example, the output of processor 1 is first given to processor 2A. The output of processor 2A is then directly given to processor 3. If processors 3 and 2A return success the same output from processor 1 is again used as input for processor 2B. The output of processor 2B is again the input of processor 3. If processor 2B returns "success" processor 2C will be executed next. If one of the processors returns "skip" then the next processors in line are skipped. For example if processor 2A returns "skip" processors 2B and 2C are skipped.

8.6.4 Examples

The first code example given in listing 8.5 gives a brief overview on how to use the processors. It is a simplified version of the converter program as it is part of the EUDAQ 2.0 framework. It starts with a processor that reads in a file. For this, the processor uses internally the "eudaq::FileReader" class, which was already part of EUDAQ 1.x. The argument given to it is the file name. The next processor is linked to this processor by using the streamer operator. The output of the fileReader is the input to the ShowEventNR processor, which will show the event number of every N-th event. This processor only inspects events, it does not modify them, therefore it can be directly chained to the next processor. Next in line is the eventSelector. It will select events based on its event number (for example events from 1 to 1000). The last processor in line is the fileWriter. It will store the events it receives on disk with the name given by the outputName. The return of this statement is a smart pointer to a batch processor. The batch processor stores all pointers to the individual processors, and will also manage their life time. By calling "init" on the batch processor, all processors owned by the batch are initialized. The run method starts a loop in the batch processor which sends empty events to the fileReader. The fileReader will then fill these empty events with events read from the file. When the file is empty the fileReader will return "stop". On "stop", the batch will quit the loop and return from the run method. On "end", the processors are reset and ready for the next use.

[1] Empty in this case means nullptr

Listing 8.5 Simplified version of the Converter executable.

```
auto batch = fileReader(fname)
  >> ShowEventNR(N)
  >> eventSelector(selectedEvents)
  >> fileWriter(outputName);


batch->init();
batch->run();
batch->end();
```

The next example, given in listing 8.6, shows the simplified implementation of the Data-Collector. Its purpose is to receive the data from the individual producer and store them in one file. Variables starting with "m_" are member variables/objects. "On OnConfigure", a new batch is created and stored as a member of the Data-Collector. It is stored in a unique pointer therefore the new batch will completely delete the old batch. The first processor added to the batch is the dataReceiver. The dataReceiver processor is already created in the constructor of the Data-Collector. This allows the producers to immediately connect to it. The dataReceiver is also given to the batch as a raw pointer. A raw pointer in EUDAQ always means that the object is owned by somebody else, therefore the batch does not try to delete it on its destruction. Therefore the producers are still connected to the same dataReceiver even after reconfiguring. As opposed to the fileReader, the dataReceiver does not necessarily know when the last event arrived. To handle this problem the waitForEORE processor counts the open streams and only allows the processors to enter the end method when all streams have been closed again with an End Of Run Event (EORE), or it timed out. The next two processors just show the event numbers and write the data to disk. On OnPrepareRun the processor chain gets initialized. Since the dataReceiver runs on its own thread, it is not necessary to call the run method. On OnStopRun the batch is stopped. This happens in two steps. First, by calling the "wait" method. This method waits for all processors to be ready to be terminated. In this case it waits only for the waitForEORE processor. The "wait" method is implicitly called by calling the run method therefore it was not in the previous example. Secondly, after the waiting is done the run is ended by calling the "end" method.

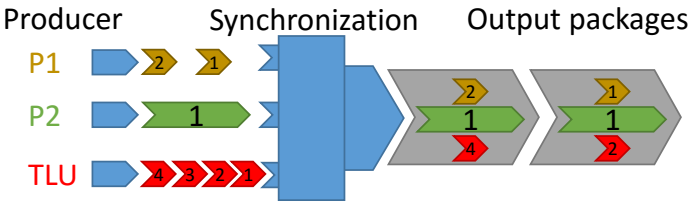
These small examples are given to illustrate the power of the new processor class and the dataflow paradigm it is based on. The imperative parts of the program are hidden inside the processors. This does not just make the code a lot more readable, it also reduces the effort of testing the code.

Listing 8.6 The use of processors in the Data-Collector.

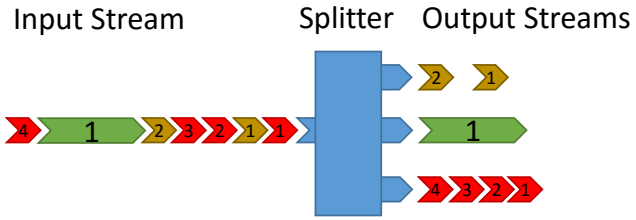
```
void DataCollector::OnConfigure(...) {
...
  m_batch = make_batch();
  *m_batch >> m_dataReceiver.get()
    >> waitForEORE(TimeOut)
    >> ShowEventNR(N)
    >> m_pwriter.get();
...
}


void DataCollector::OnPrepareRun(...) {
...
    m_batch->init();
...
}


void DataCollector::OnStopRun() {
...
  m_batch->wait();
  m_batch->end();
...
}
```



(a) Back end of the synchronization processor



(b) Front end of the Synchronization processor

Figure 8.9 Front End and Back End of the new synchronization algorithm

8.7 Asynchronous Data Streams

EUDAQ 1.x has a trigger based layout. All the information produced by any producer has to correspond to exactly one trigger. This was a very successful approach and for many use-cases this is still the best approach, but nevertheless there are cases for which it can become a burden. The typical example for this is if one has two devices with a very different integration time, like for example the FEI4 APIX detector and the MIMOSA26 detector. The FEI4 APIX has an integration time of 25 ns while the MIMOSA26 has an integration time of 114.5 μs. In EUDAQ 1.x this leads to a limitation of the trigger rate. The maximum trigger rate is set by the slowest device. To overcome this limitation, EUDAQ 2.0 allows every producer to run at its own speed. The synchronization of the individual data streams is done offline and is very flexible. It is foreseen that user groups can write their own synchronization algorithm and contribute it to the EUDAQ framework. This work will only present the one synchronization algorithm that is most similar to the synchronization done in EUDAQ 1.x. It merges individual data streams to one stream of detectorEvents[2]. This merging is very close to the merging the Data-Collector from EUDAQ 1.x performs. The one difference is that EUDAQ 2.0 is not restricted to synchronize based only on event numbers.

[2]DetectorEvents are a special kind of packages which can store multiple events from different producers

8.7.1 The Back End

Figure 8.9a shows an overview of the working principle of the new synchronization processor. It takes many data streams and produces one output stream of synchronized events. It does it by comparing each event from P1 and P2 to the events from the TLU. EUDAQ 2.0 does not make any assumptions of the data layout of user's data. Therefore the function which compares both events is part of the Data-Converter-Plugin (IsSyncWithTLU). There are three different relations the TLU event and the DUT event can have. The first being that the DUT event is early (Event_IS_EARLY). In EUDAQ 1.x terms this would correspond to an event with a larger event number than the TLU. In EUDAQ 2.0 terms it means it belongs to a later time than the TLU. In this case the current TLU event is dismissed and the next TLU event is tested. Since a new TLU event is used, the cycle of comparing the individual data streams against the TLU starts again with the first data stream. The second is that the event is late (Event_IS_LATE). In EUDAQ 1.x terms this would correspond to a DUT event with a smaller event number than the TLU. In EUDAQ 2.0 terms it means it belongs to a earlier time than the TLU. In this case the current DUT event is dismissed and the next DUT event from this data stream is tested. The third is that the events are in sync (Event_IS_Sync). In this case the event is marked as in sync and the cycle of comparing DUT and TLU events continues with the next DUT stream. When a whole cycle of comparing the DUT events with the TLU event is completed, and all have been marked as in sync, all events of this cycle are stored in a DetectorEvent and it is sent to the next processor in line.

8.7.2 The Front End

Technically it is possible to use this synchronization algorithm online in the Data-Collector. In fact to ensure compatibility with EUDAQ 1.x, it is the default. For new devices that, for example, rely on some internal timing to synchronize to the TLU, this approach is not recommended since it can cause data loss if the synchronization is not working correctly. Events that cannot be synchronized to the TLU are completely disposed of. For devices at an early level of integration it is recommended to store the data on disc without synchronizing. To handle this kind of data, the synchronization processor has a Front End which can convert the data stored in one data stream back to individual data streams (see figure 8.9b). In addition to this it can also unpack the data in already packed data streams. This allows the synchronization algorithm to resynchronize streams that have been wrongly synchronized. The Front End is designed to detect what kind of data stream is arriving at the processor and is able to convert it into a format that the back end expects as an input.

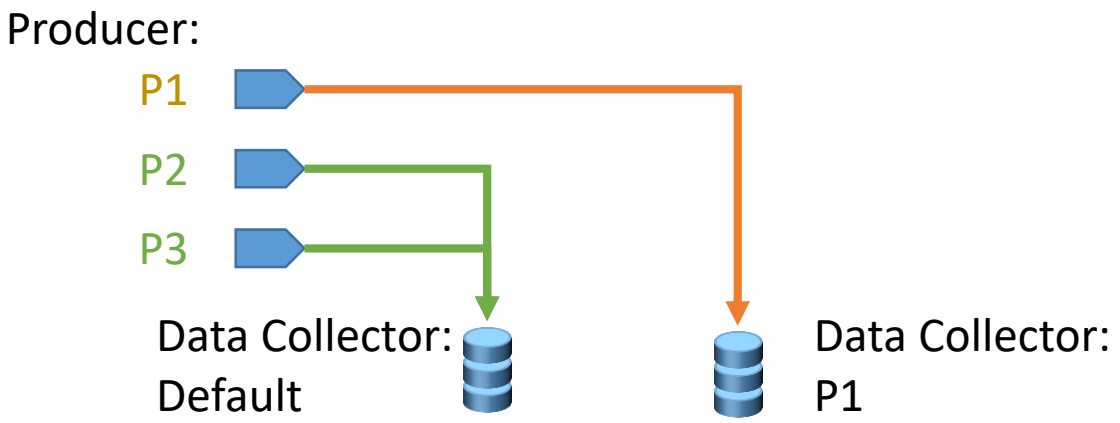


Figure 8.10 Multiple Data-Collector

8.8 Multiple Data-Collector

Another limitation of EUDAQ 1.x was that all producers had to send their data to one centralized Data-Collector. For many purposes this approach is reasonable, but at some point it will be a limiting factor for the maximal rate. To give the users more flexibility EUDAQ 2.0 allows for every producer to have its own Data-Collector[3]. Figure 8.10 shows an example for multiple Data-Collector. It shows one named Data-Collector (P1) and one default Data-Collector. The producer with the same name as the Data-Collector will connect to this Data-Collector, and the others will connect to the default data collector. The physical position of the producer and Data-Collector is not important as long as they can be reached via the network connection. This can mean the named Data-Collector P1 can be on the same machine as the producer P1, but it can also mean that the Data-Collector is on a different PC.

8.9 Summary and Outlook

The goals of EUDAQ 2.0 were to allow an easier integration of new devices which do not necessarily fit the one trigger one ReadOut Frame (ROF) approach of EUDAQ 1.x, and to remove the bottlenecks of EUDAQ 1.x to increase the maximal trigger rate by about two orders of magnitude, to 10^6 s^{-1}.

To achieve these goals it was necessary to redesign large parts of the framework. A central point of all changes was to transparently handle data streams inside the framework. For this reason a new processor base class was introduced. One of the first implemented

[3]The designed and implementation was mainly done by Hanno Perrey [**?**]

processor classes was the synchronization processor, which allows the synchronization of data streams on user defined quantities. For example, the data stream of one user's device can still be synchronized with the TLU by using the trigger number, while the data stream of the next user's device is synchronized to the TLU based on the timestamps. This allows devices with different integration times to use the framework with the highest efficiency. Since the data streams are now transparently handled with the processor classes, it is also possible for the user to implement their own synchronization class if needed. Further more, for an easier integration of DAQ systems based on the CERN ROOT framework, EUDAQ 2.0 provides an interface to ROOT. This allows one to develop the user's DAQ and EUDAQ independently. This new interface is already in use by the ITK-strip test beam community.

In order to increase the data rate two steps were necessary to undertake. Firstly, the one trigger equals one ROF approach was not anymore practical. The maximal trigger rate of the MIMOSA26 is limited by the long integration time. Therefore it was necessary to provide an algorithm for the synchronization of data streams which cover a different time frame (asynchronous data streams). Secondly, it was necessary to remove bottlenecks, such as the overhead of the event class, or the network overhead, from transferring all data to a centralized Data-Collector. With EUDAQ 2.0 it is now possible to store many ROFs in one packet as well as storing data on a local Data-Collector running on the same machine as the readout. The synchronization is, in this case, postponed to the offline analysis.

In summary, all major goals for EUDAQ 2.0 have been achieved. From the software side the project is prepared for new hardware like the miniTLU [**?**] which will increase the maximal possible trigger rate to more than 10^6 s^{-1}, and for new hardware like the CALICE calorimeter. The effort for existing groups to upgrade from EUDAQ 1.x to EUDAQ 2.0 is minimal since EUDAQ 1.x is a subset of EUDAQ 2.0.

Chapter 9

Conclusions

Building a new tracker for a major High Energy Physics detector is always a challenging task. The goal of the new Inner TracKer (ITK) is keep performance of current ID under much harsher conditions. Every single detector part, beginning from the support structures, to the silicon sensor, to the readout chips, and beyond are specifically designed for this one purpose. Due to the typically harsh radiation environment, and the uniqueness of the tracker design, there are by default no off-the-shelf solutions. Even though there is a lot of experience with silicon detectors in High Energy Physics, the performance of every new developed building block has to be thoroughly investigated under the most realistic conditions achievable. When the first ABC130s for the phase II ATLAS strip detector were manufactured, and the built-in tests for the gain of the preamplifier gave significantly smaller value than expected and non-conclusive results, it was clear that a test beam campaign was needed.

This thesis combines all aspects of performing a successful test beam campaign: From the first attempts to connect the two involved DAQ systems, to the building of laboratory test stands which can check the synchronicity of the data streams from the individual devices, up to performing the test beam campaign itself. Beyond this, the thesis also covers the necessary developments for the standard EUDET type beam telescope data acquisition system (EUDAQ) to optimally operate with an LHC style device.

The newly developed ABC130 readout chip showed, with $< 75\ \frac{\mathrm{mV}}{\mathrm{fC}}$, a much lower preamplifier gain than the anticipated $90 - 95\ \frac{\mathrm{mV}}{\mathrm{fC}}$. Since the built-in tests will always be biased, it was necessary to create a performance test that did not rely on the readout chip itself. The solution for this was to use the well understood charge spectrum of a silicon sensor as input to the ABC130. By measuring the performance of the readout chip in combination with the actual silicon strip sensor, this setup also gives insights into the performance of the whole system.

The first task was to synchronize the two DAQ systems for the telescopes (EUDAQ) and for the ATLAS strip sensors (ITSDAQ). To shift as much work as possible before the start of the test beam itself, two laboratory test stands have been designed in the context of the work presented here to check for the synchronization. One system was using timestamps and the other one was using an LED to induce signals in the system. The two methods showed that, as long as the trigger rate is slow enough, the devices stay in sync. It also showed that in order to perform test beams with rates above roughly 100s^{-1} it is necessary to integrate the TLU handshake into the readout system.

Before performing a test beam campaign a beta source measurement was performed. It was used to give some initial understanding of the behavior of the ABC130. It became especially visible that the charge sharing between silicon strips has a strong effect on the actual input charge to the preamplifier. Even with a wide pitch size of $74.5\,\mu$m, it is still necessary to correct for the charge sharing. The correction factor is extracted by performing a control measurement with an analog readout system (ALiBaVa). After correcting for the charge sharing, the gain obtained from the beta source measurement is $93.6 \pm 4.3\ \frac{\text{mV}}{\text{fC}}$ and therefore within the specifications.

Due to the limitations of the beta source measurements, especially the lack of spacial resolution, the low particle energy of only ≈ 2.3 MeV and the very limited rate of only $2\ \text{s}^{-1}$, it was necessary to perform a test beam campaign to obtain more precise data. In May 2015 a test beam campaign of two weeks was performed on two identical devices of four ATLAS12 mini sensors, with the readout by ABC130 mounted to a module hybrid. The track reconstruction achieved a pointing resolution of $11\,\mu$m, which is precise enough to perform a strip-by-strip analysis, but has some limitations for the in-strip analysis. For the analysis, three methods are investigated. It turned out that the variation between the individual readout chips is smaller than the systematic variation between the three analysis methods. The results range from $\approx 90\ \frac{\text{mV}}{\text{fC}}$ to $\approx 100\ \frac{\text{mV}}{\text{fC}}$. As such, they are consistent with the result obtained from the beta source measurement and close to the specification of the ABC130. However the test beam also showed that an observed non linearity of the preamplifier can distort the result of the gain measurement. When using an Integrated Landau-Function to fit the S-Curves, it can distort the fit value quite significantly. Also the difference between the small signal gain and the working point gain should not be underestimated.

A second major part of this work was the complete overhaul of the EUDAQ (EUDAQ 2.0) framework. In order to optimally synchronize devices with a vastly different integration time a new scheme for the synchronization was developed and implemented. Since the EUDET type beam telescopes are slow devices[1] it was necessary to create a DAQ system that can

[1] Devices with a rolling shutter readout and an integration time of $114.5\,\mu$s

easily combine those with devices with a much shorter integration time[2]. It was required to move from a trigger-based readout system to a readout system that does not have a preferred means of synchronization, but allows the users to implement their own one as part of their Data-Converter-Plugin. With this and the other changes made to the EUDAQ framework, it has become a modern and compelling DAQ system which is widely used and also adapted by new groups such as the CALICE group.

In summary, the development of new test stands for detailed studies of the ATLAS ABC130 and the integration of the different DAQ systems to use with the pixel telescopes was successful. Based on these studies, the Front End stage of the ABC130 will be optimized for the next iteration.

[2]LHC-type devices with a typical integration time of 25 ns

Acknowledgements

I want to thank Prof. Dr. Joachim Mnich and Prof. Dr. Erika Garutti for the possibility to do my dissertation at this exciting topic on the excellent research facility “Deutsches Elektronen-Synchrotron” (DESY) in collaboration with the "Universität Hamburg". I have to thank my supervisor Dr. Ingrid-Maria Gregor for the tremendous support throughout the entire PhD work.

A special thanks goes to the ATLAS group at the University of Freiburg for their strong support and the possibility to use their laboratories for the beta source measurements. Here I want to thank especially Dr. Susanne Kühn, Marc Hauser and Dr. Riccardo Mori.

Furthermore I want to thank everybody involved in the test beam campaign. Dr. Peter Philips (RAL) and Dr. Bruce Gallop (RAL) for the preparation of the DUT and the ITSDAQ software. Tobias Bisanz (Universität Göttingen) for his support during the test beam and for the reconstruction. Without his help it would not have been possible to set up the FEI4 APIX sensor in time. Also for the reconstruction of the data his deep knowledge of the EUTelescope framework was an essential part to the success of this thesis. Next I want to thank Alexander Morton (University of Glasgow) who was responsible for the implementation of the GBL software to the EUTelescope framework. This software was a key component of the data analysis. Lucrezia Bruni (NIKHEF) who devoted a lot of time on the reconstruction of the data.

I would like to express my special thanks of gratitude to my colleagues at the Test Beam group at DESY for the support, challenging discussions and lots of helpful advices for my thesis. A special thanks goes to Dr. Marcel Stanitzki, Dr. Igor Rubinsky, Dr. Jan Dreyling-Eschweiler, Dr. Hanno Perrey, Dr. Hendrik Jansen, Eda Yildirim, Dr. Simon Spannagel and Thomas Daubney.

Declaration

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Dissertationsschrift selbst verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Richard Peschke
December 2016